

## **82V3910 WANPLL – Application Programmer's Interface Reference Manual**

Version 1.1

September 23, 2013

| Version | Date               | Description                                                                                                                                           |
|---------|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1.0     | May 23, 2013       | Initial release.                                                                                                                                      |
| 1.1     | September 23, 2013 | <ul style="list-style-type: none"> <li>- Add support for APLL pre-divider value 1.</li> <li>- Update Ethernet FEC frequency configuration.</li> </ul> |
|         |                    |                                                                                                                                                       |

# Contents

|          |                                        |          |
|----------|----------------------------------------|----------|
| <b>1</b> | <b>Module Index</b>                    | <b>1</b> |
| 1.1      | Modules                                | 1        |
| <b>2</b> | <b>Data Structure Index</b>            | <b>3</b> |
| 2.1      | Data Structures                        | 3        |
| <b>3</b> | <b>File Index</b>                      | <b>5</b> |
| 3.1      | File List                              | 5        |
| <b>4</b> | <b>Module Documentation</b>            | <b>7</b> |
| 4.1      | DPLL Private header file               | 7        |
| 4.1.1    | Detailed Description                   | 8        |
| 4.1.2    | Macro Definition Documentation         | 8        |
| 4.1.2.1  | SonetGigEthSel_NUMMAX                  | 8        |
| 4.1.3    | Enumeration Type Documentation         | 8        |
| 4.1.3.1  | T_idtSonetGigEthSelector               | 8        |
| 4.1.3.2  | T_idtT0DpllSelectorA                   | 8        |
| 4.1.3.3  | T_idtT0DpllSelectorB                   | 9        |
| 4.1.3.4  | T_idtT4DpllSelectorA                   | 9        |
| 4.1.3.5  | T_idtT4DpllSelectorB                   | 9        |
| 4.1.4    | Variable Documentation                 | 9        |
| 4.1.4.1  | IDTDpll_pathConfigurationNULL_c        | 9        |
| 4.1.4.2  | IDTDpll_sonetGigEthPathConfig2Bitfield | 9        |
| 4.2      | DPLL Function Module                   | 10       |
| 4.2.1    | Detailed Description                   | 15       |
| 4.2.2    | Macro Definition Documentation         | 15       |
| 4.2.2.1  | DCO_MAX_PPT                            | 15       |
| 4.2.2.2  | DCO_MIN_PPT                            | 15       |
| 4.2.2.3  | DCO_PPT2DCOUNTS                        | 15       |
| 4.2.2.4  | IDT_TRANSLATE_EXT_TO_INT_DPLL          | 16       |
| 4.2.3    | Enumeration Type Documentation         | 16       |
| 4.2.3.1  | T_idtBandwidthLimitStage               | 16       |

|          |                                      |    |
|----------|--------------------------------------|----|
| 4.2.3.2  | T_idtCoarsePhaseLimit                | 16 |
| 4.2.3.3  | T_idtDampingFactor                   | 17 |
| 4.2.3.4  | T_idtDpllBandwidth                   | 17 |
| 4.2.3.5  | T_idtDpllHoldover                    | 18 |
| 4.2.3.6  | T_idtDpllOperMode                    | 18 |
| 4.2.3.7  | T_idtDpllOutputSelectorA             | 18 |
| 4.2.3.8  | T_idtDpllOutputSelectorB             | 19 |
| 4.2.3.9  | T_idtDpllPpsPhasePosition            | 19 |
| 4.2.3.10 | T_idtDpllPpsPulseWidth               | 19 |
| 4.2.3.11 | T_idtFinePhaseLimit                  | 20 |
| 4.2.3.12 | T_idtFreqMonFactor                   | 20 |
| 4.2.3.13 | T_idtOperDpllMode                    | 20 |
| 4.2.3.14 | T_idtPhaseSlope                      | 21 |
| 4.2.3.15 | T_idtSonetGigEthOutput               | 21 |
| 4.2.3.16 | T_idtTemp_holdover                   | 21 |
| 4.2.4    | Function Documentation               | 21 |
| 4.2.4.1  | IDTDpll_Get1stPriorityInput          | 21 |
| 4.2.4.2  | IDTDpll_Get2ndPriorityInput          | 22 |
| 4.2.4.3  | IDTDpll_Get3rdPriorityInput          | 22 |
| 4.2.4.4  | IDTDpll_GetAutoSelBandWidth          | 22 |
| 4.2.4.5  | IDTDpll_GetAutoSelInput              | 22 |
| 4.2.4.6  | IDTDpll_GetBandWidthDamping          | 23 |
| 4.2.4.7  | IDTDpll_GetCurrentDpllFreqOffset     | 24 |
| 4.2.4.8  | IDTDpll_GetCurrentPhaseError         | 24 |
| 4.2.4.9  | IDTDpll_GetCurrentSelectInput        | 24 |
| 4.2.4.10 | IDTDpll_GetDpll16E116T1Enable        | 24 |
| 4.2.4.11 | IDTDpll_GetDpllEthPathEnable         | 25 |
| 4.2.4.12 | IDTDpll_GetDpllFrozen                | 25 |
| 4.2.4.13 | IDTDpll_GetDpllHardAlarmEnable       | 25 |
| 4.2.4.14 | IDTDpll_GetDpllHardAlarmLimit        | 26 |
| 4.2.4.15 | IDTDpll_GetDpllOperMod               | 26 |
| 4.2.4.16 | IDTDpll_GetDpllOutputPathSelectorA   | 26 |
| 4.2.4.17 | IDTDpll_GetDpllOutputPathSelectorB   | 27 |
| 4.2.4.18 | IDTDpll_GetDpllSelAPathEnable        | 27 |
| 4.2.4.19 | IDTDpll_GetDpllSelBPathEnable        | 27 |
| 4.2.4.20 | IDTDpll_GetDpllSoftAlarmLimit        | 27 |
| 4.2.4.21 | IDTDpll_GetFastLossSwitch            | 28 |
| 4.2.4.22 | IDTDpll_GetForceSelInput             | 28 |
| 4.2.4.23 | IDTDpll_GetFrequencyMonitoringFactor | 28 |
| 4.2.4.24 | IDTDpll_GetHardAlarmThreshold        | 29 |

|          |                                      |    |
|----------|--------------------------------------|----|
| 4.2.4.25 | IDTDpll_GetHitlessSwitchingFeature   | 29 |
| 4.2.4.26 | IDTDpll_GetHoldoverFreq              | 29 |
| 4.2.4.27 | IDTDpll_GetHoldoverMode              | 30 |
| 4.2.4.28 | IDTDpll_GetIcpCtrl                   | 30 |
| 4.2.4.29 | IDTDpll_GetPhaseCoarseDetector       | 30 |
| 4.2.4.30 | IDTDpll_GetPhaseCoarseLimit          | 31 |
| 4.2.4.31 | IDTDpll_GetPhaseFineDetectorEnable   | 31 |
| 4.2.4.32 | IDTDpll_GetPhaseFineLimit            | 32 |
| 4.2.4.33 | IDTDpll_GetPhaseOffset               | 32 |
| 4.2.4.34 | IDTDpll_GetPhaseSlope                | 32 |
| 4.2.4.35 | IDTDpll_GetPhaseTransient            | 33 |
| 4.2.4.36 | IDTDpll_GetPpsFastLock               | 33 |
| 4.2.4.37 | IDTDpll_GetPpsPhase                  | 33 |
| 4.2.4.38 | IDTDpll_GetSoftAlarmThreshold        | 34 |
| 4.2.4.39 | IDTDpll_GetSonetGigEthOutputPath     | 34 |
| 4.2.4.40 | IDTDpll_GetTempHoldoverMode          | 34 |
| 4.2.4.41 | IDTDpll_GetTypeOfDpll                | 35 |
| 4.2.4.42 | IDTDpll_IsDpllLocked                 | 35 |
| 4.2.4.43 | IDTDpll_SetAutoSelBandWidth          | 35 |
| 4.2.4.44 | IDTDpll_SetAutoSelInput              | 36 |
| 4.2.4.45 | IDTDpll_SetBandWidthDamping          | 36 |
| 4.2.4.46 | IDTDpll_SetDpll16E116T1Enable        | 37 |
| 4.2.4.47 | IDTDpll_SetDpllEthPathEnable         | 37 |
| 4.2.4.48 | IDTDpll_SetDpllFrozen                | 37 |
| 4.2.4.49 | IDTDpll_SetDpllHardAlarmEnable       | 37 |
| 4.2.4.50 | IDTDpll_SetDpllHardAlarmLimit        | 38 |
| 4.2.4.51 | IDTDpll_SetDpllOperMod               | 38 |
| 4.2.4.52 | IDTDpll_SetDpllOutputPathSelectorA   | 38 |
| 4.2.4.53 | IDTDpll_SetDpllOutputPathSelectorB   | 39 |
| 4.2.4.54 | IDTDpll_SetDpllSelAPathEnable        | 39 |
| 4.2.4.55 | IDTDpll_SetDpllSelBPathEnable        | 39 |
| 4.2.4.56 | IDTDpll_SetDpllSoftAlarmLimit        | 39 |
| 4.2.4.57 | IDTDpll_SetFastLossSwitch            | 40 |
| 4.2.4.58 | IDTDpll_SetForceSelInput             | 40 |
| 4.2.4.59 | IDTDpll_SetFrequencyMonitoringFactor | 40 |
| 4.2.4.60 | IDTDpll_SetHardAlarmThreshold        | 41 |
| 4.2.4.61 | IDTDpll_SetHitlessSwitchingFeature   | 41 |
| 4.2.4.62 | IDTDpll_SetHoldoverFreq              | 41 |
| 4.2.4.63 | IDTDpll_SetHoldoverMode              | 41 |
| 4.2.4.64 | IDTDpll_SetIcpCtrl                   | 42 |

|          |                                    |    |
|----------|------------------------------------|----|
| 4.2.4.65 | IDTDpll_SetPhaseCoarseDetector     | 42 |
| 4.2.4.66 | IDTDpll_SetPhaseCoarseLimit        | 43 |
| 4.2.4.67 | IDTDpll_SetPhaseFineDetectorEnable | 43 |
| 4.2.4.68 | IDTDpll_SetPhaseFineLimit          | 43 |
| 4.2.4.69 | IDTDpll_SetPhaseOffset             | 44 |
| 4.2.4.70 | IDTDpll_SetPhaseSlope              | 44 |
| 4.2.4.71 | IDTDpll_SetPhaseTransient          | 44 |
| 4.2.4.72 | IDTDpll_SetPpsFastLock             | 45 |
| 4.2.4.73 | IDTDpll_SetPpsPhase                | 45 |
| 4.2.4.74 | IDTDpll_SetSoftAlarmThreshold      | 45 |
| 4.2.4.75 | IDTDpll_SetSonetGigEthOutputPath   | 46 |
| 4.2.4.76 | IDTDpll_SetTempHoldoverMode        | 46 |
| 4.3      | General Function Module            | 47 |
| 4.3.1    | Detailed Description               | 48 |
| 4.3.2    | Macro Definition Documentation     | 48 |
| 4.3.2.1  | Activate_Edge_Falling              | 48 |
| 4.3.2.2  | Activate_Edge_Rising               | 48 |
| 4.3.2.3  | INT_Active_High                    | 49 |
| 4.3.2.4  | INT_Active_Low                     | 49 |
| 4.3.2.5  | NOMINAL_2COMPL_TO_PPT              | 49 |
| 4.3.2.6  | NOMINAL_MAX_PPT                    | 49 |
| 4.3.2.7  | NOMINAL_MIN_PPT                    | 49 |
| 4.3.2.8  | NOMINAL_PPT_TO_2COMPL              | 49 |
| 4.3.3    | Enumeration Type Documentation     | 49 |
| 4.3.3.1  | networkType_t                      | 49 |
| 4.3.3.2  | phaseTimeoutMultiplier_t           | 49 |
| 4.3.4    | Function Documentation             | 50 |
| 4.3.4.1  | IDTGeneral_DeInitDriver            | 50 |
| 4.3.4.2  | IDTGeneral_GetDeviceID             | 50 |
| 4.3.4.3  | IDTGeneral_GetFlagOnTDO            | 50 |
| 4.3.4.4  | IDTGeneral_GetHardAlarm            | 50 |
| 4.3.4.5  | IDTGeneral_GetNetworkType          | 51 |
| 4.3.4.6  | IDTGeneral_GetOscActiveEdge        | 51 |
| 4.3.4.7  | IDTGeneral_GetOscFreqOffset        | 51 |
| 4.3.4.8  | IDTGeneral_GetRevertiveMode        | 52 |
| 4.3.4.9  | IDTGeneral_GetTimeoutValue         | 52 |
| 4.3.4.10 | IDTGeneral_GetUltraFastSwitch      | 52 |
| 4.3.4.11 | IDTGeneral_i2cTranslate            | 52 |
| 4.3.4.12 | IDTGeneral_InitDeviceCommon        | 53 |
| 4.3.4.13 | IDTGeneral_InitDriverCommon        | 53 |

|          |                                             |    |
|----------|---------------------------------------------|----|
| 4.3.4.14 | IDTGeneral_SetFlagOnTDO . . . . .           | 53 |
| 4.3.4.15 | IDTGeneral_SetFullyUnprotectMode . . . . .  | 53 |
| 4.3.4.16 | IDTGeneral_SetHardAlarm . . . . .           | 53 |
| 4.3.4.17 | IDTGeneral_SetNetworkType . . . . .         | 54 |
| 4.3.4.18 | IDTGeneral_SetOscActiveEdge . . . . .       | 54 |
| 4.3.4.19 | IDTGeneral_SetOscFreqOffset . . . . .       | 54 |
| 4.3.4.20 | IDTGeneral_SetProtectMode . . . . .         | 54 |
| 4.3.4.21 | IDTGeneral_SetRevertiveMode . . . . .       | 54 |
| 4.3.4.22 | IDTGeneral_SetSingleUnprotectMode . . . . . | 55 |
| 4.3.4.23 | IDTGeneral_SetTimeoutValue . . . . .        | 55 |
| 4.3.4.24 | IDTGeneral_SetUltraFastSwitch . . . . .     | 55 |
| 4.4      | Hardware Abstraction Layer Module . . . . . | 56 |
| 4.4.1    | Detailed Description . . . . .              | 56 |
| 4.4.2    | Macro Definition Documentation . . . . .    | 56 |
| 4.4.2.1  | IDT_HAL_USE_MACRO . . . . .                 | 56 |
| 4.4.2.2  | IDTHAL_GetBitfield . . . . .                | 56 |
| 4.4.2.3  | IDTHAL_ModifyBitfield . . . . .             | 57 |
| 4.4.3    | Function Documentation . . . . .            | 57 |
| 4.4.3.1  | IDTHal_Read . . . . .                       | 57 |
| 4.4.3.2  | IDTHal_Read_Ext . . . . .                   | 57 |
| 4.4.3.3  | IDTHAL_ReadBitfield . . . . .               | 57 |
| 4.4.3.4  | IDTHAL_ReadBitfield_Ext . . . . .           | 58 |
| 4.4.3.5  | IDTHal_Write . . . . .                      | 58 |
| 4.4.3.6  | IDTHal_Write_Ext . . . . .                  | 58 |
| 4.4.3.7  | IDTHAL_WriteBitfield . . . . .              | 59 |
| 4.4.3.8  | IDTHAL_WriteBitfield_Ext . . . . .          | 59 |
| 4.5      | Input Function Module . . . . .             | 60 |
| 4.5.1    | Detailed Description . . . . .              | 62 |
| 4.5.2    | Macro Definition Documentation . . . . .    | 63 |
| 4.5.2.1  | IDT_EXT_INPORT_POPULATED . . . . .          | 63 |
| 4.5.2.2  | IDT_TRANSLATE_EXT_TO_INT_INPORT . . . . .   | 63 |
| 4.5.2.3  | Input_Freq_Hard_Alarm . . . . .             | 63 |
| 4.5.2.4  | Input_No_Activity_Alarm . . . . .           | 63 |
| 4.5.2.5  | Input_Phase_Lock_Alarm . . . . .            | 63 |
| 4.5.3    | Enumeration Type Documentation . . . . .    | 63 |
| 4.5.3.1  | T_externalSyncAlarmRange . . . . .          | 63 |
| 4.5.3.2  | T_externalSyncBypass . . . . .              | 64 |
| 4.5.3.3  | T_externalSyncEdge . . . . .                | 64 |
| 4.5.3.4  | T_externalSyncEnable . . . . .              | 64 |
| 4.5.3.5  | T_externalSyncSampling . . . . .            | 65 |

|          |                                               |    |
|----------|-----------------------------------------------|----|
| 4.5.3.6  | T_idtAmiInputFrequencyBitFieldValue . . . . . | 65 |
| 4.5.3.7  | T_idtHighFrequencyDivisor . . . . .           | 65 |
| 4.5.3.8  | T_idtInputDividerMux . . . . .                | 65 |
| 4.5.3.9  | T_idtInputFrequencyBitfieldValue . . . . .    | 66 |
| 4.5.3.10 | T_idtInputFrequencyFamily . . . . .           | 66 |
| 4.5.3.11 | T_idtInputSyncFreq . . . . .                  | 67 |
| 4.5.4    | Function Documentation . . . . .              | 67 |
| 4.5.4.1  | IDTInput_DisableAllInputs . . . . .           | 67 |
| 4.5.4.2  | IDTInput_EnableAllInputs . . . . .            | 67 |
| 4.5.4.3  | IDTInput_GetActivityMonitor . . . . .         | 67 |
| 4.5.4.4  | IDTInput_GetAmiInputFrequency . . . . .       | 68 |
| 4.5.4.5  | IDTInput_GetBucketConfig . . . . .            | 68 |
| 4.5.4.6  | IDTInput_GetDivisionFactor . . . . .          | 68 |
| 4.5.4.7  | IDTInput_GetInputClockStatus . . . . .        | 68 |
| 4.5.4.8  | IDTInput_GetInputClockValidation . . . . .    | 69 |
| 4.5.4.9  | IDTInput_GetInputEnable . . . . .             | 69 |
| 4.5.4.10 | IDTInput_GetInputFreqOffset . . . . .         | 69 |
| 4.5.4.11 | IDTInput_GetInputFrequency . . . . .          | 70 |
| 4.5.4.12 | IDTInput_GetInputHFdivider . . . . .          | 70 |
| 4.5.4.13 | IDTInput_GetPreDivider . . . . .              | 70 |
| 4.5.4.14 | IDTInput_GetPriority . . . . .                | 71 |
| 4.5.4.15 | IDTInput_GetSyncAlarmRange . . . . .          | 71 |
| 4.5.4.16 | IDTInput_GetSyncBypass . . . . .              | 71 |
| 4.5.4.17 | IDTInput_GetSyncEdge . . . . .                | 72 |
| 4.5.4.18 | IDTInput_GetSyncEnable . . . . .              | 72 |
| 4.5.4.19 | IDTInput_GetSyncFrequency . . . . .           | 72 |
| 4.5.4.20 | IDTInput_GetSyncSampling . . . . .            | 72 |
| 4.5.4.21 | IDTInput_InFreqBitValue2Family . . . . .      | 73 |
| 4.5.4.22 | IDTInput_SetActivityMonitor . . . . .         | 73 |
| 4.5.4.23 | IDTInput_SetAmiInputFrequency . . . . .       | 73 |
| 4.5.4.24 | IDTInput_SetBucketConfig . . . . .            | 73 |
| 4.5.4.25 | IDTInput_SetDivisionFactor . . . . .          | 74 |
| 4.5.4.26 | IDTInput_SetInputEnable . . . . .             | 74 |
| 4.5.4.27 | IDTInput_SetInputFrequency . . . . .          | 74 |
| 4.5.4.28 | IDTInput_SetInputHFdivider . . . . .          | 75 |
| 4.5.4.29 | IDTInput_SetPreDivider . . . . .              | 75 |
| 4.5.4.30 | IDTInput_SetPriority . . . . .                | 76 |
| 4.5.4.31 | IDTInput_SetSyncAlarmRange . . . . .          | 76 |
| 4.5.4.32 | IDTInput_SetSyncBypass . . . . .              | 76 |
| 4.5.4.33 | IDTInput_SetSyncEdge . . . . .                | 76 |

|          |                                  |     |
|----------|----------------------------------|-----|
| 4.5.4.34 | IDTInput_SetSyncEnable           | 76  |
| 4.5.4.35 | IDTInput_SetSyncFrequency        | 77  |
| 4.5.4.36 | IDTInput_SetSyncSampling         | 77  |
| 4.6      | Output Function Module           | 78  |
| 4.6.1    | Detailed Description             | 79  |
| 4.6.2    | Macro Definition Documentation   | 79  |
| 4.6.2.1  | IDT_EXT_OUTPORT_POPULATED        | 79  |
| 4.6.2.2  | IDT_TRANSLATE_EXT_TO_INT_OUTPORT | 79  |
| 4.6.3    | Enumeration Type Documentation   | 79  |
| 4.6.3.1  | T_frameSync                      | 80  |
| 4.6.3.2  | T_outputInterface                | 80  |
| 4.6.3.3  | T_outputPathType                 | 80  |
| 4.6.4    | Function Documentation           | 80  |
| 4.6.4.1  | IDTOutput_DisableAllIntOutput    | 80  |
| 4.6.4.2  | IDTOutput_DisableApplInput       | 81  |
| 4.6.4.3  | IDTOutput_GetFrameSync           | 81  |
| 4.6.4.4  | IDTOutput_GetFrameSyncPulseEdge  | 81  |
| 4.6.4.5  | IDTOutput_GetOutputpInterface    | 81  |
| 4.6.4.6  | IDTOutput_GetOutputConfig        | 82  |
| 4.6.4.7  | IDTOutput_GetOutputEnable        | 82  |
| 4.6.4.8  | IDTOutput_GetOutputInvert        | 82  |
| 4.6.4.9  | IDTOutput_SetFrameSync           | 83  |
| 4.6.4.10 | IDTOutput_SetFrameSyncPulseEdge  | 83  |
| 4.6.4.11 | IDTOutput_SetOutputpInterface    | 83  |
| 4.6.4.12 | IDTOutput_SetOutputConfig        | 83  |
| 4.6.4.13 | IDTOutput_SetOutputEnable        | 84  |
| 4.6.4.14 | IDTOutput_SetOutputInvert        | 84  |
| 4.7      | Register definitions             | 85  |
| 4.7.1    | Detailed Description             | 93  |
| 4.7.2    | Enumeration Type Documentation   | 93  |
| 4.7.2.1  | anonymous enum                   | 93  |
| 4.7.2.2  | anonymous enum                   | 115 |
| 4.8      | Appl Function Module             | 118 |
| 4.8.1    | Detailed Description             | 122 |
| 4.8.2    | Macro Definition Documentation   | 122 |
| 4.8.2.1  | IDT_APLL_API_LOG_DEBUG           | 122 |
| 4.8.2.2  | IDT_APLL_API_LOG_ERR             | 122 |
| 4.8.2.3  | IDT_APLL_API_LOG_HEADER          | 122 |
| 4.8.2.4  | IDT_APLL_API_LOG_INFO            | 123 |
| 4.8.2.5  | IDT_APLL_API_TRACE               | 123 |

|          |                                          |     |
|----------|------------------------------------------|-----|
| 4.8.2.6  | IDT_GET_OUTPUT_APLL_CONFIG . . . . .     | 123 |
| 4.8.2.7  | INVALID_DIVIDER_VALUE . . . . .          | 123 |
| 4.8.3    | Typedef Documentation . . . . .          | 123 |
| 4.8.3.1  | T_idtAplDividerValue . . . . .           | 123 |
| 4.8.3.2  | T_idtAplRegAddr . . . . .                | 123 |
| 4.8.3.3  | T_idtAplRegMask . . . . .                | 124 |
| 4.8.3.4  | T_idtAplRegVal . . . . .                 | 124 |
| 4.8.4    | Enumeration Type Documentation . . . . . | 124 |
| 4.8.4.1  | T_idtAplAccessType . . . . .             | 124 |
| 4.8.4.2  | T_idtAplBypass . . . . .                 | 124 |
| 4.8.4.3  | T_idtAplConfigSel . . . . .              | 124 |
| 4.8.4.4  | T_idtAplFreqDoublerSel . . . . .         | 125 |
| 4.8.4.5  | T_idtAplId . . . . .                     | 125 |
| 4.8.4.6  | T_idtAplInputClkSrc . . . . .            | 125 |
| 4.8.4.7  | T_idtAplInputFreqType . . . . .          | 125 |
| 4.8.4.8  | T_idtAplMasterResetSync . . . . .        | 126 |
| 4.8.4.9  | T_idtAplOutput . . . . .                 | 126 |
| 4.8.4.10 | T_idtAplOutputEn . . . . .               | 126 |
| 4.8.4.11 | T_idtAplOutputFreqType . . . . .         | 126 |
| 4.8.4.12 | T_idtAplOutputSigType . . . . .          | 127 |
| 4.8.4.13 | T_idtAplOutputSupportedType . . . . .    | 127 |
| 4.8.4.14 | T_idtAplQaDiv . . . . .                  | 127 |
| 4.8.4.15 | T_idtAplQbDiv . . . . .                  | 128 |
| 4.8.4.16 | T_idtAplShutoff . . . . .                | 128 |
| 4.8.4.17 | T_idtAplXtalId . . . . .                 | 128 |
| 4.8.4.18 | T_idtAplXtalType . . . . .               | 128 |
| 4.8.5    | Function Documentation . . . . .         | 129 |
| 4.8.5.1  | IDTApl_AplInput2DpllOutput . . . . .     | 129 |
| 4.8.5.2  | IDTApl_DpllOutput2AplOutput . . . . .    | 129 |
| 4.8.5.3  | IDTApl_GetAplConfig . . . . .            | 129 |
| 4.8.5.4  | IDTApl_GetAplConfigPerOutput . . . . .   | 129 |
| 4.8.5.5  | IDTApl_GetAplFreqTableEntry . . . . .    | 130 |
| 4.8.5.6  | IDTApl_GetBypass . . . . .               | 130 |
| 4.8.5.7  | IDTApl_GetConfigListByFreq . . . . .     | 130 |
| 4.8.5.8  | IDTApl_GetConfigListByXtalType . . . . . | 130 |
| 4.8.5.9  | IDTApl_GetConfigSel . . . . .            | 131 |
| 4.8.5.10 | IDTApl_GetCurrentAplFreqConfig . . . . . | 131 |
| 4.8.5.11 | IDTApl_GetFeedbackDiv . . . . .          | 131 |
| 4.8.5.12 | IDTApl_GetFeedbackDivRange . . . . .     | 132 |
| 4.8.5.13 | IDTApl_GetFreqDoublerSel . . . . .       | 132 |

|          |                                             |            |
|----------|---------------------------------------------|------------|
| 4.8.5.14 | IDTApl_GetInputClkSrc . . . . .             | 132        |
| 4.8.5.15 | IDTApl_GetMasterResetSync . . . . .         | 133        |
| 4.8.5.16 | IDTApl_GetNonActiveApllConfig . . . . .     | 133        |
| 4.8.5.17 | IDTApl_GetOutputEnState . . . . .           | 134        |
| 4.8.5.18 | IDTApl_GetOutputSigType . . . . .           | 134        |
| 4.8.5.19 | IDTApl_GetPreDiv . . . . .                  | 135        |
| 4.8.5.20 | IDTApl_GetPreDivRange . . . . .             | 135        |
| 4.8.5.21 | IDTApl_GetQaDiv . . . . .                   | 135        |
| 4.8.5.22 | IDTApl_GetQbDiv . . . . .                   | 136        |
| 4.8.5.23 | IDTApl_GetShutoff . . . . .                 | 136        |
| 4.8.5.24 | IDTApl_GetSupportedXtal . . . . .           | 137        |
| 4.8.5.25 | IDTApl_GetXtalId . . . . .                  | 137        |
| 4.8.5.26 | IDTApl_GetXtalIdFromXtalFreq . . . . .      | 138        |
| 4.8.5.27 | IDTApl_NullFreqTableEntry . . . . .         | 138        |
| 4.8.5.28 | IDTApl_SetApllConfig . . . . .              | 138        |
| 4.8.5.29 | IDTApl_SetApllConfigPerOutput . . . . .     | 138        |
| 4.8.5.30 | IDTApl_SetBypass . . . . .                  | 139        |
| 4.8.5.31 | IDTApl_SetConfigSel . . . . .               | 139        |
| 4.8.5.32 | IDTApl_SetFeedbackDiv . . . . .             | 139        |
| 4.8.5.33 | IDTApl_SetFreqDoublerSel . . . . .          | 140        |
| 4.8.5.34 | IDTApl_SetInputClkSrc . . . . .             | 140        |
| 4.8.5.35 | IDTApl_SetMasterResetSync . . . . .         | 141        |
| 4.8.5.36 | IDTApl_SetOutputEnState . . . . .           | 141        |
| 4.8.5.37 | IDTApl_SetOutputSigType . . . . .           | 141        |
| 4.8.5.38 | IDTApl_SetPreDiv . . . . .                  | 142        |
| 4.8.5.39 | IDTApl_SetQaDiv . . . . .                   | 142        |
| 4.8.5.40 | IDTApl_SetQbDiv . . . . .                   | 143        |
| 4.8.5.41 | IDTApl_SetShutoff . . . . .                 | 143        |
| 4.8.5.42 | IDTApl_SetXtalId . . . . .                  | 144        |
| 4.8.5.43 | IDTApl_Switch2NonActiveApllConfig . . . . . | 144        |
| 4.8.5.44 | IDTApl_ValidXtalInput . . . . .             | 144        |
| 4.8.5.45 | IDTApl_XtalConfigured . . . . .             | 145        |
| <b>5</b> | <b>Data Structure Documentation</b>         | <b>147</b> |
| 5.1      | bucketReg_t Struct Reference . . . . .      | 147        |
| 5.1.1    | Detailed Description . . . . .              | 147        |
| 5.1.2    | Field Documentation . . . . .               | 147        |
| 5.1.2.1  | bucketSizeReg_m . . . . .                   | 147        |
| 5.1.2.2  | decayRateReg_m . . . . .                    | 147        |
| 5.1.2.3  | lowerThreshReg_m . . . . .                  | 147        |

|          |                                                  |     |
|----------|--------------------------------------------------|-----|
| 5.1.2.4  | upperThreshReg_m                                 | 148 |
| 5.2      | T_idtApplConfig::fbDiv_s Struct Reference        | 148 |
| 5.2.1    | Detailed Description                             | 148 |
| 5.2.2    | Field Documentation                              | 148 |
| 5.2.2.1  | fbDivConfig0                                     | 148 |
| 5.2.2.2  | fbDivConfig1                                     | 148 |
| 5.3      | globalArgs_t Struct Reference                    | 148 |
| 5.3.1    | Detailed Description                             | 149 |
| 5.3.2    | Field Documentation                              | 149 |
| 5.3.2.1  | dcoMode_m                                        | 149 |
| 5.3.2.2  | dpll_m                                           | 149 |
| 5.3.2.3  | dpllProfile_ma                                   | 149 |
| 5.3.2.4  | filename_ma                                      | 149 |
| 5.3.2.5  | inputFreq_m                                      | 149 |
| 5.3.2.6  | isSim_m                                          | 150 |
| 5.3.2.7  | modes_m                                          | 150 |
| 5.3.2.8  | numberOfXtal                                     | 150 |
| 5.3.2.9  | offset_m                                         | 150 |
| 5.3.2.10 | outFreq_m                                        | 150 |
| 5.3.2.11 | regOffset_m                                      | 150 |
| 5.3.2.12 | regValue_m                                       | 150 |
| 5.3.2.13 | sampleConfig_m                                   | 150 |
| 5.3.2.14 | xtalList                                         | 150 |
| 5.4      | T_idtApplConfig::outputConfig_s Struct Reference | 151 |
| 5.4.1    | Detailed Description                             | 151 |
| 5.4.2    | Field Documentation                              | 151 |
| 5.4.2.1  | qaConfig                                         | 151 |
| 5.4.2.2  | qbConfig                                         | 151 |
| 5.5      | T_idtApplConfig::outputDiv_s Struct Reference    | 151 |
| 5.5.1    | Detailed Description                             | 152 |
| 5.5.2    | Field Documentation                              | 152 |
| 5.5.2.1  | qaDiv                                            | 152 |
| 5.5.2.2  | qbDiv                                            | 152 |
| 5.6      | pathSelectorConfig_t Struct Reference            | 152 |
| 5.6.1    | Detailed Description                             | 152 |
| 5.6.2    | Field Documentation                              | 152 |
| 5.6.2.1  | instance_m                                       | 152 |
| 5.6.2.2  | outputDivider_m                                  | 152 |
| 5.6.2.3  | outputPathSelector_m                             | 152 |
| 5.6.2.4  | selector_m                                       | 153 |

|          |                                                             |     |
|----------|-------------------------------------------------------------|-----|
| 5.6.2.5  | value_m                                                     | 153 |
| 5.7      | pathSelectorIndex_t Struct Reference                        | 153 |
| 5.7.1    | Detailed Description                                        | 153 |
| 5.7.2    | Field Documentation                                         | 153 |
| 5.7.2.1  | size_m                                                      | 153 |
| 5.7.2.2  | start_m                                                     | 153 |
| 5.8      | T_idtApIConfig::preDiv_s Struct Reference                   | 153 |
| 5.8.1    | Detailed Description                                        | 154 |
| 5.8.2    | Field Documentation                                         | 154 |
| 5.8.2.1  | preDivConfig0                                               | 154 |
| 5.8.2.2  | preDivConfig1                                               | 154 |
| 5.9      | T_idtApIConfig::outputConfig_s::qaConfig_s Struct Reference | 154 |
| 5.9.1    | Detailed Description                                        | 154 |
| 5.9.2    | Field Documentation                                         | 154 |
| 5.9.2.1  | enOutput                                                    | 154 |
| 5.9.2.2  | outputSigType                                               | 155 |
| 5.10     | T_idtApIConfig::outputDiv_s::qaDiv_s Struct Reference       | 155 |
| 5.10.1   | Detailed Description                                        | 155 |
| 5.10.2   | Field Documentation                                         | 155 |
| 5.10.2.1 | outputDivConfig0                                            | 155 |
| 5.10.2.2 | outputDivConfig1                                            | 155 |
| 5.11     | T_idtApIConfig::outputConfig_s::qbConfig_s Struct Reference | 155 |
| 5.11.1   | Detailed Description                                        | 155 |
| 5.11.2   | Field Documentation                                         | 156 |
| 5.11.2.1 | enOutput                                                    | 156 |
| 5.11.2.2 | outputSigType                                               | 156 |
| 5.12     | T_idtApIConfig::outputDiv_s::qbDiv_s Struct Reference       | 156 |
| 5.12.1   | Detailed Description                                        | 156 |
| 5.12.2   | Field Documentation                                         | 156 |
| 5.12.2.1 | outputDivConfig0                                            | 156 |
| 5.12.2.2 | outputDivConfig1                                            | 156 |
| 5.13     | T_apIOutputFrequencyEntry Struct Reference                  | 156 |
| 5.13.1   | Detailed Description                                        | 157 |
| 5.13.2   | Field Documentation                                         | 157 |
| 5.13.2.1 | apIIdivVal_m                                                | 157 |
| 5.13.2.2 | frequencies_m                                               | 157 |
| 5.14     | T_idtApIConfig Struct Reference                             | 157 |
| 5.14.1   | Detailed Description                                        | 158 |
| 5.14.2   | Field Documentation                                         | 158 |
| 5.14.2.1 | apICfg                                                      | 158 |

|           |                                          |     |
|-----------|------------------------------------------|-----|
| 5.14.2.2  | byPass                                   | 158 |
| 5.14.2.3  | dbleSel                                  | 158 |
| 5.14.2.4  | fbDiv                                    | 158 |
| 5.14.2.5  | inputClk                                 | 158 |
| 5.14.2.6  | mrSync                                   | 158 |
| 5.14.2.7  | outputConfig                             | 158 |
| 5.14.2.8  | outputDiv                                | 158 |
| 5.14.2.9  | preDiv                                   | 158 |
| 5.14.2.10 | shutOff                                  | 158 |
| 5.14.2.11 | xtal                                     | 158 |
| 5.15      | T_idtApIConfigPerOutput Struct Reference | 159 |
| 5.15.1    | Detailed Description                     | 159 |
| 5.15.2    | Field Documentation                      | 159 |
| 5.15.2.1  | apIICfg                                  | 159 |
| 5.15.2.2  | dbleSel                                  | 159 |
| 5.15.2.3  | enOutput                                 | 159 |
| 5.15.2.4  | fbDiv                                    | 159 |
| 5.15.2.5  | inputClk                                 | 160 |
| 5.15.2.6  | outputDiv                                | 160 |
| 5.15.2.7  | preDiv                                   | 160 |
| 5.15.2.8  | sigType                                  | 160 |
| 5.15.2.9  | xtal                                     | 160 |
| 5.16      | T_idtApIFreqMapping Struct Reference     | 160 |
| 5.16.1    | Detailed Description                     | 160 |
| 5.16.2    | Field Documentation                      | 161 |
| 5.16.2.1  | apIFbDivider                             | 161 |
| 5.16.2.2  | apIInputFreq                             | 161 |
| 5.16.2.3  | apIOutputDivider                         | 161 |
| 5.16.2.4  | apIOutputFreq                            | 161 |
| 5.16.2.5  | apIOutputs                               | 161 |
| 5.16.2.6  | apIPreDivider                            | 161 |
| 5.16.2.7  | apIXtalFreq                              | 161 |
| 5.17      | T_idtApIXtal Struct Reference            | 161 |
| 5.17.1    | Detailed Description                     | 162 |
| 5.17.2    | Field Documentation                      | 162 |
| 5.17.2.1  | apIXtalFreq                              | 162 |
| 5.17.2.2  | apIXtalId                                | 162 |
| 5.18      | T_idtApIXtalList Struct Reference        | 162 |
| 5.18.1    | Detailed Description                     | 162 |
| 5.18.2    | Field Documentation                      | 163 |

|           |                                                 |     |
|-----------|-------------------------------------------------|-----|
| 5.18.2.1  | xtal1Apll1 . . . . .                            | 163 |
| 5.18.2.2  | xtal1Apll1Freq . . . . .                        | 163 |
| 5.18.2.3  | xtal2Apll2 . . . . .                            | 163 |
| 5.18.2.4  | xtal2Apll2Freq . . . . .                        | 163 |
| 5.18.2.5  | xtal3Apll1 . . . . .                            | 163 |
| 5.18.2.6  | xtal3Apll1Freq . . . . .                        | 163 |
| 5.18.2.7  | xtal4Apll2 . . . . .                            | 163 |
| 5.18.2.8  | xtal4Apll2Freq . . . . .                        | 163 |
| 5.19      | T_idtDrvAccess Struct Reference . . . . .       | 164 |
| 5.19.1    | Detailed Description . . . . .                  | 164 |
| 5.19.2    | Field Documentation . . . . .                   | 164 |
| 5.19.2.1  | deinitFunc_m . . . . .                          | 164 |
| 5.19.2.2  | initFunc_m . . . . .                            | 164 |
| 5.19.2.3  | readFunc_m . . . . .                            | 164 |
| 5.19.2.4  | userData_m . . . . .                            | 164 |
| 5.19.2.5  | writeFunc_m . . . . .                           | 164 |
| 5.20      | T_idtDrvDeviceConfig Struct Reference . . . . . | 165 |
| 5.20.1    | Detailed Description . . . . .                  | 165 |
| 5.20.2    | Field Documentation . . . . .                   | 165 |
| 5.20.2.1  | dcoModeSupport_ma . . . . .                     | 165 |
| 5.20.2.2  | inputExt2IntXlate_ma . . . . .                  | 165 |
| 5.20.2.3  | inputFreqValidFunc_m . . . . .                  | 165 |
| 5.20.2.4  | inSyncXlate_ma . . . . .                        | 166 |
| 5.20.2.5  | maxNumXtalPerApll_m . . . . .                   | 166 |
| 5.20.2.6  | numberOfApll_m . . . . .                        | 166 |
| 5.20.2.7  | numberOfSonetGigEthPath_m . . . . .             | 166 |
| 5.20.2.8  | outPutApllConfig . . . . .                      | 166 |
| 5.20.2.9  | outputExt2IntXlate_ma . . . . .                 | 166 |
| 5.20.2.10 | outputFreqValidFunc_m . . . . .                 | 166 |
| 5.20.2.11 | outSyntXlate_ma . . . . .                       | 166 |
| 5.20.2.12 | phaseSlopeLimiting_ma . . . . .                 | 166 |
| 5.20.2.13 | pllExt2IntXlate_ma . . . . .                    | 167 |
| 5.20.2.14 | pllInt2ExtXlate_ma . . . . .                    | 167 |
| 5.20.2.15 | populatedApll_ma . . . . .                      | 167 |
| 5.20.2.16 | supportedXtalFreq_ma . . . . .                  | 167 |
| 5.20.2.17 | supportedXtalId_ma . . . . .                    | 167 |
| 5.20.2.18 | t0PIIMode_m . . . . .                           | 167 |
| 5.21      | T_idtDrvHdlr Struct Reference . . . . .         | 167 |
| 5.21.1    | Detailed Description . . . . .                  | 168 |
| 5.21.2    | Field Documentation . . . . .                   | 168 |

|           |                                         |     |
|-----------|-----------------------------------------|-----|
| 5.21.2.1  | apllI2CTransData_mp                     | 168 |
| 5.21.2.2  | currOutPathCfg_mp                       | 168 |
| 5.21.2.3  | deviceConfig_m                          | 168 |
| 5.21.2.4  | deviceType_m                            | 168 |
| 5.21.2.5  | dpllI2CTransData_mp                     | 168 |
| 5.21.2.6  | drvAccess_m                             | 168 |
| 5.21.2.7  | i2cAddr                                 | 168 |
| 5.21.2.8  | i2cAddrTransFunc_m                      | 169 |
| 5.21.2.9  | name_m                                  | 169 |
| 5.21.2.10 | numberOfApplXtal_m                      | 169 |
| 5.21.2.11 | xtalList                                | 169 |
| 5.22      | T_IDTFreq2bfVal Struct Reference        | 169 |
| 5.22.1    | Detailed Description                    | 169 |
| 5.22.2    | Field Documentation                     | 169 |
| 5.22.2.1  | bfVal_m                                 | 169 |
| 5.22.2.2  | freq_m                                  | 170 |
| 5.23      | T_idtHfDivEntry Struct Reference        | 170 |
| 5.23.1    | Detailed Description                    | 170 |
| 5.23.2    | Field Documentation                     | 170 |
| 5.23.2.1  | divValue_m                              | 170 |
| 5.23.2.2  | hfDivValue_m                            | 170 |
| 5.24      | T_idtI2CtransData Struct Reference      | 170 |
| 5.24.1    | Detailed Description                    | 171 |
| 5.24.2    | Field Documentation                     | 171 |
| 5.24.2.1  | bitLocation                             | 171 |
| 5.24.2.2  | bitValue                                | 171 |
| 5.25      | T_idtOutputApplConfig Struct Reference  | 171 |
| 5.25.1    | Detailed Description                    | 171 |
| 5.25.2    | Field Documentation                     | 171 |
| 5.25.2.1  | apllInput                               | 171 |
| 5.25.2.2  | apllInst                                | 171 |
| 5.25.2.3  | apllOutput                              | 172 |
| 5.25.2.4  | extOutput                               | 172 |
| 5.26      | T_IDTPathConfiguration Struct Reference | 172 |
| 5.26.1    | Detailed Description                    | 172 |
| 5.26.2    | Field Documentation                     | 172 |
| 5.26.2.1  | networkType_m                           | 172 |
| 5.26.2.2  | sonetGigEthSel_ma                       | 172 |
| 5.26.2.3  | T0SelA_m                                | 172 |
| 5.26.2.4  | T0SelB_m                                | 173 |

|          |                                                      |            |
|----------|------------------------------------------------------|------------|
| 5.26.2.5 | T4SelA_m                                             | 173        |
| 5.26.2.6 | T4SelB_m                                             | 173        |
| 5.27     | T_idtSonetGigEthBitfield2PathConfig Struct Reference | 173        |
| 5.27.1   | Detailed Description                                 | 173        |
| 5.27.2   | Field Documentation                                  | 173        |
| 5.27.2.1 | inputDpll_m                                          | 173        |
| 5.27.2.2 | outputFreq_m                                         | 173        |
| 5.28     | T_SampleConfigEntry Struct Reference                 | 174        |
| 5.28.1   | Detailed Description                                 | 174        |
| 5.28.2   | Field Documentation                                  | 174        |
| 5.28.2.1 | configDescription_m                                  | 174        |
| 5.28.2.2 | configFunction_m                                     | 174        |
| <b>6</b> | <b>File Documentation</b>                            | <b>175</b> |
| 6.1      | 82V3910/dev/include/idt82V3910.h File Reference      | 175        |
| 6.1.1    | Function Documentation                               | 176        |
| 6.1.1.1  | IDTGeneral_InitDevice                                | 176        |
| 6.1.1.2  | IDTGeneral_InitDriver                                | 176        |
| 6.2      | 82V3910/dev/src/82V3910.c File Reference             | 176        |
| 6.2.1    | Function Documentation                               | 177        |
| 6.2.1.1  | IDTGeneral_InitDevice                                | 177        |
| 6.2.1.2  | IDTGeneral_InitDriver                                | 177        |
| 6.2.2    | Variable Documentation                               | 178        |
| 6.2.2.1  | idt82V3910Config                                     | 178        |
| 6.3      | 82V3910/sample/include/access.h File Reference       | 178        |
| 6.3.1    | Function Documentation                               | 179        |
| 6.3.1.1  | IDTSample_GetLogVerbosity                            | 179        |
| 6.3.1.2  | IDTSample_InitDriverI2c                              | 180        |
| 6.3.1.3  | IDTSample_InitDriverSimulator                        | 180        |
| 6.3.1.4  | IDTSample_ResetXtalList                              | 180        |
| 6.3.1.5  | IDTSample_SetLogVerbosity                            | 180        |
| 6.4      | 82V3910/sample/include/loadstore.h File Reference    | 181        |
| 6.4.1    | Function Documentation                               | 181        |
| 6.4.1.1  | IDTSample_LoadRgfFile                                | 181        |
| 6.4.1.2  | IDTSample_SaveRgfFile                                | 181        |
| 6.5      | 82V3910/sample/include/sample.h File Reference       | 182        |
| 6.5.1    | Function Documentation                               | 183        |
| 6.5.1.1  | IDTSample_Configure1Pps                              | 183        |
| 6.5.1.2  | IDTSample_Configure1PpsHoldover                      | 183        |
| 6.5.1.3  | IDTSample_ConfigureDcoMode                           | 183        |

|          |                                                                  |     |
|----------|------------------------------------------------------------------|-----|
| 6.5.1.4  | IDTSample_ConfigureSampleConfig1 . . . . .                       | 183 |
| 6.5.1.5  | IDTSample_ConfigureSampleConfig2 . . . . .                       | 183 |
| 6.5.1.6  | IDTSample_ConfigureSampleConfig3 . . . . .                       | 184 |
| 6.5.1.7  | IDTSample_ConfigureSampleConfig4 . . . . .                       | 184 |
| 6.5.1.8  | IDTSample_ConfigureSampleConfig5 . . . . .                       | 184 |
| 6.5.1.9  | IDTSample_GetCombinedDcoOffset . . . . .                         | 184 |
| 6.5.1.10 | IDTSample_SetCombinedDcoOffset . . . . .                         | 185 |
| 6.6      | 82V3910/sample/include/sampleAppl.h File Reference . . . . .     | 185 |
| 6.6.1    | Function Documentation . . . . .                                 | 186 |
| 6.6.1.1  | IDTSample_ConfigureSampleConfigOutput6_156_dot_25MHz . . . . .   | 186 |
| 6.6.1.2  | IDTSample_ConfigureSampleConfigureOutput10_25_MHz . . . . .      | 186 |
| 6.6.1.3  | IDTSample_ConfigureSampleConfigureOutput7_77_dot_76MHz . . . . . | 186 |
| 6.6.1.4  | IDTSample_DualMode . . . . .                                     | 187 |
| 6.6.1.5  | IDTSample_GPS . . . . .                                          | 187 |
| 6.6.1.6  | IDTSample_GPS_1PPS . . . . .                                     | 187 |
| 6.6.1.7  | IDTSample_Sonet . . . . .                                        | 188 |
| 6.6.1.8  | IDTSample_SyncE_LAN . . . . .                                    | 188 |
| 6.6.1.9  | IDTSample_SyncE_Sonet . . . . .                                  | 188 |
| 6.6.1.10 | IDTSample_SyncE_WAN . . . . .                                    | 188 |
| 6.7      | 82V3910/sample/src/access.c File Reference . . . . .             | 189 |
| 6.7.1    | Function Documentation . . . . .                                 | 190 |
| 6.7.1.1  | DeInit_SimWrapper . . . . .                                      | 190 |
| 6.7.1.2  | IDTSample_GetLogVerbosity . . . . .                              | 190 |
| 6.7.1.3  | IDTSample_InitDriverI2c . . . . .                                | 190 |
| 6.7.1.4  | IDTSample_InitDriverSimulator . . . . .                          | 190 |
| 6.7.1.5  | IDTSample_ResetXtalList . . . . .                                | 191 |
| 6.7.1.6  | IDTSample_SetLogVerbosity . . . . .                              | 191 |
| 6.7.1.7  | Init_SimWrapper . . . . .                                        | 191 |
| 6.7.1.8  | ReadByte_SimWrapper . . . . .                                    | 191 |
| 6.7.1.9  | WriteByte_SimWrapper . . . . .                                   | 192 |
| 6.7.2    | Variable Documentation . . . . .                                 | 192 |
| 6.7.2.1  | I2cAccessMethods . . . . .                                       | 192 |
| 6.7.2.2  | SimAccessMethods . . . . .                                       | 192 |
| 6.8      | 82V3910/sample/src/loadstore.c File Reference . . . . .          | 192 |
| 6.8.1    | Function Documentation . . . . .                                 | 193 |
| 6.8.1.1  | IDTSample_LoadRgfFile . . . . .                                  | 193 |
| 6.8.1.2  | IDTSample_SaveRgfFile . . . . .                                  | 193 |
| 6.9      | 82V3910/sample/src/main.c File Reference . . . . .               | 194 |
| 6.9.1    | Macro Definition Documentation . . . . .                         | 195 |
| 6.9.1.1  | MODE_DpllProfile . . . . .                                       | 195 |

|           |                                                |     |
|-----------|------------------------------------------------|-----|
| 6.9.1.2   | MODE_Input                                     | 195 |
| 6.9.1.3   | MODE_Load                                      | 195 |
| 6.9.1.4   | MODE_None                                      | 196 |
| 6.9.1.5   | MODE_Output                                    | 196 |
| 6.9.1.6   | MODE_Read                                      | 196 |
| 6.9.1.7   | MODE_Sample                                    | 196 |
| 6.9.1.8   | MODE_Save                                      | 196 |
| 6.9.1.9   | MODE_Write                                     | 196 |
| 6.9.1.10  | no_argument                                    | 196 |
| 6.9.1.11  | optional_argument                              | 196 |
| 6.9.1.12  | required_argument                              | 196 |
| 6.9.1.13  | SAMPLE_MAX_FILENAME                            | 196 |
| 6.9.2     | Typedef Documentation                          | 197 |
| 6.9.2.1   | T_DpllProfileFunc                              | 197 |
| 6.9.2.2   | T_SampleConfigDescrFunc                        | 197 |
| 6.9.2.3   | T_SampleConfigFunc                             | 197 |
| 6.9.3     | Enumeration Type Documentation                 | 197 |
| 6.9.3.1   | T_DpllProfile                                  | 197 |
| 6.9.4     | Function Documentation                         | 197 |
| 6.9.4.1   | IDTSample_DumpRegistersRgf                     | 197 |
| 6.9.4.2   | main                                           | 197 |
| 6.9.4.3   | printHelp                                      | 197 |
| 6.9.4.4   | processCmdLineArguments                        | 198 |
| 6.9.5     | Variable Documentation                         | 198 |
| 6.9.5.1   | globalArgs                                     | 198 |
| 6.9.5.2   | IDTHIApi_Frequency2String_c                    | 198 |
| 6.9.5.3   | simDebugFlag                                   | 198 |
| 6.10      | 82V3910/sample/src/sample.c File Reference     | 198 |
| 6.10.1    | Function Documentation                         | 199 |
| 6.10.1.1  | IDTSample_Configure1Pps                        | 199 |
| 6.10.1.2  | IDTSample_Configure1PpsHoldover                | 199 |
| 6.10.1.3  | IDTSample_ConfigureDcoMode                     | 199 |
| 6.10.1.4  | IDTSample_ConfigureSampleConfig1               | 200 |
| 6.10.1.5  | IDTSample_ConfigureSampleConfig2               | 200 |
| 6.10.1.6  | IDTSample_ConfigureSampleConfig3               | 200 |
| 6.10.1.7  | IDTSample_ConfigureSampleConfig4               | 200 |
| 6.10.1.8  | IDTSample_ConfigureSampleConfig5               | 201 |
| 6.10.1.9  | IDTSample_GetCombinedDcoOffset                 | 201 |
| 6.10.1.10 | IDTSample_SetCombinedDcoOffset                 | 201 |
| 6.11      | 82V3910/sample/src/sampleAppl.c File Reference | 201 |

|           |                                                      |     |
|-----------|------------------------------------------------------|-----|
| 6.11.1    | Function Documentation                               | 202 |
| 6.11.1.1  | IDTSample_ConfigureSampleConfigOutput6_156_dot_25MHz | 202 |
| 6.11.1.2  | IDTSample_ConfigureSampleConfigOutput10_25_MHz       | 203 |
| 6.11.1.3  | IDTSample_ConfigureSampleConfigOutput7_77_dot_76MHz  | 203 |
| 6.11.1.4  | IDTSample_DualMode                                   | 203 |
| 6.11.1.5  | IDTSample_GPS                                        | 203 |
| 6.11.1.6  | IDTSample_GPS_1PPS                                   | 204 |
| 6.11.1.7  | IDTSample_Sonet                                      | 204 |
| 6.11.1.8  | IDTSample_SyncE_LAN                                  | 204 |
| 6.11.1.9  | IDTSample_SyncE_Sonet                                | 205 |
| 6.11.1.10 | IDTSample_SyncE_WAN                                  | 205 |
| 6.12      | apll/access/sim/ApllReadWrite_sim.c File Reference   | 205 |
| 6.12.1    | Macro Definition Documentation                       | 206 |
| 6.12.1.1  | SIM_REG_MAP_SIZE                                     | 206 |
| 6.12.2    | Function Documentation                               | 206 |
| 6.12.2.1  | ApllDeInit_Sim                                       | 206 |
| 6.12.2.2  | ApllInit_Sim                                         | 206 |
| 6.12.2.3  | ApllRead_Sim                                         | 207 |
| 6.12.2.4  | ApllWrite_Sim                                        | 207 |
| 6.12.3    | Variable Documentation                               | 207 |
| 6.12.3.1  | simulatedRegMapApll0                                 | 207 |
| 6.12.3.2  | simulatedRegMapApll1                                 | 207 |
| 6.13      | apll/access/sim/ApllReadWrite_sim.h File Reference   | 207 |
| 6.13.1    | Function Documentation                               | 208 |
| 6.13.1.1  | ApllDeInit_Sim                                       | 208 |
| 6.13.1.2  | ApllInit_Sim                                         | 208 |
| 6.13.1.3  | ApllRead_Sim                                         | 208 |
| 6.13.1.4  | ApllWrite_Sim                                        | 208 |
| 6.14      | apll/include/ApllApi.h File Reference                | 208 |
| 6.15      | apll/include/ApllFreqProfile.h File Reference        | 210 |
| 6.16      | apll/include/ApllLogging.h File Reference            | 212 |
| 6.17      | apll/include/ApllRegDef.h File Reference             | 212 |
| 6.18      | apll/include/ApllTypeDef.h File Reference            | 213 |
| 6.19      | apll/src/ApllApi.c File Reference                    | 215 |
| 6.19.1    | Macro Definition Documentation                       | 217 |
| 6.19.1.1  | APLL1_REG_HIGH_BIT                                   | 217 |
| 6.19.1.2  | APLL1_REG_VALID_OFFSET                               | 217 |
| 6.19.1.3  | APLL2_REG_HIGH_BIT                                   | 218 |
| 6.19.1.4  | APLL2_REG_VALID_OFFSET                               | 218 |
| 6.19.1.5  | APLL_DIVIDER_LSB                                     | 218 |

|           |                                                  |     |
|-----------|--------------------------------------------------|-----|
| 6.19.1.6  | APLL_DIVIDER_MSB                                 | 218 |
| 6.19.1.7  | APLL_DIVIDER_VAL                                 | 218 |
| 6.19.1.8  | APLL_FB_DIV_LSB                                  | 218 |
| 6.19.1.9  | APLL_FB_DIV_MAX                                  | 218 |
| 6.19.1.10 | APLL_FB_DIV_MIN                                  | 218 |
| 6.19.1.11 | APLL_FB_DIV_MSB                                  | 218 |
| 6.19.1.12 | APLL_FB_DIV_VAL                                  | 218 |
| 6.19.1.13 | APLL_PRE_DIV_LSB                                 | 219 |
| 6.19.1.14 | APLL_PRE_DIV_MAX                                 | 219 |
| 6.19.1.15 | APLL_PRE_DIV_MIN                                 | 219 |
| 6.19.1.16 | APLL_PRE_DIV_MSB                                 | 219 |
| 6.19.1.17 | APLL_PRE_DIV_VAL                                 | 219 |
| 6.19.1.18 | APLL_REG_HIGH_BIT_OFFSET                         | 219 |
| 6.19.1.19 | APLL_REG_OFFSET                                  | 219 |
| 6.19.1.20 | INVALID_XTAL_SEL_VALUE                           | 219 |
| 6.20      | apll/src/ApllFreqProfile.c File Reference        | 219 |
| 6.20.1    | Macro Definition Documentation                   | 221 |
| 6.20.1.1  | APLL_FB_DIV_ETH                                  | 221 |
| 6.20.1.2  | APLL_FB_DIV_ETH_FEC                              | 221 |
| 6.20.1.3  | APLL_FB_DIV_SONET                                | 221 |
| 6.20.1.4  | APLL_OUTPUT_DIV_1                                | 221 |
| 6.20.1.5  | APLL_OUTPUT_DIV_2                                | 221 |
| 6.20.1.6  | APLL_OUTPUT_DIV_25                               | 221 |
| 6.20.1.7  | APLL_OUTPUT_DIV_4                                | 221 |
| 6.20.1.8  | APLL_OUTPUT_DIV_5                                | 221 |
| 6.20.1.9  | APLL_OUTPUT_DIV_8                                | 221 |
| 6.20.1.10 | APLL_PRE_DIV_ETH                                 | 221 |
| 6.20.1.11 | APLL_PRE_DIV_ETH_FEC                             | 221 |
| 6.20.1.12 | APLL_PRE_DIV_SONET                               | 222 |
| 6.20.1.13 | MAX_CONFIG_COMBO_PER_FREQ                        | 222 |
| 6.20.1.14 | MAX_CONFIG_COMBO_PER_VCXO_TYPE                   | 222 |
| 6.20.1.15 | NULL_APLL_FREQ_TABLE_ENTRY                       | 222 |
| 6.21      | common/access/sim/ReadWrite_sim.c File Reference | 222 |
| 6.21.1    | Macro Definition Documentation                   | 223 |
| 6.21.1.1  | DRV_MEMMAP_SIZE                                  | 223 |
| 6.21.1.2  | NONE_APLL_REG_SPACE_FLAG                         | 223 |
| 6.21.2    | Function Documentation                           | 224 |
| 6.21.2.1  | DeInit_Sim                                       | 224 |
| 6.21.2.2  | Init_Sim                                         | 224 |
| 6.21.2.3  | RByte_Sim                                        | 224 |

|          |                                                                 |     |
|----------|-----------------------------------------------------------------|-----|
| 6.21.2.4 | WRByte_Sim . . . . .                                            | 224 |
| 6.21.3   | Variable Documentation . . . . .                                | 224 |
| 6.21.3.1 | dpllOverlayRegisters . . . . .                                  | 224 |
| 6.21.3.2 | page1Registers . . . . .                                        | 224 |
| 6.21.3.3 | simDebugFlag . . . . .                                          | 225 |
| 6.21.3.4 | simulatedMemMap . . . . .                                       | 225 |
| 6.22     | common/access/sim/ReadWrite_sim.h File Reference . . . . .      | 225 |
| 6.22.1   | Function Documentation . . . . .                                | 225 |
| 6.22.1.1 | DeInit_Sim . . . . .                                            | 225 |
| 6.22.1.2 | Init_Sim . . . . .                                              | 225 |
| 6.22.1.3 | RDByte_Sim . . . . .                                            | 225 |
| 6.22.1.4 | WRByte_Sim . . . . .                                            | 226 |
| 6.23     | common/hlapi/include/inputFreq.h File Reference . . . . .       | 226 |
| 6.23.1   | Function Documentation . . . . .                                | 226 |
| 6.23.1.1 | IDTHIApi_ConfigureInput1PPS . . . . .                           | 226 |
| 6.23.1.2 | IDTHIApi_ConfigureInputAmi . . . . .                            | 227 |
| 6.23.1.3 | IDTHIApi_ConfigureInputFrequency . . . . .                      | 227 |
| 6.23.1.4 | IDTHIApi_DisableAllInputPriorities . . . . .                    | 227 |
| 6.24     | common/hlapi/include/outputFreq.h File Reference . . . . .      | 227 |
| 6.24.1   | Enumeration Type Documentation . . . . .                        | 229 |
| 6.24.1.1 | outputFrequency_t . . . . .                                     | 229 |
| 6.24.1.2 | T_sonetGigeMode . . . . .                                       | 230 |
| 6.24.2   | Function Documentation . . . . .                                | 230 |
| 6.24.2.1 | IDTHIApi_ClearProvisioning . . . . .                            | 230 |
| 6.24.2.2 | IDTHIApi_ConfigureOutput . . . . .                              | 231 |
| 6.24.2.3 | IDTHIApi_DisableAllOutputs . . . . .                            | 232 |
| 6.24.2.4 | IDTHIApi_PrintInput . . . . .                                   | 232 |
| 6.24.2.5 | IDTHIApi_PrintOutput . . . . .                                  | 232 |
| 6.25     | common/hlapi/include/outputFreq_priv.h File Reference . . . . . | 232 |
| 6.25.1   | Macro Definition Documentation . . . . .                        | 234 |
| 6.25.1.1 | CHECK_BIT . . . . .                                             | 234 |
| 6.25.1.2 | SET_BIT . . . . .                                               | 234 |
| 6.25.2   | Typedef Documentation . . . . .                                 | 234 |
| 6.25.2.1 | validFreq_t . . . . .                                           | 234 |
| 6.25.3   | Enumeration Type Documentation . . . . .                        | 235 |
| 6.25.3.1 | pathSelectors_t . . . . .                                       | 235 |
| 6.25.4   | Function Documentation . . . . .                                | 235 |
| 6.25.4.1 | IDTHIApi_ApplyFrequencies . . . . .                             | 235 |
| 6.25.4.2 | IDTHIApi_ConvertFromFreqListToFreqArray . . . . .               | 235 |
| 6.25.4.3 | IDTHIApi_IsFreqArrayInFreqList . . . . .                        | 235 |

|           |                                                    |     |
|-----------|----------------------------------------------------|-----|
| 6.25.4.4  | IDTHIApi_IsFrequencyInFreqList                     | 235 |
| 6.25.4.5  | IDTHIApi_PrintAvailFrequencies                     | 236 |
| 6.25.4.6  | IDTHIApi_PrintConfiguration                        | 236 |
| 6.25.4.7  | IDTHIApi_PrintFrequencyList                        | 236 |
| 6.25.4.8  | IDTHIApi_PrintOption                               | 236 |
| 6.25.4.9  | IDTHIApi_PrintOutputHwConfig                       | 237 |
| 6.25.4.10 | IDTHIApi_PrintSelectedOutputPath                   | 237 |
| 6.25.5    | Variable Documentation                             | 237 |
| 6.25.5.1  | IDTHIApi_Frequency2String_c                        | 237 |
| 6.26      | common/hlapi/include/profiles.h File Reference     | 237 |
| 6.26.1    | Function Documentation                             | 238 |
| 6.26.1.1  | IDTHIApi_g813Option1                               | 238 |
| 6.26.1.2  | IDTHIApi_g813Option2                               | 238 |
| 6.26.1.3  | IDTHIApi_g8262EecOption1                           | 238 |
| 6.26.1.4  | IDTHIApi_g8262EecOption2                           | 239 |
| 6.26.1.5  | IDTHIApi_gr1244Stratum3                            | 239 |
| 6.26.1.6  | IDTHIApi_gr253SonetStratum3                        | 239 |
| 6.26.1.7  | IDTHIApi_pps                                       | 239 |
| 6.26.1.8  | IDTHIApi_smc                                       | 239 |
| 6.26.1.9  | IDTHIApi_wideband                                  | 240 |
| 6.27      | common/hlapi/src/inputFreq.c File Reference        | 240 |
| 6.27.1    | Function Documentation                             | 241 |
| 6.27.1.1  | IDTHIApi_ConfigureInput1PPS                        | 241 |
| 6.27.1.2  | IDTHIApi_ConfigureInputAmi                         | 241 |
| 6.27.1.3  | IDTHIApi_ConfigureInputFrequency                   | 241 |
| 6.27.1.4  | IDTHIApi_DisableAllInputPriorities                 | 241 |
| 6.28      | common/hlapi/src/outputFreq.c File Reference       | 242 |
| 6.28.1    | Macro Definition Documentation                     | 243 |
| 6.28.1.1  | MAX_SUPPORTED_FREQUENCIES                          | 243 |
| 6.28.1.2  | OUTMUX_ENTRY                                       | 243 |
| 6.28.2    | Function Documentation                             | 243 |
| 6.28.2.1  | IDTHIApi_ClearProvisioning                         | 243 |
| 6.28.2.2  | IDTHIApi_ConfigureOutput                           | 243 |
| 6.28.2.3  | IDTHIApi_DisableAllOutputs                         | 244 |
| 6.28.2.4  | IDTHIApi_RetrieveOutputPathMuxFromHardware         | 244 |
| 6.29      | common/hlapi/src/outputFreqString.c File Reference | 245 |
| 6.29.1    | Enumeration Type Documentation                     | 246 |
| 6.29.1.1  | T_outFreqLookupIdx                                 | 246 |
| 6.29.2    | Function Documentation                             | 247 |
| 6.29.2.1  | IDTHIApi_PrintAvailFrequencies                     | 247 |

|          |                                                            |     |
|----------|------------------------------------------------------------|-----|
| 6.29.2.2 | IDTHIApi_PrintConfiguration . . . . .                      | 247 |
| 6.29.2.3 | IDTHIApi_PrintFrequencyList . . . . .                      | 247 |
| 6.29.2.4 | IDTHIApi_PrintInput . . . . .                              | 247 |
| 6.29.2.5 | IDTHIApi_PrintInputHwConfig . . . . .                      | 247 |
| 6.29.2.6 | IDTHIApi_PrintOption . . . . .                             | 247 |
| 6.29.2.7 | IDTHIApi_PrintOutput . . . . .                             | 248 |
| 6.29.2.8 | IDTHIApi_PrintOutputHwConfig . . . . .                     | 248 |
| 6.29.2.9 | IDTHIApi_PrintSelectedOutputPath . . . . .                 | 248 |
| 6.29.3   | Variable Documentation . . . . .                           | 248 |
| 6.29.3.1 | IDTHIApi_Frequency2String_c . . . . .                      | 248 |
| 6.30     | common/hlapi/src/outputFreqUtil.c File Reference . . . . . | 248 |
| 6.30.1   | Function Documentation . . . . .                           | 249 |
| 6.30.1.1 | IDTHIApi_ApplyFrequencies . . . . .                        | 249 |
| 6.30.1.2 | IDTHIApi_ConvertFromFreqListToFreqArray . . . . .          | 250 |
| 6.30.1.3 | IDTHIApi_IsFreqArrayInFreqList . . . . .                   | 250 |
| 6.30.1.4 | IDTHIApi_IsFrequencyInFreqList . . . . .                   | 250 |
| 6.31     | common/hlapi/src/profiles.c File Reference . . . . .       | 250 |
| 6.31.1   | Detailed Description . . . . .                             | 251 |
| 6.31.2   | Function Documentation . . . . .                           | 252 |
| 6.31.2.1 | IDTHIApi_g813Option1 . . . . .                             | 252 |
| 6.31.2.2 | IDTHIApi_g813Option2 . . . . .                             | 252 |
| 6.31.2.3 | IDTHIApi_g8262EecOption1 . . . . .                         | 252 |
| 6.31.2.4 | IDTHIApi_g8262EecOption2 . . . . .                         | 252 |
| 6.31.2.5 | IDTHIApi_gr1244Stratum3 . . . . .                          | 252 |
| 6.31.2.6 | IDTHIApi_gr253SonetStratum3 . . . . .                      | 253 |
| 6.31.2.7 | IDTHIApi_pps . . . . .                                     | 253 |
| 6.31.2.8 | IDTHIApi_smc . . . . .                                     | 253 |
| 6.31.2.9 | IDTHIApi_wideband . . . . .                                | 253 |
| 6.32     | common/include/Apll.h File Reference . . . . .             | 253 |
| 6.33     | common/include/Define.h File Reference . . . . .           | 255 |
| 6.33.1   | Macro Definition Documentation . . . . .                   | 257 |
| 6.33.1.1 | APLL_CONFIGURED . . . . .                                  | 257 |
| 6.33.1.2 | IDT_DRV_MAX_APLL_XTAL . . . . .                            | 257 |
| 6.33.1.3 | IDT_DRV_MAX_INPUT . . . . .                                | 258 |
| 6.33.1.4 | IDT_DRV_MAX_OUTPUT . . . . .                               | 258 |
| 6.33.1.5 | IDT_DRV_MAX_XTAL_PER_APLL . . . . .                        | 258 |
| 6.33.1.6 | MPU_I2C_MODE . . . . .                                     | 258 |
| 6.33.1.7 | MPU_SERIAL_MODE . . . . .                                  | 258 |
| 6.33.1.8 | NULL_APLL_CONFIG . . . . .                                 | 258 |
| 6.33.2   | Typedef Documentation . . . . .                            | 258 |

|          |                                             |     |
|----------|---------------------------------------------|-----|
| 6.33.2.1 | T_idtAccessDeInitFunc                       | 258 |
| 6.33.2.2 | T_idtAccessInitFunc                         | 259 |
| 6.33.2.3 | T_idtI2CAddrTransFunc                       | 259 |
| 6.33.2.4 | T_idtInputFreqValid                         | 259 |
| 6.33.2.5 | T_idtOutputFreqValid                        | 259 |
| 6.33.2.6 | T_idtReadFunc                               | 260 |
| 6.33.2.7 | T_idtWriteFunc                              | 260 |
| 6.33.3   | Enumeration Type Documentation              | 260 |
| 6.33.3.1 | T_idtDeviceType                             | 260 |
| 6.33.3.2 | T_idtDpllInstance                           | 260 |
| 6.33.3.3 | T_idtDpllType                               | 261 |
| 6.34     | common/include/Dpll.h File Reference        | 261 |
| 6.35     | common/include/DpllCfg.h File Reference     | 262 |
| 6.36     | common/include/General.h File Reference     | 268 |
| 6.37     | common/include/Hal.h File Reference         | 271 |
| 6.38     | common/include/Input.h File Reference       | 272 |
| 6.39     | common/include/Output.h File Reference      | 275 |
| 6.40     | common/include/RegisterDef.h File Reference | 277 |
| 6.40.1   | Detailed Description                        | 285 |
| 6.41     | common/src/Apl.c File Reference             | 285 |
| 6.42     | common/src/DpllCfg.c File Reference         | 286 |
| 6.42.1   | Macro Definition Documentation              | 291 |
| 6.42.1.1 | INVALID_REGISTER_OFFSET                     | 291 |
| 6.42.2   | Variable Documentation                      | 291 |
| 6.42.2.1 | IDTDpll_bandwidthLimitRegisters_a           | 291 |
| 6.43     | common/src/General.c File Reference         | 291 |
| 6.44     | common/src/Hal.c File Reference             | 293 |
| 6.44.1   | Macro Definition Documentation              | 294 |
| 6.44.1.1 | DEBUG_IDT_HAL                               | 294 |
| 6.45     | common/src/Input.c File Reference           | 294 |
| 6.46     | common/src/Output.c File Reference          | 297 |
| 6.46.1   | Enumeration Type Documentation              | 298 |
| 6.46.1.1 | anonymous enum                              | 298 |
| 6.46.1.2 | T_idtOutputPathBfVal                        | 299 |



# Chapter 1

## Module Index

### 1.1 Modules

Here is a list of all modules:

|                                             |     |
|---------------------------------------------|-----|
| DPLL Private header file . . . . .          | 7   |
| DPLL Function Module . . . . .              | 10  |
| General Function Module . . . . .           | 47  |
| Hardware Abstraction Layer Module . . . . . | 56  |
| Input Function Module . . . . .             | 60  |
| Output Function Module . . . . .            | 78  |
| Register definitions . . . . .              | 85  |
| ApII Function Module . . . . .              | 118 |



## Chapter 2

# Data Structure Index

### 2.1 Data Structures

Here are the data structures with brief descriptions:

|                                                            |                                                                                                                        |     |
|------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------|-----|
| <a href="#">bucketReg_t</a>                                | Structure used to store leaky bucket register offsets . . . . .                                                        | 147 |
| <a href="#">T_idtApIConfig::fbDiv_s</a>                    | . . . . .                                                                                                              | 148 |
| <a href="#">globalArgs_t</a>                               | Structure to define program command line arguments . . . . .                                                           | 148 |
| <a href="#">T_idtApIConfig::outputConfig_s</a>             | . . . . .                                                                                                              | 151 |
| <a href="#">T_idtApIConfig::outputDiv_s</a>                | . . . . .                                                                                                              | 151 |
| <a href="#">pathSelectorConfig_t</a>                       | Structure containing select configuration. Used to translate between frequency and output path configuration . . . . . | 152 |
| <a href="#">pathSelectorIndex_t</a>                        | Structure to index to path selector configs . . . . .                                                                  | 153 |
| <a href="#">T_idtApIConfig::preDiv_s</a>                   | . . . . .                                                                                                              | 153 |
| <a href="#">T_idtApIConfig::outputConfig_s::qaConfig_s</a> | . . . . .                                                                                                              | 154 |
| <a href="#">T_idtApIConfig::outputDiv_s::qaDiv_s</a>       | . . . . .                                                                                                              | 155 |
| <a href="#">T_idtApIConfig::outputConfig_s::qbConfig_s</a> | . . . . .                                                                                                              | 155 |
| <a href="#">T_idtApIConfig::outputDiv_s::qbDiv_s</a>       | . . . . .                                                                                                              | 156 |
| <a href="#">T_apIOutputFrequencyEntry</a>                  | Structure used to translate APLL configuration and output port frequency . . . . .                                     | 156 |
| <a href="#">T_idtApIConfig</a>                             | Structure containing APLL full configuration . . . . .                                                                 | 157 |
| <a href="#">T_idtApIConfigPerOutput</a>                    | Structure containing APLL per output configuration . . . . .                                                           | 159 |
| <a href="#">T_idtApIConfigFreqMapping</a>                  | Type definition for mapping output frequency to APLL configuration . . . . .                                           | 160 |
| <a href="#">T_idtApIXtal</a>                               | Structure containing APLL crystal id and frequency information . . . . .                                               | 161 |
| <a href="#">T_idtApIXtalList</a>                           | Wrapper structure containing APLL crystal id and frequency information . . . . .                                       | 162 |
| <a href="#">T_idtDrvAccess</a>                             | Initialization structure detailing device access information for device driver . . . . .                               | 164 |
| <a href="#">T_idtDrvDeviceConfig</a>                       | Device type/variant information . . . . .                                                                              | 165 |
| <a href="#">T_idtDrvHdlr</a>                               | Device driver handler . . . . .                                                                                        | 167 |
| <a href="#">T_IDTFreq2bfVal</a>                            | Structure to contain translation information for frequency to bitfield value . . . . .                                 | 169 |

|                                                     |                                                                              |     |
|-----------------------------------------------------|------------------------------------------------------------------------------|-----|
| <a href="#">T_idtHfDivEntry</a>                     | Structure to use as high frequency (HF) divider information . . . . .        | 170 |
| <a href="#">T_idtI2CtransData</a>                   | I2C address translation information used internally by the driver . . . . .  | 170 |
| <a href="#">T_idtOutputApIConfig</a>                | Structure containing APLL inforatmion . . . . .                              | 171 |
| <a href="#">T_IDTPathConfiguration</a>              | Structure containing output path configuration . . . . .                     | 172 |
| <a href="#">T_idtSonetGigEthBitfield2PathConfig</a> | Structure Translate from Sonet/GigE bitfield to path configuration . . . . . | 173 |
| <a href="#">T_SampleConfigEntry</a>                 | Sample configuration information . . . . .                                   | 174 |

## Chapter 3

# File Index

### 3.1 File List

Here is a list of all files with brief descriptions:

|                                                  |     |
|--------------------------------------------------|-----|
| 82V3910/dev/include/idt82V3910.h . . . . .       | 175 |
| 82V3910/dev/src/82V3910.c . . . . .              | 176 |
| 82V3910/sample/include/access.h . . . . .        | 178 |
| 82V3910/sample/include/loadstore.h . . . . .     | 181 |
| 82V3910/sample/include/sample.h . . . . .        | 182 |
| 82V3910/sample/include/sampleAppl.h . . . . .    | 185 |
| 82V3910/sample/src/access.c . . . . .            | 189 |
| 82V3910/sample/src/loadstore.c . . . . .         | 192 |
| 82V3910/sample/src/main.c . . . . .              | 194 |
| 82V3910/sample/src/sample.c . . . . .            | 198 |
| 82V3910/sample/src/sampleAppl.c . . . . .        | 201 |
| appl/access/sim/ApplReadWrite_sim.c . . . . .    | 205 |
| appl/access/sim/ApplReadWrite_sim.h . . . . .    | 207 |
| appl/include/ApplApi.h . . . . .                 | 208 |
| appl/include/ApplFreqProfile.h . . . . .         | 210 |
| appl/include/ApplLogging.h . . . . .             | 212 |
| appl/include/ApplRegDef.h . . . . .              | 212 |
| appl/include/ApplTypeDef.h . . . . .             | 213 |
| appl/src/ApplApi.c . . . . .                     | 215 |
| appl/src/ApplFreqProfile.c . . . . .             | 219 |
| common/access/sim/ReadWrite_sim.c . . . . .      | 222 |
| common/access/sim/ReadWrite_sim.h . . . . .      | 225 |
| common/hlapi/include/inputFreq.h . . . . .       | 226 |
| common/hlapi/include/outputFreq.h . . . . .      | 227 |
| common/hlapi/include/outputFreq_priv.h . . . . . | 232 |
| common/hlapi/include/profiles.h . . . . .        | 237 |
| common/hlapi/src/inputFreq.c . . . . .           | 240 |
| common/hlapi/src/outputFreq.c . . . . .          | 242 |
| common/hlapi/src/outputFreqString.c . . . . .    | 245 |
| common/hlapi/src/outputFreqUtil.c . . . . .      | 248 |
| common/hlapi/src/profiles.c . . . . .            | 250 |
| common/include/Appl.h . . . . .                  | 253 |
| common/include/Define.h . . . . .                | 255 |
| common/include/Dpll.h . . . . .                  | 261 |
| common/include/DpllCnfg.h . . . . .              | 262 |
| common/include/General.h . . . . .               | 268 |
| common/include/Hal.h . . . . .                   | 271 |
| common/include/Input.h . . . . .                 | 272 |

|                                                                                            |     |
|--------------------------------------------------------------------------------------------|-----|
| common/include/Output.h . . . . .                                                          | 275 |
| common/include/RegisterDef.h . . . . .                                                     |     |
| This header file contains register and bitfield information for accessing device . . . . . | 277 |
| common/src/Apll.c . . . . .                                                                | 285 |
| common/src/DpllCnfg.c . . . . .                                                            | 286 |
| common/src/General.c . . . . .                                                             | 291 |
| common/src/Hal.c . . . . .                                                                 | 293 |
| common/src/Input.c . . . . .                                                               | 294 |
| common/src/Output.c . . . . .                                                              | 297 |

## Chapter 4

# Module Documentation

### 4.1 DPLL Private header file

Functions in this group are related to DPLL provisioning.

#### Data Structures

- struct [T\\_IDTPathConfiguration](#)  
*Structure containing output path configuration.*

#### Macros

- #define [SonetGigEthSel\\_NUMMAX](#) 2  
*Constant defining number of instances of SonetGigEth selectors.*

#### Enumerations

- enum [T\\_idtT0DpllSelectorA](#) {  
[T0DpllSelA\\_16E1](#), [T0DpllSelA\\_16T1](#), [T0DpllSelA\\_GSM](#), [T0DpllSelA\\_OBSAI](#),  
[T0DpllSelA\\_MAX](#) }  
*Enumerated type listing T0 DPLL selector A options.*
- enum [T\\_idtT0DpllSelectorB](#) {  
[T0DpllSelB\\_12E1](#), [T0DpllSelB\\_GPS](#), [T0DpllSelB\\_E3](#), [T0DpllSelB\\_T3](#),  
[T0DpllSelB\\_MAX](#) }  
*Enumerated type listing T0 DPLL selector B options.*
- enum [T\\_idtT4DpllSelectorA](#) {  
[T4DpllSelA\\_16E1](#), [T4DpllSelA\\_16T1](#), [T4DpllSelA\\_GSM](#), [T4DpllSelA\\_GPS](#),  
[T4DpllSelA\\_MAX](#) }  
*Enumerated type listing T4 DPLL selector A options.*
- enum [T\\_idtT4DpllSelectorB](#) {  
[T4DpllSelB\\_12E1](#), [T4DpllSelB\\_24T1](#), [T4DpllSelB\\_E3](#), [T4DpllSelB\\_T3](#),  
[T4DpllSelB\\_MAX](#) }  
*Enumerated type listing T4 DPLL selector B options.*
- enum [T\\_idtSonetGigEthSelector](#) {  
[SonetGigEthSel\\_T0DpllSonet](#) = 0, [SonetGigEthSel\\_T4DpllSonet](#) = 4, [SonetGigEthSel\\_T0Dpll10GbE](#) = 8,  
[SonetGigEthSel\\_T0Dpll10GbE\\_6664](#) = 9,  
[SonetGigEthSel\\_T4Dpll10GbE](#) = 12, [SonetGigEthSel\\_T4Dpll10GbE\\_6664](#) = 13, [SonetGigEthSel\\_MAX](#) }  
*Enumerated type listing Sonet/GigE path selection options.*

## Variables

- const [T\\_IDTPathConfiguration](#) [IDTDpll\\_pathConfigurationNULL\\_c](#)  
*Output path configuration no path constant.*
- const [T\\_idtSonetGigEthSelector](#) [IDTDpll\\_sonetGigEthPathConfig2Bitfield](#) [[Dpll\\_MAX](#)][[SonetGigEthOutput\\_MAX](#)]  
*Table to translate from Sonet/GigE path configuration to bitfield value.*

### 4.1.1 Detailed Description

Functions in this group are related to DPLL provisioning.

### 4.1.2 Macro Definition Documentation

#### 4.1.2.1 #define SonetGigEthSel\_NUMMAX 2

Constant defining number of instances of SonetGigEth selectors.

Definition at line 111 of file Dpll.h.

### 4.1.3 Enumeration Type Documentation

#### 4.1.3.1 enum T\_idtSonetGigEthSelector

Enumerated type listing Sonet/GigE path selection options.

Enumerator

***SonetGigEthSel\_T0DpllSonet*** From T0, 622.08MHz  
***SonetGigEthSel\_T4DpllSonet*** From T4, 622.08MHz  
***SonetGigEthSel\_T0Dpll10GbE*** From T0, 625.00MHz  
***SonetGigEthSel\_T0Dpll10GbE\_6664*** From T0, 625\*66/64MHz  
***SonetGigEthSel\_T4Dpll10GbE*** From T4, 625.00MHz  
***SonetGigEthSel\_T4Dpll10GbE\_6664*** From T4, 625\*66/64MHz  
***SonetGigEthSel\_MAX***

Definition at line 98 of file Dpll.h.

#### 4.1.3.2 enum T\_idtT0DpllSelectorA

Enumerated type listing T0 DPLL selector A options.

Enumerator

***T0DpllSelA\_16E1*** 32.768MHz  
***T0DpllSelA\_16T1*** 24.704MHz  
***T0DpllSelA\_GSM*** 26MHz  
***T0DpllSelA\_OBSAI*** 30.72MHz  
***T0DpllSelA\_MAX***

Definition at line 58 of file Dpll.h.

## 4.1.3.3 enum T\_idtT0DpllSelectorB

Enumerated type listing T0 DPLL selector B options.

Enumerator

***T0DpllSelB\_12E1*** 24.576MHz  
***T0DpllSelB\_GPS*** GPS  
***T0DpllSelB\_E3*** 34.368MHz  
***T0DpllSelB\_T3*** 44.736MHz  
***T0DpllSelB\_MAX***

Definition at line 68 of file Dpll.h.

## 4.1.3.4 enum T\_idtT4DpllSelectorA

Enumerated type listing T4 DPLL selector A options.

Enumerator

***T4DpllSelA\_16E1*** 32.768MHz  
***T4DpllSelA\_16T1*** 24.704MHz  
***T4DpllSelA\_GSM*** 26MHz  
***T4DpllSelA\_GPS*** GPS  
***T4DpllSelA\_MAX***

Definition at line 78 of file Dpll.h.

## 4.1.3.5 enum T\_idtT4DpllSelectorB

Enumerated type listing T4 DPLL selector B options.

Enumerator

***T4DpllSelB\_12E1*** 24.576MHz  
***T4DpllSelB\_24T1*** 37.056MHz  
***T4DpllSelB\_E3*** 34.368MHz  
***T4DpllSelB\_T3*** 44.736MHz  
***T4DpllSelB\_MAX***

Definition at line 88 of file Dpll.h.

## 4.1.4 Variable Documentation

## 4.1.4.1 const T\_IDTPathConfiguration IDTDpll\_pathConfigurationNULL\_c

Output path configuration no path constant.

Definition at line 62 of file DpllCnfg.c.

## 4.1.4.2 const T\_idtSonetGigEthSelector IDTDpll\_sonetGigEthPathConfig2Bitfield[Dpll\_MAX][SonetGigEthOutput\_MAX]

Table to translate from Sonet/GigE path configuration to bitfield value.

Definition at line 91 of file DpllCnfg.c.

## 4.2 DPLL Function Module

Functions in this group are related to DPPL provisioning.

### Macros

- `#define DCO_PPT2DCOUNITS 11`  
*Conversion from parts per trillion to register unit value for DCO offset.*
- `#define DCO_MAX_PPT 92274677`  
*Max DCO value.*
- `#define DCO_MIN_PPT -92274688`  
*Max DCO value.*
- `#define IDT_TRANSLATE_EXT_TO_INT_DPLL(hdlr, ext, internal)`  
*Utility macro to check if DPLL instance is supported and to translated from DPLL instance to DPLL type.*

### Enumerations

- `enum T_idtFreqMonFactor {`  
`FreqMonFactor_0p0032 = 0, FreqMonFactor_0p0064 = 1, FreqMonFactor_0p0127 = 2, FreqMonFactor_0p0257 = 3,`  
`FreqMonFactor_0p0514 = 4, FreqMonFactor_0p1030 = 5, FreqMonFactor_0p2060 = 6, FreqMonFactor_0p4120 = 7,`  
`FreqMonFactor_0p8230 = 8, FreqMonFactor_1p6460 = 9, FreqMonFactor_3p2920 = 10, FreqMonFactor_3p8100 = 11,`  
`FreqMonFactor_4p6000 = 12, FreqMonFactor_MAX }`  
*Enumerated list indicating frequency monitoring factor.*
- `enum T_idtBandwidthLimitStage { dpIIBandwidthLimitStart, dpIIBandwidthLimitAcquire, dpIIBandwidthLimitLocked, dpIIBandwidthLimit_MAX }`  
*Enumerated list used for selecting bandwidth limiting stages.*
- `enum T_idtDampingFactor {`  
`DampingFactor_1p2 = 1, DampingFactor_2p5 = 2, DampingFactor_5 = 3, DampingFactor_10 = 4,`  
`DampingFactor_20 = 5, DampingFactor_MAX }`  
*Enumerated list showing options for bandwidth damping factor.*
- `enum T_idtDpIIBandwidth {`  
`dpIIBandwidth_0p5mHz = 0, dpIIBandwidth_1mHz = 1, dpIIBandwidth_2mHz = 2, dpIIBandwidth_4mHz = 3,`  
`dpIIBandwidth_8mHz = 4, dpIIBandwidth_15mHz = 5, dpIIBandwidth_30mHz = 6, dpIIBandwidth_60mHz = 7,`  
`dpIIBandwidth_0p1Hz = 8, dpIIBandwidth_0p3Hz = 9, dpIIBandwidth_0p6Hz = 10, dpIIBandwidth_1p2Hz = 11,`  
`dpIIBandwidth_2p5Hz = 12, dpIIBandwidth_4Hz = 13, dpIIBandwidth_8Hz = 14, dpIIBandwidth_18Hz = 15,`  
`dpIIBandwidth_35Hz = 16, dpIIBandwidth_70Hz = 17, dpIIBandwidth_560Hz = 18, dpIIBandwidth_MAX }`  
*List of options for DPLL bandwidth.*
- `enum T_idtPhaseSlope {`  
`phaseSlope_gr1244st3 = 0, phaseSlope_gr1244st3e = 1, phaseSlope_gr813 = 2, phaseSlope_noLimit = 3,`  
`phaseSlope_MAX }`  
*Enumerated type listing phase slope.*
- `enum T_idtDpIIOperMode {`  
`Automatic_Mode = 0, Force_FreeRun_Mode = 1, Force_Holdover_Mode = 2, Force_Locked_Mode = 4,`  
`Force_Pre_Locked2_Mode = 5, Force_Pre_Locked_Mode = 6, Force_Lost_Phase_Mode = 7, Dco_Mode = 8,`  
`DpIIOperMode_MAX }`  
*Enumerated type for DPLL operation mode.*

- enum `T_idtCoarsePhaseLimit` {  
`coarsePhaseLimit_1` = 0, `coarsePhaseLimit_3` = 1, `coarsePhaseLimit_7` = 2, `coarsePhaseLimit_15` = 3,  
`coarsePhaseLimit_31` = 4, `coarsePhaseLimit_63` = 5, `coarsePhaseLimit_127` = 6, `coarsePhaseLimit_255` = 7,  
`coarsePhaseLimit_511` = 8, `coarsePhaseLimit_1023` = 9, `coarsePhaseLimit_MAX` }  
*Enumerated type listing coarse phase limits.*
- enum `T_idtFinePhaseLimit` {  
`finePhaseLimit_45_90degree` = 1, `finePhaseLimit_90_180degree` = 2, `finePhaseLimit_180_360degree` = 3,  
`finePhaseLimit_20_25nanosec` = 4,  
`finePhaseLimit_60_65nanosec` = 5, `finePhaseLimit_120_125nanosec` = 6, `finePhaseLimit_950_955nanosec` = 7, `finePhaseLimit_MAX` }  
*Enumerated type listing fine phase limits.*
- enum `T_idtDpllHoldover` {  
`Holdover_Instantaneous` = 0, `Holdover_Slow_Average` = 2, `Holdover_Fast_Average` = 3, `Holdover_Manual` = 4,  
`Holdover_MAX` }  
*Enumerated type listing holdover modes.*
- enum `T_idtTemp_holdover` {  
`Temp_Same_As_Full_Holdover` = 0, `Temp_Holdover_Instantaneous` = 1, `Temp_Holdover_Fast_Average` = 2,  
`Temp_Holdover_Slow_Average` = 3,  
`Temp_Holdover_MAX` }  
*Enumerated type listing temp-holdover modes.*
- enum `T_idtOperDpllMode` {  
`OperDpllMode_FreeRun` = 1, `OperDpllMode_HoldOver` = 2, `OperDpllMode_Locked` = 4, `OperDpllMode_Pre-Locked2` = 5,  
`OperDpllMode_PreLocked` = 6, `OperDpllMode_LostPhase` = 7 }  
*Enumerated type listing DPLL operational modes.*
- enum `T_idtSonetGigEthOutput` { `SonetGigEthOutput_Sonet`, `SonetGigEthOutput_10GbE`, `SonetGigEthOutput_10GbE_6664`, `SonetGigEthOutput_MAX` }  
*Enumerated type listing possible Sonet/GigE output frequencies.*
- enum `T_idtDpllOutputSelectorA` {  
`DPLLOutputSelA_16E1`, `DPLLOutputSelA_16T1`, `DPLLOutputSelA_GSM`, `DPLLOutputSelA_OBSAI`,  
`DPLLOutputSelA_GPS`, `DPLLOutputSelA_MAX` }  
*Enumerated type listing DPLL selector A paths.*
- enum `T_idtDpllOutputSelectorB` {  
`DPLLOutputSelB_12E1`, `DPLLOutputSelB_GPS`, `DPLLOutputSelB_E3`, `DPLLOutputSelB_T3`,  
`DPLLOutputSelB_24T1`, `DPLLOutputSelB_MAX` }  
*Enumerated type listing DPLL selector B paths.*
- enum `T_idtDpllPpsPhasePostion` { `PpsPhasePostion_0degree`, `PpsPhasePostion_50ns`, `PpsPhasePostion_100ns`, `PpsPhasePostion_MAX` }  
*Enumerated type listing PPS phase options.*
- enum `T_idtDpllPpsPulseWidth` {  
`PpsPulseWidth_0pt5sec`, `PpsPulseWidth_10ns`, `PpsPulseWidth_200ns`, `PpsPulseWidth_400ns`,  
`PpsPulseWidth_800ns`, `PpsPulseWidth_1us`, `PpsPulseWidth_20us`, `PpsPulseWidth_50us`,  
`PpsPulseWidth_100us`, `PpsPulseWidth_200us`, `PpsPulseWidth_1pt2ms`, `PpsPulseWidth_800us`,  
`PpsPulseWidth_MAX` }  
*Enumerated type listing PPS pulse width.*

## Functions

- `T_idtDpllType IDTDpll_GetTypeOfDpll` (`T_idtDrvHdlr` hdlr, `T_idtDpllInstance` nDpll)  
*Function to determine type of DPLL give a DPLL instance.*
- `void IDTDpll_SetAutoSelInput` (`T_idtDrvHdlr` hdlr, `T_idtDpllInstance` nDpll, `T_osUInt8` mode)  
*Set the DPLL as automatic reference clock selection.*

- T\_osUInt8 [IDTDpll\\_GetAutoSelInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get automatic input selection status.*
- void [IDTDpll\\_SetForceSelInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nInputNo)  
*Set the DPLL to force to lock to a specified reference clock.*
- T\_osUInt8 [IDTDpll\\_GetForceSelInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Set the DPLL to force to lock to a specified reference clock.*
- T\_osUInt8 [IDTDpll\\_Get1stPriorityInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get input number of a valid clock with the highest priority.*
- T\_osUInt8 [IDTDpll\\_Get2ndPriorityInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get input number of a valid clock with the second highest priority.*
- T\_osUInt8 [IDTDpll\\_Get3rdPriorityInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get input number of a valid clock with the third highest priority.*
- T\_osUInt8 [IDTDpll\\_GetCurrentSelectInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get the current selected input clock.*
- void [IDTDpll\\_SetDpllOperMod](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtDpllOperMode nMode)  
*Set DPLL working at the specified operating mode.*
- T\_idtDpllOperMode [IDTDpll\\_GetDpllOperMod](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Set DPLL working at the specified operating mode.*
- T\_osUInt8 [IDTDpll\\_IsDpllLocked](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Inquire if the DPLL is in lock state.*
- void [IDTDpll\\_SetBandWidthDamping](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtBandwidthLimitStage stage, T\_idtDpllBandwidth nBandWidth, T\_idtDampingFactor nDamping)  
*Set the bandwidth and damping factor used for bandwidth limiting.*
- void [IDTDpll\\_GetBandWidthDamping](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtBandwidthLimitStage stage, T\_idtDpllBandwidth \*nBandWidth\_p, T\_idtDampingFactor \*nDamping\_p)  
*Get configured bandwidth and damping factor for bandwidth limiting.*
- void [IDTDpll\\_SetAutoSelBandWidth](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nEnable)  
*Set device to select a suitable bandwidth and damping factor automatically.*
- T\_osUInt8 [IDTDpll\\_GetAutoSelBandWidth](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get auto bandwidth/damping factor status automatically.*
- void [IDTDpll\\_SetDpllFrozen](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nEnable)  
*Set DPLL integral path value frozen.*
- T\_osUInt8 [IDTDpll\\_GetDpllFrozen](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get DPLL integral path value frozen.*
- void [IDTDpll\\_SetPhaseCoarseDetector](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nEnable, T\_osUInt8 nWide, T\_osUInt8 nMultiPhase, T\_osUInt8 nMultiPh8kEn)  
*Set the coarse phase lose detector enable.*
- void [IDTDpll\\_GetPhaseCoarseDetector](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 \*nEnable\_p, T\_osUInt8 \*nWide\_p, T\_osUInt8 \*nMultiPhase\_p, T\_osUInt8 \*nMultiPh8kEn\_p)  
*Set the coarse phase lose detector enable.*
- void [IDTDpll\\_SetPhaseCoarseLimit](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtCoarsePhaseLimit nLimit)  
*Set the limitation of the coarse phase lose detector.*
- T\_idtCoarsePhaseLimit [IDTDpll\\_GetPhaseCoarseLimit](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get the limitation of the coarse phase lose detector.*
- void [IDTDpll\\_SetPhaseFineDetectorEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nEnable)  
*Set the fine phase lose detector enable.*
- T\_osUInt8 [IDTDpll\\_GetPhaseFineDetectorEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get the fine phase lose detector enable.*
- void [IDTDpll\\_SetPhaseFineLimit](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtFinePhaseLimit nLimit)

- Set the limitation of the fine phase lose detector.*

  - [T\\_idtFinePhaseLimit IDTDpll\\_GetPhaseFinelimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
- Set the limitation of the fine phase lose detector.*

  - [void IDTDpll\\_SetFastLossSwitch](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)
- Set reference clock switch if a fast loss occurs.*

  - [T\\_osUInt8 IDTDpll\\_GetFastLossSwitch](#) (T\_idtDrvHdlr hdlr)
- Get reference clock switch if a fast loss occurs.*

  - [void IDTDpll\\_SetHoldoverMode](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtDpllHoldover](#) nMode, T\_osUInt8 nReadMode)
- Set calculation mode of holdover frequency and read back mode.*

  - [void IDTDpll\\_GetHoldoverMode](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtDpllHoldover](#) \*nMode\_p, T\_osUInt8 \*nReadMode\_p)
- Get calculation mode of holdover frequency and read back mode.*

  - [void IDTDpll\\_SetTempHoldoverMode](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtTemp\\_holdover](#) nMode)
- Set calculation mode of temporary holdover frequency.*

  - [T\\_idtTemp\\_holdover IDTDpll\\_GetTempHoldoverMode](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
- Set calculation mode of temporary holdover frequency.*

  - [long IDTDpll\\_GetHoldoverFreq](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
- Get holdover frequency offset (in parts per trillion). In DCO mode, this function returns current DCO offset (in parts per trillion).*

  - [void IDTDpll\\_SetHoldoverFreq](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, long pptOffset)
- Set DCO frequency to device. Note this function should only be called when DCO mode is enabled.*

  - [long IDTDpll\\_GetCurrentDpllFreqOffset](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
- Get current frequency offset of DPLL in parts per trillion (PPT)*

  - [void IDTDpll\\_SetDpllSoftAlarmLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, T\_osUInt8 nLimit)
- Set the limitation of Soft alarm of DPLL.*

  - [T\\_osUInt8 IDTDpll\\_GetDpllSoftAlarmLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
- Get the limitation of Soft alarm of DPLL.*

  - [void IDTDpll\\_SetDpllHardAlarmLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, T\_osUInt16 nLimit)
- Set the limitation of Hard alarm of DPLL.*

  - [T\\_osUInt16 IDTDpll\\_GetDpllHardAlarmLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
- Get the limitation of Hard alarm of DPLL.*

  - [void IDTDpll\\_SetDpllHardAlarmEnable](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)
- Set DPLL in unlocked state when a Hard alarm raised.*

  - [T\\_osUInt8 IDTDpll\\_GetDpllHardAlarmEnable](#) (T\_idtDrvHdlr hdlr)
- Get DPLL in unlocked state when a Hard alarm raised.*

  - [T\\_osUInt16 IDTDpll\\_GetCurrentPhaseError](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
- Get the current phase error of DPLL.*

  - [void IDTDpll\\_SetSonetGigEthOutputPath](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 instance, [T\\_idtDpllInstance](#) inputDpll, [T\\_idtSonetGigEthOutput](#) outputFreq)
- Set Sonet/GigE path configuration.*

  - [void IDTDpll\\_GetSonetGigEthOutputPath](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 instance, [T\\_idtDpllInstance](#) \*inputDpll\_p, [T\\_idtSonetGigEthOutput](#) \*outputFreq\_p)
- Retrieve Sonet/GigE path configuration.*

  - [void IDTDpll\\_SetDpllOutputPathSelectorA](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtDpllOutputSelectorA](#) path)
- Set DPLL output path selector A.*

  - [T\\_idtDpllOutputSelectorA IDTDpll\\_GetDpllOutputPathSelectorA](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
- Get DPLL output path selector A.*

- void [IDTDpll\\_SetDpllOutputPathSelectorB](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtDpllOutputSelectorB path)  
Set DPLL output path selector B.
- T\_idtDpllOutputSelectorB [IDTDpll\\_GetDpllOutputPathSelectorB](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
Get DPLL output path selector B.
- void [IDTDpll\\_SetDpllEthPathEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nDisable)  
Set ETH Enable/Disable.
- T\_osUInt8 [IDTDpll\\_GetDpllEthPathEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
Get ETH Enable/Disable.
- void [IDTDpll\\_SetDpllSelBPathEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nDisable)  
Set DPLL output path selector B Enable/Disable.
- T\_osUInt8 [IDTDpll\\_GetDpllSelBPathEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
Get DPLL output path selector B Enable/Disable.
- void [IDTDpll\\_SetDpll16E116T1Enable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nDisable)  
Set 16E1/16T1 Enable/Disable.
- T\_osUInt8 [IDTDpll\\_GetDpll16E116T1Enable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
Get 16E1/16T1 Enable/Disable.
- void [IDTDpll\\_SetDpllSelAPathEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nDisable)  
Set DPLL output path selector A Enable/Disable.
- T\_osUInt8 [IDTDpll\\_GetDpllSelAPathEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
Set DPLL output path selector A Enable/Disable.
- void [IDTDpll\\_SetFrequencyMonitoringFactor](#) (T\_idtDrvHdlr hdlr, T\_idtFreqMonFactor freqMonFactor)  
Function to set the frequency monitoring factor. Hard and soft alarm thresholds use this factor as their base unit.
- T\_idtFreqMonFactor [IDTDpll\\_GetFrequencyMonitoringFactor](#) (T\_idtDrvHdlr hdlr)  
Function to get the frequency monitoring factor. Hard and soft alarm thresholds use this factor as their base unit.
- void [IDTDpll\\_SetHardAlarmThreshold](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 accepting, T\_osUInt8 rejecting)  
Set the frequency hard alarm accepting thresholds of reference clock monitoring.
- void [IDTDpll\\_GetHardAlarmThreshold](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 \*accepting\_p, T\_osUInt8 \*rejecting\_p)  
Get the frequency hard alarm accepting thresholds of reference clock monitoring.
- void [IDTDpll\\_SetSoftAlarmThreshold](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 accepting, T\_osUInt8 rejecting)  
Set the frequency soft alarm accepting thresholds of reference clock monitoring.
- void [IDTDpll\\_GetSoftAlarmThreshold](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 \*accepting\_p, T\_osUInt8 \*rejecting\_p)  
Get the frequency soft alarm accepting thresholds of reference clock monitoring.
- void [IDTDpll\\_SetIcpCtrl](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nIcp)  
Set charge pump control.
- T\_osUInt8 [IDTDpll\\_GetIcpCtrl](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
Get charge pump control.
- void [IDTDpll\\_SetPhaseSlope](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtPhaseSlope nPhaseSlope)  
Set DPLL phase slope for DPLLs.
- T\_idtPhaseSlope [IDTDpll\\_GetPhaseSlope](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
Get DPLL phase slope for DPLLs.
- void [IDTDpll\\_SetPpsFastLock](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnableFreq, T\_osUInt8 nEnablePhase)  
Enable/Disable 1 PPS fast lock.
- void [IDTDpll\\_GetPpsFastLock](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 \*nEnableFreq\_p, T\_osUInt8 \*nEnablePhase\_p)  
Get enable/disable 1 PPS fast lock status.
- void [IDTDpll\\_SetPhaseTransient](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
Set the configuration when phase transient occurs.
- T\_osUInt8 [IDTDpll\\_GetPhaseTransient](#) (T\_idtDrvHdlr hdlr)

*Get the configuration when phase transient occurs.*

- void [IDTDpll\\_SetHitlessSwitchingFeature](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable, T\_osUInt8 nFrozen)

*Set the configuration of hitless switching feature.*

- void [IDTDpll\\_GetHitlessSwitchingFeature](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 \*nEnable\_p, T\_osUInt8 \*nFrozen\_p)

*Get the configuration of hitless switching feature.*

- void [IDTDpll\\_SetPhaseOffset](#) (T\_idtDrvHdlr hdlr, T\_osUInt16 nOffset)

*Set the phase offset value.*

- T\_osUInt16 [IDTDpll\\_GetPhaseOffset](#) (T\_idtDrvHdlr hdlr)

*Get the phase offset value.*

- void [IDTDpll\\_SetPpsPhase](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtDpllPpsPhasePostion](#) nPhase, [T\\_idtDpllPpsPulseWidth](#) nPulse)

*Set PPS phase and pulse for DPLL.*

- void [IDTDpll\\_GetPpsPhase](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtDpllPpsPhasePostion](#) \*nPhase\_p, [T\\_idtDpllPpsPulseWidth](#) \*nPulse\_p)

*Get PPS phase and pulse for DPLL.*

### 4.2.1 Detailed Description

Functions in this group are related to DPPL provisioning.

### 4.2.2 Macro Definition Documentation

#### 4.2.2.1 #define DCO\_MAX\_PPT 92274677

Max DCO value.

See Also

[IDTDpll\\_SetHoldoverFreq](#)/[IDTDpll\\_GetHoldoverFreq](#)

Definition at line 262 of file DpllCnfg.h.

#### 4.2.2.2 #define DCO\_MIN\_PPT -92274688

Max DCO value.

See Also

[IDTDpll\\_SetHoldoverFreq](#)/[IDTDpll\\_GetHoldoverFreq](#)

Definition at line 266 of file DpllCnfg.h.

#### 4.2.2.3 #define DCO\_PPT2DCOUNITS 11

Conversion from parts per trillion to register unit value for DCO offset.

Definition at line 258 of file DpllCnfg.h.

#### 4.2.2.4 #define IDT\_TRANSLATE\_EXT\_TO\_INT\_DPLL( *hdlr*, *ext*, *internal* )

Value:

```
do {
    OS_ASSERT((ext) < DPLL_INSTANCE_MAX);
    OS_ASSERTS((hdlr->deviceConfig_m->pllExt2IntXlate_ma[(ext)] < Dpll_MAX), \
        ("Unsupported DPLL instance %d\n", (ext)));
    internal = hdlr->deviceConfig_m->pllExt2IntXlate_ma[(ext)];
} while(0)
```

Utility macro to check if DPLL instance is supported and to translated from DPLL instance to DPLL type.

Parameters

|                 |                                      |
|-----------------|--------------------------------------|
| <i>hdlr</i>     | Device driver handle                 |
| <i>ext</i>      | External DPLL instance number (0..N) |
| <i>internal</i> | Internal DPLL instance               |

Definition at line 304 of file DpllCnfg.h.

### 4.2.3 Enumeration Type Documentation

#### 4.2.3.1 enum T\_idtBandwidthLimitStage

Enumerated list used for selecting bandwidth limiting stages.

Enumerator

***dpllBandwidthLimitStart*** Start stage bandwidth limiting stage  
***dpllBandwidthLimitAcquire*** Acquire stage bandwidth limiting stage  
***dpllBandwidthLimitLocked*** Locked stage bandwidth limiting stage  
***dpllBandwidthLimit\_MAX***

Definition at line 79 of file DpllCnfg.h.

#### 4.2.3.2 enum T\_idtCoarsePhaseLimit

Enumerated type listing coarse phase limits.

Enumerator

***coarsePhaseLimit\_1*** +/- 1 UI  
***coarsePhaseLimit\_3*** +/- 3 UI  
***coarsePhaseLimit\_7*** +/- 7 UI  
***coarsePhaseLimit\_15*** +/- 15 UI  
***coarsePhaseLimit\_31*** +/- 31 UI  
***coarsePhaseLimit\_63*** +/- 63 UI  
***coarsePhaseLimit\_127*** +/- 127 UI  
***coarsePhaseLimit\_255*** +/- 255 UI  
***coarsePhaseLimit\_511*** +/- 511 UI  
***coarsePhaseLimit\_1023*** +/- 1023 UI (Only valid for T0)  
***coarsePhaseLimit\_MAX***

Definition at line 159 of file DpllCnfg.h.

## 4.2.3.3 enum T\_idtDampingFactor

Enumerated list showing options for bandwidth damping factor.

Enumerator

***DampingFactor\_1p2*** 1.2  
***DampingFactor\_2p5*** 2.5  
***DampingFactor\_5*** 5  
***DampingFactor\_10*** 10  
***DampingFactor\_20*** 20  
***DampingFactor\_MAX***

Definition at line 91 of file DpllCnfg.h.

## 4.2.3.4 enum T\_idtDpllBandwidth

List of options for DPLL bandwidth.

Enumerator

***dpllBandwidth\_0p5mHz*** 0.5 mHz  
***dpllBandwidth\_1mHz*** 1 mHz  
***dpllBandwidth\_2mHz*** 2 mHz  
***dpllBandwidth\_4mHz*** 4 mHz  
***dpllBandwidth\_8mHz*** 8 mHz  
***dpllBandwidth\_15mHz*** 15 mHz  
***dpllBandwidth\_30mHz*** 30 mHz  
***dpllBandwidth\_60mHz*** 60 mHz  
***dpllBandwidth\_0p1Hz*** 0.1 Hz  
***dpllBandwidth\_0p3Hz*** 0.3 Hz  
***dpllBandwidth\_0p6Hz*** 0.6 Hz  
***dpllBandwidth\_1p2Hz*** 1.2 Hz  
***dpllBandwidth\_2p5Hz*** 2.5 Hz  
***dpllBandwidth\_4Hz*** 4 Hz  
***dpllBandwidth\_8Hz*** 8 Hz  
***dpllBandwidth\_18Hz*** 18 Hz  
***dpllBandwidth\_35Hz*** 35 Hz  
***dpllBandwidth\_70Hz*** 70 Hz  
***dpllBandwidth\_560Hz*** 560 Hz  
***dpllBandwidth\_MAX***

Definition at line 104 of file DpllCnfg.h.

#### 4.2.3.5 enum T\_idtDpllHoldover

Enumerated type listing holdover modes.

Enumerator

***Holdover\_Instantaneous*** Automatic instantaneous mode  
***Holdover\_Slow\_Average*** Automatic slow average mode  
***Holdover\_Fast\_Average*** Automatic fast average mode  
***Holdover\_Manual*** Manual mode  
***Holdover\_MAX***

Definition at line 191 of file DpllCnfg.h.

#### 4.2.3.6 enum T\_idtDpllOperMode

Enumerated type for DPLL operation mode.

Enumerator

***Automatic\_Mode*** Automatic mode  
***Force\_FreeRun\_Mode*** Force to Free-Run mode  
***Force\_Holdover\_Mode*** Force to Holdover mode  
***Force\_Locked\_Mode*** Force to Locked mode  
***Force\_Pre\_Locked2\_Mode*** Force to Pre-Locked2 mode  
***Force\_Pre\_Locked\_Mode*** Force to Pre-Lock mode  
***Force\_Lost\_Phase\_Mode*** Force to Lost-Phase mode  
***Dco\_Mode*** DCO mode  
***DpllOperMode\_MAX***

Definition at line 143 of file DpllCnfg.h.

#### 4.2.3.7 enum T\_idtDpllOutputSelectorA

Enumerated type listing DPLL selector A paths.

Enumerator

***DPLLOutputSelA\_16E1*** 16E1 Output path  
***DPLLOutputSelA\_16T1*** 16T1 Output path  
***DPLLOutputSelA\_GSM*** GSM Output path  
***DPLLOutputSelA\_OBSAI*** OBSAI Output path  
***DPLLOutputSelA\_GPS*** GPS Output path  
***DPLLOutputSelA\_MAX***

Definition at line 235 of file DpllCnfg.h.

## 4.2.3.8 enum T\_idtDpllOutputSelectorB

Enumerated type listing DPLL selector B paths.

Enumerator

***DPLLOutputSelB\_12E1*** 12E1 Output path  
***DPLLOutputSelB\_GPS*** GPS Output path  
***DPLLOutputSelB\_E3*** E3 Output path  
***DPLLOutputSelB\_T3*** T3 Output path  
***DPLLOutputSelB\_24T1*** 24T1 Output path  
***DPLLOutputSelB\_MAX***

Definition at line 247 of file DpllCnfg.h.

## 4.2.3.9 enum T\_idtDpllPpsPhasePostion

Enumerated type listing PPS phase options.

Enumerator

***PpsPhasePostion\_0degree*** 0 degrees  
***PpsPhasePostion\_50ns*** 50 nanoseconds  
***PpsPhasePostion\_100ns*** 100 nanaoseconds  
***PpsPhasePostion\_MAX***

Definition at line 271 of file DpllCnfg.h.

## 4.2.3.10 enum T\_idtDpllPpsPulseWidth

Enumerated type listing PPS pulse width.

Enumerator

***PpsPulseWidth\_Opt5sec*** 0.5 seconds  
***PpsPulseWidth\_10ns*** 10 nanoseconds  
***PpsPulseWidth\_200ns*** 200 nanoseconds  
***PpsPulseWidth\_400ns*** 400 nanoseconds  
***PpsPulseWidth\_800ns*** 800 nanoseconds  
***PpsPulseWidth\_1us*** 1 microsecond  
***PpsPulseWidth\_20us*** 20 microseconds  
***PpsPulseWidth\_50us*** 20 microseconds  
***PpsPulseWidth\_100us*** 100 microseconds  
***PpsPulseWidth\_200us*** 200 microseconds  
***PpsPulseWidth\_1pt2ms***  
***PpsPulseWidth\_800us*** 1.2 miliseonds 800 microseconds  
***PpsPulseWidth\_MAX***

Definition at line 281 of file DpllCnfg.h.

#### 4.2.3.11 enum T\_idtFinePhaseLimit

Enumerated type listing fine phase limits.

Enumerator

***finePhaseLimit\_45\_90degree*** +/- 45-90 degrees  
***finePhaseLimit\_90\_180degree*** +/- 90-180 degrees  
***finePhaseLimit\_180\_360degree*** +/- 180-360 degrees  
***finePhaseLimit\_20\_25nanosec*** +/- 20-25 nanoseconds  
***finePhaseLimit\_60\_65nanosec*** +/- 60-65 nanoseconds  
***finePhaseLimit\_120\_125nanosec*** +/- 120-125 nanoseconds  
***finePhaseLimit\_950\_955nanosec*** +/- 950-955 nanoseconds  
***finePhaseLimit\_MAX***

Definition at line 178 of file DpllCnfg.h.

#### 4.2.3.12 enum T\_idtFreqMonFactor

Enumerated list indicating frequency monitoring factor.

Enumerator

***FreqMonFactor\_0p0032*** 0.0032 ppm  
***FreqMonFactor\_0p0064*** 0.0064 ppm  
***FreqMonFactor\_0p0127*** 0.0127 ppm  
***FreqMonFactor\_0p0257*** 0.0257 ppm  
***FreqMonFactor\_0p0514*** 0.0514 ppm  
***FreqMonFactor\_0p1030*** 0.103 ppm  
***FreqMonFactor\_0p2060*** 0.206 ppm  
***FreqMonFactor\_0p4120*** 0.412 ppm  
***FreqMonFactor\_0p8230*** 0.823 ppm  
***FreqMonFactor\_1p6460*** 1.646 ppm  
***FreqMonFactor\_3p2920*** 3.292 ppm  
***FreqMonFactor\_3p8100*** 3.81 ppm  
***FreqMonFactor\_4p6000*** 4.6 ppm  
***FreqMonFactor\_MAX***

Definition at line 58 of file DpllCnfg.h.

#### 4.2.3.13 enum T\_idtOperDpllMode

Enumerated type listing DPLL operational modes.

Enumerator

***OperDpllMode\_FreeRun*** Free-Run Mode  
***OperDpllMode\_HoldOver*** HoldOver Mode  
***OperDpllMode\_Locked*** Locked Mode  
***OperDpllMode\_PreLocked2*** Pre-Locked 2 Mode  
***OperDpllMode\_PreLocked*** Pre-Locked Mode  
***OperDpllMode\_LostPhase*** Lost Phase Mode

Definition at line 211 of file DpllCnfg.h.

## 4.2.3.14 enum T\_idtPhaseSlope

Enumerated type listing phase slope.

Enumerator

***phaseSlope\_gr1244st3*** GR-1244 Stratum3 61us/s  
***phaseSlope\_gr1244st3e*** GR-1244 Stratum3E, G2-253 Stratum 3 (855ns/s)  
***phaseSlope\_gr813*** GR-813,G.8262 (7.5us/s)  
***phaseSlope\_noLimit*** No limit  
***phaseSlope\_MAX***

Definition at line 131 of file DpllCnfg.h.

## 4.2.3.15 enum T\_idtSonetGigEthOutput

Enumerated type listing possible Sonet/GigE output frequencies.

Enumerator

***SonetGigEthOutput\_Sonet*** SONET/SDH 622.08MHz output  
***SonetGigEthOutput\_10GbE*** Ethernet 625 MHz output  
***SonetGigEthOutput\_10GbE\_6664*** Ethernet OTN 625(66/64) MHz output  
***SonetGigEthOutput\_MAX***

Definition at line 224 of file DpllCnfg.h.

## 4.2.3.16 enum T\_idtTemp\_holdover

Enumerated type listing temp-holdover modes.

Enumerator

***Temp\_Same\_As\_Full\_Holdover*** Same as the DPLL holdover mode  
***Temp\_Holdover\_Instantaneous*** Automatic instantaneous  
***Temp\_Holdover\_Fast\_Average*** Automatic fast average  
***Temp\_Holdover\_Slow\_Average*** Automatic slow average  
***Temp\_Holdover\_MAX***

Definition at line 201 of file DpllCnfg.h.

## 4.2.4 Function Documentation

4.2.4.1 T\_osUInt8 IDTDpll\_Get1stPriorityInput ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get input number of a valid clock with the highest priority.

Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

**Returns**

nInput The number of input clock with highest priority

Definition at line 296 of file DpllCnfg.c.

#### 4.2.4.2 T\_osUInt8 IDTDpll.Get2ndPriorityInput ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get input number of a valid clock with the second highest priority.

**Parameters**

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

**Returns**

nInput The number of input clock with second highest priority

Definition at line 318 of file DpllCnfg.c.

#### 4.2.4.3 T\_osUInt8 IDTDpll.Get3rdPriorityInput ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get input number of a valid clock with the third highest priority.

**Parameters**

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

**Returns**

nInput The number of input clock with third highest priority

Definition at line 340 of file DpllCnfg.c.

#### 4.2.4.4 T\_osUInt8 IDTDpll.GetAutoSelBandWidth ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get auto bandwidth/damping factor status automatically.

**Parameters**

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

**Returns**

0-manual bandwidth/damping factor mode, 1-automatic bandwidth/damping factor mode

Definition at line 723 of file DpllCnfg.c.

#### 4.2.4.5 T\_osUInt8 IDTDpll.GetAutoSelInput ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get automatic input selection status.

## Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

## Returns

0-In manual mode, 1-In auto mode

Definition at line 218 of file DpllCnfg.c.

**4.2.4.6 void IDTDpll\_GetBandWidthDamping ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_idtBandwidthLimitStage *stage*, T\_idtDpllBandwidth \* *nBandWidth\_p*, T\_idtDampingFactor \* *nDamping\_p* )**

Get configured bandwidth and damping factor for bandwidth limiting.

## Parameters

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>         | Device driver handler                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <i>nDpll</i>        | DPLL instance                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <i>stage</i>        | Bandwidth stage                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <i>nBandWidth_p</i> | the bandwidth of DPLL <ul style="list-style-type: none"> <li>• dpllBandwidth_0p5mHz - 0.5 mHz</li> <li>• dpllBandwidth_1mHz - 1 mHz</li> <li>• dpllBandwidth_2mHz - 2 mHz</li> <li>• dpllBandwidth_4mHz - 4 mHz</li> <li>• dpllBandwidth_8mHz - 8 mHz</li> <li>• dpllBandwidth_15mHz - 15 mHz</li> <li>• dpllBandwidth_30mHz - 30 mHz</li> <li>• dpllBandwidth_60mHz - 60 mHz</li> <li>• dpllBandwidth_0p1Hz - 0.1 Hz</li> <li>• dpllBandwidth_0p3Hz - 0.3 Hz</li> <li>• dpllBandwidth_0p6Hz - 0.6 Hz</li> <li>• dpllBandwidth_1p2Hz - 1.2 Hz</li> <li>• dpllBandwidth_2p5Hz - 2.5 Hz</li> <li>• dpllBandwidth_4Hz - 4 Hz</li> <li>• dpllBandwidth_8Hz - 8 Hz</li> <li>• dpllBandwidth_18Hz - 18 Hz</li> <li>• dpllBandwidth_35Hz - 35 Hz</li> <li>• dpllBandwidth_70Hz - 70 Hz</li> <li>• dpllBandwidth_560Hz - 560 Hz</li> </ul> |
| <i>nDamping_p</i>   | the damping factor of DPLL <ul style="list-style-type: none"> <li>• DampingFactor_1p2 - 1.2</li> <li>• DampingFactor_2p5 - 2.5</li> <li>• DampingFactor_5 - 5</li> <li>• DampingFactor_10 - 10</li> <li>• DampingFactor_20 - 20</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

Definition at line 639 of file DpllCnfg.c.

#### 4.2.4.7 long IDTDpll\_GetCurrentDpllFreqOffset ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get current frequency offset of DPLL in parts per trillion (PPT)

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

##### Returns

Frequency offset in parts per trillion

Definition at line 1347 of file DpllCnfg.c.

#### 4.2.4.8 T\_osUInt16 IDTDpll\_GetCurrentPhaseError ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get the current phase error of DPLL.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

##### Returns

nError: a 2' complement signed integer represents current phase error. It is a multiple, the real phase error is:  
nError \* 0.0014

Definition at line 1510 of file DpllCnfg.c.

#### 4.2.4.9 T\_osUInt8 IDTDpll\_GetCurrentSelectInput ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get the current selected input clock.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

##### Returns

nInput the number of selected input clock

Definition at line 362 of file DpllCnfg.c.

#### 4.2.4.10 T\_osUInt8 IDTDpll\_GetDpll16E116T1Enable ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get 16E1/16T1 Enable/Disable.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

## Returns

- 0= Enable
- 1= Disable

Definition at line 1992 of file DpllCnfg.c.

#### 4.2.4.11 T\_osUInt8 IDTDpll\_GetDpllEthPathEnable ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get ETH Enable/Disable.

## Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

## Returns

- 0= Enable
- 1= Disable

Definition at line 1846 of file DpllCnfg.c.

#### 4.2.4.12 T\_osUInt8 IDTDpll\_GetDpllFrozen ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get DPLL integral path value frozen.

## Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

## Returns

- 0= Disable the integral path frozen
- 1= Enable the integral path frozen

Definition at line 780 of file DpllCnfg.c.

#### 4.2.4.13 T\_osUInt8 IDTDpll\_GetDpllHardAlarmEnable ( T\_idtDrvHdlr *hdlr* )

Get DPLL in unlocked state when a Hard alarm raised.

## Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

## Returns

- 0= Ignore Hard alarm, DPLL continues operation
- 1= DPLL is unlock if Hard alarm raised

Definition at line 1495 of file DpllCnfg.c.

#### 4.2.4.14 T\_uint16 IDTDpll\_GetDpllHardAlarmLimit ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get the limitation of Hard alarm of DPLL.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

##### Returns

a 2' complement signed integer represents limitation of hard alarm. It is a multiple, the real limitation is: nLimit \* 0.0014

Definition at line 1458 of file DpllCnfg.c.

#### 4.2.4.15 T\_idtDpllOperMode IDTDpll\_GetDpllOperMod ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Set DPLL working at the specified operating mode.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

##### Returns

nMode

- Automatic\_Mode = Automatic mode
- Force\_FreeRun\_Mode = Force to Free-Run mode
- Force\_Holdover\_Mode = Force to Holdover mode
- Force\_Locked\_Mode = Force to Locked mode
- Force\_Pre\_Locked2\_Mode = Force to Pre-Locked2 mode
- Force\_Pre\_Locked\_Mode = Force to Pre-Lock mode
- Force\_Lost\_Phase\_Mode = Force to Lost-Phase mode
- Dco\_Mode = DCO mode

Definition at line 454 of file DpllCnfg.c.

#### 4.2.4.16 T\_idtDpllOutputSelectorA IDTDpll\_GetDpllOutputPathSelectorA ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get DPLL output path selector A.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

##### Returns

DPLL selector path

Definition at line 1687 of file DpllCnfg.c.

#### 4.2.4.17 T\_idtDpllOutputSelectorB IDTDpll\_GetDpllOutputPathSelectorB ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get DPLL output path selector B.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

##### Returns

DPLL selector path

Definition at line 1771 of file DpllCnfg.c.

#### 4.2.4.18 T\_osUInt8 IDTDpll\_GetDpllSelAPathEnable ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Set DPLL output path selector A Enable/Disable.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

##### Returns

- 0= Enable
- 1= Disable

Definition at line 2064 of file DpllCnfg.c.

#### 4.2.4.19 T\_osUInt8 IDTDpll\_GetDpllSelBPathEnable ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get DPLL output path selector B Enable/Disable.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

##### Returns

- 0= Enable
- 1= Disable

Definition at line 1920 of file DpllCnfg.c.

#### 4.2.4.20 T\_osUInt8 IDTDpll\_GetDpllSoftAlarmLimit ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get the limitation of Soft alarm of DPLL.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

**Returns**

a 2' complement signed integer represents limitation of soft alarm. It is a multiple, the real limitation is:  $nLimit * 0.724$

Definition at line 1410 of file DpllCnfg.c.

#### 4.2.4.21 T\_osUInt8 IDTDpll\_GetFastLossSwitch ( T\_idtDrvHdlr *hdlr* )

Get reference clock switch if a fast loss occurs.

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

**Returns**

- 0= Disable switch, DPLL enters temporary holdover state
- 1= Enable switch, DPLL enters Phase-loss state

Definition at line 1116 of file DpllCnfg.c.

#### 4.2.4.22 T\_osUInt8 IDTDpll\_GetForceSelInput ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Set the DPLL to force to lock to a specified reference clock.

**Parameters**

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

**Returns**

Port number of selected input clock

Definition at line 271 of file DpllCnfg.c.

#### 4.2.4.23 T\_idtFreqMonFactor IDTDpll\_GetFrequencyMonitoringFactor ( T\_idtDrvHdlr *hdlr* )

Function to get the frequency monitoring factor. Hard and soft alarm thresholds use this factor as their base unit.

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

**Returns**

Frequency monitoring factor value.

- FreqMonFactor\_0p0032 - 0.0032 ppm
- FreqMonFactor\_0p0064 - 0.0064 ppm
- FreqMonFactor\_0p0127 - 0.0127 ppm
- FreqMonFactor\_0p0257 - 0.0257 ppm
- FreqMonFactor\_0p0514 - 0.0514 ppm
- FreqMonFactor\_0p1030 - 0.103 ppm

- FreqMonFactor\_0p2060 - 0.206 ppm
- FreqMonFactor\_0p4120 - 0.412 ppm
- FreqMonFactor\_0p8230 - 0.823 ppm
- FreqMonFactor\_1p6460 - 1.646 ppm
- FreqMonFactor\_3p2920 - 3.29 ppm
- FreqMonFactor\_3p8100 - 3.81 ppm
- FreqMonFactor\_4p6000 - 4.6 ppm

Definition at line 2146 of file DpllCnfg.c.

#### 4.2.4.24 void IDTDpll\_GetHardAlarmThreshold ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 \* *accepting\_p*, T\_osUInt8 \* *rejecting\_p* )

Get the frequency hard alarm accepting thresholds of reference clock monitoring.

##### Parameters

|                    |                                                                                                                                          |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>        | Device driver handler                                                                                                                    |
| <i>accepting_p</i> | Accepting threshold (ppm) is a multiple of frequency monitoring factor value. Accepting Threshold = (AcceptingValue + 1)*(FreqMonFactor) |
| <i>rejecting_p</i> | Rejecting threshold (ppm) is a multiple of frequency monitoring factor value. Rejecting Threshold = (RejectingValue + 1)*(FreqMonFactor) |

Definition at line 2195 of file DpllCnfg.c.

#### 4.2.4.25 void IDTDpll\_GetHitlessSwitchingFeature ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 \* *nEnable\_p*, T\_osUInt8 \* *nFrozen\_p* )

Get the configuration of hitless switching feature.

##### Parameters

|                  |                                                                                                                                                                                 |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>      | Device driver handler                                                                                                                                                           |
| <i>nEnable_p</i> | <ul style="list-style-type: none"> <li>• 0= Disable the hitless switching feature;</li> <li>• 1= Enable the hitless switching feature;</li> </ul>                               |
| <i>nFrozen_p</i> | <ul style="list-style-type: none"> <li>• 0= hitless switching feature is not frozen;</li> <li>• 1= HS feature is frozen, ignore the further hitless switching events</li> </ul> |

Definition at line 2546 of file DpllCnfg.c.

#### 4.2.4.26 long IDTDpll\_GetHoldoverFreq ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get holdover frequency offset (in parts per trillion). In DCO mode, this function returns current DCO offset (in parts per trillion).

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

**Returns**

Parts per trillion

Definition at line 1255 of file DpllCnfg.c.

4.2.4.27 void IDTDpll\_GetHoldoverMode ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_idtDpllHoldover \* *nMode\_p*, T\_osUInt8 \* *nReadMode\_p* )

Get calculation mode of holdover frequency and read back mode.

**Parameters**

|                    |                                                                                                                                                                                                                                                           |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>        | Device driver handler                                                                                                                                                                                                                                     |
| <i>nDpll</i>       | DPLL instance                                                                                                                                                                                                                                             |
| <i>nMode_p</i>     | <ul style="list-style-type: none"> <li>Holdover_Instantaneous = automatic instantaneous mode</li> <li>Holdover_Slow_Average = automatic slow average</li> <li>Holdover_Fast_Average = Automatic fast average</li> <li>Holdover_Manual = Manual</li> </ul> |
| <i>nReadMode_p</i> | <ul style="list-style-type: none"> <li>0= Don't read value from device</li> <li>1= Read value calculated by device back to the holdover frequency register(0x5D,0x5E and 0x5F)</li> </ul>                                                                 |

Definition at line 1176 of file DpllCnfg.c.

4.2.4.28 T\_osUInt8 IDTDpll\_GetIcpCtrl ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get charge pump control.

**Parameters**

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

**Returns**

Charge pump current selection ( 0 - 0x1F ); 10 = 40uA;

Definition at line 2312 of file DpllCnfg.c.

4.2.4.29 void IDTDpll\_GetPhaseCoarseDetector ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_osUInt8 \* *nEnable\_p*, T\_osUInt8 \* *nWide\_p*, T\_osUInt8 \* *nMultiPhase\_p*, T\_osUInt8 \* *nMultiPh8kEn\_p* )

Set the coarse phase lose detector enable.

**Parameters**

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

|                       |                                                                                                                                                                |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>nEnable_p</i>      | <ul style="list-style-type: none"> <li>• 0= Disable the coarse phase detector</li> <li>• 1= Enable the coarse phase detector</li> </ul>                        |
| <i>nWide_p</i>        | <ul style="list-style-type: none"> <li>• 0= Wide range disable</li> <li>• 1= Wide range enable</li> </ul>                                                      |
| <i>nMultiPhase_p</i>  | <ul style="list-style-type: none"> <li>• 0= Limit to +/- 1UI</li> <li>• 1= Limit to PH_LOS_COARSE_LIMT register</li> </ul>                                     |
| <i>nMultiPh8kEn_p</i> | <ul style="list-style-type: none"> <li>• 0= Limit to +/- 1UI when reference clock 8/4/2KHz clock</li> <li>• 1= Limit to PH_LOS_COARSE_LIMT register</li> </ul> |

Definition at line 876 of file DpllCnfg.c.

#### 4.2.4.30 T\_idtCoarsePhaseLimit IDTDpll\_GetPhaseCoarseLimit ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get the limitation of the coarse phase lose detector.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

##### Returns

Limitation of coarse phase loss detector.

- coarsePhaseLimit\_1 - +/- 1 UI
- coarsePhaseLimit\_3 - +/- 3 UI
- coarsePhaseLimit\_7 - +/- 7 UI
- coarsePhaseLimit\_15 - +/- 15 UI
- coarsePhaseLimit\_31 - +/- 31 UI
- coarsePhaseLimit\_63 - +/- 63 UI
- coarsePhaseLimit\_127 - +/- 127 UI
- coarsePhaseLimit\_255 - +/- 255 UI
- coarsePhaseLimit\_511 - +/- 511 UI
- coarsePhaseLimit\_1023 - +/- 1023 UI T0 only

Definition at line 968 of file DpllCnfg.c.

#### 4.2.4.31 T\_osUInt8 IDTDpll\_GetPhaseFineDetectorEnable ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get the fine phase lose detector enable.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

**Returns**

- 0= Disable the fine phase detector
- 1= Enable the fine phase detector

Definition at line 1016 of file DpllCnfg.c.

#### 4.2.4.32 **T\_idtFinePhaseLimit** IDTDpll\_GetPhaseFineLimit ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Set the limitation of the fine phase lose detector.

**Parameters**

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

**Returns**

Limitation of fine phase loss detector

- finePhaseLimit\_45\_90degree = +/- 45-90 degrees
- finePhaseLimit\_90\_180degree = +/- 90-180 degrees
- finePhaseLimit\_180\_360degree = +/- 180-360 degrees
- finePhaseLimit\_20\_25nanosec = +/- 20-25 nanoseconds
- finePhaseLimit\_60\_65nanosec = +/- 60-65 nanoseconds
- finePhaseLimit\_120\_125nanosec = +/- 120-125 nanoseconds
- finePhaseLimit\_950\_955nanosec = +/- 950-955 nanoseconds

Definition at line 1077 of file DpllCnfg.c.

#### 4.2.4.33 **T\_osUInt16** IDTDpll\_GetPhaseOffset ( T\_idtDrvHdlr *hdlr* )

Get the phase offset value.

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

**Returns**

Phase offset= nOffset \* 0x61 ns;

Definition at line 2584 of file DpllCnfg.c.

#### 4.2.4.34 **T\_idtPhaseSlope** IDTDpll\_GetPhaseSlope ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Get DPLL phase slope for DPLLs.

## Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

## Returns

Phase slope

- phaseSlope\_gr1244st3 = GR-1244 ST3 (61us/s)
- phaseSlope\_gr1244st3e = GR-1244 ST2,3E,ST3 (855ns/s)
- phaseSlope\_gr813 = GR-813,G.8262 (7.5us/s)
- phaseSlope\_noLimit = No limitation

Definition at line 2397 of file DpllCnfg.c.

4.2.4.35 T\_osUInt8 IDTDpll\_GetPhaseTransient ( T\_idtDrvHdlr *hdlr* )

Get the configuration when phase transient occurs.

## Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

## Returns

- 0= Disable the phase transient monitor
- 1= Enable the phase transient monitor

Definition at line 2503 of file DpllCnfg.c.

4.2.4.36 void IDTDpll\_GetPpsFastLock ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 \* *nEnableFreq\_p*, T\_osUInt8 \* *nEnablePhase\_p* )

Get enable/disable 1 PPS fast lock status.

## Parameters

|                       |                                            |
|-----------------------|--------------------------------------------|
| <i>hdlr</i>           | Device driver handler                      |
| <i>nEnableFreq_p</i>  | Enable (1) / Disable (0) frequency locking |
| <i>nEnablePhase_p</i> | Enable (1) / Disable (0) phase locking     |

Definition at line 2461 of file DpllCnfg.c.

4.2.4.37 void IDTDpll\_GetPpsPhase ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_idtDpllPpsPhasePostion \* *nPhase\_p*, T\_idtDpllPpsPulseWidth \* *nPulse\_p* )

Get PPS phase and pulse for DPLL.

## Parameters

|                 |                       |
|-----------------|-----------------------|
| <i>hdlr</i>     | Device driver handler |
| <i>nDpll</i>    | DPLL instance         |
| <i>nPhase_p</i> | Phase position        |

See Also

[T\\_idtDpllPpsPhasePostion](#)

Parameters

|                 |             |
|-----------------|-------------|
| <i>nPulse_p</i> | Pulse width |
|-----------------|-------------|

See Also

[T\\_idtDpllPpsPulseWidth](#)

Definition at line 2661 of file DpllCnfg.c.

**4.2.4.38** void IDTDpll\_GetSoftAlarmThreshold ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 \* *accepting\_p*, T\_osUInt8 \* *rejecting\_p* )

Get the frequency soft alarm accepting thresholds of reference clock monitoring.

Parameters

|                    |                                                                                                                                          |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>        | Device driver handler                                                                                                                    |
| <i>accepting_p</i> | Accepting threshold (ppm) is a multiple of frequency monitoring factor value. Accepting Threshold = (AcceptingValue + 1)*(FreqMonFactor) |
| <i>rejecting_p</i> | Rejecting threshold (ppm) is a multiple of frequency monitoring factor value. Rejecting Threshold = (RejectingValue + 1)*(FreqMonFactor) |

Definition at line 2249 of file DpllCnfg.c.

**4.2.4.39** void IDTDpll\_GetSonetGigEthOutputPath ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *instance*, T\_idtDpllInstance \* *inputDpll\_p*, T\_idtSonetGigEthOutput \* *outputFreq\_p* )

Retrieve Sonet/GigE path configuration.

Parameters

|                     |                                                 |
|---------------------|-------------------------------------------------|
| <i>hdlr</i>         | Device driver handler                           |
| <i>instance</i>     | Sonet/GigE path instance 0..1                   |
| <i>inputDpll_p</i>  | Provisioned DPLL instance of Sonet/GigE path    |
| <i>outputFreq_p</i> | Provisioned output frequency of Sonet/GigE path |

Definition at line 1609 of file DpllCnfg.c.

**4.2.4.40** T\_idtTemp\_holdover IDTDpll\_GetTempHoldoverMode ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Set calculation mode of temporary holdover frequency.

Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Returns

- Temp\_Same\_As\_Full\_Holdover = Same as the DPLL holdover mode
- Temp\_Holdover\_Instantaneous = Automatic instantaneous mode
- Temp\_Holdover\_Fast\_Average = Automatic fast average

- Temp\_Holdover\_Slow\_Average = Automatic slow average

Definition at line 1234 of file DpllCnfg.c.

#### 4.2.4.41 T\_idtDpllType IDTDpll\_GetTypeOfDpll ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Function to determine type of DPLL give a DPLL instance.

This functions is useful in determining supported features of each DPLL

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

##### Returns

Dpll\_T0- T0 DPLL, Dpll\_T4- T4 DPLL, Dpll\_MAX- DPLL instance not supported

Definition at line 176 of file DpllCnfg.c.

#### 4.2.4.42 T\_osUInt8 IDTDpll\_IsDpllLocked ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

Inquire if the DPLL is in lock state.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

##### Returns

nState

- 0= DPLL is in unlocked state
- 1= DPLL is in locked state

Definition at line 498 of file DpllCnfg.c.

#### 4.2.4.43 void IDTDpll\_SetAutoSelBandWidth ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_osUInt8 *nEnable* )

Set device to select a suitable bandwidth and damping factor automatically.

##### Parameters

|                |                                                                                                                                                   |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>    | Device driver handler                                                                                                                             |
| <i>nDpll</i>   | DPLL instance                                                                                                                                     |
| <i>nEnable</i> | <ul style="list-style-type: none"> <li>• 0= Only locked bandwidth and damping factor selected</li> <li>• 1= enable automatic selection</li> </ul> |

Definition at line 694 of file DpllCnfg.c.

4.2.4.44 void IDTDpll\_SetAutoSelInput ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_osUInt8 *mode* )

Set the DPLL as automatic reference clock selection.

#### Parameters

|              |                                  |
|--------------|----------------------------------|
| <i>hdlr</i>  | Device driver handler            |
| <i>nDpll</i> | DPLL instance                    |
| <i>mode</i>  | 0-In manual mode, 1-In auto mode |

Definition at line 190 of file DpllCnfg.c.

4.2.4.45 void IDTDpll\_SetBandWidthDamping ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_idtBandwidthLimitStage *stage*, T\_idtDpllBandwidth *nBandWidth*, T\_idtDampingFactor *nDamping* )

Set the bandwidth and damping factor used for bandwidth limiting.

#### Parameters

|                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <i>nDpll</i>      | DPLL instance                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <i>stage</i>      | Bandwidth stage                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <i>nBandWidth</i> | the bandwidth of DPLL <ul style="list-style-type: none"> <li>• dppllBandwidth_0p5mHz - 0.5 mHz</li> <li>• dppllBandwidth_1mHz - 1 mHz</li> <li>• dppllBandwidth_2mHz - 2 mHz</li> <li>• dppllBandwidth_4mHz - 4 mHz</li> <li>• dppllBandwidth_8mHz - 8 mHz</li> <li>• dppllBandwidth_15mHz - 15 mHz</li> <li>• dppllBandwidth_30mHz - 30 mHz</li> <li>• dppllBandwidth_60mHz - 60 mHz</li> <li>• dppllBandwidth_0p1Hz - 0.1 Hz</li> <li>• dppllBandwidth_0p3Hz - 0.3 Hz</li> <li>• dppllBandwidth_0p6Hz - 0.6 Hz</li> <li>• dppllBandwidth_1p2Hz - 1.2 Hz</li> <li>• dppllBandwidth_2p5Hz - 2.5 Hz</li> <li>• dppllBandwidth_4Hz - 4 Hz</li> <li>• dppllBandwidth_8Hz - 8 Hz</li> <li>• dppllBandwidth_18Hz - 18 Hz</li> <li>• dppllBandwidth_35Hz - 35 Hz</li> <li>• dppllBandwidth_70Hz - 70 Hz</li> <li>• dppllBandwidth_560Hz - 560 Hz</li> </ul> |
| <i>nDamping</i>   | the damping factor of DPLL <ul style="list-style-type: none"> <li>• DampingFactor_1p2 - 1.2</li> <li>• DampingFactor_2p5 - 2.5</li> <li>• DampingFactor_5 - 5</li> <li>• DampingFactor_10 - 10</li> <li>• DampingFactor_20 - 20</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

Definition at line 550 of file DpllCnfg.c.

4.2.4.46 void IDTDpll\_SetDpll16E116T1Enable ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_osUInt8 *nDisable* )

Set 16E1/16T1 Enable/Disable.

#### Parameters

|                 |                                                                                     |
|-----------------|-------------------------------------------------------------------------------------|
| <i>hdlr</i>     | Device driver handler                                                               |
| <i>nDpll</i>    | DPLL instance                                                                       |
| <i>nDisable</i> | <ul style="list-style-type: none"> <li>• 0= Enable</li> <li>• 1= Disable</li> </ul> |

Definition at line 1956 of file DpllCnfg.c.

4.2.4.47 void IDTDpll\_SetDpllEthPathEnable ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_osUInt8 *nDisable* )

Set ETH Enable/Disable.

#### Parameters

|                 |                                                                                     |
|-----------------|-------------------------------------------------------------------------------------|
| <i>hdlr</i>     | Device driver handler                                                               |
| <i>nDpll</i>    | DPLL instance                                                                       |
| <i>nDisable</i> | <ul style="list-style-type: none"> <li>• 0= Enable</li> <li>• 1= Disable</li> </ul> |

Definition at line 1810 of file DpllCnfg.c.

4.2.4.48 void IDTDpll\_SetDpllFrozen ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_osUInt8 *nEnable* )

Set DPLL integral path value frozen.

#### Parameters

|                |                                                                                                                                       |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>    | Device driver handler                                                                                                                 |
| <i>nDpll</i>   | DPLL instance                                                                                                                         |
| <i>nEnable</i> | <ul style="list-style-type: none"> <li>• 0= Disable the integral path frozen</li> <li>• 1= Enable the integral path frozen</li> </ul> |

Definition at line 750 of file DpllCnfg.c.

4.2.4.49 void IDTDpll\_SetDpllHardAlarmEnable ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nEnable* )

Set DPLL in unlocked state when a Hard alarm raised.

#### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

|                |                                                                                                                                                      |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>nEnable</i> | <ul style="list-style-type: none"> <li>• 0= Ignore Hard alarm, DPLL continues operation</li> <li>• 1= DPLL is unlock if Hard alarm raised</li> </ul> |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------|

Definition at line 1479 of file DpllCnfg.c.

4.2.4.50 void IDTDpll\_SetDpllHardAlarmLimit ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_osUInt16 *nLimit* )

Set the limitation of Hard alarm of DPLL.

#### Parameters

|               |                                                                                                                                 |
|---------------|---------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>   | Device driver handler                                                                                                           |
| <i>nDpll</i>  | DPLL instance                                                                                                                   |
| <i>nLimit</i> | A 2' complement signed integer represents limitation of hard alarm. It is a multiple, the real limitation is: $nLimit * 0.0014$ |

Definition at line 1430 of file DpllCnfg.c.

4.2.4.51 void IDTDpll\_SetDpllOperMod ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_idtDpllOperMode *nMode* )

Set DPLL working at the specified operating mode.

#### Parameters

|              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>  | Device driver handler                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <i>nDpll</i> | DPLL instance                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <i>nMode</i> | <ul style="list-style-type: none"> <li>• Automatic_Mode = Automatic mode</li> <li>• Force_FreeRun_Mode = Force to Free-Run mode</li> <li>• Force_Holdover_Mode = Force to Holdover mode</li> <li>• Force_Locked_Mode = Force to Locked mode</li> <li>• Force_Pre_Locked2_Mode = Force to Pre-Locked2 mode</li> <li>• Force_Pre_Locked_Mode = Force to Pre-Lock mode</li> <li>• Force_Lost_Phase_Mode = Force to Lost-Phase mode</li> <li>• Dco_Mode = DCO mode</li> </ul> |

Definition at line 390 of file DpllCnfg.c.

4.2.4.52 void IDTDpll\_SetDpllOutputPathSelectorA ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_idtDpllOutputSelectorA *path* )

Set DPLL output path selector A.

#### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |
| <i>path</i>  | DPLL selector path    |

Definition at line 1646 of file DpllCnfg.c.

4.2.4.53 void IDTDpll\_SetDpllOutputPathSelectorB ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_idtDpllOutputSelectorB *path* )

Set DPLL output path selector B.

#### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |
| <i>path</i>  | DPLL selector path    |

Definition at line 1724 of file DpllCnfg.c.

4.2.4.54 void IDTDpll\_SetDpllSelAPathEnable ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_osUInt8 *nDisable* )

Set DPLL output path selector A Enable/Disable.

#### Parameters

|                 |                                                                                     |
|-----------------|-------------------------------------------------------------------------------------|
| <i>hdlr</i>     | Device driver handler                                                               |
| <i>nDpll</i>    | DPLL instance                                                                       |
| <i>nDisable</i> | <ul style="list-style-type: none"> <li>• 0= Enable</li> <li>• 1= Disable</li> </ul> |

Definition at line 2028 of file DpllCnfg.c.

4.2.4.55 void IDTDpll\_SetDpllSelBPathEnable ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_osUInt8 *nDisable* )

Set DPLL output path selector B Enable/Disable.

#### Parameters

|                 |                                                                                     |
|-----------------|-------------------------------------------------------------------------------------|
| <i>hdlr</i>     | Device driver handler                                                               |
| <i>nDpll</i>    | DPLL instance                                                                       |
| <i>nDisable</i> | <ul style="list-style-type: none"> <li>• 0= Enable</li> <li>• 1= Disable</li> </ul> |

Definition at line 1884 of file DpllCnfg.c.

4.2.4.56 void IDTDpll\_SetDpllSoftAlarmLimit ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_osUInt8 *nLimit* )

Set the limitation of Soft alarm of DPLL.

#### Parameters

|               |                                                                                                                                |
|---------------|--------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>   | Device driver handler                                                                                                          |
| <i>nDpll</i>  | DPLL instance                                                                                                                  |
| <i>nLimit</i> | A 2' complement signed integer represents limitation of soft alarm. It is a multiple, the real limitation is: $nLimit * 0.724$ |

Definition at line 1386 of file DpllCnfg.c.

#### 4.2.4.57 void IDTDpll\_SetFastLossSwitch ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nEnable* )

Set reference clock switch if a fast loss occurs.

##### Parameters

|                |                                                                                                                                                                       |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>    | Device driver handler                                                                                                                                                 |
| <i>nEnable</i> | <ul style="list-style-type: none"> <li>• 0= Disable switch, DPLL enters temporary holdover state</li> <li>• 1= Enable switch, DPLL enters Phase-loss state</li> </ul> |

Definition at line 1100 of file DpllCnfg.c.

#### 4.2.4.58 void IDTDpll\_SetForceSelInput ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_osUInt8 *nInputNo* )

Set the DPLL to force to lock to a specified reference clock.

##### Parameters

|                 |                                       |
|-----------------|---------------------------------------|
| <i>hdlr</i>     | Device driver handler                 |
| <i>nDpll</i>    | DPLL instance                         |
| <i>nInputNo</i> | The number of a specified input clock |

Definition at line 243 of file DpllCnfg.c.

#### 4.2.4.59 void IDTDpll\_SetFrequencyMonitoringFactor ( T\_idtDrvHdlr *hdlr*, T\_idtFreqMonFactor *freqMonFactor* )

Function to set the frequency monitoring factor. Hard and soft alarm thresholds use this factor as their base unit.

##### Parameters

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>          | Device driver handler                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <i>freqMonFactor</i> | <p>Frequency monitoring factor value.</p> <ul style="list-style-type: none"> <li>• FreqMonFactor_0p0032 - 0.0032 ppm</li> <li>• FreqMonFactor_0p0064 - 0.0064 ppm</li> <li>• FreqMonFactor_0p0127 - 0.0127 ppm</li> <li>• FreqMonFactor_0p0257 - 0.0257 ppm</li> <li>• FreqMonFactor_0p0514 - 0.0514 ppm</li> <li>• FreqMonFactor_0p1030 - 0.103 ppm</li> <li>• FreqMonFactor_0p2060 - 0.206 ppm</li> <li>• FreqMonFactor_0p4120 - 0.412 ppm</li> <li>• FreqMonFactor_0p8230 - 0.823 ppm</li> <li>• FreqMonFactor_1p6460 - 1.646 ppm</li> <li>• FreqMonFactor_3p2920 - 3.29 ppm</li> <li>• FreqMonFactor_3p8100 - 3.81 ppm</li> <li>• FreqMonFactor_4p6000 - 4.6 ppm</li> </ul> |

Definition at line 2114 of file DpllCnfg.c.

#### 4.2.4.60 void IDTDpll\_SetHardAlarmThreshold ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *accepting*, T\_osUInt8 *rejecting* )

Set the frequency hard alarm accepting thresholds of reference clock monitoring.

##### Parameters

|                  |                                                                                                                                          |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>      | Device driver handler                                                                                                                    |
| <i>accepting</i> | Accepting threshold (ppm) is a multiple of frequency monitoring factor value. Accepting Threshold = (AcceptingValue + 1)*(FreqMonFactor) |
| <i>rejecting</i> | Rejecting threshold (ppm) is a multiple of frequency monitoring factor value. Rejecting Threshold = (RejectingValue + 1)*(FreqMonFactor) |

Definition at line 2166 of file DpllCnfg.c.

#### 4.2.4.61 void IDTDpll\_SetHitlessSwitchingFeature ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nEnable*, T\_osUInt8 *nFrozen* )

Set the configuration of hitless switching feature.

##### Parameters

|                |                                                                                                                                                                                 |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>    | Device driver handler                                                                                                                                                           |
| <i>nEnable</i> | <ul style="list-style-type: none"> <li>• 0= Disable the hitless switching feature;</li> <li>• 1= Enable the hitless switching feature;</li> </ul>                               |
| <i>nFrozen</i> | <ul style="list-style-type: none"> <li>• 0= hitless switching feature is not frozen;</li> <li>• 1= HS feature is frozen, ignore the further hitless switching events</li> </ul> |

Definition at line 2521 of file DpllCnfg.c.

#### 4.2.4.62 void IDTDpll\_SetHoldoverFreq ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, long *pptOffset* )

Set DCO frequency to device. Note this function should only be called when DCO mode is enabled.

##### Parameters

|                  |                            |
|------------------|----------------------------|
| <i>hdlr</i>      | Device driver handler      |
| <i>nDpll</i>     | DPLL instance              |
| <i>pptOffset</i> | Parts per trillion offset. |

Definition at line 1295 of file DpllCnfg.c.

#### 4.2.4.63 void IDTDpll\_SetHoldoverMode ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_idtDpllHoldover *nMode*, T\_osUInt8 *nReadMode* )

Set calculation mode of holdover frequency and read back mode.

## Parameters

|                  |                                                                                                                                                                                                                                                                   |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>      | Device driver handler                                                                                                                                                                                                                                             |
| <i>nDpll</i>     | DPLL instance                                                                                                                                                                                                                                                     |
| <i>nMode</i>     | <ul style="list-style-type: none"> <li>• Holdover_Instantaneous = automatic instantaneous mode</li> <li>• Holdover_Slow_Average = automatic slow average</li> <li>• Holdover_Fast_Average = Automatic fast average</li> <li>• Holdover_Manual = Manual</li> </ul> |
| <i>nReadMode</i> | <ul style="list-style-type: none"> <li>• 0= Don't read value from device</li> <li>• 1= Read value calculated by device back to the holdover frequency register(0x5D,0x5E and 0x5F)</li> </ul>                                                                     |

Definition at line 1136 of file DpllCnfg.c.

4.2.4.64 void IDTDpll.SetIcpCtrl ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_osUInt8 *nIcp* )

Set charge pump control.

## Parameters

|              |                                                        |
|--------------|--------------------------------------------------------|
| <i>hdlr</i>  | Device driver handler                                  |
| <i>nDpll</i> | DPLL instance                                          |
| <i>nIcp</i>  | Charge pump current selection ( 0 - 0x1F ); 10 = 40uA; |

Definition at line 2269 of file DpllCnfg.c.

4.2.4.65 void IDTDpll.SetPhaseCoarseDetector ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_osUInt8 *nEnable*, T\_osUInt8 *nWide*, T\_osUInt8 *nMultiPhase*, T\_osUInt8 *nMultiPh8kEn* )

Set the coarse phase lose detector enable.

## Parameters

|                    |                                                                                                                                         |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>        | Device driver handler                                                                                                                   |
| <i>nDpll</i>       | DPLL instance                                                                                                                           |
| <i>nEnable</i>     | <ul style="list-style-type: none"> <li>• 0= Disable the coarse phase detector</li> <li>• 1= Enable the coarse phase detector</li> </ul> |
| <i>nWide</i>       | <ul style="list-style-type: none"> <li>• 0= Wide range disable</li> <li>• 1= Wide range enable</li> </ul>                               |
| <i>nMultiPhase</i> | <ul style="list-style-type: none"> <li>• 0= Limit to +/- 1UI</li> <li>• 1= Limit to PH_LOS_COARSE_LIMT register</li> </ul>              |

|                     |                                                                                                                                                                |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>nMultiPh8kEn</i> | <ul style="list-style-type: none"> <li>• 0= Limit to +/- 1UI when reference clock 8/4/2KHz clock</li> <li>• 1= Limit to PH_LOS_COARSE_LIMT register</li> </ul> |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|

Definition at line 813 of file DpllCnfg.c.

**4.2.4.66** void IDTDpll\_SetPhaseCoarseLimit ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_idtCoarsePhaseLimit *nLimit* )

Set the limitation of the coarse phase lose detector.

#### Parameters

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>   | Device driver handler                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <i>nDpll</i>  | DPLL instance                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <i>nLimit</i> | Limitation of coarse phase loss detector. <ul style="list-style-type: none"> <li>• coarsePhaseLimit_1 - +/- 1 UI</li> <li>• coarsePhaseLimit_3 - +/- 3 UI</li> <li>• coarsePhaseLimit_7 - +/- 7 UI</li> <li>• coarsePhaseLimit_15 - +/- 15 UI</li> <li>• coarsePhaseLimit_31 - +/- 31 UI</li> <li>• coarsePhaseLimit_63 - +/- 63 UI</li> <li>• coarsePhaseLimit_127 - +/- 127 UI</li> <li>• coarsePhaseLimit_255 - +/- 255 UI</li> <li>• coarsePhaseLimit_511 - +/- 511 UI</li> <li>• coarsePhaseLimit_1023 - +/- 1023 UI T0 only</li> </ul> |

Definition at line 934 of file DpllCnfg.c.

**4.2.4.67** void IDTDpll\_SetPhaseFineDetectorEnable ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_osUInt8 *nEnable* )

Set the fine phase lose detector enable.

#### Parameters

|                |                                                                                                                                     |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>    | Device driver handler                                                                                                               |
| <i>nDpll</i>   | DPLL instance                                                                                                                       |
| <i>nEnable</i> | <ul style="list-style-type: none"> <li>• 0= Disable the fine phase detector</li> <li>• 1= Enable the fine phase detector</li> </ul> |

Definition at line 991 of file DpllCnfg.c.

**4.2.4.68** void IDTDpll\_SetPhaseFineLimit ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_idtFinePhaseLimit *nLimit* )

Set the limitation of the fine phase lose detector.

## Parameters

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>   | Device driver handler                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| <i>nDpll</i>  | DPLL instance                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <i>nLimit</i> | Limitation of fine phase loss detector <ul style="list-style-type: none"> <li>• finePhaseLimit_45_90degree = +/- 45-90 degrees</li> <li>• finePhaseLimit_90_180degree = +/- 90-180 degrees</li> <li>• finePhaseLimit_180_360degree = +/- 180-360 degrees</li> <li>• finePhaseLimit_20_25nanosec = +/- 20-25 nanoseconds</li> <li>• finePhaseLimit_60_65nanosec = +/- 60-65 nanoseconds</li> <li>• finePhaseLimit_120_125nanosec = +/- 120-125 nanoseconds</li> <li>• finePhaseLimit_950_955nanosec = +/- 950-955 nanoseconds</li> </ul> |

Definition at line 1045 of file DpllCnfg.c.

4.2.4.69 void IDTDpll\_SetPhaseOffset ( T\_idtDrvHdlr *hdlr*, T\_osUInt16 *nOffset* )

Set the phase offset value.

## Parameters

|                |                                         |
|----------------|-----------------------------------------|
| <i>hdlr</i>    | Device driver handler                   |
| <i>nOffset</i> | Phase offset= <i>nOffset</i> * 0x61 ns; |

Definition at line 2566 of file DpllCnfg.c.

4.2.4.70 void IDTDpll\_SetPhaseSlope ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_idtPhaseSlope *nPhaseSlope* )

Set DPLL phase slope for DPLLs.

## Parameters

|                    |                                                                                                                                                                                                                                                                     |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>        | Device driver handler                                                                                                                                                                                                                                               |
| <i>nDpll</i>       | DPLL instance                                                                                                                                                                                                                                                       |
| <i>nPhaseSlope</i> | <ul style="list-style-type: none"> <li>• phaseSlope_gr1244st3 = GR-1244 ST3 (61us/s)</li> <li>• phaseSlope_gr1244st3e = GR-1244 ST2,3E,ST3 (855ns/s)</li> <li>• phaseSlope_gr813 = GR-813,G.8262 (7.5us/s)</li> <li>• phaseSlope_noLimit = No limitation</li> </ul> |

Definition at line 2351 of file DpllCnfg.c.

4.2.4.71 void IDTDpll\_SetPhaseTransient ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nEnable* )

Set the configuration when phase transient occurs.

## Parameters

|                |                                                                                                                                             |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>    | Device driver handler                                                                                                                       |
| <i>nEnable</i> | <ul style="list-style-type: none"> <li>• 0= Disable the phase transient monitor</li> <li>• 1= Enable the phase transient monitor</li> </ul> |

Definition at line 2486 of file DpllCnfg.c.

4.2.4.72 void IDTDpll\_SetPpsFastLock ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nEnableFreq*, T\_osUInt8 *nEnablePhase* )

Enable/Disable 1 PPS fast lock.

## Parameters

|                     |                                            |
|---------------------|--------------------------------------------|
| <i>hdlr</i>         | Device driver handler                      |
| <i>nEnableFreq</i>  | Enable (1) / Disable (0) frequency locking |
| <i>nEnablePhase</i> | Enable (1) / Disable (0) phase locking     |

Definition at line 2433 of file DpllCnfg.c.

4.2.4.73 void IDTDpll\_SetPpsPhase ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_idtDpllPpsPhasePostion *nPhase*, T\_idtDpllPpsPulseWidth *nPulse* )

Set PPS phase and pulse for DPLL.

## Parameters

|               |                       |
|---------------|-----------------------|
| <i>hdlr</i>   | Device driver handler |
| <i>nDpll</i>  | DPLL instance         |
| <i>nPhase</i> | Phase position        |

## See Also

[T\\_idtDpllPpsPhasePostion](#)

## Parameters

|               |             |
|---------------|-------------|
| <i>nPulse</i> | Pulse width |
|---------------|-------------|

## See Also

[T\\_idtDpllPpsPulseWidth](#)

Definition at line 2605 of file DpllCnfg.c.

4.2.4.74 void IDTDpll\_SetSoftAlarmThreshold ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *accepting*, T\_osUInt8 *rejecting* )

Set the frequency soft alarm accepting thresholds of reference clock monitoring.

## Parameters

|                  |                                                                                                                                          |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>      | Device driver handler                                                                                                                    |
| <i>accepting</i> | Accepting threshold (ppm) is a multiple of frequency monitoring factor value. Accepting Threshold = (AcceptingValue + 1)*(FreqMonFactor) |

|                  |                                                                                                                                          |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>rejecting</i> | Rejecting threshold (ppm) is a multiple of frequency monitoring factor value. Rejecting Threshold = (RejectingValue + 1)*(FreqMonFactor) |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------|

Definition at line 2220 of file DpllCnfg.c.

4.2.4.75 void IDTDpll.SetSonetGigEthOutputPath ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *instance*, T\_idtDpllInstance *inputDpll*, T\_idtSonetGigEthOutput *outputFreq* )

Set Sonet/GigE path configuration.

#### Parameters

|                   |                               |
|-------------------|-------------------------------|
| <i>hdlr</i>       | Device driver handler         |
| <i>instance</i>   | Sonet/GigE path instance 0..1 |
| <i>inputDpll</i>  | Dpll instance                 |
| <i>outputFreq</i> | Output frequency              |

Definition at line 1535 of file DpllCnfg.c.

4.2.4.76 void IDTDpll.SetTempHoldoverMode ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll*, T\_idtTemp\_holdover *nMode* )

Set calculation mode of temporary holdover frequency.

#### Parameters

|              |                                                                                                                                                                                                                                                                                                             |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>  | Device driver handler                                                                                                                                                                                                                                                                                       |
| <i>nDpll</i> | DPLL instance                                                                                                                                                                                                                                                                                               |
| <i>nMode</i> | <ul style="list-style-type: none"> <li>Temp_Same_As_Full_Holdover = Same as the DPLL holdover mode</li> <li>Temp_Holdover_Instantaneous = Automatic instantaneous mode</li> <li>Temp_Holdover_Fast_Average = Automatic fast average</li> <li>Temp_Holdover_Slow_Average = Automatic slow average</li> </ul> |

Definition at line 1209 of file DpllCnfg.c.

## 4.3 General Function Module

Functions in this group are related to device wide provisioning and status.

### Data Structures

- struct [T\\_idtI2CtransData](#)  
*I2C address translation information used internally by the driver.*

### Macros

- #define [Activate\\_Edge\\_Rising](#) 0
- #define [Activate\\_Edge\\_Falling](#) 1
- #define [INT\\_Active\\_Low](#) 0
- #define [INT\\_Active\\_High](#) 1
- #define [NOMINAL\\_PPT\\_TO\\_2COMPL](#)(ppt) (T\_osUInt32)(((ppt)\*10)/884)  
*Conversion from parts per trillion to register unit value for nominal (oscillator) offset.*
- #define [NOMINAL\\_2COMPL\\_TO\\_PPT](#)(twoCompl) (long)((((twoCompl)\*884)/10)  
*Conversion from parts per trillion to register unit value for nominal (oscillator) offset.*
- #define [NOMINAL\\_MAX\\_PPT](#) 94893  
*Maximum oscillator offset value.*
- #define [NOMINAL\\_MIN\\_PPT](#) -94893  
*Minimum oscillator offset value.*

### Enumerations

- enum [networkType\\_t](#) { [Network\\_SDH](#) = 0, [Network\\_SONET](#) = 1, [Network\\_MAX](#) }
- Enumerated type listing types of selectable network type.
- enum [phaseTimeoutMultiplier\\_t](#) {  
[phaseTimeoutMultiplier\\_2](#) = 0, [phaseTimeoutMultiplier\\_4](#) = 1, [phaseTimeoutMultiplier\\_8](#) = 2, [phaseTimeoutMultiplier\\_16](#) = 3,  
[phaseTimeoutMultiplier\\_MAX](#) }
- Enumerated type listing phase alarm timeout multipliers.

### Functions

- T\_osUInt16 [IDTGeneral\\_GetDeviceID](#) (T\_idtDrvHdlr hdlr)  
*Get ID number of the device.*
- void [IDTGeneral\\_SetOscFreqOffset](#) (T\_idtDrvHdlr hdlr, long pptOffset)  
*Set compensation to the oscillator offset.*
- long [IDTGeneral\\_GetOscFreqOffset](#) (T\_idtDrvHdlr hdlr)  
*Get compensated oscillator offset.*
- void [IDTGeneral\\_SetNetworkType](#) (T\_idtDrvHdlr hdlr, [networkType\\_t](#) nType)  
*Set the network type, SDH or SONET.*
- [networkType\\_t](#) [IDTGeneral\\_GetNetworkType](#) (T\_idtDrvHdlr hdlr)  
*Get the network type, SDH or SONET.*
- void [IDTGeneral\\_SetRevertiveMode](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
*Set the device running in Revertive mode or not.*
- T\_osUInt8 [IDTGeneral\\_GetRevertiveMode](#) (T\_idtDrvHdlr hdlr)  
*Get the device running in Revertive mode or not.*

- void [IDTGeneral\\_SetTimeoutValue](#) (T\_idtDrvHdlr hdlr, [phaseTimeoutMultiplier\\_t](#) nMultiFactor, T\_osUInt8 nTimeout)  
*Set the value of time out of phase alarm Timeout(second)= nMultiFactor \* nTimeout.*
- void [IDTGeneral\\_GetTimeoutValue](#) (T\_idtDrvHdlr hdlr, [phaseTimeoutMultiplier\\_t](#) \*nMultiFactor\_p, T\_osUInt8 \*nTimeout\_p)  
*Get the value of time out of phase alarm Timeout(second)= nMultiFactor \* nTimeout.*
- void [IDTGeneral\\_SetOscActiveEdge](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEdge)  
*Set the activate edge of main oscillator.*
- T\_osUInt8 [IDTGeneral\\_GetOscActiveEdge](#) (T\_idtDrvHdlr hdlr)  
*Get the activate edge of main oscillator.*
- void [IDTGeneral\\_SetProtectMode](#) (T\_idtDrvHdlr hdlr)  
*Enable protected register access mode.*
- void [IDTGeneral\\_SetSingleUnprotectMode](#) (T\_idtDrvHdlr hdlr)  
*Set single access register access mode.*
- void [IDTGeneral\\_SetFullyUnprotectMode](#) (T\_idtDrvHdlr hdlr)  
*Set fully un-protected register access mode.*
- void [IDTGeneral\\_SetFlagOnTDO](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
*Set TDO pin to indicate the fail of selected clock.*
- T\_osUInt8 [IDTGeneral\\_GetFlagOnTDO](#) (T\_idtDrvHdlr hdlr)  
*Get TDO pin to indicate the fail of selected clock.*
- void [IDTGeneral\\_SetUltraFastSwitch](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
*Set Ultra-fast-switch enable or not.*
- T\_osUInt8 [IDTGeneral\\_GetUltraFastSwitch](#) (T\_idtDrvHdlr hdlr)  
*Get Ultra-fast-switch enable or not.*
- void [IDTGeneral\\_SetHardAlarm](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
*Set Hard-alarm enable when the input exceeds the threshold.*
- T\_osUInt8 [IDTGeneral\\_GetHardAlarm](#) (T\_idtDrvHdlr hdlr)  
*Get Hard-alarm enable when the input exceeds the threshold.*
- T\_osUInt8 [IDTGeneral\\_i2cTranslate](#) (T\_osUInt8 i2cAddr, void \*transData)  
*I2C address translation function used internally by the driver.*
- void [IDTGeneral\\_InitDeviceCommon](#) (T\_idtDrvHdlr hdlr)
- T\_idtDrvHdlr [IDTGeneral\\_InitDriverCommon](#) (const char \*name, [T\\_idtDrvAccess](#) drvAccess, [T\\_idtDeviceType](#) deviceType, const [T\\_idtDrvDeviceConfig](#) \*deviceConfig, int useHighLevelApi)
- void [IDTGeneral\\_DeInitDriver](#) (T\_idtDrvHdlr hdlr)  
*De-initialize driver handler.*

### 4.3.1 Detailed Description

Functions in this group are related to device wide provisioning and status.

### 4.3.2 Macro Definition Documentation

#### 4.3.2.1 #define Activate\_Edge\_Falling 1

Definition at line 85 of file General.h.

#### 4.3.2.2 #define Activate\_Edge\_Rising 0

Definition at line 84 of file General.h.

4.3.2.3 `#define INT_Active_High 1`

Definition at line 89 of file General.h.

4.3.2.4 `#define INT_Active_Low 0`

Definition at line 88 of file General.h.

4.3.2.5 `#define NOMINAL_2COMPL_TO_PPT( twoCompl ) (long)((((twoCompl)*884)/10)`

Conversion from parts per trillion to register unit value for nominal (oscillator) offset.

Definition at line 97 of file General.h.

4.3.2.6 `#define NOMINAL_MAX_PPT 94893`

Maximum oscillator offset value.

Definition at line 100 of file General.h.

4.3.2.7 `#define NOMINAL_MIN_PPT -94893`

Minimum oscillator offset value.

Definition at line 103 of file General.h.

4.3.2.8 `#define NOMINAL_PPT_TO_2COMPL( ppt ) (T_osUInt32)((((ppt)*10)/884)`

Conversion from parts per trillion to register unit value for nominal (oscillator) offset.

Definition at line 93 of file General.h.

## 4.3.3 Enumeration Type Documentation

4.3.3.1 `enum networkType_t`

Enumerated type listing types of selectable network type.

Enumerator

***Network\_SDH*** SDH network

***Network\_SONET*** SONET network

***Network\_MAX***

Definition at line 58 of file General.h.

4.3.3.2 `enum phaseTimeoutMultiplier_t`

Enumerated type listing phase alarm timeout multipliers.

Enumerator

***phaseTimeoutMultiplier\_2*** 2x Multiplier

***phaseTimeoutMultiplier\_4*** 4x Multiplier

***phaseTimeoutMultiplier\_8*** 8x Multiplier  
***phaseTimeoutMultiplier\_16*** 18x Multiplier  
***phaseTimeoutMultiplier\_MAX***

Definition at line 66 of file General.h.

#### 4.3.4 Function Documentation

##### 4.3.4.1 void IDTGeneral\_DeInitDriver ( T\_idtDrvHdlr *hdlr* )

De-initialize driver handler.

De-allocates handler. Handler is invalid after calling this function

###### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 568 of file General.c.

##### 4.3.4.2 T\_osUInt16 IDTGeneral\_GetDeviceID ( T\_idtDrvHdlr *hdlr* )

Get ID number of the device.

###### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

###### Returns

16-bit ID number

Definition at line 58 of file General.c.

##### 4.3.4.3 T\_osUInt8 IDTGeneral\_GetFlagOnTDO ( T\_idtDrvHdlr *hdlr* )

Get TDO pin to indicate the fail of selected clock.

###### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

###### Returns

- 0= Not enable to use TDO pin
- 1= enable to use the pin;

Definition at line 354 of file General.c.

##### 4.3.4.4 T\_osUInt8 IDTGeneral\_GetHardAlarm ( T\_idtDrvHdlr *hdlr* )

Get Hard-alarm enable when the input exceeds the threshold.

## Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

## Returns

- 0= Disable Hard-alarm feature
- 1= Enable the feature

Definition at line 414 of file General.c.

**4.3.4.5 networkType\_t IDTGeneral\_GetNetworkType ( T\_idtDrvHdlr *hdlr* )**

Get the network type, SDH or SONET.

## Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

## Returns

nType

- Network\_SDH = SDH network
- Network\_SONET = SONET network

Definition at line 172 of file General.c.

**4.3.4.6 T\_osUInt8 IDTGeneral\_GetOscActiveEdge ( T\_idtDrvHdlr *hdlr* )**

Get the activate edge of main oscillator.

## Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

## Returns

- 0= Rising edge is activate edge
- 1= Falling edge is activate edge

Definition at line 286 of file General.c.

**4.3.4.7 long IDTGeneral\_GetOscFreqOffset ( T\_idtDrvHdlr *hdlr* )**

Get compensated oscillator offset.

## Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

## Returns

Crystal oscillator frequency offset (in parts per trillion)

Definition at line 117 of file General.c.

#### 4.3.4.8 T\_osUInt8 IDTGeneral\_GetRevertiveMode ( T\_idtDrvHdlr *hdlr* )

Get the device running in Revertive mode or not.

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

##### Returns

nEnable

- 0= Works in Non-Revertive mode
- 1= Works in Revertive mode

Definition at line 203 of file General.c.

#### 4.3.4.9 void IDTGeneral\_GetTimeoutValue ( T\_idtDrvHdlr *hdlr*, phaseTimeoutMultiplier\_t \* *nMultiFactor\_p*, T\_osUInt8 \* *nTimeout\_p* )

Get the value of time out of phase alarm Timeout(second)= nMultiFactor \* nTimeout.

##### Parameters

|                       |                                                                                                                                                                                                                                                |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>           | Device driver handler                                                                                                                                                                                                                          |
| <i>nMultiFactor_p</i> | <ul style="list-style-type: none"> <li>• phaseTimeoutMultiplier_2 - Multiple 2</li> <li>• phaseTimeoutMultiplier_4 - Multiple 4</li> <li>• phaseTimeoutMultiplier_8 - Multiple 8</li> <li>• phaseTimeoutMultiplier_16 - Multiple 16</li> </ul> |
| <i>nTimeout_p</i>     | a 6-bit unsigned integer, unit is second;                                                                                                                                                                                                      |

Definition at line 252 of file General.c.

#### 4.3.4.10 T\_osUInt8 IDTGeneral\_GetUltraFastSwitch ( T\_idtDrvHdlr *hdlr* )

Get Ultra-fast-switch enable or not.

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

##### Returns

- 0= The feature is disabled
- 1= The feature is enabled

Definition at line 384 of file General.c.

#### 4.3.4.11 T\_osUInt8 IDTGeneral\_i2cTranslate ( T\_osUInt8 *i2cAddr*, void \* *transData* )

I2C address translation function used internally by the driver.

## Parameters

|                  |                         |
|------------------|-------------------------|
| <i>i2cAddr</i>   | I2C slave address       |
| <i>transData</i> | Translation information |

## Returns

Translated address

Definition at line 428 of file General.c.

4.3.4.12 void IDTGeneral\_InitDeviceCommon ( T\_idtDrvHdlr *hdlr* )

Definition at line 455 of file General.c.

4.3.4.13 T\_idtDrvHdlr IDTGeneral\_InitDriverCommon ( const char \* *name*, T\_idtDrvAccess *drvAccess*, T\_idtDeviceType *deviceType*, const T\_idtDrvDeviceConfig \* *deviceConfig*, int *useHighLevelApi* )

Definition at line 508 of file General.c.

4.3.4.14 void IDTGeneral\_SetFlagOnTDO ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nEnable* )

Set TDO pin to indicate the fail of selected clock.

## Parameters

|                |                                                                                                                       |
|----------------|-----------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>    | Device driver handler                                                                                                 |
| <i>nEnable</i> | <ul style="list-style-type: none"> <li>• 0= Not enable to use TDO pin</li> <li>• 1= enable to use the pin;</li> </ul> |

Definition at line 338 of file General.c.

4.3.4.15 void IDTGeneral\_SetFullyUnprotectMode ( T\_idtDrvHdlr *hdlr* )

Set fully un-protected register access mode.

## Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 324 of file General.c.

4.3.4.16 void IDTGeneral\_SetHardAlarm ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nEnable* )

Set Hard-alarm enable when the input exceeds the threshold.

## Parameters

|                |                                                                                                                    |
|----------------|--------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>    | Device driver handler                                                                                              |
| <i>nEnable</i> | <ul style="list-style-type: none"> <li>• 0= Disable Hard-alarm feature</li> <li>• 1= Enable the feature</li> </ul> |

Definition at line 398 of file General.c.

**4.3.4.17** void IDTGeneral\_SetNetworkType ( T\_idtDrvHdlr *hdlr*, networkType\_t *nType* )

Set the network type, SDH or SONET.

**Parameters**

|              |                                                                                                                        |
|--------------|------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>  | Device driver handler                                                                                                  |
| <i>nType</i> | <ul style="list-style-type: none"> <li>• Network_SDH = SDH network</li> <li>• Network_SONET = SONET network</li> </ul> |

Definition at line 152 of file General.c.

**4.3.4.18** void IDTGeneral\_SetOscActiveEdge ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nEdge* )

Set the activate edge of main oscillator.

**Parameters**

|              |                                                                                                                                 |
|--------------|---------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>  | Device driver handler                                                                                                           |
| <i>nEdge</i> | <ul style="list-style-type: none"> <li>• 0= Rising edge is activate edge</li> <li>• 1= Falling edge is activate edge</li> </ul> |

Definition at line 270 of file General.c.

**4.3.4.19** void IDTGeneral\_SetOscFreqOffset ( T\_idtDrvHdlr *hdlr*, long *pptOffset* )

Set compensation to the oscillator offset.

**Parameters**

|                  |                                                             |
|------------------|-------------------------------------------------------------|
| <i>hdlr</i>      | Device driver handler                                       |
| <i>pptOffset</i> | Crystal oscillator frequency offset (in parts per trillion) |

Definition at line 78 of file General.c.

**4.3.4.20** void IDTGeneral\_SetProtectMode ( T\_idtDrvHdlr *hdlr* )

Enable protected register access mode.

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 298 of file General.c.

**4.3.4.21** void IDTGeneral\_SetRevertiveMode ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nEnable* )

Set the device running in Revertive mode or not.

## Parameters

|                |                                                                                                                          |
|----------------|--------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>    | Device driver handler                                                                                                    |
| <i>nEnable</i> | <ul style="list-style-type: none"> <li>• 0= Works in Non-Revertive mode</li> <li>• 1= Works in Revertive mode</li> </ul> |

Definition at line 186 of file General.c.

#### 4.3.4.22 void IDTGeneral\_SetSingleUnprotectMode ( T\_idtDrvHdlr *hdlr* )

Set single access register access mode.

## Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 311 of file General.c.

#### 4.3.4.23 void IDTGeneral\_SetTimeoutValue ( T\_idtDrvHdlr *hdlr*, phaseTimeoutMultiplier\_t *nMultiFactor*, T\_osUInt8 *nTimeout* )

Set the value of time out of phase alarm Timeout(second)= nMultiFactor \* nTimeout.

## Parameters

|                     |                                                                                                                                                                                                                                                |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>         | Device driver handler                                                                                                                                                                                                                          |
| <i>nMultiFactor</i> | <ul style="list-style-type: none"> <li>• phaseTimeoutMultiplier_2 - Multiple 2</li> <li>• phaseTimeoutMultiplier_4 - Multiple 4</li> <li>• phaseTimeoutMultiplier_8 - Multiple 8</li> <li>• phaseTimeoutMultiplier_16 - Multiple 16</li> </ul> |
| <i>nTimeout</i>     | a 6-bit unsigned integer, unit is second;                                                                                                                                                                                                      |

Definition at line 221 of file General.c.

#### 4.3.4.24 void IDTGeneral\_SetUltraFastSwitch ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nEnable* )

Set Ultra-fast-switch enable or not.

## Parameters

|                |                                                                                                                     |
|----------------|---------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>    | Device driver handler                                                                                               |
| <i>nEnable</i> | <ul style="list-style-type: none"> <li>• 0= The feature is disabled</li> <li>• 1= The feature is enabled</li> </ul> |

Definition at line 368 of file General.c.

## 4.4 Hardware Abstraction Layer Module

Functions in this group are related to hardware abstraction layer.

### Macros

- `#define IDT_HAL_USE_MACRO 0`
- `#define IDTHAL_ModifyBitfield(regVal, mask, lsb, data)`  
*Define function to set bitfield value of read data from device.*
- `#define IDTHAL_GetBitfield(regVal, mask, lsb) (((regVal) & (mask)) >> (lsb))`  
*Define function to get bitfield value of read data from device.*

### Functions

- void `IDTHAL_WriteBitfield` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 mask, T\_osUInt8 lsb, T\_osUInt8 data)  
*Define function to write a bitfield to a device.*
- void `IDTHAL_WriteBitfield_Ext` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 mask, T\_osUInt8 lsb, T\_osUInt8 data)  
*Define function to write a bitfield to a device.*
- T\_osUInt8 `IDTHAL_ReadBitfield` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 mask, T\_osUInt8 lsb)  
*Define function to read a bitfield from device.*
- T\_osUInt8 `IDTHAL_ReadBitfield_Ext` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 mask, T\_osUInt8 lsb)  
*Define function to read a bitfield from device.*
- T\_osUInt8 `IDTHal_Read` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset)  
*Function to read from device.*
- void `IDTHal_Write` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 data)  
*Define function to write to device.*
- T\_osUInt8 `IDTHal_Read_Ext` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset)  
*Function to read from device apll address space.*
- void `IDTHal_Write_Ext` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 data)  
*Function to write to device apll address space.*

### 4.4.1 Detailed Description

Functions in this group are related to hardware abstraction layer.

### 4.4.2 Macro Definition Documentation

#### 4.4.2.1 `#define IDT_HAL_USE_MACRO 0`

Definition at line 53 of file Hal.h.

#### 4.4.2.2 `#define IDTHAL_GetBitfield( regVal, mask, lsb ) (((regVal) & (mask)) >> (lsb))`

Define function to get bitfield value of read data from device.

This macro function can be used after a register is read from a device. This macro **doesn't** access the device.

#### Parameters

|               |                       |
|---------------|-----------------------|
| <i>regVal</i> | register data value.  |
| <i>mask</i>   | Bitfield mask         |
| <i>lsb</i>    | Least significant bit |

Definition at line 80 of file Hal.h.

#### 4.4.2.3 #define IDTHAL\_ModifyBitfield( *regVal*, *mask*, *lsb*, *data* )

**Value:**

```
regVal = ((regVal & ~(mask)) | \
          (((data) << (lsb)) & (mask)))
```

Define function to set bitfield value of read data from device.

This macro function can be used after a register is read from a device. This macro **doesn't** access the device.

**Parameters**

|               |                       |
|---------------|-----------------------|
| <i>regVal</i> | register data value.  |
| <i>mask</i>   | Bitfield mask         |
| <i>lsb</i>    | Least significant bit |
| <i>data</i>   | Bit field data to set |

Definition at line 66 of file Hal.h.

### 4.4.3 Function Documentation

#### 4.4.3.1 T\_osUInt8 IDTHal\_Read ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *offset* )

Function to read from device.

**Parameters**

|               |                         |
|---------------|-------------------------|
| <i>hdlr</i>   | Device driver handler   |
| <i>offset</i> | Register address offset |

Function to read from device.

**Parameters**

|               |                         |
|---------------|-------------------------|
| <i>hdlr</i>   | Device driver handler   |
| <i>offset</i> | Register address offset |

Definition at line 152 of file Hal.c.

#### 4.4.3.2 T\_osUInt8 IDTHal\_Read.Ext ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *offset* )

Function to read from device apll address space.

**Parameters**

|               |                         |
|---------------|-------------------------|
| <i>hdlr</i>   | Device driver handler   |
| <i>offset</i> | Register address offset |

Definition at line 209 of file Hal.c.

#### 4.4.3.3 T\_osUInt8 IDTHAL\_ReadBitfield ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *offset*, T\_osUInt8 *mask*, T\_osUInt8 *lsb* )

Define function to read a bitfield from device.

**Parameters**

|               |                         |
|---------------|-------------------------|
| <i>hdlr</i>   | Device driver handler   |
| <i>offset</i> | Register address offset |
| <i>mask</i>   | Bitfield mask           |
| <i>lsb</i>    | Least significant bit   |

**Returns**

Bit field value

Definition at line 108 of file Hal.c.

#### 4.4.3.4 T\_osUInt8 IDTHAL\_ReadBitfield\_Ext ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *offset*, T\_osUInt8 *mask*, T\_osUInt8 *lsb* )

Define function to read a bitfield from device.

**Parameters**

|               |                         |
|---------------|-------------------------|
| <i>hdlr</i>   | Device driver handler   |
| <i>offset</i> | Register address offset |
| <i>mask</i>   | Bitfield mask           |
| <i>lsb</i>    | Least significant bit   |

**Returns**

Bit field value

Definition at line 131 of file Hal.c.

#### 4.4.3.5 void IDTHal\_Write ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *offset*, T\_osUInt8 *data* )

Define function to write to device.

**Parameters**

|               |                         |
|---------------|-------------------------|
| <i>hdlr</i>   | Device driver handler   |
| <i>offset</i> | Register address offset |
| <i>data</i>   | Data to write           |

Define function to write to device.

**Parameters**

|               |                         |
|---------------|-------------------------|
| <i>hdlr</i>   | Device driver handler   |
| <i>offset</i> | Register address offset |
| <i>data</i>   | Data to write           |

Definition at line 181 of file Hal.c.

#### 4.4.3.6 void IDTHal\_Write\_Ext ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *offset*, T\_osUInt8 *data* )

Function to write to device apII address space.

## Parameters

|               |                         |
|---------------|-------------------------|
| <i>hdlr</i>   | Device driver handler   |
| <i>offset</i> | Register address offset |
| <i>data</i>   | Data to write           |

Definition at line 238 of file Hal.c.

4.4.3.7 void IDTHAL\_WriteBitfield ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *offset*, T\_osUInt8 *mask*, T\_osUInt8 *lsb*, T\_osUInt8 *data* )

Define function to write a bitfield to a device.

## Parameters

|               |                         |
|---------------|-------------------------|
| <i>hdlr</i>   | Device driver handler   |
| <i>offset</i> | Register address offset |
| <i>mask</i>   | Bitfield mask           |
| <i>lsb</i>    | Least significant bit   |
| <i>data</i>   | Bit field data to write |

Definition at line 60 of file Hal.c.

4.4.3.8 void IDTHAL\_WriteBitfield\_Ext ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *offset*, T\_osUInt8 *mask*, T\_osUInt8 *lsb*, T\_osUInt8 *data* )

Define function to write a bitfield to a device.

## Parameters

|               |                         |
|---------------|-------------------------|
| <i>hdlr</i>   | Device driver handler   |
| <i>offset</i> | Register address offset |
| <i>mask</i>   | Bitfield mask           |
| <i>lsb</i>    | Least significant bit   |
| <i>data</i>   | Bit field data to write |

Definition at line 84 of file Hal.c.

## 4.5 Input Function Module

Functions in this group are related to input port provisioning.

### Macros

- `#define Input_Phase_Lock_Alarm 1`
- `#define Input_No_Activity_Alarm 2`
- `#define Input_Freq_Hard_Alarm 4`
- `#define IDT_TRANSLATE_EXT_TO_INT_INPORT(hdlr, ext, internal)`  
*Utility macro to check if input port is supported and to translated from external port to internal port.*
- `#define IDT_EXT_INPORT_POPULATED(hdlr, ext) (hdlr->deviceConfig_m->inputExt2IntXlate_ma[(ext)])`  
*Utility macro to check if input port is populated.*

### Enumerations

- `enum T_idtInputDividerMux { Divider_Bypass = 0, Divider_Lock8k = 1, Divider_DivN = 2, Divider_MAX }`  
*Enumerated type listing input divider modes.*
- `enum T_idtHighFrequencyDivisor { HF_Bypass = 0, HF_Divide_4 = 1, HF_Divide_5 = 2, HF_MAX }`  
*Enumerated type listing high frequency (> 155.52MHz) dividers.*
- `enum T_idtInputFrequencyBitfieldValue { Freq_8K = 0, Freq_1544M = 1, Freq_2048M = 1, Freq_648M = 2, Freq_1944M = 3, Freq_2592M = 4, Freq_3888M = 5, Freq_2K = 9, Freq_4K = 10, Freq_1PPS = 11, Freq_625M = 12, Freq_10M = 13, Freq_MAX }`  
*Enumerated type listing input frequencies.*
- `enum T_idtInputFrequencyFamily { InFreqFamily_8K = 0, InFreqFamily_1544M = 1, InFreqFamily_2048M = 2, InFreqFamily_648M = 3, InFreqFamily_1944M = 4, InFreqFamily_2592M = 5, InFreqFamily_3888M = 6, InFreqFamily_2K = 7, InFreqFamily_4K = 8, InFreqFamily_1PPS = 9, InFreqFamily_625M = 10, InFreqFamily_10M = 11, InFreqFamily_MAX }`  
*Enumerated type listing input frequency families.*
- `enum T_idtAmiInputFrequencyBitFieldValue { AmiFreq_64kHzPlus8kHz, AmiFreq_64kHzPlus400Hz, AmiFreq_MAX }`  
*Enumerated type listing AMI Input frequencies.*
- `enum T_idtInputSyncFreq { SYNC_8kHz, SYNC_1PPS, SYNC_4kHz, SYNC_2kHz, SYNC_MAX }`  
*Enumerated type listing possible input sync frequencies.*
- `enum T_externalSyncSampling { externalSyncSampling_onTarget, externalSyncSampling_0dot5Early, externalSyncSampling_1dot0Late, externalSyncSampling_0dot5Late, externalSyncSampling_MAX }`  
*Enumerated type listing sampling of input external sync.*
- `enum T_externalSyncEdge { externalSyncEdge_FallingEdge, externalSyncEdge_RisingEdge, externalSyncEdge_MAX }`  
*Enumerated type listing possible external sync alignment options.*
- `enum T_externalSyncAlarmRange { externalSyncAlarmRange_1, externalSyncAlarmRange_2, externalSyncAlarmRange_3, externalSyncAlarmRange_4, externalSyncAlarmRange_5, externalSyncAlarmRange_6, externalSyncAlarmRange_7, externalSyncAlarmRange_8, externalSyncAlarmRange_MAX }`

*Enumerate type listing ranges for external sync alarm.*

- enum `T_externalSyncBypass` { `externalSyncBypass_ExSync1`, `externalSyncBypass_AutoSel`, `externalSyncBypass_MAX` }

*Enumerated type listing output synchronization with respect to.*

- enum `T_externalSyncEnable` { `externalSyncEnable_Disable`, `externalSyncEnable_Enable`, `externalSyncEnable_Auto`, `externalSyncEnable_MAX` }

*Enumerated type listing enable options for external input sync.*

## Functions

- void `IDTInput_SetPriority` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo, `T_idtDpllInstance` nDpll, `T_osUInt8` nPriority)  
*Set specified port to the priority.*
- `T_osUInt8` `IDTInput_GetPriority` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo, `T_idtDpllInstance` nDpll)  
*Get the current priority of the specified port.*
- void `IDTInput_SetInputFrequency` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo, `T_idtInputFrequencyBitfieldValue` nFrequency)  
*Set the frequency of the input clock at the specified port.*
- `T_idtInputFrequencyBitfieldValue` `IDTInput_GetInputFrequency` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo)  
*Get frequency of the input clock at the specified port.*
- void `IDTInput_SetAmiInputFrequency` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo, `T_idtAmiInputFrequencyBitFieldValue` amiSignal)  
*Provision AMI signal for input port.*
- `T_idtAmiInputFrequencyBitFieldValue` `IDTInput_GetAmiInputFrequency` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo)  
*Function to retrieve AMI input frequency for input port.*
- void `IDTInput_SetActivityMonitor` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo, `T_osUInt8` nBucket)  
*Select one of the four activity monitor configurations for the specified input port.*
- `T_osUInt8` `IDTInput_GetActivityMonitor` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo)  
*Retrieve selected activity monitoring configuration.*
- void `IDTInput_SetBucketConfig` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nBucket, `T_osUInt8` nUpper, `T_osUInt8` nLower, `T_osUInt8` nSize, `T_osUInt8` nDecay)  
*Set the configuration to Leaky Bucket activity monitoring.*
- void `IDTInput_GetBucketConfig` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nBucket, `T_osUInt8` \*nUpper\_p, `T_osUInt8` \*nLower\_p, `T_osUInt8` \*nSize\_p, `T_osUInt8` \*nDecay\_p)  
*Get activity monitoring configuration (Leaky Bucket)*
- void `IDTInput_SetPreDivider` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo, `T_idtInputDividerMux` nDivider)  
*Assign a pre-divider for the specified input.*
- `T_idtInputDividerMux` `IDTInput_GetPreDivider` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo)  
*Get configured input port pre-divider.*
- void `IDTInput_SetDivisionFactor` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo, `T_osUInt16` iFactor)  
*Set the dividen factor to pre-divider assigned to a certain input.*
- `T_osUInt16` `IDTInput_GetDivisionFactor` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo)  
*Get the dividen factor to pre-divider assigned to a certain input.*
- void `IDTInput_SetInputHFdivider` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo, `T_idtHighFrequencyDivisor` nUsage)  
*Set the usage of high frequency (> 155.52MHz) divider.*
- `T_idtHighFrequencyDivisor` `IDTInput_GetInputHFdivider` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo)  
*Get high frequency configuration divider.*
- `T_osUInt8` `IDTInput_GetInputFreqOffset` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo)  
*Get the frequency offset of the specified input clock. The returned value is updated every 16 seconds.*
- `T_osUInt8` `IDTInput_GetInputClockStatus` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nInputNo)

*Get the status of a specified input clock.*

- `T_osUInt8 IDTInput_GetInputClockValidation (T_idtDrvHdlr hdlr, T_osUInt8 nInputNo)`

*Get the clock validation status of an input port.*

- `void IDTInput_EnableAllInputs (T_idtDrvHdlr hdlr)`

*Enable to select anyone of input clocks.*

- `void IDTInput_DisableAllInputs (T_idtDrvHdlr hdlr)`

*Disable to select anyone of input clocks.*

- `void IDTInput_SetInputEnable (T_idtDrvHdlr hdlr, T_osUInt8 nInputNo, T_osUInt8 enable)`

*Enable/Disable selected input.*

- `T_osUInt8 IDTInput_GetInputEnable (T_idtDrvHdlr hdlr, T_osUInt8 nInputNo)`

*Retrieve enable/disable status of selected input.*

- `void IDTInput_SetSyncFrequency (T_idtDrvHdlr hdlr, T_idtInputSyncFreq syncFreq)`

*Function to select input external sync frequency.*

- `T_idtInputSyncFreq IDTInput_GetSyncFrequency (T_idtDrvHdlr hdlr)`

*Function to retrieve select input external sync frequency.*

- `void IDTInput_SetSyncSampling (T_idtDrvHdlr hdlr, T_osUInt8 exSyncN, T_externalSyncSampling sampling)`

*Function to provision sampling configuration for external sync.*

- `T_externalSyncSampling IDTInput_GetSyncSampling (T_idtDrvHdlr hdlr, T_osUInt8 exSyncN)`

*Function to retrieve sampling configuration for external sync.*

- `void IDTInput_SetSyncEdge (T_idtDrvHdlr hdlr, T_externalSyncEdge edge)`

*Function to provision synchronization alignment.*

- `T_externalSyncEdge IDTInput_GetSyncEdge (T_idtDrvHdlr hdlr)`

*Function to retrieve synchronization alignment.*

- `void IDTInput_SetSyncAlarmRange (T_idtDrvHdlr hdlr, T_externalSyncAlarmRange range)`

*Function to provision the sync allowable range before alarming.*

- `T_externalSyncAlarmRange IDTInput_GetSyncAlarmRange (T_idtDrvHdlr hdlr)`

*Function to retrieve the sync allowable range before alarming.*

- `void IDTInput_SetSyncBypass (T_idtDrvHdlr hdlr, T_externalSyncBypass bypass)`

*Function to provision sync bypass mode.*

- `T_externalSyncBypass IDTInput_GetSyncBypass (T_idtDrvHdlr hdlr)`

*Function to retrieve sync bypass mode.*

- `void IDTInput_SetSyncEnable (T_idtDrvHdlr hdlr, T_externalSyncEnable enable)`

*Function to provision enable mode for input external sync.*

- `T_externalSyncEnable IDTInput_GetSyncEnable (T_idtDrvHdlr hdlr)`

*Function to retrieve external sync enable mode.*

- `T_idtInputFrequencyFamily IDTInput_InFreqBitValue2Family (T_idtDrvHdlr hdlr, T_idtInputFrequencyBitfield-Value inFreqBitValue)`

*Translate input frequency bit value to input frequency family.*

#### 4.5.1 Detailed Description

Functions in this group are related to input port provisioning. This module also include frequency monitoring controls too.

## 4.5.2 Macro Definition Documentation

### 4.5.2.1 #define IDT\_EXT\_INPORT\_POPULATED( *hdlr*, *ext* ) (hdlr->deviceConfig\_m->inputExt2IntXlate\_ma[(ext)])

Utility macro to check if input port is populated.

#### Parameters

|             |                            |
|-------------|----------------------------|
| <i>hdlr</i> | Device driver handle       |
| <i>ext</i>  | External input port number |

#### Returns

a positive number - populated, 0 - not populated

Definition at line 223 of file Input.h.

### 4.5.2.2 #define IDT\_TRANSLATE\_EXT\_TO\_INT\_INPORT( *hdlr*, *ext*, *internal* )

#### Value:

```
do {
    OS_ASSERT((ext) <= IDT_DRV_MAX_INPUT);
    OS_ASSERT((ext) > 0);
    OS_ASSERTS(hdlr->deviceConfig_m->inputExt2IntXlate_ma[(ext)],
               ("Unsupported input port %d\n", (ext)));
    internal = hdlr->deviceConfig_m->inputExt2IntXlate_ma[(ext)];
} while(0)
```

Utility macro to check if input port is supported and to translated from external port to internal port.

#### Parameters

|                 |                            |
|-----------------|----------------------------|
| <i>hdlr</i>     | Device driver handle       |
| <i>ext</i>      | External input port number |
| <i>internal</i> | Internal input number      |

Definition at line 208 of file Input.h.

### 4.5.2.3 #define Input\_Freq\_Hard\_Alarm 4

Definition at line 79 of file Input.h.

### 4.5.2.4 #define Input\_No\_Activity\_Alarm 2

Definition at line 78 of file Input.h.

### 4.5.2.5 #define Input\_Phase\_Lock\_Alarm 1

Definition at line 77 of file Input.h.

## 4.5.3 Enumeration Type Documentation

### 4.5.3.1 enum T\_externalSyncAlarmRange

Enumerate type listing ranges for external sync alarm.

## Enumerator

***externalSyncAlarmRange\_1*** +/- 1 UI range  
***externalSyncAlarmRange\_2*** +/- 2 UI range  
***externalSyncAlarmRange\_3*** +/- 3 UI range  
***externalSyncAlarmRange\_4*** +/- 4 UI range  
***externalSyncAlarmRange\_5*** +/- 5 UI range  
***externalSyncAlarmRange\_6*** +/- 6 UI range  
***externalSyncAlarmRange\_7*** +/- 7 UI range  
***externalSyncAlarmRange\_8*** +/- 8 UI range  
***externalSyncAlarmRange\_MAX***

Definition at line 161 of file Input.h.

## 4.5.3.2 enum T\_externalSyncBypass

Enumerated type listing output synchronization with respect to.

## Enumerator

***externalSyncBypass\_ExSync1*** Synchronized to external sync 1  
***externalSyncBypass\_AutoSel*** Auto synchronize using DPLL #1. If DPLL #1 select IN1 or IN4 then synchronize to external sync1 If DPLL #1 select IN2 then synchronize to external sync 2. If no input selected by DPLL #1, external sync are not synchronized  
***externalSyncBypass\_MAX***

Definition at line 176 of file Input.h.

## 4.5.3.3 enum T\_externalSyncEdge

Enumerated type listing possible external sync alignment options.

## Enumerator

***externalSyncEdge\_FallingEdge*** External sync align to falling edge  
***externalSyncEdge\_RisingEdge*** External sync align to rising edge  
***externalSyncEdge\_MAX***

Definition at line 151 of file Input.h.

## 4.5.3.4 enum T\_externalSyncEnable

Enumerated type listing enable options for external input sync.

## Enumerator

***externalSyncEnable\_Disable*** Disable external sync  
***externalSyncEnable\_Enable*** Enable external sync  
***externalSyncEnable\_Auto*** Enable/Disable external sync based on DPLL #1 locking to (internal) input 4  
***externalSyncEnable\_MAX***

Definition at line 193 of file Input.h.

## 4.5.3.5 enum T\_externalSyncSampling

Enumerated type listing sampling of input external sync.

Enumerator

***externalSyncSampling\_onTarget*** On target (0 UI)  
***externalSyncSampling\_0dot5Early*** 0.5 UI early  
***externalSyncSampling\_1dot0Late*** 1.0 UI late  
***externalSyncSampling\_0dot5Late*** 0.5 UI late  
***externalSyncSampling\_MAX***

Definition at line 139 of file Input.h.

## 4.5.3.6 enum T\_idtAmiInputFrequencyBitFieldValue

Enumerated type listing AMI Input frequencies.

Enumerator

***AmiFreq\_64kHzPlus8kHz*** 64 kHz + 8 kHz signal  
***AmiFreq\_64kHzPlus400Hz*** 64 kHz + 400 Hz signal  
***AmiFreq\_MAX***

Definition at line 118 of file Input.h.

## 4.5.3.7 enum T\_idtHighFrequencyDivisor

Enumerated type listing high frequency (>155.52MHz) dividers.

Enumerator

***HF\_Bypass*** Bypass high frequency divider  
***HF\_Divide\_4*** HF Divide by 4  
***HF\_Divide\_5*** HF Divide by 5  
***HF\_MAX***

Definition at line 68 of file Input.h.

## 4.5.3.8 enum T\_idtInputDividerMux

Enumerated type listing input divider modes.

Enumerator

***Divider\_Bypass*** Bypass divider mode  
***Divider\_Lock8k*** Lock 8kHz divider mode  
***Divider\_DivN*** Divide by N (DivN) mode  
***Divider\_MAX***

Definition at line 59 of file Input.h.

#### 4.5.3.9 enum T\_idtInputFrequencyBitfieldValue

Enumerated type listing input frequencies.

Enumerator

*Freq\_8K*  
*Freq\_1544M*  
*Freq\_2048M*  
*Freq\_648M*  
*Freq\_1944M*  
*Freq\_2592M*  
*Freq\_3888M*  
*Freq\_2K*  
*Freq\_4K*  
*Freq\_1PPS*  
*Freq\_625M*  
*Freq\_10M*  
*Freq\_MAX*

Definition at line 82 of file Input.h.

#### 4.5.3.10 enum T\_idtInputFrequencyFamily

Enumerated type listing input frequency families.

Enumerator

*InFreqFamily\_8K*  
*InFreqFamily\_1544M*  
*InFreqFamily\_2048M*  
*InFreqFamily\_648M*  
*InFreqFamily\_1944M*  
*InFreqFamily\_2592M*  
*InFreqFamily\_3888M*  
*InFreqFamily\_2K*  
*InFreqFamily\_4K*  
*InFreqFamily\_1PPS*  
*InFreqFamily\_625M*  
*InFreqFamily\_10M*  
*InFreqFamily\_MAX*

Definition at line 100 of file Input.h.

## 4.5.3.11 enum T\_idtInputSyncFreq

Enumerated type listing possible input sync frequencies.

Enumerator

**SYNC\_8kHz** 8 kHz sync input  
**SYNC\_1PPS** 1 PPS sync input  
**SYNC\_4kHz** 4 kHz sync input  
**SYNC\_2kHz** 2 kHz sync input  
**SYNC\_MAX**

Definition at line 128 of file Input.h.

## 4.5.4 Function Documentation

4.5.4.1 void IDTInput\_DisableAllInputs ( T\_idtDrvHdlr *hdlr* )

Disable to select anyone of input clocks.

Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 790 of file Input.c.

4.5.4.2 void IDTInput\_EnableAllInputs ( T\_idtDrvHdlr *hdlr* )

Enable to select anyone of input clocks.

Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 771 of file Input.c.

4.5.4.3 T\_osUInt8 IDTInput\_GetActivityMonitor ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo* )

Retrieve selected activity monitoring configuration.

Parameters

|                 |                       |
|-----------------|-----------------------|
| <i>hdlr</i>     | Device driver handler |
| <i>nInputNo</i> | the specified port    |

Returns

nBucket the bucket number assigned to the specified port.

- 0= bucket 0, controlled by the registers from 31H to 34H
- 1= bucket 1, controlled by the registers from 35H to 38H
- 2= bucket 2, controlled by the registers from 39H to 3cH
- 3= bucket 3, controlled by the registers from 3dH to 40H

Definition at line 363 of file Input.c.

#### 4.5.4.4 T\_idtAmilInputFrequencyBitFieldValue IDTInput.GetAmilInputFrequency ( T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo )

Function to retrieve AMI input frequency for input port.

##### Parameters

|                 |                                 |
|-----------------|---------------------------------|
| <i>hdlr</i>     | Device driver handler           |
| <i>nInputNo</i> | the specified input port number |

##### Returns

AMI input frequency

Definition at line 1238 of file Input.c.

#### 4.5.4.5 void IDTInput.GetBucketConfig ( T\_idtDrvHdlr hdlr, T\_osUInt8 nBucket, T\_osUInt8 \* nUpper\_p, T\_osUInt8 \* nLower\_p, T\_osUInt8 \* nSize\_p, T\_osUInt8 \* nDecay\_p )

Get activity monitoring configuration (Leaky Bucket)

##### Parameters

|                 |                                                                                                                                                                                                                                                                                              |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>     | Device driver handler                                                                                                                                                                                                                                                                        |
| <i>nBucket</i>  | the bucket number to provision 0-3                                                                                                                                                                                                                                                           |
| <i>nUpper_p</i> | Upper threshold of Leaky Bucket                                                                                                                                                                                                                                                              |
| <i>nLower_p</i> | Lower threshold of Leaky Bucket                                                                                                                                                                                                                                                              |
| <i>nSize_p</i>  | Bucket size of Leaky Bucket                                                                                                                                                                                                                                                                  |
| <i>nDecay_p</i> | Decay rate of Leaky bucket <ul style="list-style-type: none"> <li>• 0x0 = Bucket decrease by 1 in every 128 ms</li> <li>• 0x1 = Bucket decrease by 1 in every 256 ms</li> <li>• 0x2 = Bucket decrease by 1 in every 512 ms</li> <li>• 0x3 = Bucket decrease by 1 in every 1024 ms</li> </ul> |

Definition at line 431 of file Input.c.

#### 4.5.4.6 T\_osUInt16 IDTInput.GetDivisionFactor ( T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo )

Get the dividen factor to pre-divider assigned to a certain input.

##### Parameters

|                 |                                                |
|-----------------|------------------------------------------------|
| <i>hdlr</i>     | Device driver handler                          |
| <i>nInputNo</i> | the input port number to which the pre-divider |

##### Returns

division factor as a preset of the specified divider (24 bit 2's compliment)

Definition at line 568 of file Input.c.

#### 4.5.4.7 T\_osUInt8 IDTInput.GetInputClockStatus ( T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo )

Get the status of a specified input clock.

## Parameters

|                 |                           |
|-----------------|---------------------------|
| <i>hdlr</i>     | Device driver handler     |
| <i>nInputNo</i> | number of the input clock |

## Returns

nStatus.Bit0: =1, Phase lock alarm. =0, No phase lock alarm. nStatus.Bit1: =1, Clock no-activity alarm. =0, No clock no-activity alarm. nStatus.Bit2: =1, Clock frequency hard alarm. =0, No clock frequency hard alarm.

Definition at line 710 of file Input.c.

**4.5.4.8 T\_osUInt8 IDTInput\_GetInputClockValidation ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo* )**

Get the clock validation status of an input port.

## Parameters

|                 |                       |
|-----------------|-----------------------|
| <i>hdlr</i>     | Device driver handler |
| <i>nInputNo</i> | Input port            |

## Returns

0-Invalid clock, 1-Valid clock

Definition at line 738 of file Input.c.

**4.5.4.9 T\_osUInt8 IDTInput\_GetInputEnable ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo* )**

Retrieve enable/disable status of selected input.

## Parameters

|                 |                           |
|-----------------|---------------------------|
| <i>hdlr</i>     | Device driver handler     |
| <i>nInputNo</i> | number of the input clock |

## Returns

enable 1-Disable 0-Enable input port

Definition at line 849 of file Input.c.

**4.5.4.10 T\_osUInt8 IDTInput\_GetInputFreqOffset ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo* )**

Get the frequency offset of the specified input clock. The returned value is updated every 16 seconds.

## Parameters

|                 |                                                               |
|-----------------|---------------------------------------------------------------|
| <i>hdlr</i>     | Device driver handler                                         |
| <i>nInputNo</i> | number of the input clock whose frequency offset is returned. |

## Returns

Frequency offset

Definition at line 675 of file Input.c.

#### 4.5.4.11 T\_idtInputFrequencyBitfieldValue IDTInput\_GetInputFrequency ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo* )

Get frequency of the input clock at the specified port.

##### Parameters

|                 |                                 |
|-----------------|---------------------------------|
| <i>hdlr</i>     | Device driver handler           |
| <i>nInputNo</i> | the specified input port number |

##### Returns

nFrequency code of input clock frequency.

- Freq\_8K= 8 kHz
- Freq\_1544M/Freq\_2048M= 1.544/2.048 MHz
- Freq\_648M= 6.48 MHz
- Freq\_1944M= 19.44 MHz
- Freq\_2592M= 25.92 MHz
- Freq\_3888M= 38.88 MHz
- Freq\_2K= 2 kHz
- Freq\_4K= 4 kHz
- Freq\_1PPS= 1 PPS
- Freq\_625M= 6.25 MHz
- Freq\_10M= 10 MHz

Definition at line 299 of file Input.c.

#### 4.5.4.12 T\_idtHighFrequencyDivisor IDTInput\_GetInputHFdivider ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo* )

Get high frequency configuration divider.

##### Parameters

|                 |                       |
|-----------------|-----------------------|
| <i>hdlr</i>     | Device driver handler |
| <i>nInputNo</i> | Input port number     |

##### Returns

- HF\_Bypass : The HF divider is bypassed
- HF\_Divide\_4 : The HF divider's division factor is 4
- HF\_Divide\_5 : The HF divider's division factor is 5

Definition at line 640 of file Input.c.

#### 4.5.4.13 T\_idtInputDividerMux IDTInput\_GetPreDivider ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo* )

Get configured input port pre-divider.

##### Parameters

|                 |                                 |
|-----------------|---------------------------------|
| <i>hdlr</i>     | Device driver handler           |
| <i>nInputNo</i> | the specified input port number |

**Returns**

pre-divider assigned to the specified input

- 0 = bypass both dividers
- 1 = Lock-8k divider
- 2 = DivN divider

Definition at line 502 of file Input.c.

#### 4.5.4.14 T\_osUInt8 IDTInput\_GetPriority ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo*, T\_idtDpllInstance *nDpll* )

Get the current priority of the specified port.

**Parameters**

|                 |                                                               |
|-----------------|---------------------------------------------------------------|
| <i>hdlr</i>     | Device driver handler                                         |
| <i>nInputNo</i> | Input port number (consult data sheet for valid port numbers) |
| <i>nDpll</i>    | DPLL instance                                                 |

**Returns**

Priority of the port

Definition at line 211 of file Input.c.

#### 4.5.4.15 T\_externalSyncAlarmRange IDTInput\_GetSyncAlarmRange ( T\_idtDrvHdlr *hdlr* )

Function to retrieve the sync allowable range before alarming.

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

**Returns**

Allowable range

Definition at line 1053 of file Input.c.

#### 4.5.4.16 T\_externalSyncBypass IDTInput\_GetSyncBypass ( T\_idtDrvHdlr *hdlr* )

Function to retrieve sync bypass mode.

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

**Returns**

Bypass mode

Definition at line 1091 of file Input.c.

#### 4.5.4.17 T\_externalSyncEdge IDTInput\_GetSyncEdge ( T\_idtDrvHdlr *hdlr* )

Function to retrieve synchronization alignment.

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

##### Returns

Edge to synchronize against

Definition at line 1014 of file Input.c.

#### 4.5.4.18 T\_externalSyncEnable IDTInput\_GetSyncEnable ( T\_idtDrvHdlr *hdlr* )

Function to retrieve external sync enable mode.

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

##### Returns

Enable mode

Definition at line 1165 of file Input.c.

#### 4.5.4.19 T\_idtInputSyncFreq IDTInput\_GetSyncFrequency ( T\_idtDrvHdlr *hdlr* )

Function to retrieve select input external sync frequency.

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

##### Returns

Input external sync

Definition at line 902 of file Input.c.

#### 4.5.4.20 T\_externalSyncSampling IDTInput\_GetSyncSampling ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *exSyncN* )

Function to retrieve sampling configuration for external sync.

##### Parameters

|                |                                         |
|----------------|-----------------------------------------|
| <i>hdlr</i>    | Device driver handler                   |
| <i>exSyncN</i> | External sync (1-EXT SYNC1, 2-EX SYNC2) |

##### Returns

Sampling configuration

Definition at line 957 of file Input.c.

#### 4.5.4.21 `T_idtInputFrequencyFamily` `IDTInput_InFreqBitValue2Family` ( `T_idtDrvHdlr` *hdlr*, `T_idtInputFrequencyBitfieldValue` *inFreqBitValue* )

Translate input frequency bit value to input frequency family.

##### Parameters

|                       |                           |
|-----------------------|---------------------------|
| <i>hdlr</i>           | Device driver handler     |
| <i>inFreqBitValue</i> | Input frequency bit value |

##### Returns

Input frequency family

Definition at line 95 of file Input.c.

#### 4.5.4.22 `void` `IDTInput_SetActivityMonitor` ( `T_idtDrvHdlr` *hdlr*, `T_osUInt8` *nInputNo*, `T_osUInt8` *nBucket* )

Select one of the four activity monitor configurations for the specified input port.

##### Parameters

|                 |                                                                                                                                                                                                                                                                                                                                                                            |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>     | Device driver handler                                                                                                                                                                                                                                                                                                                                                      |
| <i>nInputNo</i> | the specified port                                                                                                                                                                                                                                                                                                                                                         |
| <i>nBucket</i>  | the bucket number assigned to the specified port. <ul style="list-style-type: none"> <li>• 0= bucket 0, controlled by the registers from 31H to 34H</li> <li>• 1= bucket 1, controlled by the registers from 35H to 38H</li> <li>• 2= bucket 2, controlled by the registers from 39H to 3cH</li> <li>• 3= bucket 3, controlled by the registers from 3dH to 40H</li> </ul> |

Definition at line 329 of file Input.c.

#### 4.5.4.23 `void` `IDTInput_SetAmiInputFrequency` ( `T_idtDrvHdlr` *hdlr*, `T_osUInt8` *nInputNo*, `T_idtAmiInputFrequencyBitFieldValue` *amiSignal* )

Provision AMI signal for input port.

##### Parameters

|                  |                                                                                                                                                                                                 |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>      | Device driver handler                                                                                                                                                                           |
| <i>nInputNo</i>  | the specified input port number                                                                                                                                                                 |
| <i>amiSignal</i> | AMI signal <ul style="list-style-type: none"> <li>• <code>AmiFreq_64kHzPlus8kHz</code>= 64 kHz + 8 kHz signal</li> <li>• <code>AmiFreq_64kHzPlus400Hz</code>= 64 kHz + 400 Hz signal</li> </ul> |

Definition at line 1195 of file Input.c.

#### 4.5.4.24 `void` `IDTInput_SetBucketConfig` ( `T_idtDrvHdlr` *hdlr*, `T_osUInt8` *nBucket*, `T_osUInt8` *nUpper*, `T_osUInt8` *nLower*, `T_osUInt8` *nSize*, `T_osUInt8` *nDecay* )

Set the configuration to Leaky Bucket activity monitoring.

## Parameters

|                |                                                                                                                                                                                                                                                                                              |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>    | Device driver handler                                                                                                                                                                                                                                                                        |
| <i>nBucket</i> | the bucket number to provision 0-3                                                                                                                                                                                                                                                           |
| <i>nUpper</i>  | Upper threshold of Leaky Bucket                                                                                                                                                                                                                                                              |
| <i>nLower</i>  | Lower threshold of Leaky Bucket                                                                                                                                                                                                                                                              |
| <i>nSize</i>   | Bucket size of Leaky Bucket                                                                                                                                                                                                                                                                  |
| <i>nDecay</i>  | Decay rate of Leaky bucket <ul style="list-style-type: none"> <li>• 0x0 = Bucket decrease by 1 in every 128 ms</li> <li>• 0x1 = Bucket decrease by 1 in every 256 ms</li> <li>• 0x2 = Bucket decrease by 1 in every 512 ms</li> <li>• 0x3 = Bucket decrease by 1 in every 1024 ms</li> </ul> |

Definition at line 393 of file Input.c.

4.5.4.25 void IDTInput\_SetDivisionFactor ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo*, T\_osUInt16 *nFactor* )

Set the dividen factor to pre-divider assigned to a certain input.

## Parameters

|                 |                                                      |
|-----------------|------------------------------------------------------|
| <i>hdlr</i>     | Device driver handler                                |
| <i>nInputNo</i> | the input port number to which the pre-divider       |
| <i>nFactor</i>  | division factor as a preset of the specified divider |

Definition at line 531 of file Input.c.

4.5.4.26 void IDTInput\_SetInputEnable ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo*, T\_osUInt8 *enable* )

Enable/Disable selected input.

## Parameters

|                 |                               |
|-----------------|-------------------------------|
| <i>hdlr</i>     | Device driver handler         |
| <i>nInputNo</i> | number of the input clock     |
| <i>enable</i>   | 1-Disable 0-Enable input port |

Definition at line 811 of file Input.c.

4.5.4.27 void IDTInput\_SetInputFrequency ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo*, T\_idtInputFrequencyBitfieldValue *nFrequency* )

Set the frequency of the input clock at the specified port.

## Parameters

|                   |                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                                                                                                                                                                                                                                                                                                            |
| <i>nInputNo</i>   | the specified input port number                                                                                                                                                                                                                                                                                                                                                                                                  |
| <i>nFrequency</i> | code of input clock frequency. <ul style="list-style-type: none"> <li>• Freq_8K= 8 kHz</li> <li>• Freq_1544M/Freq_2048M= 1.544/2.048 MHz</li> <li>• Freq_648M= 6.48 MHz</li> <li>• Freq_1944M= 19.44 MHz</li> <li>• Freq_2592M= 25.92 MHz</li> <li>• Freq_3888M= 38.88 MHz</li> <li>• Freq_2K= 2 kHz</li> <li>• Freq_4K= 4 kHz</li> <li>• Freq_1PPS= 1 PPS</li> <li>• Freq_625M= 6.25 MHz</li> <li>• Freq_10M= 10 MHz</li> </ul> |

Definition at line 255 of file Input.c.

4.5.4.28 void IDTInput\_SetInputHFdivider ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo*, T\_idtHighFrequencyDivisor *nUsage* )

Set the usage of high frequency (> 155.52MHz) divider.

## Parameters

|                 |                                                                                                                                                                                                                          |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>     | Device driver handler                                                                                                                                                                                                    |
| <i>nInputNo</i> | Input port number                                                                                                                                                                                                        |
| <i>nUsage</i>   | <ul style="list-style-type: none"> <li>• HF_Bypass : The HF divider is bypassed</li> <li>• HF_Divide_4 : The HF divider's division factor is 4</li> <li>• HF_Divide_5 : The HF divider's division factor is 5</li> </ul> |

Definition at line 600 of file Input.c.

4.5.4.29 void IDTInput\_SetPreDivider ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo*, T\_idtInputDividerMux *nDivider* )

Assign a pre-divider for the specified input.

## Parameters

|                 |                                                                                                                                                                                     |
|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>     | Device driver handler                                                                                                                                                               |
| <i>nInputNo</i> | the specified input port number                                                                                                                                                     |
| <i>nDivider</i> | pre-divider assigned to the specified input <ul style="list-style-type: none"> <li>• 0 = bypass both dividers</li> <li>• 1 = Lock-8k divider</li> <li>• 2 = DivN divider</li> </ul> |

Definition at line 466 of file Input.c.

**4.5.4.30 void IDTInput\_SetPriority ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo*, T\_idtDpllInstance *nDpll*, T\_osUInt8 *nPriority* )**

Set specified port to the priority.

**Parameters**

|                  |                                                               |
|------------------|---------------------------------------------------------------|
| <i>hdlr</i>      | Device driver handler                                         |
| <i>nInputNo</i>  | Input port number (consult data sheet for valid port numbers) |
| <i>nDpll</i>     | DPLL instance                                                 |
| <i>nPriority</i> | the priority of the specified port                            |

Definition at line 170 of file Input.c.

**4.5.4.31 void IDTInput\_SetSyncAlarmRange ( T\_idtDrvHdlr *hdlr*, T\_externalSyncAlarmRange *range* )**

Function to provision the sync allowable range before alarming.

**Parameters**

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>range</i> | Allowable range       |

Definition at line 1030 of file Input.c.

**4.5.4.32 void IDTInput\_SetSyncBypass ( T\_idtDrvHdlr *hdlr*, T\_externalSyncBypass *bypass* )**

Function to provision sync bypass mode.

**Parameters**

|               |                       |
|---------------|-----------------------|
| <i>hdlr</i>   | Device driver handler |
| <i>bypass</i> | Bypass mode           |

Definition at line 1069 of file Input.c.

**4.5.4.33 void IDTInput\_SetSyncEdge ( T\_idtDrvHdlr *hdlr*, T\_externalSyncEdge *edge* )**

Function to provision synchronization alignment.

**Parameters**

|             |                             |
|-------------|-----------------------------|
| <i>hdlr</i> | Device driver handler       |
| <i>edge</i> | Edge to synchronize against |

Definition at line 992 of file Input.c.

**4.5.4.34 void IDTInput\_SetSyncEnable ( T\_idtDrvHdlr *hdlr*, T\_externalSyncEnable *enable* )**

Function to provision enable mode for input external sync.

**Parameters**

|               |                       |
|---------------|-----------------------|
| <i>hdlr</i>   | Device driver handler |
| <i>enable</i> | Enable mode           |

Definition at line 1106 of file Input.c.

4.5.4.35 void IDTInput\_SetSyncFrequency ( T\_idtDrvHdlr *hdlr*, T\_idtInputSyncFreq *syncFreq* )

Function to select input external sync frequency.

Parameters

|                 |                       |
|-----------------|-----------------------|
| <i>hdlr</i>     | Device driver handler |
| <i>syncFreq</i> | Input external sync   |

Definition at line 881 of file Input.c.

4.5.4.36 void IDTInput\_SetSyncSampling ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *exSyncN*, T\_externalSyncSampling *sampling* )

Function to provision sampling configuration for external sync.

Parameters

|                 |                                         |
|-----------------|-----------------------------------------|
| <i>hdlr</i>     | Device driver handler                   |
| <i>exSyncN</i>  | External sync (1-EXT SYNC1, 2-EX SYNC2) |
| <i>sampling</i> | Sampling configuration                  |

Definition at line 918 of file Input.c.

## 4.6 Output Function Module

Functions in this group are related to output port provisioning.

### Macros

- `#define IDT_TRANSLATE_EXT_TO_INT_OUTPORT(hdrlr, ext, internal)`  
*Utility macro to check if output port is supported and to translated from external port to internal port.*
- `#define IDT_EXT_OUTPORT_POPULATED(hdrlr, ext) (hdrlr->deviceConfig_m->outputExt2IntXlate_ma[(ext)])`  
*Utility macro to check if output port is populated.*

### Enumerations

- enum `T_outputPathType` {  
OutputPath\_SonetGigEth, OutputPath\_7776kHz, OutputPath\_Eth, OutputPath\_16E1\_16T1,  
OutputPath\_12E1\_E3\_T3, OutputPath\_GSM\_16E1\_16T1, OutputPath\_GPS, OutputPath\_OBSAI,  
OutputPath\_24T1, OutputPath\_MAX }  
*Enumerated type listing possible output port sources (from DPLL(s)) This enumerated type is used in conjunction with DPLL instance to select output DPLL frequency to output dividers. The following table outlines.*
- enum `T_outputInterface` {  
OUTPUTIF\_AMI, OUTPUTIF\_CMOS, OUTPUTIF\_LVDS, OUTPUTIF\_PECL,  
OUTPUTIF\_MAX }  
*Enumerated type listing type of supported output port interfaces.*
- enum `T_frameSync` { frameSync\_FRSYNC, frameSync\_MFRSYNC, frameSync\_MAX }  
*Enumerated type listing supported frame syncs.*

### Functions

- void `IDTOutput_SetOutputConfig` (T\_idtDrvHdrlr hdrlr, T\_osUInt8 nOutN, T\_idtDpllInstance nDpll, T\_outputPathType nPath, T\_osUInt8 nFactor, T\_idtApplConfigPerOutput \*applConfig)  
*Set the configuration of Output port.*
- void `IDTOutput_GetOutputConfig` (T\_idtDrvHdrlr hdrlr, T\_osUInt8 nOutN, T\_idtDpllInstance \*nDpll\_p, T\_outputPathType \*nPath\_p, T\_osUInt8 \*nFactor\_p, T\_idtApplConfigPerOutput \*applConfig)  
*Get the configuration of Output port.*
- void `IDTOutput_SetOutputInvert` (T\_idtDrvHdrlr hdrlr, T\_osUInt8 nOutN, T\_osUInt8 invert)  
*Set the specified output port to generate a invert signal.*
- T\_osUInt8 `IDTOutput_GetOutputInvert` (T\_idtDrvHdrlr hdrlr, T\_osUInt8 nOutN)  
*Get output signal inversion status.*
- void `IDTOutput_SetOutputEnable` (T\_idtDrvHdrlr hdrlr, T\_osUInt8 nOutN, T\_osUInt8 nEnable)  
*Enable/Disable output port. Consult data sheet for supported ports.*
- T\_osUInt8 `IDTOutput_GetOutputEnable` (T\_idtDrvHdrlr hdrlr, T\_osUInt8 nOutN)  
*Get enable/disable status of output port. Consult data sheet for supported ports.*
- void `IDTOutput_SetOutputInterface` (T\_idtDrvHdrlr hdrlr, T\_osUInt8 nOutN, T\_outputInterface outIf)  
*Set output port interface type. Consult data sheet for supported interfaces per port.*
- T\_outputInterface `IDTOutput_GetOutputInterface` (T\_idtDrvHdrlr hdrlr, T\_osUInt8 nOutN)  
*Get output port interface type. Consult data sheet for supported interfaces per port.*
- void `IDTOutput_SetFrameSync` (T\_idtDrvHdrlr hdrlr, T\_frameSync frameSync, T\_osUInt8 enable, T\_osUInt8 invert, T\_osUInt8 pulse)  
*Function to control output frame syncs.*
- void `IDTOutput_GetFrameSync` (T\_idtDrvHdrlr hdrlr, T\_frameSync frameSync, T\_osUInt8 \*enable\_p, T\_osUInt8 \*invert\_p, T\_osUInt8 \*pulse\_p)

*Function to retrieve output frame sync configuration.*

- void [IDTOutput\\_SetFrameSyncPulseEdge](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 isRisingEdge)

*Function provision trigger for frame sync pulses.*

- T\_osUInt8 [IDTOutput\\_GetFrameSyncPulseEdge](#) (T\_idtDrvHdlr hdlr)

*Function retrieve trigger for frame sync pulses.*

- void [IDTOutput\\_DisableApIInput](#) (T\_idtDrvHdlr hdlr, [T\\_idtOutputApIConfig](#) outputApIConfig)

*Function disables APLL input.*

- void [IDTOutput\\_DisableAllIntOutput](#) (T\_idtDrvHdlr hdlr)

*Function disables all internal outputs.*

#### 4.6.1 Detailed Description

Functions in this group are related to output port provisioning.

#### 4.6.2 Macro Definition Documentation

##### 4.6.2.1 #define IDT\_EXT\_OUTPORT\_POPULATED( hdlr, ext ) (hdlr->deviceConfig\_m->outputExt2IntXlate\_ma[(ext)])

Utility macro to check if output port is populated.

##### Parameters

|             |                             |
|-------------|-----------------------------|
| <i>hdlr</i> | Device driver handle        |
| <i>ext</i>  | External output port number |

##### Returns

a positive number - populated, 0 - not populated

Definition at line 79 of file Output.h.

##### 4.6.2.2 #define IDT\_TRANSLATE\_EXT\_TO\_INT\_OUTPORT( hdlr, ext, internal )

##### Value:

```
do {
    OS_ASSERT((ext)<=IDT_DRV_MAX_OUTPUT); \
    OS_ASSERT((ext)>0); \
    OS_ASSERTS(hdlr->deviceConfig_m->outputExt2IntXlate_ma[(ext)], \
        ("Unsupported output port %d\n", (ext))); \
    internal = hdlr->deviceConfig_m->outputExt2IntXlate_ma[(ext)]; \
} while(0)
```

Utility macro to check if output port is supported and to translated from external port to internal port.

##### Parameters

|                 |                             |
|-----------------|-----------------------------|
| <i>hdlr</i>     | Device driver handle        |
| <i>ext</i>      | External output port number |
| <i>internal</i> | Internal output number      |

Definition at line 64 of file Output.h.

#### 4.6.3 Enumeration Type Documentation

#### 4.6.3.1 enum T\_frameSync

Enumerated type listing supported frame syncs.

Enumerator

***frameSync\_FRSYNC*** FR Sync (8 kHz / 1 PPS)  
***frameSync\_MFRSYNC*** MFR Sync (2kHz / 1 PPS)  
***frameSync\_MAX***

Definition at line 121 of file Output.h.

#### 4.6.3.2 enum T\_outputInterface

Enumerated type listing type of supported output port interfaces.

Enumerator

***OUTPUTIF\_AMI*** AMI  
***OUTPUTIF\_CMOS*** CMOS  
***OUTPUTIF\_LVDS*** LVDS  
***OUTPUTIF\_PECL*** PECL  
***OUTPUTIF\_MAX***

Definition at line 110 of file Output.h.

#### 4.6.3.3 enum T\_outputPathType

Enumerated type listing possible output port sources (from DPLL(s)) This enumerated type is used in conjunction with DPLL instance to select output DPLL frequency to output dividers. The following table outlines.

Enumerator

***OutputPath\_SonetGigEth*** From corresponding Sonet/GigE path  
***OutputPath\_7776kHz*** From SONET/SDH path 77.76MHz  
***OutputPath\_Eth*** From Ethernet path 25MHz  
***OutputPath\_16E1\_16T1*** From 16E1/16T1 path 32.768MHz/24.704MHz  
***OutputPath\_12E1\_E3\_T3*** From 12E1/E3/T3 path 24.576MHz/34.368MHz/44.736MHz  
***OutputPath\_GSM\_16E1\_16T1*** From GSM/16E1/16T1 path 26MHz/32.768MHz/16.384MHz  
***OutputPath\_GPS*** From GPS path 40MHz  
***OutputPath\_OBSAI*** From OBSAI path 30.72MHz  
***OutputPath\_24T1*** From 24T1 path 37.056MHz  
***OutputPath\_MAX***

Definition at line 89 of file Output.h.

### 4.6.4 Function Documentation

#### 4.6.4.1 void IDTOutput\_DisableAllIntOutput ( T\_idtDrvHdlr hdlr )

Function disables all internal outputs.

Should ONLY be called from device init function to disable internal outputs for better jitter performance

## Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 1030 of file Output.c.

#### 4.6.4.2 void IDTOutput\_DisableApIInput ( T\_idtDrvHdlr *hdlr*, T\_idtOutputApIConfig *outputApIConfig* )

Function disables APLL input.

Only valid for output path has APLL configured

## Parameters

|                        |                                |
|------------------------|--------------------------------|
| <i>hdlr</i>            | Device driver handler          |
| <i>outputApIConfig</i> | APLL output path configuration |

Definition at line 1004 of file Output.c.

#### 4.6.4.3 void IDTOutput\_GetFrameSync ( T\_idtDrvHdlr *hdlr*, T\_frameSync *frameSync*, T\_osUInt8 \* *enable\_p*, T\_osUInt8 \* *invert\_p*, T\_osUInt8 \* *pulse\_p* )

Function to retrieve output frame sync configuration.

## Parameters

|                  |                                                                                        |
|------------------|----------------------------------------------------------------------------------------|
| <i>hdlr</i>      | Device driver handler                                                                  |
| <i>frameSync</i> | Select between FR and MFR frame syncs                                                  |
| <i>enable_p</i>  | Enable frame sync                                                                      |
| <i>invert_p</i>  | Invert output frame sync (0-noninverted, 1-invert)                                     |
| <i>pulse_p</i>   | Select between clock (50:50 clock cycle) or pulse (dictated by IN3) (0-clock, 1-pulse) |

Definition at line 919 of file Output.c.

#### 4.6.4.4 T\_osUInt8 IDTOutput\_GetFrameSyncPulseEdge ( T\_idtDrvHdlr *hdlr* )

Function retrieve trigger for frame sync pulses.

Only valid for frame syncs when provisioned for pulse.

## Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

## Returns

Frame sync pulse triggered by rising edge (0-falling edge, 1-rising edge)

Definition at line 986 of file Output.c.

#### 4.6.4.5 T\_outputInterface IDTOutput\_GetOutputInterface ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nOutN* )

Get output port interface type. Consult data sheet for supported interfaces per port.

## Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nOutN</i> | Output port number    |

**Returns**

Interface type.

Definition at line 793 of file Output.c.

**4.6.4.6** void IDTOutput\_GetOutputConfig ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nOutN*, T\_idtDpllInstance \* *nDpll\_p*, T\_outputPathType \* *nPath\_p*, T\_osUInt8 \* *nFactor\_p*, T\_idtApIConfigPerOutput \* *apIConfig* )

Get the configuration of Output port.

**Parameters**

|                  |                                                                              |
|------------------|------------------------------------------------------------------------------|
| <i>hdlr</i>      | Device driver handler                                                        |
| <i>nOutN</i>     | Output port that want to be used                                             |
| <i>nDpll_p</i>   | DPLL instance                                                                |
| <i>nPath_p</i>   | Select a specified path as input of output divider                           |
| <i>nFactor_p</i> | Dividen factor of a certain output divider. Look up the table in datasheet   |
| <i>apIConfig</i> | APLL configuration. Set to NULL if the output port does not go through APLL. |

Definition at line 405 of file Output.c.

**4.6.4.7** T\_osUInt8 IDTOutput\_GetOutputEnable ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nOutN* )

Get enable/disable status of output port. Consult data sheet for supported ports.

**Parameters**

|              |                                  |
|--------------|----------------------------------|
| <i>hdlr</i>  | Device driver handler            |
| <i>nOutN</i> | Output port that want to be used |

**Returns**

nEnable

- 0=Disable port
- 1=Enable port

Definition at line 653 of file Output.c.

**4.6.4.8** T\_osUInt8 IDTOutput\_GetOutputInvert ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nOutN* )

Get output signal inversion status.

**Parameters**

|              |                                  |
|--------------|----------------------------------|
| <i>hdlr</i>  | Device driver handler            |
| <i>nOutN</i> | Output port that want to be used |

**Returns**

0-do no invert, 1-invert output signal

Definition at line 555 of file Output.c.

4.6.4.9 void IDTOutput\_SetFrameSync ( T\_idtDrvHdlr *hdlr*, T\_frameSync *frameSync*, T\_osUInt8 *enable*, T\_osUInt8 *invert*, T\_osUInt8 *pulse* )

Function to control output frame syncs.

#### Parameters

|                  |                                                                                        |
|------------------|----------------------------------------------------------------------------------------|
| <i>hdlr</i>      | Device driver handler                                                                  |
| <i>frameSync</i> | Select between FR and MFR frame syncs                                                  |
| <i>enable</i>    | Enable frame sync                                                                      |
| <i>invert</i>    | Invert output frame sync (0-noninverted, 1-invert)                                     |
| <i>pulse</i>     | Select between clock (50:50 clock cycle) or pulse (dictated by IN3) (0-clock, 1-pulse) |

Definition at line 860 of file Output.c.

4.6.4.10 void IDTOutput\_SetFrameSyncPulseEdge ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *isRisingEdge* )

Function provision trigger for frame sync pulses.

Only valid for frame syncs when provisioned for pulse.

#### Parameters

|                     |                                            |
|---------------------|--------------------------------------------|
| <i>hdlr</i>         | Device driver handler                      |
| <i>isRisingEdge</i> | Frame sync pulse triggered by rising edge. |

Definition at line 962 of file Output.c.

4.6.4.11 void IDTOutput\_SetOutputpulInterface ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nOutN*, T\_outputInterface *outIf* )

Set output port interface type. Consult data sheet for supported interfaces per port.

#### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nOutN</i> | Output port number    |
| <i>outIf</i> | Interface type.       |

Definition at line 706 of file Output.c.

4.6.4.12 void IDTOutput\_SetOutputConfig ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nOutN*, T\_idtDpllInstance *nDpll*, T\_outputPathType *nPath*, T\_osUInt8 *nFactor*, T\_idtApllConfigPerOutput \* *apllConfig* )

Set the configuration of Output port.

#### Parameters

|                   |                                                                            |
|-------------------|----------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                      |
| <i>nOutN</i>      | Output port that want to be used                                           |
| <i>nDpll</i>      | DPLL instance                                                              |
| <i>nPath</i>      | Select a specified path as input of output divider                         |
| <i>nFactor</i>    | Dividen factor of a certain output divider. Look up the table in datasheet |
| <i>apllConfig</i> | APLL configuration                                                         |

Definition at line 182 of file Output.c.

#### 4.6.4.13 void IDTOutput.SetOutputEnable ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nOutN*, T\_osUInt8 *nEnable* )

Enable/Disable output port. Consult data sheet for supported ports.

##### Parameters

|                |                                                                                          |
|----------------|------------------------------------------------------------------------------------------|
| <i>hdlr</i>    | Device driver handler                                                                    |
| <i>nOutN</i>   | Output port that want to be used                                                         |
| <i>nEnable</i> | <ul style="list-style-type: none"><li>• 0=Disable port</li><li>• 1=Enable port</li></ul> |

Definition at line 596 of file Output.c.

#### 4.6.4.14 void IDTOutput.SetOutputInvert ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nOutN*, T\_osUInt8 *invert* )

Set the specified output port to generate a invert signal.

##### Parameters

|               |                                        |
|---------------|----------------------------------------|
| <i>hdlr</i>   | Device driver handler                  |
| <i>nOutN</i>  | Output port that want to be used       |
| <i>invert</i> | 0-do no invert, 1-invert output signal |

Definition at line 506 of file Output.c.

## 4.7 Register definitions

### Enumerations

```

• enum {
    REG_ID_LOW = 0x00, REG_ID_HIGH = 0x01, REG_MPU_PIN_STS = 0x02, REG_NOMINAL_FREQ_LO-
    W_CNFG = 0x04,
    REG_NOMINAL_FREQ_MID_CNFG = 0x05, REG_NOMINAL_FREQ_HIGH_CNFG = 0x06, REG_T4_T0_
    SEL_CNFG = 0x07, REG_PHASE_ALARM_TIME_OUT_CNFG = 0x08,
    REG_INPUT_MODE_CNFG = 0x09, REG_DIFFERENTIAL_IN_OUT_CNFG = 0x0A, REG_MON_SW_HS-
    _CNFG = 0x0B, REG_INTERRUPT_CNFG = 0x0C,
    REG_INTERRUPTS1_STS = 0x0D, REG_INTERRUPTS2_STS = 0x0E, REG_INTERRUPTS3_STS = 0x0F,
    REG_INTERRUPTS1_MASK_CNFG = 0x10,
    REG_INTERRUPTS2_MASK_CNFG = 0x11, REG_INTERRUPTS3_MASK_CNFG = 0x12, REG_MS_SL_
    CTRL_CNFG = 0x13, REG_IN1_CNFG = 0x14,
    REG_IN2_CNFG = 0x15, REG_IN3_CNFG = 0x16, REG_IN4_CNFG = 0x17, REG_IN5_IN6_HF_DIV_CN-
    FG = 0x18,
    REG_IN5_CNFG = 0x19, REG_IN6_CNFG = 0x1A, REG_IN7_CNFG = 0x1B, REG_IN8_CNFG = 0x1C,
    REG_IN9_CNFG = 0x1D, REG_IN10_CNFG = 0x1E, REG_IN11_CNFG = 0x1F, REG_IN12_CNFG = 0x20,
    REG_IN13_CNFG = 0x21, REG_IN14_CNFG = 0x22, REG_PRE_DIV_CH_CNFG = 0x23, REG_PRE_DIV-
    N_LOW_CNFG = 0x24,
    REG_PRE_DIVN_HIGH_CNFG = 0x25, REG_IN1_IN2_SEL_PRIORITY_CNFG = 0x26, REG_IN3_IN4_S-
    EL_PRIORITY_CNFG = 0x27, REG_IN5_IN6_SEL_PRIORITY_CNFG = 0x28,
    REG_IN7_IN8_SEL_PRIORITY_CNFG = 0x29, REG_IN9_IN10_SEL_PRIORITY_CNFG = 0x2A, REG_I-
    N11_IN12_SEL_PRIORITY_CNFG = 0x2B, REG_IN13_IN14_SEL_PRIORITY_CNFG = 0x2C,
    REG_PAGE_POINTER_CNFG = 0x2D, REG_FREQ_MON_FACTOR_CNFG = 0x2E, REG_HARD_FREQ-
    _MON_THRESHOLD_CNFG = 0x2F, REG_SOFT_FREQ_MON_THRESHOLD_CNFG = 0x30,
    REG_UPPER_THRESHOLD_0_CNFG = 0x31, REG_LOWER_THRESHOLD_0_CNFG = 0x32, REG_BU-
    CKET_SIZE_0_CNFG = 0x33, REG_DECAY_RATE_0_CNFG = 0x34,
    REG_UPPER_THRESHOLD_1_CNFG = 0x35, REG_LOWER_THRESHOLD_1_CNFG = 0x36, REG_BU-
    CKET_SIZE_1_CNFG = 0x37, REG_DECAY_RATE_1_CNFG = 0x38,
    REG_UPPER_THRESHOLD_2_CNFG = 0x39, REG_LOWER_THRESHOLD_2_CNFG = 0x3A, REG_BU-
    CKET_SIZE_2_CNFG = 0x3B, REG_DECAY_RATE_2_CNFG = 0x3C,
    REG_UPPER_THRESHOLD_3_CNFG = 0x3D, REG_LOWER_THRESHOLD_3_CNFG = 0x3E, REG_BU-
    CKET_SIZE_3_CNFG = 0x3F, REG_DECAY_RATE_3_CNFG = 0x40,
    REG_IN_FREQ_READ_CH_CNFG = 0x41, REG_IN_FREQ_READ_STS = 0x42, REG_IN1_IN2_STS =
    0x43, REG_IN3_IN4_STS = 0x44,
    REG_IN5_IN6_STS = 0x45, REG_IN7_IN8_STS = 0x46, REG_IN9_IN10_STS = 0x47, REG_IN11_IN12_S-
    TS = 0x48,
    REG_IN13_IN14_STS = 0x49, REG_INPUT_VALID1_STS = 0x4A, REG_INPUT_VALID2_STS = 0x4B,
    REG_REMOTE_INPUT_VALID1_CNFG = 0x4C,
    REG_REMOTE_INPUT_VALID2_CNFG = 0x4D, REG_PRIORITY_TABLE1_STS = 0x4E, REG_PRIORITY-
    _TABLE2_STS = 0x4F, REG_T0_INPUT_SEL_CNFG = 0x50,
    REG_T4_INPUT_SEL_CNFG = 0x51, REG_OPERATING_STS = 0x52, REG_T0_OPERATION_MODE_C-
    NFG = 0x53, REG_T4_OPERATION_MODE_CNFG = 0x54,
    REG_T0_DPLL_APLL_PATH_CNFG = 0x55, REG_T0_DPLL_START_BW_DAMPING_CNFG = 0x56, R-
    EG_T0_DPLL_ACQ_BW_DAMPING_CNFG = 0x57, REG_T0_DPLL_LOCKED_BW_DAMPING_CNFG =
    0x58,
    REG_T0_BW_OVERSHOOT_CNFG = 0x59, REG_PHASE_LOSS_COARSE_LIMIT_CNFG = 0x5A, REG-
    _PHASE_LOSS_FINE_LIMIT_CNFG = 0x5B, REG_T0_HOLDOVER_MODE_CNFG = 0x5C,
    REG_HOLDOVER_FREQ_LOW_CNFG = 0x5D, REG_HOLDOVER_FREQ_MID_CNFG = 0x5E, REG_HO-
    LDOVER_FREQ_HIGH_CNFG = 0x5F, REG_T4_DPLL_APLL_PATH_CNFG = 0x60,
    REG_T4_DPLL_LOCKED_BW_DAMPING_CNFG = 0x61, REG_CURRENT_FREQ_LOW_STS = 0x62,
    REG_CURRENT_FREQ_MID_STS = 0x63, REG_CURRENT_FREQ_HIGH_STS = 0x64,
    REG_DPLL_FREQ_SOFT_LIMIT_CNFG = 0x65, REG_DPLL_FREQ_HARD_LIMIT_LOW_CNFG = 0x66,
    REG_DPLL_FREQ_HARD_LIMIT_MID_CNFG = 0x67, REG_CURRENT_DPLL_PHASE_LOW_STS =
    0x68,
    REG_CURRENT_DPLL_PHASE_HIGH_STS = 0x69, REG_OUT1_FREQ_CNFG = 0x6B, REG_OUT2_FR-

```

```

EQ_CNFG = 0x6C, REG_OUT3_FREQ_CNFG = 0x6D,
REG_OUT4_FREQ_CNFG = 0x6E, REG_OUT5_FREQ_CNFG = 0x6F, REG_OUT6_FREQ_CNFG = 0x70,
REG_OUT7_FREQ_CNFG = 0x71,
REG_OUT8_FREQ_CNFG = 0x72, REG_OUT9_FREQ_CNFG = 0x73, REG_FR_MFR_SYNC_CNFG =
0x74, REG_PHASE_MON_CNFG = 0x78,
REG_PHASE_OFFSET_LOW_CNFG = 0x7A, REG_PHASE_OFFSET_HIGH_CNFG = 0x7B, REG_SYNC-
_MONITOR_CNFG = 0x7C, REG_SYNC_PHASE_CNFG = 0x7D,
REG_PROTECTION_CNFG = 0x7E, REG_MPU_SEL_CNFG = 0x7F, REG_DFS_OFF_CNFG = 0x30, RE-
G_T0_PPS_CNFG = 0x31,
REG_PH_SLOPE_CNFG = 0x32, REG_T0_ICP_CTRL_CNFG = 0x33, REG_T4_ICP_CTRL_CNFG = 0x34,
REG_T4_PPS_CNFG = 0x3E,
BF_ID_A_SIZE = 8, BF_ID_A_MASK = 0xFF, BF_ID_A_LSB = 0, BF_ID_B_SIZE = 8,
BF_ID_B_MASK = 0xFF, BF_ID_B_LSB = 0, BF_MPU_PIN_STS_SIZE = 1, BF_MPU_PIN_STS_MASK =
0x01,
BF_MPU_PIN_STS_LSB = 0, BF_NOMINAL_FREQ_VALUE_A_SIZE = 8, BF_NOMINAL_FREQ_VALUE -
A_MASK = 0xFF, BF_NOMINAL_FREQ_VALUE_A_LSB = 0,
BF_NOMINAL_FREQ_VALUE_B_SIZE = 8, BF_NOMINAL_FREQ_VALUE_B_MASK = 0xFF, BF_NOMIN-
AL_FREQ_VALUE_B_LSB = 0, BF_NOMINAL_FREQ_VALUE_C_SIZE = 8,
BF_NOMINAL_FREQ_VALUE_C_MASK = 0xFF, BF_NOMINAL_FREQ_VALUE_C_LSB = 0, BF_T4_T0_S-
EL_SIZE = 1, BF_T4_T0_SEL_MASK = 0x10,
BF_T4_T0_SEL_LSB = 4, BF_MULTI_FACTOR_SIZE = 2, BF_MULTI_FACTOR_MASK = 0xC0, BF_MUL-
TI_FACTOR_LSB = 6,
BF_TIMEOUT_VALUE_SIZE = 6, BF_TIMEOUT_VALUE_MASK = 0x3F, BF_TIMEOUT_VALUE_LSB = 0,
BF_AUTO_EXT_SYNC_EN_SIZE = 1,
BF_AUTO_EXT_SYNC_EN_MASK = 0x80, BF_AUTO_EXT_SYNC_EN_LSB = 7, BF_EXT_SYNC_EN_SI-
ZE = 1, BF_EXT_SYNC_EN_MASK = 0x40,
BF_EXT_SYNC_EN_LSB = 6, BF_PH_ALARM_TIMEOUT_SIZE = 1, BF_PH_ALARM_TIMEOUT_MASK =
0x20, BF_PH_ALARM_TIMEOUT_LSB = 5,
BF_SYNC_FREQ_SIZE = 2, BF_SYNC_FREQ_MASK = 0x18, BF_SYNC_FREQ_LSB = 3, BF_IN_SONE-
T_SDH_SIZE = 1,
BF_IN_SONET_SDH_MASK = 0x04, BF_IN_SONET_SDH_LSB = 2, BF_MASTER_SLAVE_SIZE = 1, BF-
_MASTER_SLAVE_MASK = 0x02,
BF_MASTER_SLAVE_LSB = 1, BF_REVERTIVE_MODE_SIZE = 1, BF_REVERTIVE_MODE_MASK =
0x01, BF_REVERTIVE_MODE_LSB = 0,
BF_OSC_EDGE_SIZE = 1, BF_OSC_EDGE_MASK = 0x04, BF_OSC_EDGE_LSB = 2, BF_OUT7_PECL -
LVDS_SIZE = 1,
BF_OUT7_PECL_LVDS_MASK = 0x02, BF_OUT7_PECL_LVDS_LSB = 1, BF_OUT6_PECL_LVDS_SIZE
= 1, BF_OUT6_PECL_LVDS_MASK = 0x01,
BF_OUT6_PECL_LVDS_LSB = 0, BF_FREQ_MON_CLK_SIZE = 1, BF_FREQ_MON_CLK_MASK = 0x80,
BF_FREQ_MON_CLK_LSB = 7,
BF_LOS_FLAG_TO_TDO_SIZE = 1, BF_LOS_FLAG_TO_TDO_MASK = 0x40, BF_LOS_FLAG_TO_TDO-
_LSB = 6, BF_ULTR_FAST_SW_SIZE = 1,
BF_ULTR_FAST_SW_MASK = 0x20, BF_ULTR_FAST_SW_LSB = 5, BF_EXT_SW_SIZE = 1, BF_EXT_S-
W_MASK = 0x10,
BF_EXT_SW_LSB = 4, BF_HS_FREQ_SIZE = 1, BF_HS_FREQ_MASK = 0x08, BF_HS_FREQ_LSB = 3,
BF_HS_EN_SIZE = 1, BF_HS_EN_MASK = 0x04, BF_HS_EN_LSB = 2, BF_FREQ_MON_HARD_EN_SIZE
= 1,
BF_FREQ_MON_HARD_EN_MASK = 0x01, BF_FREQ_MON_HARD_EN_LSB = 0, BF_HZ_EN_SIZE = 1,
BF_HZ_EN_MASK = 0x02,
BF_HZ_EN_LSB = 1, BF_INT_POL_SIZE = 1, BF_INT_POL_MASK = 0x01, BF_INT_POL_LSB = 0,
BF_IN8_SIZE = 1, BF_IN8_MASK = 0x80, BF_IN8_LSB = 7, BF_IN7_SIZE = 1,
BF_IN7_MASK = 0x40, BF_IN7_LSB = 6, BF_IN6_SIZE = 1, BF_IN6_MASK = 0x20,
BF_IN6_LSB = 5, BF_IN5_SIZE = 1, BF_IN5_MASK = 0x10, BF_IN5_LSB = 4,
BF_IN4_SIZE = 1, BF_IN4_MASK = 0x08, BF_IN4_LSB = 3, BF_IN3_SIZE = 1,
BF_IN3_MASK = 0x04, BF_IN3_LSB = 2, BF_IN2_SIZE = 1, BF_IN2_MASK = 0x02,
BF_IN2_LSB = 1, BF_IN1_SIZE = 1, BF_IN1_MASK = 0x01, BF_IN1_LSB = 0,
BF_T0_OPERATING_MODE_STS_SIZE = 1, BF_T0_OPERATING_MODE_STS_MASK = 0x80, BF_T0 -

```

```

OPERATING_MODE_STS_LSB = 7, BF_TO_MAIN_REF_FAIL_SIZE = 1,
BF_TO_MAIN_REF_FAIL_MASK = 0x40, BF_TO_MAIN_REF_FAIL_LSB = 6, BF_IN14_SIZE = 1, BF_IN14_
_MASK = 0x20,
BF_IN14_LSB = 5, BF_IN13_SIZE = 1, BF_IN13_MASK = 0x10, BF_IN13_LSB = 4,
BF_IN12_SIZE = 1, BF_IN12_MASK = 0x08, BF_IN12_LSB = 3, BF_IN11_SIZE = 1,
BF_IN11_MASK = 0x04, BF_IN11_LSB = 2, BF_IN10_SIZE = 1, BF_IN10_MASK = 0x02,
BF_IN10_LSB = 1, BF_IN9_SIZE = 1, BF_IN9_MASK = 0x01, BF_IN9_LSB = 0,
BF_EX_SYNC_ALARM_SIZE = 1, BF_EX_SYNC_ALARM_MASK = 0x80, BF_EX_SYNC_ALARM_LSB =
7, BF_T4_STS_SIZE = 1,
BF_T4_STS_MASK = 0x40, BF_T4_STS_LSB = 6, BF_INPUT_TO_T4_SIZE = 1, BF_INPUT_TO_T4_MA-
SK = 0x10,
BF_INPUT_TO_T4_LSB = 4, BF_AMI2_VIOL_SIZE = 1, BF_AMI2_VIOL_MASK = 0x08, BF_AMI2_VIOL_-
LSB = 3,
BF_AMI2_LOS_SIZE = 1, BF_AMI2_LOS_MASK = 0x04, BF_AMI2_LOS_LSB = 2, BF_AMI1_VIOL_SIZE =
1,
BF_AMI1_VIOL_MASK = 0x02, BF_AMI1_VIOL_LSB = 1, BF_AMI1_LOS_SIZE = 1, BF_AMI1_LOS_MASK
= 0x01,
BF_AMI1_LOS_LSB = 0, BF_PPS_FAST_FREQ_LOCK_EN_SIZE = 1, BF_PPS_FAST_FREQ_LOCK_EN_-
_MASK = 0x10, BF_PPS_FAST_FREQ_LOCK_EN_LSB = 4,
BF_PPS_FAST_PH_LOCK_EN_SIZE = 1, BF_PPS_FAST_PH_LOCK_EN_MASK = 0x08, BF_PPS_FAST_-
_PH_LOCK_EN_LSB = 3, BF_MS_SL_CTRL_SIZE = 1,
BF_MS_SL_CTRL_MASK = 0x01, BF_MS_SL_CTRL_LSB = 0, BF_400HZ_SEL_SIZE = 1, BF_400HZ_SE-
L_MASK = 0x40,
BF_400HZ_SEL_LSB = 6, BF_BUCKET_SEL_SIZE = 2, BF_BUCKET_SEL_MASK = 0x30, BF_BUCKET_-
SEL_LSB = 4,
BF_IN_FREQ_SIZE = 4, BF_IN_FREQ_MASK = 0x0F, BF_IN_FREQ_LSB = 0, BF_DIRECT_DIV_SIZE = 1,
BF_DIRECT_DIV_MASK = 0x80, BF_DIRECT_DIV_LSB = 7, BF_LOCK_8K_SIZE = 1, BF_LOCK_8K_MA-
SK = 0x40,
BF_LOCK_8K_LSB = 6, BF_IN6_DIV_SIZE = 2, BF_IN6_DIV_MASK = 0xC0, BF_IN6_DIV_LSB = 6,
BF_IN5_DIV_SIZE = 2, BF_IN5_DIV_MASK = 0x03, BF_IN5_DIV_LSB = 0, BF_PRE_DIV_CH_VALUE_SI-
ZE = 4,
BF_PRE_DIV_CH_VALUE_MASK = 0x0F, BF_PRE_DIV_CH_VALUE_LSB = 0, BF_PRE_DIVN_VALUE_-
A_SIZE = 8, BF_PRE_DIVN_VALUE_A_MASK = 0xFF,
BF_PRE_DIVN_VALUE_A_LSB = 0, BF_PRE_DIVN_VALUE_B_SIZE = 7, BF_PRE_DIVN_VALUE_B_MA-
SK = 0x7F, BF_PRE_DIVN_VALUE_B_LSB = 0,
BF_IN2_SEL_PRIORITY_SIZE = 4, BF_IN2_SEL_PRIORITY_MASK = 0xF0, BF_IN2_SEL_PRIORITY_LS-
B = 4, BF_IN1_SEL_PRIORITY_SIZE = 4,
BF_IN1_SEL_PRIORITY_MASK = 0x0F, BF_IN1_SEL_PRIORITY_LSB = 0, BF_IN4_SEL_PRIORITY_SIZE
= 4, BF_IN4_SEL_PRIORITY_MASK = 0xF0,
BF_IN4_SEL_PRIORITY_LSB = 4, BF_IN3_SEL_PRIORITY_SIZE = 4, BF_IN3_SEL_PRIORITY_MASK =
0x0F, BF_IN3_SEL_PRIORITY_LSB = 0,
BF_IN6_SEL_PRIORITY_SIZE = 4, BF_IN6_SEL_PRIORITY_MASK = 0xF0, BF_IN6_SEL_PRIORITY_LS-
B = 4, BF_IN5_SEL_PRIORITY_SIZE = 4,
BF_IN5_SEL_PRIORITY_MASK = 0x0F, BF_IN5_SEL_PRIORITY_LSB = 0, BF_IN8_SEL_PRIORITY_SIZE
= 4, BF_IN8_SEL_PRIORITY_MASK = 0xF0,
BF_IN8_SEL_PRIORITY_LSB = 4, BF_IN7_SEL_PRIORITY_SIZE = 4, BF_IN7_SEL_PRIORITY_MASK =
0x0F, BF_IN7_SEL_PRIORITY_LSB = 0,
BF_IN10_SEL_PRIORITY_SIZE = 4, BF_IN10_SEL_PRIORITY_MASK = 0xF0, BF_IN10_SEL_PRIORITY_-
LSB = 4, BF_IN9_SEL_PRIORITY_SIZE = 4,
BF_IN9_SEL_PRIORITY_MASK = 0x0F, BF_IN9_SEL_PRIORITY_LSB = 0, BF_IN12_SEL_PRIORITY_SI-
ZE = 4, BF_IN12_SEL_PRIORITY_MASK = 0xF0,
BF_IN12_SEL_PRIORITY_LSB = 4, BF_IN11_SEL_PRIORITY_SIZE = 4, BF_IN11_SEL_PRIORITY_MAS-
K = 0x0F, BF_IN11_SEL_PRIORITY_LSB = 0,
BF_IN14_SEL_PRIORITY_SIZE = 4, BF_IN14_SEL_PRIORITY_MASK = 0xF0, BF_IN14_SEL_PRIORITY_-
LSB = 4, BF_IN13_SEL_PRIORITY_SIZE = 4,
BF_IN13_SEL_PRIORITY_MASK = 0x0F, BF_IN13_SEL_PRIORITY_LSB = 0, BF_PAGE_POINTER_SIZE
= 1, BF_PAGE_POINTER_MASK = 0x01,
BF_PAGE_POINTER_LSB = 0, BF_FREQ_MON_FACTOR_SIZE = 4, BF_FREQ_MON_FACTOR_MASK =

```

```

0x0F, BF_FREQ_MON_FACTOR_LSB = 0,
BF_HARD_FREQ_MON_THRESHOLD_A_SIZE = 4, BF_HARD_FREQ_MON_THRESHOLD_A_MASK =
0xF0, BF_HARD_FREQ_MON_THRESHOLD_A_LSB = 4, BF_HARD_FREQ_MON_THRESHOLD_B_SIZE
= 4,
BF_HARD_FREQ_MON_THRESHOLD_B_MASK = 0x0F, BF_HARD_FREQ_MON_THRESHOLD_B_LSB
= 0, BF_SOFT_FREQ_MON_THRESHOLD_A_SIZE = 4, BF_SOFT_FREQ_MON_THRESHOLD_A_MASK
= 0xF0,
BF_SOFT_FREQ_MON_THRESHOLD_A_LSB = 4, BF_SOFT_FREQ_MON_THRESHOLD_B_SIZE = 4,
BF_SOFT_FREQ_MON_THRESHOLD_B_MASK = 0x0F, BF_SOFT_FREQ_MON_THRESHOLD_B_LSB =
0,
BF_UPPER_THRESHOLD_DATA_SIZE = 8, BF_UPPER_THRESHOLD_DATA_MASK = 0xFF, BF_UPPE-
R_THRESHOLD_DATA_LSB = 0, BF_LOWER_THRESHOLD_DATA_SIZE = 8,
BF_LOWER_THRESHOLD_DATA_MASK = 0xFF, BF_LOWER_THRESHOLD_DATA_LSB = 0, BF_BUCK-
ET_SIZE_DATA_SIZE = 8, BF_BUCKET_SIZE_DATA_MASK = 0xFF,
BF_BUCKET_SIZE_DATA_LSB = 0, BF_DECAY_RATE_DATA_SIZE = 2, BF_DECAY_RATE_DATA_MA-
SK = 0x03, BF_DECAY_RATE_DATA_LSB = 0,
BF_IN_FREQ_READ_CH_SIZE = 4, BF_IN_FREQ_READ_CH_MASK = 0x0F, BF_IN_FREQ_READ_CH_-
LSB = 0, BF_IN_FREQ_VALUE_SIZE = 8,
BF_IN_FREQ_VALUE_MASK = 0xFF, BF_IN_FREQ_VALUE_LSB = 0, BF_IN2_FREQ_SOFT_ALARM_SI-
ZE = 1, BF_IN2_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN2_FREQ_SOFT_ALARM_LSB = 7, BF_IN2_FREQ_HARD_ALARM_SIZE = 1, BF_IN2_FREQ_HAR-
D_ALARM_MASK = 0x40, BF_IN2_FREQ_HARD_ALARM_LSB = 6,
BF_IN2_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN2_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN2_NO_-
ACTIVITY_ALARM_LSB = 5, BF_IN2_PH_LOCK_ALARM_SIZE = 1,
BF_IN2_PH_LOCK_ALARM_MASK = 0x10, BF_IN2_PH_LOCK_ALARM_LSB = 4, BF_IN1_FREQ_SOFT-
_ALARM_SIZE = 1, BF_IN1_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN1_FREQ_SOFT_ALARM_LSB = 3, BF_IN1_FREQ_HARD_ALARM_SIZE = 1, BF_IN1_FREQ_HAR-
D_ALARM_MASK = 0x04, BF_IN1_FREQ_HARD_ALARM_LSB = 2,
BF_IN1_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN1_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN1_NO_-
ACTIVITY_ALARM_LSB = 1, BF_IN1_PH_LOCK_ALARM_SIZE = 1,
BF_IN1_PH_LOCK_ALARM_MASK = 0x01, BF_IN1_PH_LOCK_ALARM_LSB = 0, BF_IN4_FREQ_SOFT-
_ALARM_SIZE = 1, BF_IN4_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN4_FREQ_SOFT_ALARM_LSB = 7, BF_IN4_FREQ_HARD_ALARM_SIZE = 1, BF_IN4_FREQ_HAR-
D_ALARM_MASK = 0x40, BF_IN4_FREQ_HARD_ALARM_LSB = 6,
BF_IN4_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN4_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN4_NO_-
ACTIVITY_ALARM_LSB = 5, BF_IN4_PH_LOCK_ALARM_SIZE = 1,
BF_IN4_PH_LOCK_ALARM_MASK = 0x10, BF_IN4_PH_LOCK_ALARM_LSB = 4, BF_IN3_FREQ_SOFT-
_ALARM_SIZE = 1, BF_IN3_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN3_FREQ_SOFT_ALARM_LSB = 3, BF_IN3_FREQ_HARD_ALARM_SIZE = 1, BF_IN3_FREQ_HAR-
D_ALARM_MASK = 0x04, BF_IN3_FREQ_HARD_ALARM_LSB = 2,
BF_IN3_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN3_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN3_NO_-
ACTIVITY_ALARM_LSB = 1, BF_IN3_PH_LOCK_ALARM_SIZE = 1,
BF_IN3_PH_LOCK_ALARM_MASK = 0x01, BF_IN3_PH_LOCK_ALARM_LSB = 0, BF_IN6_FREQ_SOFT-
_ALARM_SIZE = 1, BF_IN6_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN6_FREQ_SOFT_ALARM_LSB = 7, BF_IN6_FREQ_HARD_ALARM_SIZE = 1, BF_IN6_FREQ_HAR-
D_ALARM_MASK = 0x40, BF_IN6_FREQ_HARD_ALARM_LSB = 6,
BF_IN6_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN6_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN6_NO_-
ACTIVITY_ALARM_LSB = 5, BF_IN6_PH_LOCK_ALARM_SIZE = 1,
BF_IN6_PH_LOCK_ALARM_MASK = 0x10, BF_IN6_PH_LOCK_ALARM_LSB = 4, BF_IN5_FREQ_SOFT-
_ALARM_SIZE = 1, BF_IN5_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN5_FREQ_SOFT_ALARM_LSB = 3, BF_IN5_FREQ_HARD_ALARM_SIZE = 1, BF_IN5_FREQ_HAR-
D_ALARM_MASK = 0x04, BF_IN5_FREQ_HARD_ALARM_LSB = 2,
BF_IN5_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN5_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN5_NO_-
ACTIVITY_ALARM_LSB = 1, BF_IN5_PH_LOCK_ALARM_SIZE = 1,
BF_IN5_PH_LOCK_ALARM_MASK = 0x01, BF_IN5_PH_LOCK_ALARM_LSB = 0, BF_IN8_FREQ_SOFT-
_ALARM_SIZE = 1, BF_IN8_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN8_FREQ_SOFT_ALARM_LSB = 7, BF_IN8_FREQ_HARD_ALARM_SIZE = 1, BF_IN8_FREQ_HAR-

```

```

D_ALARM_MASK = 0x40, BF_IN8_FREQ_HARD_ALARM_LSB = 6,
BF_IN8_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN8_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN8_NO_
ACTIVITY_ALARM_LSB = 5, BF_IN8_PH_LOCK_ALARM_SIZE = 1,
BF_IN8_PH_LOCK_ALARM_MASK = 0x10, BF_IN8_PH_LOCK_ALARM_LSB = 4, BF_IN7_FREQ_SOFT_
_ALARM_SIZE = 1, BF_IN7_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN7_FREQ_SOFT_ALARM_LSB = 3, BF_IN7_FREQ_HARD_ALARM_SIZE = 1, BF_IN7_FREQ_HAR_
D_ALARM_MASK = 0x04, BF_IN7_FREQ_HARD_ALARM_LSB = 2,
BF_IN7_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN7_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN7_NO_
ACTIVITY_ALARM_LSB = 1, BF_IN7_PH_LOCK_ALARM_SIZE = 1,
BF_IN7_PH_LOCK_ALARM_MASK = 0x01, BF_IN7_PH_LOCK_ALARM_LSB = 0, BF_IN10_FREQ_SOF_
T_ALARM_SIZE = 1, BF_IN10_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN10_FREQ_SOFT_ALARM_LSB = 7, BF_IN10_FREQ_HARD_ALARM_SIZE = 1, BF_IN10_FREQ_H_
ARD_ALARM_MASK = 0x40, BF_IN10_FREQ_HARD_ALARM_LSB = 6,
BF_IN10_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN10_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN10_
NO_ACTIVITY_ALARM_LSB = 5, BF_IN10_PH_LOCK_ALARM_SIZE = 1,
BF_IN10_PH_LOCK_ALARM_MASK = 0x10, BF_IN10_PH_LOCK_ALARM_LSB = 4, BF_IN9_FREQ_SO_
FT_ALARM_SIZE = 1, BF_IN9_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN9_FREQ_SOFT_ALARM_LSB = 3, BF_IN9_FREQ_HARD_ALARM_SIZE = 1, BF_IN9_FREQ_HAR_
D_ALARM_MASK = 0x04, BF_IN9_FREQ_HARD_ALARM_LSB = 2,
BF_IN9_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN9_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN9_NO_
ACTIVITY_ALARM_LSB = 1, BF_IN9_PH_LOCK_ALARM_SIZE = 1,
BF_IN9_PH_LOCK_ALARM_MASK = 0x01, BF_IN9_PH_LOCK_ALARM_LSB = 0, BF_IN12_FREQ_SOF_
T_ALARM_SIZE = 1, BF_IN12_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN12_FREQ_SOFT_ALARM_LSB = 7, BF_IN12_FREQ_HARD_ALARM_SIZE = 1, BF_IN12_FREQ_H_
ARD_ALARM_MASK = 0x40, BF_IN12_FREQ_HARD_ALARM_LSB = 6,
BF_IN12_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN12_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN12_
NO_ACTIVITY_ALARM_LSB = 5, BF_IN12_PH_LOCK_ALARM_SIZE = 1,
BF_IN12_PH_LOCK_ALARM_MASK = 0x10, BF_IN12_PH_LOCK_ALARM_LSB = 4, BF_IN11_FREQ_S_
OFT_ALARM_SIZE = 1, BF_IN11_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN11_FREQ_SOFT_ALARM_LSB = 3, BF_IN11_FREQ_HARD_ALARM_SIZE = 1, BF_IN11_FREQ_H_
ARD_ALARM_MASK = 0x04, BF_IN11_FREQ_HARD_ALARM_LSB = 2,
BF_IN11_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN11_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN11_
NO_ACTIVITY_ALARM_LSB = 1, BF_IN11_PH_LOCK_ALARM_SIZE = 1,
BF_IN11_PH_LOCK_ALARM_MASK = 0x01, BF_IN11_PH_LOCK_ALARM_LSB = 0, BF_IN14_FREQ_S_
OFT_ALARM_SIZE = 1, BF_IN14_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN14_FREQ_SOFT_ALARM_LSB = 7, BF_IN14_FREQ_HARD_ALARM_SIZE = 1, BF_IN14_FREQ_H_
ARD_ALARM_MASK = 0x40, BF_IN14_FREQ_HARD_ALARM_LSB = 6,
BF_IN14_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN14_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN14_
NO_ACTIVITY_ALARM_LSB = 5, BF_IN14_PH_LOCK_ALARM_SIZE = 1,
BF_IN14_PH_LOCK_ALARM_MASK = 0x10, BF_IN14_PH_LOCK_ALARM_LSB = 4, BF_IN13_FREQ_S_
OFT_ALARM_SIZE = 1, BF_IN13_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN13_FREQ_SOFT_ALARM_LSB = 3, BF_IN13_FREQ_HARD_ALARM_SIZE = 1, BF_IN13_FREQ_H_
ARD_ALARM_MASK = 0x04, BF_IN13_FREQ_HARD_ALARM_LSB = 2,
BF_IN13_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN13_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN13_
NO_ACTIVITY_ALARM_LSB = 1, BF_IN13_PH_LOCK_ALARM_SIZE = 1,
BF_IN13_PH_LOCK_ALARM_MASK = 0x01, BF_IN13_PH_LOCK_ALARM_LSB = 0, BF_HIGHEST_PRI_
ORITY_VALIDATED_SIZE = 4, BF_HIGHEST_PRIORITY_VALIDATED_MASK = 0xF0,
BF_HIGHEST_PRIORITY_VALIDATED_LSB = 4, BF_CURRENTLY_SELECTED_INPUTS_SIZE = 4, BF_
CURRENTLY_SELECTED_INPUTS_MASK = 0x0F, BF_CURRENTLY_SELECTED_INPUTS_LSB = 0,
BF_THIRD_HIGHEST_PRIORITY_VALIDATED_SIZE = 4, BF_THIRD_HIGHEST_PRIORITY_VALIDATE_
D_MASK = 0xF0, BF_THIRD_HIGHEST_PRIORITY_VALIDATED_LSB = 4, BF_SECOND_HIGHEST_PRI_
ORITY_VALIDATED_SIZE = 4,
BF_SECOND_HIGHEST_PRIORITY_VALIDATED_MASK = 0x0F, BF_SECOND_HIGHEST_PRIORITY_V_
ALIDATED_LSB = 0, BF_T0_INPUT_SEL_SIZE = 4, BF_T0_INPUT_SEL_MASK = 0x0F,
BF_T0_INPUT_SEL_LSB = 0, BF_T4_LOCK_T0_SIZE = 1, BF_T4_LOCK_T0_MASK = 0x40, BF_T4_LOC_
K_T0_LSB = 6,
BF_T0_FOR_T4_SIZE = 1, BF_T0_FOR_T4_MASK = 0x20, BF_T0_FOR_T4_LSB = 5, BF_T4_TEST_T0_

```

```

PH_SIZE = 1,
BF_T4_TEST_T0_PH_MASK = 0x10, BF_T4_TEST_T0_PH_LSB = 4, BF_T4_INPUT_SEL_SIZE = 4, BF_T4_INPUT_SEL_MASK = 0x0F,
BF_T4_INPUT_SEL_LSB = 0, BF_EX_SYNC_ALARM_MON_SIZE = 1, BF_EX_SYNC_ALARM_MON_MASK = 0x80, BF_EX_SYNC_ALARM_MON_LSB = 7,
BF_T4_DPLL_LOCK_SIZE = 1, BF_T4_DPLL_LOCK_MASK = 0x40, BF_T4_DPLL_LOCK_LSB = 6, BF_T0_DPLL_SOFT_FREQ_ALARM_SIZE = 1,
BF_T0_DPLL_SOFT_FREQ_ALARM_MASK = 0x20, BF_T0_DPLL_SOFT_FREQ_ALARM_LSB = 5, BF_T4_DPLL_SOFT_FREQ_ALARM_SIZE = 1, BF_T4_DPLL_SOFT_FREQ_ALARM_MASK = 0x10,
BF_T4_DPLL_SOFT_FREQ_ALARM_LSB = 4, BF_T0_DPLL_LOCK_SIZE = 1, BF_T0_DPLL_LOCK_MASK = 0x08, BF_T0_DPLL_LOCK_LSB = 3,
BF_T0_DPLL_OPERATING_MODE_SIZE = 3, BF_T0_DPLL_OPERATING_MODE_MASK = 0x07, BF_T0_DPLL_OPERATING_MODE_LSB = 0, BF_T0_WDCO_SIZE = 1,
BF_T0_WDCO_MASK = 0x08, BF_T0_WDCO_LSB = 3, BF_T0_OPERATING_MODE_SIZE = 3, BF_T0_OPERATING_MODE_MASK = 0x07,
BF_T0_OPERATING_MODE_LSB = 0, BF_T4_WDCO_SIZE = 1, BF_T4_WDCO_MASK = 0x08, BF_T4_WDCO_LSB = 3,
BF_T4_OPERATING_MODE_SIZE = 3, BF_T4_OPERATING_MODE_MASK = 0x07, BF_T4_OPERATING_MODE_LSB = 0, BF_T0_APLL_PATH_SIZE = 4,
BF_T0_APLL_PATH_MASK = 0xF0, BF_T0_APLL_PATH_LSB = 4, BF_T0_GSM_OBSAI_16E1_16T1_SEL_SIZE = 2, BF_T0_GSM_OBSAI_16E1_16T1_SEL_MASK = 0x0C,
BF_T0_GSM_OBSAI_16E1_16T1_SEL_LSB = 2, BF_T0_12E1_GPS_E3_T3_SEL_SIZE = 2, BF_T0_12E1_GPS_E3_T3_SEL_MASK = 0x03, BF_T0_12E1_GPS_E3_T3_SEL_LSB = 0,
BF_T0_DPLL_START_DAMPING_SIZE = 3, BF_T0_DPLL_START_DAMPING_MASK = 0xE0, BF_T0_DPLL_START_DAMPING_LSB = 5, BF_T0_DPLL_START_BW_SIZE = 5,
BF_T0_DPLL_START_BW_MASK = 0x1F, BF_T0_DPLL_START_BW_LSB = 0, BF_T0_DPLL_ACQ_DAMPING_SIZE = 3, BF_T0_DPLL_ACQ_DAMPING_MASK = 0xE0,
BF_T0_DPLL_ACQ_DAMPING_LSB = 5, BF_T0_DPLL_ACQ_BW_SIZE = 5, BF_T0_DPLL_ACQ_BW_MASK = 0x1F, BF_T0_DPLL_ACQ_BW_LSB = 0,
BF_T0_DPLL_LOCKED_DAMPING_SIZE = 3, BF_T0_DPLL_LOCKED_DAMPING_MASK = 0xE0, BF_T0_DPLL_LOCKED_DAMPING_LSB = 5, BF_T0_DPLL_LOCKED_BW_SIZE = 5,
BF_T0_DPLL_LOCKED_BW_MASK = 0x1F, BF_T0_DPLL_LOCKED_BW_LSB = 0, BF_AUTO_BW_SEL_SIZE = 1, BF_AUTO_BW_SEL_MASK = 0x80,
BF_AUTO_BW_SEL_LSB = 7, BF_T0_LIMIT_SIZE = 1, BF_T0_LIMIT_MASK = 0x08, BF_T0_LIMIT_LSB = 3,
BF_COARSE_PH_LOS_LIMT_EN_SIZE = 1, BF_COARSE_PH_LOS_LIMT_EN_MASK = 0x80, BF_COARSE_PH_LOS_LIMT_EN_LSB = 7, BF_WIDE_EN_SIZE = 1,
BF_WIDE_EN_MASK = 0x40, BF_WIDE_EN_LSB = 6, BF_MULTI_PH_APP_SIZE = 1, BF_MULTI_PH_APP_MASK = 0x20,
BF_MULTI_PH_APP_LSB = 5, BF_MULTI_PH_8K_4K_2K_EN_SIZE = 1, BF_MULTI_PH_8K_4K_2K_EN_MASK = 0x10, BF_MULTI_PH_8K_4K_2K_EN_LSB = 4,
BF_PH_LOS_COARSE_LIMT_SIZE = 4, BF_PH_LOS_COARSE_LIMT_MASK = 0x0F, BF_PH_LOS_COARSE_LIMT_LSB = 0, BF_FINE_PH_LOS_LIMT_EN_SIZE = 1,
BF_FINE_PH_LOS_LIMT_EN_MASK = 0x80, BF_FINE_PH_LOS_LIMT_EN_LSB = 7, BF_FAST_LOS_SW_SIZE = 1, BF_FAST_LOS_SW_MASK = 0x40,
BF_FAST_LOS_SW_LSB = 6, BF_PH_LOS_FINE_LIMT_SIZE = 3, BF_PH_LOS_FINE_LIMT_MASK = 0x07, BF_PH_LOS_FINE_LIMT_LSB = 0,
BF_MAN_HOLDOVER_SIZE = 1, BF_MAN_HOLDOVER_MASK = 0x80, BF_MAN_HOLDOVER_LSB = 7, BF_AUTO_AVG_SIZE = 1,
BF_AUTO_AVG_MASK = 0x40, BF_AUTO_AVG_LSB = 6, BF_FAST_AVG_SIZE = 1, BF_FAST_AVG_MASK = 0x20,
BF_FAST_AVG_LSB = 5, BF_READ_AVG_SIZE = 1, BF_READ_AVG_MASK = 0x10, BF_READ_AVG_LSB = 4,
BF_TEMP_HOLDOVER_MODE_SIZE = 2, BF_TEMP_HOLDOVER_MODE_MASK = 0x0C, BF_TEMP_HOLDOVER_MODE_LSB = 2, BF_HOLDOVER_FREQ_A_SIZE = 8,
BF_HOLDOVER_FREQ_A_MASK = 0xFF, BF_HOLDOVER_FREQ_A_LSB = 0, BF_HOLDOVER_FREQ_B_SIZE = 8, BF_HOLDOVER_FREQ_B_MASK = 0xFF,
BF_HOLDOVER_FREQ_B_LSB = 0, BF_HOLDOVER_FREQ_C_SIZE = 8, BF_HOLDOVER_FREQ_C_MASK = 0xFF, BF_HOLDOVER_FREQ_C_LSB = 0

```

```

ASK = 0xFF, BF_HOLD OVER_FREQ_C_LSB = 0,
BF_T4_APLL_PATH_SIZE = 4, BF_T4_APLL_PATH_MASK = 0xF0, BF_T4_APLL_PATH_LSB = 4, BF_
T4_GSM_OBSAI_16E1_16T1_SEL_SIZE = 2,
BF_T4_GSM_OBSAI_16E1_16T1_SEL_MASK = 0x0C, BF_T4_GSM_OBSAI_16E1_16T1_SEL_LSB = 2,
BF_T4_12E1_GPS_E3_T3_SEL_SIZE = 2, BF_T4_12E1_GPS_E3_T3_SEL_MASK = 0x03,
BF_T4_12E1_GPS_E3_T3_SEL_LSB = 0, BF_T4_DPLL_LOCKED_DAMPING_SIZE = 3, BF_T4_DPLL_L-
OCKED_DAMPING_MASK = 0xE0, BF_T4_DPLL_LOCKED_DAMPING_LSB = 5,
BF_T4_DPLL_LOCKED_BW_SIZE = 2, BF_T4_DPLL_LOCKED_BW_MASK = 0x03, BF_T4_DPLL_LOCK-
ED_BW_LSB = 0, BF_CURRENT_DPLL_FREQ_A_SIZE = 8,
BF_CURRENT_DPLL_FREQ_A_MASK = 0xFF, BF_CURRENT_DPLL_FREQ_A_LSB = 0, BF_CURRENT-
_DPLL_FREQ_B_SIZE = 8, BF_CURRENT_DPLL_FREQ_B_MASK = 0xFF,
BF_CURRENT_DPLL_FREQ_B_LSB = 0, BF_CURRENT_DPLL_FREQ_C_SIZE = 8, BF_CURRENT_DP-
LL_FREQ_C_MASK = 0xFF, BF_CURRENT_DPLL_FREQ_C_LSB = 0,
BF_FREQ_LIMT_PH_LOS_SIZE = 1, BF_FREQ_LIMT_PH_LOS_MASK = 0x80, BF_FREQ_LIMT_PH_LO-
S_LSB = 7, BF_DPLL_FREQ_SOFT_LIMT_SIZE = 7,
BF_DPLL_FREQ_SOFT_LIMT_MASK = 0x7F, BF_DPLL_FREQ_SOFT_LIMT_LSB = 0, BF_DPLL_FREQ_-
HARD_LIMT_A_SIZE = 8, BF_DPLL_FREQ_HARD_LIMT_A_MASK = 0xFF,
BF_DPLL_FREQ_HARD_LIMT_A_LSB = 0, BF_DPLL_FREQ_HARD_LIMT_B_SIZE = 8, BF_DPLL_FRE-
Q_HARD_LIMT_B_MASK = 0xFF, BF_DPLL_FREQ_HARD_LIMT_B_LSB = 0,
BF_CURRENT_PH_DATA_A_SIZE = 8, BF_CURRENT_PH_DATA_A_MASK = 0xFF, BF_CURRENT_PH-
_DATA_A_LSB = 0, BF_CURRENT_PH_DATA_B_SIZE = 8,
BF_CURRENT_PH_DATA_B_MASK = 0xFF, BF_CURRENT_PH_DATA_B_LSB = 0, BF_OUT1_PATH_S-
EL_SIZE = 4, BF_OUT1_PATH_SEL_MASK = 0xF0,
BF_OUT1_PATH_SEL_LSB = 4, BF_OUT1_DIVIDER_SIZE = 4, BF_OUT1_DIVIDER_MASK = 0x0F, BF_-
OUT1_DIVIDER_LSB = 0,
BF_OUT2_PATH_SEL_SIZE = 4, BF_OUT2_PATH_SEL_MASK = 0xF0, BF_OUT2_PATH_SEL_LSB = 4,
BF_OUT2_DIVIDER_SIZE = 4,
BF_OUT2_DIVIDER_MASK = 0x0F, BF_OUT2_DIVIDER_LSB = 0, BF_OUT3_PATH_SEL_SIZE = 4, BF_-
OUT3_PATH_SEL_MASK = 0xF0,
BF_OUT3_PATH_SEL_LSB = 4, BF_OUT3_DIVIDER_SIZE = 4, BF_OUT3_DIVIDER_MASK = 0x0F, BF_-
OUT3_DIVIDER_LSB = 0,
BF_OUT4_PATH_SEL_SIZE = 4, BF_OUT4_PATH_SEL_MASK = 0xF0, BF_OUT4_PATH_SEL_LSB = 4,
BF_OUT4_DIVIDER_SIZE = 4,
BF_OUT4_DIVIDER_MASK = 0x0F, BF_OUT4_DIVIDER_LSB = 0, BF_OUT5_PATH_SEL_SIZE = 4, BF_-
OUT5_PATH_SEL_MASK = 0xF0,
BF_OUT5_PATH_SEL_LSB = 4, BF_OUT5_DIVIDER_SIZE = 4, BF_OUT5_DIVIDER_MASK = 0x0F, BF_-
OUT5_DIVIDER_LSB = 0,
BF_OUT6_PATH_SEL_SIZE = 4, BF_OUT6_PATH_SEL_MASK = 0xF0, BF_OUT6_PATH_SEL_LSB = 4,
BF_OUT6_DIVIDER_SIZE = 4,
BF_OUT6_DIVIDER_MASK = 0x0F, BF_OUT6_DIVIDER_LSB = 0, BF_OUT7_PATH_SEL_SIZE = 4, BF_-
OUT7_PATH_SEL_MASK = 0xF0,
BF_OUT7_PATH_SEL_LSB = 4, BF_OUT7_DIVIDER_SIZE = 4, BF_OUT7_DIVIDER_MASK = 0x0F, BF_-
OUT7_DIVIDER_LSB = 0,
BF_OUT8_PATH_SEL_SIZE = 1, BF_OUT8_PATH_SEL_MASK = 0x80, BF_OUT8_PATH_SEL_LSB = 7,
BF_OUT8_EN_SIZE = 1,
BF_OUT8_EN_MASK = 0x40, BF_OUT8_EN_LSB = 6, BF_T4_INPUT_FAIL_SIZE = 1, BF_T4_INPUT_FA-
IL_MASK = 0x20,
BF_T4_INPUT_FAIL_LSB = 5, BF_AMI_OUT_DUTY_SIZE = 1, BF_AMI_OUT_DUTY_MASK = 0x10, BF_-
AMI_OUT_DUTY_LSB = 4,
BF_400HZ_OUT_SEL_SIZE = 1, BF_400HZ_OUT_SEL_MASK = 0x08, BF_400HZ_OUT_SEL_LSB = 3,
BF_OUT9_INV_SIZE = 1,
BF_OUT9_INV_MASK = 0x04, BF_OUT9_INV_LSB = 2, BF_OUT7_INV_SIZE = 1, BF_OUT7_INV_MASK
= 0x02,
BF_OUT7_INV_LSB = 1, BF_OUT6_INV_SIZE = 1, BF_OUT6_INV_MASK = 0x01, BF_OUT6_INV_LSB =
0,
BF_OUT9_PATH_SEL_SIZE = 1, BF_OUT9_PATH_SEL_MASK = 0x80, BF_OUT9_PATH_SEL_LSB = 7,
BF_OUT9_EN_SIZE = 1,
BF_OUT9_EN_MASK = 0x40, BF_OUT9_EN_LSB = 6, BF_OUT5_INV_SIZE = 1, BF_OUT5_INV_MASK =

```

```

0x10,
BF_OUT5_INV_LSB = 4, BF_OUT4_INV_SIZE = 1, BF_OUT4_INV_MASK = 0x08, BF_OUT4_INV_LSB =
3,
BF_OUT3_INV_SIZE = 1, BF_OUT3_INV_MASK = 0x04, BF_OUT3_INV_LSB = 2, BF_OUT2_INV_SIZE =
1,
BF_OUT2_INV_MASK = 0x02, BF_OUT2_INV_LSB = 1, BF_OUT1_INV_SIZE = 1, BF_OUT1_INV_MASK
= 0x01,
BF_OUT1_INV_LSB = 0, BF_IN_2K_4K_8K_INV_SIZE = 1, BF_IN_2K_4K_8K_INV_MASK = 0x80, BF_IN-
_2K_4K_8K_INV_LSB = 7,
BF_8K_EN_SIZE = 1, BF_8K_EN_MASK = 0x40, BF_8K_EN_LSB = 6, BF_2K_EN_SIZE = 1,
BF_2K_EN_MASK = 0x20, BF_2K_EN_LSB = 5, BF_2K_8K_PUL_POSITION_SIZE = 1, BF_2K_8K_PUL_-
POSITION_MASK = 0x10,
BF_2K_8K_PUL_POSITION_LSB = 4, BF_8K_INV_SIZE = 1, BF_8K_INV_MASK = 0x08, BF_8K_INV_LSB
= 3,
BF_8K_PUL_SIZE = 1, BF_8K_PUL_MASK = 0x04, BF_8K_PUL_LSB = 2, BF_2K_INV_SIZE = 1,
BF_2K_INV_MASK = 0x02, BF_2K_INV_LSB = 1, BF_2K_PUL_SIZE = 1, BF_2K_PUL_MASK = 0x01,
BF_2K_PUL_LSB = 0, BF_IN_NOISE_WINDOW_SIZE = 1, BF_IN_NOISE_WINDOW_MASK = 0x80, BF_I-
N_NOISE_WINDOW_LSB = 7,
BF_PH_OFFSET_A_SIZE = 8, BF_PH_OFFSET_A_MASK = 0xFF, BF_PH_OFFSET_A_LSB = 0, BF_PH_-
OFFSET_EN_SIZE = 1,
BF_PH_OFFSET_EN_MASK = 0x80, BF_PH_OFFSET_EN_LSB = 7, BF_PH_OFFSET_B_SIZE = 2, BF_-
PH_OFFSET_B_MASK = 0x03,
BF_PH_OFFSET_B_LSB = 0, BF_SYNC_BYPASS_SIZE = 1, BF_SYNC_BYPASS_MASK = 0x80, BF_SY-
NC_BYPASS_LSB = 7,
BF_SYNC_MON_LIMT_SIZE = 3, BF_SYNC_MON_LIMT_MASK = 0x70, BF_SYNC_MON_LIMT_LSB = 4,
BF_SYNC_EDGE_SIZE = 1,
BF_SYNC_EDGE_MASK = 0x40, BF_SYNC_EDGE_LSB = 6, BF_SYNC_PH2_SIZE = 2, BF_SYNC_PH2-
_MASK = 0x0C,
BF_SYNC_PH2_LSB = 2, BF_SYNC_PH1_SIZE = 2, BF_SYNC_PH1_MASK = 0x03, BF_SYNC_PH1_LSB
= 0,
BF_PROTECTION_DATA_SIZE = 8, BF_PROTECTION_DATA_MASK = 0xFF, BF_PROTECTION_DATA-
_LSB = 0, BF_MPU_SEL_CNFG_SIZE = 3,
BF_MPU_SEL_CNFG_MASK = 0x07, BF_MPU_SEL_CNFG_LSB = 0, BF_T0_DFS_OFF_3_SIZE = 1,
BF_T0_DFS_OFF_3_MASK = 0x80,
BF_T0_DFS_OFF_3_LSB = 7, BF_T0_DFS_OFF_2_SIZE = 1, BF_T0_DFS_OFF_2_MASK = 0x40, BF_-
T0_DFS_OFF_2_LSB = 6,
BF_T0_DFS_OFF_1_SIZE = 1, BF_T0_DFS_OFF_1_MASK = 0x20, BF_T0_DFS_OFF_1_LSB = 5, BF_-
T0_DFS_OFF_0_SIZE = 1,
BF_T0_DFS_OFF_0_MASK = 0x10, BF_T0_DFS_OFF_0_LSB = 4, BF_T4_DFS_OFF_3_SIZE = 1, BF_-
T4_DFS_OFF_3_MASK = 0x08,
BF_T4_DFS_OFF_3_LSB = 3, BF_T4_DFS_OFF_2_SIZE = 1, BF_T4_DFS_OFF_2_MASK = 0x04, BF_-
T4_DFS_OFF_2_LSB = 2,
BF_T4_DFS_OFF_1_SIZE = 1, BF_T4_DFS_OFF_1_MASK = 0x02, BF_T4_DFS_OFF_1_LSB = 1, BF_-
T4_DFS_OFF_0_SIZE = 1,
BF_T4_DFS_OFF_0_MASK = 0x01, BF_T4_DFS_OFF_0_LSB = 0, BF_PPS_PHASE_SIZE = 2, BF_PPS-
_PHASE_MASK = 0x30,
BF_PPS_PHASE_LSB = 4, BF_PPS_PULSE_SIZE = 4, BF_PPS_PULSE_MASK = 0x0F, BF_PPS_PULS-
E_LSB = 0,
BF_T0_PH_SLOPE_SIZE = 2, BF_T0_PH_SLOPE_MASK = 0x0C, BF_T0_PH_SLOPE_LSB = 2, BF_T4_-
PH_SLOPE_SIZE = 2,
BF_T4_PH_SLOPE_MASK = 0x03, BF_T4_PH_SLOPE_LSB = 0, BF_ICP_CTRL_CODE_SIZE = 5, BF_IC-
P_CTRL_CODE_MASK = 0x1F,
BF_ICP_CTRL_CODE_LSB = 0, REGMAP_MAX }

```

*Enumerated type listing register offsets and bit field constants.*

• enum {

```

REG_CONTROL = 0x00, REG_CLK_SEL = 0x01, REG_PDSEL0_LSB = 0x02, REG_PDSEL0_MSB = 0x03,
REG_PDSEL1_LSB = 0x04, REG_PDSEL1_MSB = 0x05, REG_M0_LSB = 0x06, REG_M0_MSB = 0x07,
REG_M1_LSB = 0x08, REG_M1_MSB = 0x09, REG_ODBSELA0 = 0x0A, REG_ODBSELA1 = 0x0B,
REG_ODBSELB0 = 0x0C, REG_ODBSELB1 = 0x0D, REG_VCXO_SEL = 0x0E, REG_CONFIG_QA = 0x0F,
REG_CONFIG_QB = 0x10, BF_CONFIG_SIZE = 1, BF_CONFIG_MASK = 0x02, BF_CONFIG_LSB = 1,
BF_CLK_SEL_SIZE = 1, BF_CLK_SEL_MASK = 0x01, BF_CLK_SEL_LSB = 0, BF_PDSEL0_LSB_SIZE =
8,
BF_PDSEL0_LSB_MASK = 0xFF, BF_PDSEL0_LSB_LSB = 0, BF_PDSEL0_MSB_SIZE = 7, BF_PDSEL0-
_MSB_MASK = 0x7F,
BF_PDSEL0_MSB_LSB = 0, BF_PDSEL1_LSB_SIZE = 8, BF_PDSEL1_LSB_MASK = 0xFF, BF_PDSEL1-
_LSB_LSB = 0,
BF_PDSEL1_MSB_SIZE = 7, BF_PDSEL1_MSB_MASK = 0x7F, BF_PDSEL1_MSB_LSB = 0, BF_M0_LS-
B_SIZE = 8,
BF_M0_LSB_MASK = 0xFF, BF_M0_LSB_LSB = 0, BF_M0_MSB_SIZE = 7, BF_M0_MSB_MASK = 0x7F,
BF_M0_MSB_LSB = 0, BF_M1_LSB_SIZE = 8, BF_M1_LSB_MASK = 0xFF, BF_M1_LSB_LSB = 0,
BF_M1_MSB_SIZE = 7, BF_M1_MSB_MASK = 0x7F, BF_M1_MSB_LSB = 0, BF_ODBSELA0_SIZE = 3,
BF_ODBSELA0_MASK = 0x07, BF_ODBSELA0_LSB = 0, BF_ODBSELA1_SIZE = 3, BF_ODBSELA1_MA-
SK = 0x07,
BF_ODBSELA1_LSB = 0, BF_ODBSELB0_SIZE = 3, BF_ODBSELB0_MASK = 0x07, BF_ODBSELB0_LSB
= 0,
BF_ODBSELB1_SIZE = 3, BF_ODBSELB1_MASK = 0x07, BF_ODBSELB1_LSB = 0, BF_SEL_DOUBLE_-
SIZE = 1,
BF_SEL_DOUBLE_MASK = 0x80, BF_SEL_DOUBLE_LSB = 7, BF_MR_SYNC_SIZE = 1, BF_MR_SYNC-
_MASK = 0x40,
BF_MR_SYNC_LSB = 6, BF_BYPASS_SIZE = 1, BF_BYPASS_MASK = 0x20, BF_BYPASS_LSB = 5,
BF_SHUTOFF_SIZE = 1, BF_SHUTOFF_MASK = 0x10, BF_SHUTOFF_LSB = 4, BF_VCXO_SEL_SIZE =
1,
BF_VCXO_SEL_MASK = 0x01, BF_VCXO_SEL_LSB = 0, BF_LVDS_QA_SIZE = 1, BF_LVDS_QA_MASK
= 0x02,
BF_LVDS_QA_LSB = 1, BF_OE_A_SIZE = 1, BF_OE_A_MASK = 0x01, BF_OE_A_LSB = 0,
BF_LVDS_QB_SIZE = 1, BF_LVDS_QB_MASK = 0x02, BF_LVDS_QB_LSB = 1, BF_OE_B_SIZE = 1,
BF_OE_B_MASK = 0x01, BF_OE_B_LSB = 0, REGMAP_APLL_MAX }

```

*Enumerated type listing register offsets and bit field constants.*

#### 4.7.1 Detailed Description

#### 4.7.2 Enumeration Type Documentation

##### 4.7.2.1 anonymous enum

Enumerated type listing register offsets and bit field constants.

##### Enumerator

**REG\_ID\_LOW** Device ID Register 1, Low byte

**REG\_ID\_HIGH** Device ID Register 2, High byte

**REG\_MPU\_PIN\_STS** MPU Pin Status Register

**REG\_NOMINAL\_FREQ\_LOW\_CNFG** Crystal Oscillator Frequency Offset Calibration Configuration Register  
1,value[7:0]

**REG\_NOMINAL\_FREQ\_MID\_CNFG** Crystal Oscillator Frequency Offset Calibration Configuration Register  
2,value[15:8]

**REG\_NOMINAL\_FREQ\_HIGH\_CNFG** Crystal Oscillator Frequency Offset Calibration Configuration Register  
3,value[23:16]

**REG\_T4\_T0\_SEL\_CNFG** T0/T4 Register Selection Configuration

**REG\_PHASE\_ALARM\_TIME\_OUT\_CNFG** Phase Lock Alarm Time Out Configuration Register

**REG\_INPUT\_MODE\_CNFG** Input Mode Control Register

**REG\_DIFFERENTIAL\_IN\_OUT\_CNFG** Differential Input/Output Port Configuration Register

**REG\_MON\_SW\_HS\_CNFG** Frequency Monitoring, Master Clock Switch, Input Clock Switch, HS Configuration Register

**REG\_INTERRUPT\_CNFG** Interrupt Configuration Register

**REG\_INTERRUPTS1\_STS** Interrupt Status Register 1

**REG\_INTERRUPTS2\_STS** Interrupt Status Register 2

**REG\_INTERRUPTS3\_STS** Interrupt Status Register 3

**REG\_INTERRUPTS1\_MASK\_CNFG** Interrupt Mask Register 1

**REG\_INTERRUPTS2\_MASK\_CNFG** Interrupt Mask Register 2

**REG\_INTERRUPTS3\_MASK\_CNFG** Interrupt Mask Register 3

**REG\_MS\_SL\_CTRL\_CNFG** Master Slave Control

**REG\_IN1\_CNFG** Input 1 Configuration Register

**REG\_IN2\_CNFG** Input 2 Configuration Register

**REG\_IN3\_CNFG** Input 3 Configuration Register

**REG\_IN4\_CNFG** Input 4 Configuration Register

**REG\_IN5\_IN6\_HF\_DIV\_CNFG** Input 5 and Input 6 High Frequency divider Configuration Register

**REG\_IN5\_CNFG** Input 5 Configuration Register

**REG\_IN6\_CNFG** Input 6 Configuration Register

**REG\_IN7\_CNFG** Input 7 Configuration Register

**REG\_IN8\_CNFG** Input 8 Configuration Register

**REG\_IN9\_CNFG** Input 9 Configuration Register

**REG\_IN10\_CNFG** Input 10 Configuration Register

**REG\_IN11\_CNFG** Input 11 Configuration Register

**REG\_IN12\_CNFG** Input 12 Configuration Register

**REG\_IN13\_CNFG** Input 13 Configuration Register

**REG\_IN14\_CNFG** Input 14 Configuration Register

**REG\_PRE\_DIV\_CH\_CNFG** Input Clock Select for Frequency division Configuration Register

**REG\_PRE\_DIVN\_LOW\_CNFG** DivN Divider Configuration Register 1, Low byte

**REG\_PRE\_DIVN\_HIGH\_CNFG** DivN Divider Configuration Register 2, High byte

**REG\_IN1\_IN2\_SEL\_PRIORITY\_CNFG** Input 1 and Input 2 Priority Configuration Register

**REG\_IN3\_IN4\_SEL\_PRIORITY\_CNFG** Input 3 and Input 4 Priority Configuration Register

**REG\_IN5\_IN6\_SEL\_PRIORITY\_CNFG** Input 5 and Input 6 Priority Configuration Register

**REG\_IN7\_IN8\_SEL\_PRIORITY\_CNFG** Input 7 and Input 8 Priority Configuration Register

**REG\_IN9\_IN10\_SEL\_PRIORITY\_CNFG** Input 9 and Input 10 Priority Configuration Register

**REG\_IN11\_IN12\_SEL\_PRIORITY\_CNFG** Input 11 and Input 12 Priority Configuration Register

**REG\_IN13\_IN14\_SEL\_PRIORITY\_CNFG** Input 13 and Input 14 Priority Configuration Register

**REG\_PAGE\_POINTER\_CNFG** Page Pointer Configuration Register

**REG\_FREQ\_MON\_FACTOR\_CNFG** Frequency Monitor Factor Configuration Register

**REG\_HARD\_FREQ\_MON\_THRESHOLD\_CNFG** Hard Frequency Alarm Threshold for the Non-Selected Input Clocks Configuration Register

**REG\_SOFT\_FREQ\_MON\_THRESHOLD\_CNFG** Soft Frequency Alarm Threshold for the Selected Input Clock Configuration Register

**REG\_UPPER\_THRESHOLD\_0\_CNFG** Upper Threshold 0 Configuration Register

**REG\_LOWER\_THRESHOLD\_0\_CNFG** Lower Threshold 0 Configuration Register

**REG\_BUCKET\_SIZE\_0\_CNFG** Leaky Bucket Size 0 Configuration Register  
**REG\_DECAY\_RATE\_0\_CNFG** Decay Rate 0 Configuration Register  
**REG\_UPPER\_THRESHOLD\_1\_CNFG** Upper Threshold 1 Configuration Register  
**REG\_LOWER\_THRESHOLD\_1\_CNFG** Lower Threshold 1 Configuration Register  
**REG\_BUCKET\_SIZE\_1\_CNFG** Leaky Bucket Size 1 Configuration Register  
**REG\_DECAY\_RATE\_1\_CNFG** Decay Rate 1 Configuration Register  
**REG\_UPPER\_THRESHOLD\_2\_CNFG** Upper Threshold 2 Configuration Register  
**REG\_LOWER\_THRESHOLD\_2\_CNFG** Lower Threshold 2 Configuration Register  
**REG\_BUCKET\_SIZE\_2\_CNFG** Leaky Bucket Size 2 Configuration Register  
**REG\_DECAY\_RATE\_2\_CNFG** Decay Rate 2 Configuration Register  
**REG\_UPPER\_THRESHOLD\_3\_CNFG** Upper Threshold 3 Configuration Register  
**REG\_LOWER\_THRESHOLD\_3\_CNFG** Lower Threshold 3 Configuration Register  
**REG\_BUCKET\_SIZE\_3\_CNFG** Leaky Bucket Size 3 Configuration Register  
**REG\_DECAY\_RATE\_3\_CNFG** Decay Rate 3 Configuration Register  
**REG\_IN\_FREQ\_READ\_CH\_CNFG** Input Clock Selection for Frequency Monitoring Result Read Register  
**REG\_IN\_FREQ\_READ\_STS** Input Clock Frequency Offset Register  
**REG\_IN1\_IN2\_STS** Input 1 and Input 2 Status Register  
**REG\_IN3\_IN4\_STS** Input 3 and Input 4 Status Register  
**REG\_IN5\_IN6\_STS** Input 5 and Input 6 Status Register  
**REG\_IN7\_IN8\_STS** Input 7 and Input 8 Status Register  
**REG\_IN9\_IN10\_STS** Input 9 and Input 10 Status Register  
**REG\_IN11\_IN12\_STS** Input 11 and Input 12 Status Register  
**REG\_IN13\_IN14\_STS** Input 13 and Input 14 Status Register  
**REG\_INPUT\_VALID1\_STS** Input Clock Validation Register 1  
**REG\_INPUT\_VALID2\_STS** Input Clock Validation Register 2  
**REG\_REMOTE\_INPUT\_VALID1\_CNFG** Remote Input Clock Validation Register 1  
**REG\_REMOTE\_INPUT\_VALID2\_CNFG** Remote Input Clock Validation Register 2  
**REG\_PRIORITY\_TABLE1\_STS** The Highest Priority Valid Input Clock and the Selected Input Clock Register  
  
**REG\_PRIORITY\_TABLE2\_STS** The Second Highest Priority Valid Input Clock and the Selected Input Clock Register  
**REG\_T0\_INPUT\_SEL\_CNFG** T0 Input Clock Selection Configuration Register  
**REG\_T4\_INPUT\_SEL\_CNFG** T4 Input Clock Selection Configuration Register  
**REG\_OPERATING\_STS** Operating Status Register  
**REG\_T0\_OPERATION\_MODE\_CNFG** T0 DPLL Operation Mode Selection Register  
**REG\_T4\_OPERATION\_MODE\_CNFG** T4 DPLL Operation Mode Selection Register  
**REG\_T0\_DPLL\_APLL\_PATH\_CNFG** T0 DPLL APLL Path Configuration Register  
**REG\_T0\_DPLL\_START\_BW\_DAMPING\_CNFG** T0 DPLL Start Bandwidth and Damping Factor Configuration Register  
**REG\_T0\_DPLL\_ACQ\_BW\_DAMPING\_CNFG** T0 DPLL Acquisition Bandwidth and Damping Factor Configuration Register  
**REG\_T0\_DPLL\_LOCKED\_BW\_DAMPING\_CNFG** T0 DPLL Locked Bandwidth and Damping Factor Configuration Register  
**REG\_T0\_BW\_OVERSHOOT\_CNFG** T0 DPLL Bandwidth Overshoot Configuration Register  
**REG\_PHASE\_LOSS\_COARSE\_LIMIT\_CNFG** Coarse Phase Lock Detector Configuration Register  
**REG\_PHASE\_LOSS\_FINE\_LIMIT\_CNFG** Fine Phase Lock Detector Configuration Register

**REG\_T0\_HOLD OVER\_MODE\_CNFG** T0 DPLL Holdover Mode Configuration Register  
**REG\_HOLD OVER\_FREQ\_LOW\_CNFG** DPLL Holdover Frequency Register 1  
**REG\_HOLD OVER\_FREQ\_MID\_CNFG** DPLL Holdover Frequency Register 2  
**REG\_HOLD OVER\_FREQ\_HIGH\_CNFG** DPLL Holdover Frequency Register 3  
**REG\_T4\_DPLL\_APLL\_PATH\_CNFG** T4 DPLL APLL Path Configuration Register  
**REG\_T4\_DPLL\_LOCKED\_BW\_DAMPING\_CNFG** T4 DPLL Locked Bandwidth and Damping Factor Configuration Register  
**REG\_CURRENT\_FREQ\_LOW\_STS** DPLL Current Frequency Register 1  
**REG\_CURRENT\_FREQ\_MID\_STS** DPLL Current Frequency Register 2  
**REG\_CURRENT\_FREQ\_HIGH\_STS** DPLL Current Frequency Register 3  
**REG\_DPLL\_FREQ\_SOFT\_LIMIT\_CNFG** DPLL Soft Frequency Alarm Limit Configuration Register  
**REG\_DPLL\_FREQ\_HARD\_LIMIT\_LOW\_CNFG** DPLL Hard Frequency Alarm Limit Configuration Register 1  
  
**REG\_DPLL\_FREQ\_HARD\_LIMIT\_MID\_CNFG** DPLL Hard Frequency Alarm Limit Configuration Register 2  
**REG\_CURRENT\_DPLL\_PHASE\_LOW\_STS** Current DPLL Phase Register 1  
**REG\_CURRENT\_DPLL\_PHASE\_HIGH\_STS** Current DPLL Phase Register 2  
**REG\_OUT1\_FREQ\_CNFG** Output 1 Configuration Register  
**REG\_OUT2\_FREQ\_CNFG** Output 2 Configuration Register  
**REG\_OUT3\_FREQ\_CNFG** Output 3 Configuration Register  
**REG\_OUT4\_FREQ\_CNFG** Output 4 Configuration Register  
**REG\_OUT5\_FREQ\_CNFG** Output 5 Configuration Register  
**REG\_OUT6\_FREQ\_CNFG** APLL 1 Clock Frequency Configuration  
**REG\_OUT7\_FREQ\_CNFG** APLL 2 Clock Frequency Configuration  
**REG\_OUT8\_FREQ\_CNFG** Output 8 Configuration Register  
**REG\_OUT9\_FREQ\_CNFG** Output 9 Configuration Register  
**REG\_FR\_MFR\_SYNC\_CNFG** Frame Sync and Multi-frame Sync Output Configuration Register  
**REG\_PHASE\_MON\_CNFG** Phase Transient Monitor Configuration Register  
**REG\_PHASE\_OFFSET\_LOW\_CNFG** Phase Offset Control Register 1  
**REG\_PHASE\_OFFSET\_HIGH\_CNFG** Phase Offset Control Register 2  
**REG\_SYNC\_MONITOR\_CNFG** Sync Monitor Configuration Register  
**REG\_SYNC\_PHASE\_CNFG** Sync Phase Configuration Register  
**REG\_PROTECTION\_CNFG** Registers Access Mode Selection Register  
**REG\_MPU\_SEL\_CNFG** MPU Interface Mode Selection Register  
**REG\_DFS\_OFF\_CNFG** Digital Frequency Synthesizer Configuration Register  
**REG\_T0\_PPS\_CNFG** T0 1 Pulse Per Second Configuration Register  
**REG\_PH\_SLOPE\_CNFG** Phase Slope limiting Register  
**REG\_T0\_ICP\_CTRL\_CNFG** T0 APLL Charge Pump Current Configuration Register  
**REG\_T4\_ICP\_CTRL\_CNFG** T4 APLL Charge Pump Current Configuration Register  
**REG\_T4\_PPS\_CNFG** T4 1 Pulse Per Second Configuration Register  
**BF\_ID\_A\_SIZE** ID\_LOW  
**BF\_ID\_A\_MASK**  
**BF\_ID\_A\_LSB**  
**BF\_ID\_B\_SIZE** ID\_HIGH  
**BF\_ID\_B\_MASK**  
**BF\_ID\_B\_LSB**

***BF\_MPU\_PIN\_STS\_SIZE*** MPU\_PIN\_STS  
***BF\_MPU\_PIN\_STS\_MASK***  
***BF\_MPU\_PIN\_STS\_LSB***  
***BF\_NOMINAL\_FREQ\_VALUE\_A\_SIZE*** NOMINAL\_FREQ\_LOW\_CNFG  
***BF\_NOMINAL\_FREQ\_VALUE\_A\_MASK***  
***BF\_NOMINAL\_FREQ\_VALUE\_A\_LSB***  
***BF\_NOMINAL\_FREQ\_VALUE\_B\_SIZE*** NOMINAL\_FREQ\_MID\_CNFG  
***BF\_NOMINAL\_FREQ\_VALUE\_B\_MASK***  
***BF\_NOMINAL\_FREQ\_VALUE\_B\_LSB***  
***BF\_NOMINAL\_FREQ\_VALUE\_C\_SIZE*** NOMINAL\_FREQ\_HIGH\_CNFG  
***BF\_NOMINAL\_FREQ\_VALUE\_C\_MASK***  
***BF\_NOMINAL\_FREQ\_VALUE\_C\_LSB***  
***BF\_T4\_T0\_SEL\_SIZE*** T4\_T0\_SEL\_CNFG  
***BF\_T4\_T0\_SEL\_MASK***  
***BF\_T4\_T0\_SEL\_LSB***  
***BF\_MULTI\_FACTOR\_SIZE*** PHASE\_ALARM\_TIME\_OUT\_CNFG  
***BF\_MULTI\_FACTOR\_MASK***  
***BF\_MULTI\_FACTOR\_LSB***  
***BF\_TIMEOUT\_VALUE\_SIZE*** PHASE\_ALARM\_TIME\_OUT\_CNFG  
***BF\_TIMEOUT\_VALUE\_MASK***  
***BF\_TIMEOUT\_VALUE\_LSB***  
***BF\_AUTO\_EXT\_SYNC\_EN\_SIZE*** INPUT\_MODE\_CNFG  
***BF\_AUTO\_EXT\_SYNC\_EN\_MASK***  
***BF\_AUTO\_EXT\_SYNC\_EN\_LSB***  
***BF\_EXT\_SYNC\_EN\_SIZE*** INPUT\_MODE\_CNFG  
***BF\_EXT\_SYNC\_EN\_MASK***  
***BF\_EXT\_SYNC\_EN\_LSB***  
***BF\_PH\_ALARM\_TIMEOUT\_SIZE*** INPUT\_MODE\_CNFG  
***BF\_PH\_ALARM\_TIMEOUT\_MASK***  
***BF\_PH\_ALARM\_TIMEOUT\_LSB***  
***BF\_SYNC\_FREQ\_SIZE*** INPUT\_MODE\_CNFG  
***BF\_SYNC\_FREQ\_MASK***  
***BF\_SYNC\_FREQ\_LSB***  
***BF\_IN\_SONET\_SDH\_SIZE*** INPUT\_MODE\_CNFG  
***BF\_IN\_SONET\_SDH\_MASK***  
***BF\_IN\_SONET\_SDH\_LSB***  
***BF\_MASTER\_SLAVE\_SIZE*** INPUT\_MODE\_CNFG  
***BF\_MASTER\_SLAVE\_MASK***  
***BF\_MASTER\_SLAVE\_LSB***  
***BF\_REVERTIVE\_MODE\_SIZE*** INPUT\_MODE\_CNFG  
***BF\_REVERTIVE\_MODE\_MASK***  
***BF\_REVERTIVE\_MODE\_LSB***  
***BF\_OSC\_EDGE\_SIZE*** DIFFERENTIAL\_IN\_OUT\_CNFG  
***BF\_OSC\_EDGE\_MASK***  
***BF\_OSC\_EDGE\_LSB***

***BF\_OUT7\_PECL\_LVDS\_SIZE*** DIFFERENTIAL\_IN\_OUT\_CNFG  
***BF\_OUT7\_PECL\_LVDS\_MASK***  
***BF\_OUT7\_PECL\_LVDS\_LSB***  
***BF\_OUT6\_PECL\_LVDS\_SIZE*** DIFFERENTIAL\_IN\_OUT\_CNFG  
***BF\_OUT6\_PECL\_LVDS\_MASK***  
***BF\_OUT6\_PECL\_LVDS\_LSB***  
***BF\_FREQ\_MON\_CLK\_SIZE*** MON\_SW\_HS\_CNFG  
***BF\_FREQ\_MON\_CLK\_MASK***  
***BF\_FREQ\_MON\_CLK\_LSB***  
***BF\_LOS\_FLAG\_TO\_TDO\_SIZE*** MON\_SW\_HS\_CNFG  
***BF\_LOS\_FLAG\_TO\_TDO\_MASK***  
***BF\_LOS\_FLAG\_TO\_TDO\_LSB***  
***BF\_ULTR\_FAST\_SW\_SIZE*** MON\_SW\_HS\_CNFG  
***BF\_ULTR\_FAST\_SW\_MASK***  
***BF\_ULTR\_FAST\_SW\_LSB***  
***BF\_EXT\_SW\_SIZE*** MON\_SW\_HS\_CNFG  
***BF\_EXT\_SW\_MASK***  
***BF\_EXT\_SW\_LSB***  
***BF\_HS\_FREZ\_SIZE*** MON\_SW\_HS\_CNFG  
***BF\_HS\_FREZ\_MASK***  
***BF\_HS\_FREZ\_LSB***  
***BF\_HS\_EN\_SIZE*** MON\_SW\_HS\_CNFG  
***BF\_HS\_EN\_MASK***  
***BF\_HS\_EN\_LSB***  
***BF\_FREQ\_MON\_HARD\_EN\_SIZE*** MON\_SW\_HS\_CNFG  
***BF\_FREQ\_MON\_HARD\_EN\_MASK***  
***BF\_FREQ\_MON\_HARD\_EN\_LSB***  
***BF\_HZ\_EN\_SIZE*** INTERRUPT\_CNFG  
***BF\_HZ\_EN\_MASK***  
***BF\_HZ\_EN\_LSB***  
***BF\_INT\_POL\_SIZE*** INTERRUPT\_CNFG  
***BF\_INT\_POL\_MASK***  
***BF\_INT\_POL\_LSB***  
***BF\_IN8\_SIZE*** INTERRUPTS1\_STS INTERRUPTS1\_MASK\_CNFG INPUT\_VALID1\_STS REMOTE\_INPUT-  
VALID1\_CNFG  
***BF\_IN8\_MASK***  
***BF\_IN8\_LSB***  
***BF\_IN7\_SIZE*** INTERRUPTS1\_STS INTERRUPTS1\_MASK\_CNFG INPUT\_VALID1\_STS REMOTE\_INPUT-  
VALID1\_CNFG  
***BF\_IN7\_MASK***  
***BF\_IN7\_LSB***  
***BF\_IN6\_SIZE*** INTERRUPTS1\_STS INTERRUPTS1\_MASK\_CNFG INPUT\_VALID1\_STS REMOTE\_INPUT-  
VALID1\_CNFG  
***BF\_IN6\_MASK***  
***BF\_IN6\_LSB***

**BF\_IN5\_SIZE** INTERRUPTS1\_STS INTERRUPTS1\_MASK\_CNFG INPUT\_VALID1\_STS REMOTE\_INPUT\_VALID1\_CNFG

**BF\_IN5\_MASK**

**BF\_IN5\_LSB**

**BF\_IN4\_SIZE** INTERRUPTS1\_STS INTERRUPTS1\_MASK\_CNFG INPUT\_VALID1\_STS REMOTE\_INPUT\_VALID1\_CNFG

**BF\_IN4\_MASK**

**BF\_IN4\_LSB**

**BF\_IN3\_SIZE** INTERRUPTS1\_STS INTERRUPTS1\_MASK\_CNFG INPUT\_VALID1\_STS REMOTE\_INPUT\_VALID1\_CNFG

**BF\_IN3\_MASK**

**BF\_IN3\_LSB**

**BF\_IN2\_SIZE** INTERRUPTS1\_STS INTERRUPTS1\_MASK\_CNFG INPUT\_VALID1\_STS REMOTE\_INPUT\_VALID1\_CNFG

**BF\_IN2\_MASK**

**BF\_IN2\_LSB**

**BF\_IN1\_SIZE** INTERRUPTS1\_STS INTERRUPTS1\_MASK\_CNFG INPUT\_VALID1\_STS REMOTE\_INPUT\_VALID1\_CNFG

**BF\_IN1\_MASK**

**BF\_IN1\_LSB**

**BF\_T0\_OPERATING\_MODE\_STS\_SIZE** INTERRUPTS2\_STS INTERRUPTS2\_MASK\_CNFG

**BF\_T0\_OPERATING\_MODE\_STS\_MASK**

**BF\_T0\_OPERATING\_MODE\_STS\_LSB**

**BF\_T0\_MAIN\_REF\_FAIL\_SIZE** INTERRUPTS2\_STS INTERRUPTS2\_MASK\_CNFG

**BF\_T0\_MAIN\_REF\_FAIL\_MASK**

**BF\_T0\_MAIN\_REF\_FAIL\_LSB**

**BF\_IN14\_SIZE** INTERRUPTS2\_STS INTERRUPTS2\_MASK\_CNFG INPUT\_VALID2\_STS REMOTE\_INPUT\_VALID2\_CNFG

**BF\_IN14\_MASK**

**BF\_IN14\_LSB**

**BF\_IN13\_SIZE** INTERRUPTS2\_STS INTERRUPTS2\_MASK\_CNFG INPUT\_VALID2\_STS REMOTE\_INPUT\_VALID2\_CNFG

**BF\_IN13\_MASK**

**BF\_IN13\_LSB**

**BF\_IN12\_SIZE** INTERRUPTS2\_STS INTERRUPTS2\_MASK\_CNFG INPUT\_VALID2\_STS REMOTE\_INPUT\_VALID2\_CNFG

**BF\_IN12\_MASK**

**BF\_IN12\_LSB**

**BF\_IN11\_SIZE** INTERRUPTS2\_STS INTERRUPTS2\_MASK\_CNFG INPUT\_VALID2\_STS REMOTE\_INPUT\_VALID2\_CNFG

**BF\_IN11\_MASK**

**BF\_IN11\_LSB**

**BF\_IN10\_SIZE** INTERRUPTS2\_STS INTERRUPTS2\_MASK\_CNFG INPUT\_VALID2\_STS REMOTE\_INPUT\_VALID2\_CNFG

**BF\_IN10\_MASK**

**BF\_IN10\_LSB**

**BF\_IN9\_SIZE** INTERRUPTS2\_STS INTERRUPTS2\_MASK\_CNFG INPUT\_VALID2\_STS REMOTE\_INPUT\_VALID2\_CNFG

***BF\_IN9\_MASK***

***BF\_IN9\_LSB***

***BF\_EX\_SYNC\_ALARM\_SIZE*** INTERRUPTS3\_STS INTERRUPTS3\_MASK\_CNFG

***BF\_EX\_SYNC\_ALARM\_MASK***

***BF\_EX\_SYNC\_ALARM\_LSB***

***BF\_T4\_STS\_SIZE*** INTERRUPTS3\_STS INTERRUPTS3\_MASK\_CNFG

***BF\_T4\_STS\_MASK***

***BF\_T4\_STS\_LSB***

***BF\_INPUT\_TO\_T4\_SIZE*** INTERRUPTS3\_STS INTERRUPTS3\_MASK\_CNFG

***BF\_INPUT\_TO\_T4\_MASK***

***BF\_INPUT\_TO\_T4\_LSB***

***BF\_AMI2\_VIOL\_SIZE*** INTERRUPTS3\_STS INTERRUPTS3\_MASK\_CNFG

***BF\_AMI2\_VIOL\_MASK***

***BF\_AMI2\_VIOL\_LSB***

***BF\_AMI2\_LOS\_SIZE*** INTERRUPTS3\_STS INTERRUPTS3\_MASK\_CNFG

***BF\_AMI2\_LOS\_MASK***

***BF\_AMI2\_LOS\_LSB***

***BF\_AMI1\_VIOL\_SIZE*** INTERRUPTS3\_STS INTERRUPTS3\_MASK\_CNFG

***BF\_AMI1\_VIOL\_MASK***

***BF\_AMI1\_VIOL\_LSB***

***BF\_AMI1\_LOS\_SIZE*** INTERRUPTS3\_STS INTERRUPTS3\_MASK\_CNFG

***BF\_AMI1\_LOS\_MASK***

***BF\_AMI1\_LOS\_LSB***

***BF\_PPS\_FAST\_FREQ\_LOCK\_EN\_SIZE*** MS\_SL\_CTRL\_CNFG

***BF\_PPS\_FAST\_FREQ\_LOCK\_EN\_MASK***

***BF\_PPS\_FAST\_FREQ\_LOCK\_EN\_LSB***

***BF\_PPS\_FAST\_PH\_LOCK\_EN\_SIZE*** MS\_SL\_CTRL\_CNFG

***BF\_PPS\_FAST\_PH\_LOCK\_EN\_MASK***

***BF\_PPS\_FAST\_PH\_LOCK\_EN\_LSB***

***BF\_MS\_SL\_CTRL\_SIZE*** MS\_SL\_CTRL\_CNFG

***BF\_MS\_SL\_CTRL\_MASK***

***BF\_MS\_SL\_CTRL\_LSB***

***BF\_400HZ\_SEL\_SIZE*** IN1\_CNFG IN2\_CNFG

***BF\_400HZ\_SEL\_MASK***

***BF\_400HZ\_SEL\_LSB***

***BF\_BUCKET\_SEL\_SIZE*** IN1\_CNFG IN2\_CNFG IN3\_CNFG IN4\_CNFG IN5\_CNFG IN6\_CNFG IN7\_CNFG  
IN8\_CNFG IN9\_CNFG IN10\_CNFG IN11\_CNFG IN12\_CNFG IN13\_CNFG IN14\_CNFG

***BF\_BUCKET\_SEL\_MASK***

***BF\_BUCKET\_SEL\_LSB***

***BF\_IN\_FREQ\_SIZE*** IN1\_CNFG IN2\_CNFG IN3\_CNFG IN4\_CNFG IN5\_CNFG IN6\_CNFG IN7\_CNFG IN8-  
\_CNFG IN9\_CNFG IN10\_CNFG IN11\_CNFG IN12\_CNFG IN13\_CNFG IN14\_CNFG

***BF\_IN\_FREQ\_MASK***

***BF\_IN\_FREQ\_LSB***

***BF\_DIRECT\_DIV\_SIZE*** IN3\_CNFG IN4\_CNFG IN5\_CNFG IN6\_CNFG IN7\_CNFG IN8\_CNFG IN9\_CNFG  
IN10\_CNFG IN11\_CNFG IN12\_CNFG IN13\_CNFG IN14\_CNFG

***BF\_DIRECT\_DIV\_MASK***

***BF\_DIRECT\_DIV\_LSB***

***BF\_LOCK\_8K\_SIZE*** IN3\_CNFG IN4\_CNFG IN5\_CNFG IN6\_CNFG IN7\_CNFG IN8\_CNFG IN9\_CNFG I-  
N10\_CNFG IN11\_CNFG IN12\_CNFG IN13\_CNFG IN14\_CNFG

***BF\_LOCK\_8K\_MASK***

***BF\_LOCK\_8K\_LSB***

***BF\_IN6\_DIV\_SIZE*** IN5\_IN6\_HF\_DIV\_CNFG

***BF\_IN6\_DIV\_MASK***

***BF\_IN6\_DIV\_LSB***

***BF\_IN5\_DIV\_SIZE*** IN5\_IN6\_HF\_DIV\_CNFG

***BF\_IN5\_DIV\_MASK***

***BF\_IN5\_DIV\_LSB***

***BF\_PRE\_DIV\_CH\_VALUE\_SIZE*** PRE\_DIV\_CH\_CNFG

***BF\_PRE\_DIV\_CH\_VALUE\_MASK***

***BF\_PRE\_DIV\_CH\_VALUE\_LSB***

***BF\_PRE\_DIVN\_VALUE\_A\_SIZE*** PRE\_DIVN\_LOW\_CNFG

***BF\_PRE\_DIVN\_VALUE\_A\_MASK***

***BF\_PRE\_DIVN\_VALUE\_A\_LSB***

***BF\_PRE\_DIVN\_VALUE\_B\_SIZE*** PRE\_DIVN\_HIGH\_CNFG

***BF\_PRE\_DIVN\_VALUE\_B\_MASK***

***BF\_PRE\_DIVN\_VALUE\_B\_LSB***

***BF\_IN2\_SEL\_PRIORITY\_SIZE*** IN1\_IN2\_SEL\_PRIORITY\_CNFG

***BF\_IN2\_SEL\_PRIORITY\_MASK***

***BF\_IN2\_SEL\_PRIORITY\_LSB***

***BF\_IN1\_SEL\_PRIORITY\_SIZE*** IN1\_IN2\_SEL\_PRIORITY\_CNFG

***BF\_IN1\_SEL\_PRIORITY\_MASK***

***BF\_IN1\_SEL\_PRIORITY\_LSB***

***BF\_IN4\_SEL\_PRIORITY\_SIZE*** IN3\_IN4\_SEL\_PRIORITY\_CNFG

***BF\_IN4\_SEL\_PRIORITY\_MASK***

***BF\_IN4\_SEL\_PRIORITY\_LSB***

***BF\_IN3\_SEL\_PRIORITY\_SIZE*** IN3\_IN4\_SEL\_PRIORITY\_CNFG

***BF\_IN3\_SEL\_PRIORITY\_MASK***

***BF\_IN3\_SEL\_PRIORITY\_LSB***

***BF\_IN6\_SEL\_PRIORITY\_SIZE*** IN5\_IN6\_SEL\_PRIORITY\_CNFG

***BF\_IN6\_SEL\_PRIORITY\_MASK***

***BF\_IN6\_SEL\_PRIORITY\_LSB***

***BF\_IN5\_SEL\_PRIORITY\_SIZE*** IN5\_IN6\_SEL\_PRIORITY\_CNFG

***BF\_IN5\_SEL\_PRIORITY\_MASK***

***BF\_IN5\_SEL\_PRIORITY\_LSB***

***BF\_IN8\_SEL\_PRIORITY\_SIZE*** IN7\_IN8\_SEL\_PRIORITY\_CNFG

***BF\_IN8\_SEL\_PRIORITY\_MASK***

***BF\_IN8\_SEL\_PRIORITY\_LSB***

***BF\_IN7\_SEL\_PRIORITY\_SIZE*** IN7\_IN8\_SEL\_PRIORITY\_CNFG

***BF\_IN7\_SEL\_PRIORITY\_MASK***

***BF\_IN7\_SEL\_PRIORITY\_LSB***

***BF\_IN10\_SEL\_PRIORITY\_SIZE*** IN9\_IN10\_SEL\_PRIORITY\_CNFG  
***BF\_IN10\_SEL\_PRIORITY\_MASK***  
***BF\_IN10\_SEL\_PRIORITY\_LSB***  
***BF\_IN9\_SEL\_PRIORITY\_SIZE*** IN9\_IN10\_SEL\_PRIORITY\_CNFG  
***BF\_IN9\_SEL\_PRIORITY\_MASK***  
***BF\_IN9\_SEL\_PRIORITY\_LSB***  
***BF\_IN12\_SEL\_PRIORITY\_SIZE*** IN11\_IN12\_SEL\_PRIORITY\_CNFG  
***BF\_IN12\_SEL\_PRIORITY\_MASK***  
***BF\_IN12\_SEL\_PRIORITY\_LSB***  
***BF\_IN11\_SEL\_PRIORITY\_SIZE*** IN11\_IN12\_SEL\_PRIORITY\_CNFG  
***BF\_IN11\_SEL\_PRIORITY\_MASK***  
***BF\_IN11\_SEL\_PRIORITY\_LSB***  
***BF\_IN14\_SEL\_PRIORITY\_SIZE*** IN13\_IN14\_SEL\_PRIORITY\_CNFG  
***BF\_IN14\_SEL\_PRIORITY\_MASK***  
***BF\_IN14\_SEL\_PRIORITY\_LSB***  
***BF\_IN13\_SEL\_PRIORITY\_SIZE*** IN13\_IN14\_SEL\_PRIORITY\_CNFG  
***BF\_IN13\_SEL\_PRIORITY\_MASK***  
***BF\_IN13\_SEL\_PRIORITY\_LSB***  
***BF\_PAGE\_POINTER\_SIZE*** PAGE\_POINTER\_CNFG  
***BF\_PAGE\_POINTER\_MASK***  
***BF\_PAGE\_POINTER\_LSB***  
***BF\_FREQ\_MON\_FACTOR\_SIZE*** FREQ\_MON\_FACTOR\_CNFG  
***BF\_FREQ\_MON\_FACTOR\_MASK***  
***BF\_FREQ\_MON\_FACTOR\_LSB***  
***BF\_HARD\_FREQ\_MON\_THRESHOLD\_A\_SIZE*** HARD\_FREQ\_MON\_THRESHOLD\_CNFG  
***BF\_HARD\_FREQ\_MON\_THRESHOLD\_A\_MASK***  
***BF\_HARD\_FREQ\_MON\_THRESHOLD\_A\_LSB***  
***BF\_HARD\_FREQ\_MON\_THRESHOLD\_B\_SIZE*** HARD\_FREQ\_MON\_THRESHOLD\_CNFG  
***BF\_HARD\_FREQ\_MON\_THRESHOLD\_B\_MASK***  
***BF\_HARD\_FREQ\_MON\_THRESHOLD\_B\_LSB***  
***BF\_SOFT\_FREQ\_MON\_THRESHOLD\_A\_SIZE*** SOFT\_FREQ\_MON\_THRESHOLD\_CNFG  
***BF\_SOFT\_FREQ\_MON\_THRESHOLD\_A\_MASK***  
***BF\_SOFT\_FREQ\_MON\_THRESHOLD\_A\_LSB***  
***BF\_SOFT\_FREQ\_MON\_THRESHOLD\_B\_SIZE*** SOFT\_FREQ\_MON\_THRESHOLD\_CNFG  
***BF\_SOFT\_FREQ\_MON\_THRESHOLD\_B\_MASK***  
***BF\_SOFT\_FREQ\_MON\_THRESHOLD\_B\_LSB***  
***BF\_UPPER\_THRESHOLD\_DATA\_SIZE*** UPPER\_THRESHOLD\_0\_CNFG UPPER\_THRESHOLD\_1\_CNFG  
 UPPER\_THRESHOLD\_2\_CNFG UPPER\_THRESHOLD\_3\_CNFG  
***BF\_UPPER\_THRESHOLD\_DATA\_MASK***  
***BF\_UPPER\_THRESHOLD\_DATA\_LSB***  
***BF\_LOWER\_THRESHOLD\_DATA\_SIZE*** LOWER\_THRESHOLD\_0\_CNFG LOWER\_THRESHOLD\_1\_CN-  
 FG LOWER\_THRESHOLD\_2\_CNFG LOWER\_THRESHOLD\_3\_CNFG  
***BF\_LOWER\_THRESHOLD\_DATA\_MASK***  
***BF\_LOWER\_THRESHOLD\_DATA\_LSB***

***BF\_BUCKET\_SIZE\_DATA\_SIZE*** BUCKET\_SIZE\_0\_CNFG BUCKET\_SIZE\_1\_CNFG BUCKET\_SIZE\_2\_CNFG BUCKET\_SIZE\_3\_CNFG

***BF\_BUCKET\_SIZE\_DATA\_MASK***

***BF\_BUCKET\_SIZE\_DATA\_LSB***

***BF\_DECAY\_RATE\_DATA\_SIZE*** DECAY\_RATE\_0\_CNFG DECAY\_RATE\_1\_CNFG DECAY\_RATE\_2\_CNFG DECAY\_RATE\_3\_CNFG

***BF\_DECAY\_RATE\_DATA\_MASK***

***BF\_DECAY\_RATE\_DATA\_LSB***

***BF\_IN\_FREQ\_READ\_CH\_SIZE*** IN\_FREQ\_READ\_CH\_CNFG

***BF\_IN\_FREQ\_READ\_CH\_MASK***

***BF\_IN\_FREQ\_READ\_CH\_LSB***

***BF\_IN\_FREQ\_VALUE\_SIZE*** IN\_FREQ\_READ\_STS

***BF\_IN\_FREQ\_VALUE\_MASK***

***BF\_IN\_FREQ\_VALUE\_LSB***

***BF\_IN2\_FREQ\_SOFT\_ALARM\_SIZE*** IN1\_IN2\_STS

***BF\_IN2\_FREQ\_SOFT\_ALARM\_MASK***

***BF\_IN2\_FREQ\_SOFT\_ALARM\_LSB***

***BF\_IN2\_FREQ\_HARD\_ALARM\_SIZE*** IN1\_IN2\_STS

***BF\_IN2\_FREQ\_HARD\_ALARM\_MASK***

***BF\_IN2\_FREQ\_HARD\_ALARM\_LSB***

***BF\_IN2\_NO\_ACTIVITY\_ALARM\_SIZE*** IN1\_IN2\_STS

***BF\_IN2\_NO\_ACTIVITY\_ALARM\_MASK***

***BF\_IN2\_NO\_ACTIVITY\_ALARM\_LSB***

***BF\_IN2\_PH\_LOCK\_ALARM\_SIZE*** IN1\_IN2\_STS

***BF\_IN2\_PH\_LOCK\_ALARM\_MASK***

***BF\_IN2\_PH\_LOCK\_ALARM\_LSB***

***BF\_IN1\_FREQ\_SOFT\_ALARM\_SIZE*** IN1\_IN2\_STS

***BF\_IN1\_FREQ\_SOFT\_ALARM\_MASK***

***BF\_IN1\_FREQ\_SOFT\_ALARM\_LSB***

***BF\_IN1\_FREQ\_HARD\_ALARM\_SIZE*** IN1\_IN2\_STS

***BF\_IN1\_FREQ\_HARD\_ALARM\_MASK***

***BF\_IN1\_FREQ\_HARD\_ALARM\_LSB***

***BF\_IN1\_NO\_ACTIVITY\_ALARM\_SIZE*** IN1\_IN2\_STS

***BF\_IN1\_NO\_ACTIVITY\_ALARM\_MASK***

***BF\_IN1\_NO\_ACTIVITY\_ALARM\_LSB***

***BF\_IN1\_PH\_LOCK\_ALARM\_SIZE*** IN1\_IN2\_STS

***BF\_IN1\_PH\_LOCK\_ALARM\_MASK***

***BF\_IN1\_PH\_LOCK\_ALARM\_LSB***

***BF\_IN4\_FREQ\_SOFT\_ALARM\_SIZE*** IN3\_IN4\_STS

***BF\_IN4\_FREQ\_SOFT\_ALARM\_MASK***

***BF\_IN4\_FREQ\_SOFT\_ALARM\_LSB***

***BF\_IN4\_FREQ\_HARD\_ALARM\_SIZE*** IN3\_IN4\_STS

***BF\_IN4\_FREQ\_HARD\_ALARM\_MASK***

***BF\_IN4\_FREQ\_HARD\_ALARM\_LSB***

***BF\_IN4\_NO\_ACTIVITY\_ALARM\_SIZE*** IN3\_IN4\_STS

***BF\_IN4\_NO\_ACTIVITY\_ALARM\_MASK***  
***BF\_IN4\_NO\_ACTIVITY\_ALARM\_LSB***  
***BF\_IN4\_PH\_LOCK\_ALARM\_SIZE*** IN3\_IN4\_STS  
***BF\_IN4\_PH\_LOCK\_ALARM\_MASK***  
***BF\_IN4\_PH\_LOCK\_ALARM\_LSB***  
***BF\_IN3\_FREQ\_SOFT\_ALARM\_SIZE*** IN3\_IN4\_STS  
***BF\_IN3\_FREQ\_SOFT\_ALARM\_MASK***  
***BF\_IN3\_FREQ\_SOFT\_ALARM\_LSB***  
***BF\_IN3\_FREQ\_HARD\_ALARM\_SIZE*** IN3\_IN4\_STS  
***BF\_IN3\_FREQ\_HARD\_ALARM\_MASK***  
***BF\_IN3\_FREQ\_HARD\_ALARM\_LSB***  
***BF\_IN3\_NO\_ACTIVITY\_ALARM\_SIZE*** IN3\_IN4\_STS  
***BF\_IN3\_NO\_ACTIVITY\_ALARM\_MASK***  
***BF\_IN3\_NO\_ACTIVITY\_ALARM\_LSB***  
***BF\_IN3\_PH\_LOCK\_ALARM\_SIZE*** IN3\_IN4\_STS  
***BF\_IN3\_PH\_LOCK\_ALARM\_MASK***  
***BF\_IN3\_PH\_LOCK\_ALARM\_LSB***  
***BF\_IN6\_FREQ\_SOFT\_ALARM\_SIZE*** IN5\_IN6\_STS  
***BF\_IN6\_FREQ\_SOFT\_ALARM\_MASK***  
***BF\_IN6\_FREQ\_SOFT\_ALARM\_LSB***  
***BF\_IN6\_FREQ\_HARD\_ALARM\_SIZE*** IN5\_IN6\_STS  
***BF\_IN6\_FREQ\_HARD\_ALARM\_MASK***  
***BF\_IN6\_FREQ\_HARD\_ALARM\_LSB***  
***BF\_IN6\_NO\_ACTIVITY\_ALARM\_SIZE*** IN5\_IN6\_STS  
***BF\_IN6\_NO\_ACTIVITY\_ALARM\_MASK***  
***BF\_IN6\_NO\_ACTIVITY\_ALARM\_LSB***  
***BF\_IN6\_PH\_LOCK\_ALARM\_SIZE*** IN5\_IN6\_STS  
***BF\_IN6\_PH\_LOCK\_ALARM\_MASK***  
***BF\_IN6\_PH\_LOCK\_ALARM\_LSB***  
***BF\_IN5\_FREQ\_SOFT\_ALARM\_SIZE*** IN5\_IN6\_STS  
***BF\_IN5\_FREQ\_SOFT\_ALARM\_MASK***  
***BF\_IN5\_FREQ\_SOFT\_ALARM\_LSB***  
***BF\_IN5\_FREQ\_HARD\_ALARM\_SIZE*** IN5\_IN6\_STS  
***BF\_IN5\_FREQ\_HARD\_ALARM\_MASK***  
***BF\_IN5\_FREQ\_HARD\_ALARM\_LSB***  
***BF\_IN5\_NO\_ACTIVITY\_ALARM\_SIZE*** IN5\_IN6\_STS  
***BF\_IN5\_NO\_ACTIVITY\_ALARM\_MASK***  
***BF\_IN5\_NO\_ACTIVITY\_ALARM\_LSB***  
***BF\_IN5\_PH\_LOCK\_ALARM\_SIZE*** IN5\_IN6\_STS  
***BF\_IN5\_PH\_LOCK\_ALARM\_MASK***  
***BF\_IN5\_PH\_LOCK\_ALARM\_LSB***  
***BF\_IN8\_FREQ\_SOFT\_ALARM\_SIZE*** IN7\_IN8\_STS  
***BF\_IN8\_FREQ\_SOFT\_ALARM\_MASK***  
***BF\_IN8\_FREQ\_SOFT\_ALARM\_LSB***  
***BF\_IN8\_FREQ\_HARD\_ALARM\_SIZE*** IN7\_IN8\_STS

***BF\_IN8\_FREQ\_HARD\_ALARM\_MASK***  
***BF\_IN8\_FREQ\_HARD\_ALARM\_LSB***  
***BF\_IN8\_NO\_ACTIVITY\_ALARM\_SIZE*** IN7\_IN8\_STS  
***BF\_IN8\_NO\_ACTIVITY\_ALARM\_MASK***  
***BF\_IN8\_NO\_ACTIVITY\_ALARM\_LSB***  
***BF\_IN8\_PH\_LOCK\_ALARM\_SIZE*** IN7\_IN8\_STS  
***BF\_IN8\_PH\_LOCK\_ALARM\_MASK***  
***BF\_IN8\_PH\_LOCK\_ALARM\_LSB***  
***BF\_IN7\_FREQ\_SOFT\_ALARM\_SIZE*** IN7\_IN8\_STS  
***BF\_IN7\_FREQ\_SOFT\_ALARM\_MASK***  
***BF\_IN7\_FREQ\_SOFT\_ALARM\_LSB***  
***BF\_IN7\_FREQ\_HARD\_ALARM\_SIZE*** IN7\_IN8\_STS  
***BF\_IN7\_FREQ\_HARD\_ALARM\_MASK***  
***BF\_IN7\_FREQ\_HARD\_ALARM\_LSB***  
***BF\_IN7\_NO\_ACTIVITY\_ALARM\_SIZE*** IN7\_IN8\_STS  
***BF\_IN7\_NO\_ACTIVITY\_ALARM\_MASK***  
***BF\_IN7\_NO\_ACTIVITY\_ALARM\_LSB***  
***BF\_IN7\_PH\_LOCK\_ALARM\_SIZE*** IN7\_IN8\_STS  
***BF\_IN7\_PH\_LOCK\_ALARM\_MASK***  
***BF\_IN7\_PH\_LOCK\_ALARM\_LSB***  
***BF\_IN10\_FREQ\_SOFT\_ALARM\_SIZE*** IN9\_IN10\_STS  
***BF\_IN10\_FREQ\_SOFT\_ALARM\_MASK***  
***BF\_IN10\_FREQ\_SOFT\_ALARM\_LSB***  
***BF\_IN10\_FREQ\_HARD\_ALARM\_SIZE*** IN9\_IN10\_STS  
***BF\_IN10\_FREQ\_HARD\_ALARM\_MASK***  
***BF\_IN10\_FREQ\_HARD\_ALARM\_LSB***  
***BF\_IN10\_NO\_ACTIVITY\_ALARM\_SIZE*** IN9\_IN10\_STS  
***BF\_IN10\_NO\_ACTIVITY\_ALARM\_MASK***  
***BF\_IN10\_NO\_ACTIVITY\_ALARM\_LSB***  
***BF\_IN10\_PH\_LOCK\_ALARM\_SIZE*** IN9\_IN10\_STS  
***BF\_IN10\_PH\_LOCK\_ALARM\_MASK***  
***BF\_IN10\_PH\_LOCK\_ALARM\_LSB***  
***BF\_IN9\_FREQ\_SOFT\_ALARM\_SIZE*** IN9\_IN10\_STS  
***BF\_IN9\_FREQ\_SOFT\_ALARM\_MASK***  
***BF\_IN9\_FREQ\_SOFT\_ALARM\_LSB***  
***BF\_IN9\_FREQ\_HARD\_ALARM\_SIZE*** IN9\_IN10\_STS  
***BF\_IN9\_FREQ\_HARD\_ALARM\_MASK***  
***BF\_IN9\_FREQ\_HARD\_ALARM\_LSB***  
***BF\_IN9\_NO\_ACTIVITY\_ALARM\_SIZE*** IN9\_IN10\_STS  
***BF\_IN9\_NO\_ACTIVITY\_ALARM\_MASK***  
***BF\_IN9\_NO\_ACTIVITY\_ALARM\_LSB***  
***BF\_IN9\_PH\_LOCK\_ALARM\_SIZE*** IN9\_IN10\_STS  
***BF\_IN9\_PH\_LOCK\_ALARM\_MASK***  
***BF\_IN9\_PH\_LOCK\_ALARM\_LSB***  
***BF\_IN12\_FREQ\_SOFT\_ALARM\_SIZE*** IN11\_IN12\_STS

***BF\_IN12\_FREQ\_SOFT\_ALARM\_MASK***  
***BF\_IN12\_FREQ\_SOFT\_ALARM\_LSB***  
***BF\_IN12\_FREQ\_HARD\_ALARM\_SIZE*** IN11\_IN12\_STS  
***BF\_IN12\_FREQ\_HARD\_ALARM\_MASK***  
***BF\_IN12\_FREQ\_HARD\_ALARM\_LSB***  
***BF\_IN12\_NO\_ACTIVITY\_ALARM\_SIZE*** IN11\_IN12\_STS  
***BF\_IN12\_NO\_ACTIVITY\_ALARM\_MASK***  
***BF\_IN12\_NO\_ACTIVITY\_ALARM\_LSB***  
***BF\_IN12\_PH\_LOCK\_ALARM\_SIZE*** IN11\_IN12\_STS  
***BF\_IN12\_PH\_LOCK\_ALARM\_MASK***  
***BF\_IN12\_PH\_LOCK\_ALARM\_LSB***  
***BF\_IN11\_FREQ\_SOFT\_ALARM\_SIZE*** IN11\_IN12\_STS  
***BF\_IN11\_FREQ\_SOFT\_ALARM\_MASK***  
***BF\_IN11\_FREQ\_SOFT\_ALARM\_LSB***  
***BF\_IN11\_FREQ\_HARD\_ALARM\_SIZE*** IN11\_IN12\_STS  
***BF\_IN11\_FREQ\_HARD\_ALARM\_MASK***  
***BF\_IN11\_FREQ\_HARD\_ALARM\_LSB***  
***BF\_IN11\_NO\_ACTIVITY\_ALARM\_SIZE*** IN11\_IN12\_STS  
***BF\_IN11\_NO\_ACTIVITY\_ALARM\_MASK***  
***BF\_IN11\_NO\_ACTIVITY\_ALARM\_LSB***  
***BF\_IN11\_PH\_LOCK\_ALARM\_SIZE*** IN11\_IN12\_STS  
***BF\_IN11\_PH\_LOCK\_ALARM\_MASK***  
***BF\_IN11\_PH\_LOCK\_ALARM\_LSB***  
***BF\_IN14\_FREQ\_SOFT\_ALARM\_SIZE*** IN13\_IN14\_STS  
***BF\_IN14\_FREQ\_SOFT\_ALARM\_MASK***  
***BF\_IN14\_FREQ\_SOFT\_ALARM\_LSB***  
***BF\_IN14\_FREQ\_HARD\_ALARM\_SIZE*** IN13\_IN14\_STS  
***BF\_IN14\_FREQ\_HARD\_ALARM\_MASK***  
***BF\_IN14\_FREQ\_HARD\_ALARM\_LSB***  
***BF\_IN14\_NO\_ACTIVITY\_ALARM\_SIZE*** IN13\_IN14\_STS  
***BF\_IN14\_NO\_ACTIVITY\_ALARM\_MASK***  
***BF\_IN14\_NO\_ACTIVITY\_ALARM\_LSB***  
***BF\_IN14\_PH\_LOCK\_ALARM\_SIZE*** IN13\_IN14\_STS  
***BF\_IN14\_PH\_LOCK\_ALARM\_MASK***  
***BF\_IN14\_PH\_LOCK\_ALARM\_LSB***  
***BF\_IN13\_FREQ\_SOFT\_ALARM\_SIZE*** IN13\_IN14\_STS  
***BF\_IN13\_FREQ\_SOFT\_ALARM\_MASK***  
***BF\_IN13\_FREQ\_SOFT\_ALARM\_LSB***  
***BF\_IN13\_FREQ\_HARD\_ALARM\_SIZE*** IN13\_IN14\_STS  
***BF\_IN13\_FREQ\_HARD\_ALARM\_MASK***  
***BF\_IN13\_FREQ\_HARD\_ALARM\_LSB***  
***BF\_IN13\_NO\_ACTIVITY\_ALARM\_SIZE*** IN13\_IN14\_STS  
***BF\_IN13\_NO\_ACTIVITY\_ALARM\_MASK***  
***BF\_IN13\_NO\_ACTIVITY\_ALARM\_LSB***  
***BF\_IN13\_PH\_LOCK\_ALARM\_SIZE*** IN13\_IN14\_STS

***BF\_IN13\_PH\_LOCK\_ALARM\_MASK***  
***BF\_IN13\_PH\_LOCK\_ALARM\_LSB***  
***BF\_HIGHEST\_PRIORITY\_VALIDATED\_SIZE*** PRIORITY\_TABLE1\_STS  
***BF\_HIGHEST\_PRIORITY\_VALIDATED\_MASK***  
***BF\_HIGHEST\_PRIORITY\_VALIDATED\_LSB***  
***BF\_CURRENTLY\_SELECTED\_INPUTS\_SIZE*** PRIORITY\_TABLE1\_STS  
***BF\_CURRENTLY\_SELECTED\_INPUTS\_MASK***  
***BF\_CURRENTLY\_SELECTED\_INPUTS\_LSB***  
***BF\_THIRD\_HIGHEST\_PRIORITY\_VALIDATED\_SIZE*** PRIORITY\_TABLE2\_STS  
***BF\_THIRD\_HIGHEST\_PRIORITY\_VALIDATED\_MASK***  
***BF\_THIRD\_HIGHEST\_PRIORITY\_VALIDATED\_LSB***  
***BF\_SECOND\_HIGHEST\_PRIORITY\_VALIDATED\_SIZE*** PRIORITY\_TABLE2\_STS  
***BF\_SECOND\_HIGHEST\_PRIORITY\_VALIDATED\_MASK***  
***BF\_SECOND\_HIGHEST\_PRIORITY\_VALIDATED\_LSB***  
***BF\_T0\_INPUT\_SEL\_SIZE*** T0\_INPUT\_SEL\_CNFG  
***BF\_T0\_INPUT\_SEL\_MASK***  
***BF\_T0\_INPUT\_SEL\_LSB***  
***BF\_T4\_LOCK\_T0\_SIZE*** T4\_INPUT\_SEL\_CNFG  
***BF\_T4\_LOCK\_T0\_MASK***  
***BF\_T4\_LOCK\_T0\_LSB***  
***BF\_T0\_FOR\_T4\_SIZE*** T4\_INPUT\_SEL\_CNFG  
***BF\_T0\_FOR\_T4\_MASK***  
***BF\_T0\_FOR\_T4\_LSB***  
***BF\_T4\_TEST\_T0\_PH\_SIZE*** T4\_INPUT\_SEL\_CNFG  
***BF\_T4\_TEST\_T0\_PH\_MASK***  
***BF\_T4\_TEST\_T0\_PH\_LSB***  
***BF\_T4\_INPUT\_SEL\_SIZE*** T4\_INPUT\_SEL\_CNFG  
***BF\_T4\_INPUT\_SEL\_MASK***  
***BF\_T4\_INPUT\_SEL\_LSB***  
***BF\_EX\_SYNC\_ALARM\_MON\_SIZE*** OPERATING\_STS  
***BF\_EX\_SYNC\_ALARM\_MON\_MASK***  
***BF\_EX\_SYNC\_ALARM\_MON\_LSB***  
***BF\_T4\_DPLL\_LOCK\_SIZE*** OPERATING\_STS  
***BF\_T4\_DPLL\_LOCK\_MASK***  
***BF\_T4\_DPLL\_LOCK\_LSB***  
***BF\_T0\_DPLL\_SOFT\_FREQ\_ALARM\_SIZE*** OPERATING\_STS  
***BF\_T0\_DPLL\_SOFT\_FREQ\_ALARM\_MASK***  
***BF\_T0\_DPLL\_SOFT\_FREQ\_ALARM\_LSB***  
***BF\_T4\_DPLL\_SOFT\_FREQ\_ALARM\_SIZE*** OPERATING\_STS  
***BF\_T4\_DPLL\_SOFT\_FREQ\_ALARM\_MASK***  
***BF\_T4\_DPLL\_SOFT\_FREQ\_ALARM\_LSB***  
***BF\_T0\_DPLL\_LOCK\_SIZE*** OPERATING\_STS  
***BF\_T0\_DPLL\_LOCK\_MASK***  
***BF\_T0\_DPLL\_LOCK\_LSB***  
***BF\_T0\_DPLL\_OPERATING\_MODE\_SIZE*** OPERATING\_STS

***BF\_T0\_DPLL\_OPERATING\_MODE\_MASK***  
***BF\_T0\_DPLL\_OPERATING\_MODE\_LSB***  
***BF\_T0\_WDCO\_SIZE*** T0\_OPERATION\_MODE\_CNFG  
***BF\_T0\_WDCO\_MASK***  
***BF\_T0\_WDCO\_LSB***  
***BF\_T0\_OPERATING\_MODE\_SIZE*** T0\_OPERATION\_MODE\_CNFG  
***BF\_T0\_OPERATING\_MODE\_MASK***  
***BF\_T0\_OPERATING\_MODE\_LSB***  
***BF\_T4\_WDCO\_SIZE*** T4\_OPERATION\_MODE\_CNFG  
***BF\_T4\_WDCO\_MASK***  
***BF\_T4\_WDCO\_LSB***  
***BF\_T4\_OPERATING\_MODE\_SIZE*** T4\_OPERATION\_MODE\_CNFG  
***BF\_T4\_OPERATING\_MODE\_MASK***  
***BF\_T4\_OPERATING\_MODE\_LSB***  
***BF\_T0\_APLL\_PATH\_SIZE*** T0\_DPLL\_APLL\_PATH\_CNFG  
***BF\_T0\_APLL\_PATH\_MASK***  
***BF\_T0\_APLL\_PATH\_LSB***  
***BF\_T0\_GSM\_OBSAI\_16E1\_16T1\_SEL\_SIZE*** T0\_DPLL\_APLL\_PATH\_CNFG  
***BF\_T0\_GSM\_OBSAI\_16E1\_16T1\_SEL\_MASK***  
***BF\_T0\_GSM\_OBSAI\_16E1\_16T1\_SEL\_LSB***  
***BF\_T0\_12E1\_GPS\_E3\_T3\_SEL\_SIZE*** T0\_DPLL\_APLL\_PATH\_CNFG  
***BF\_T0\_12E1\_GPS\_E3\_T3\_SEL\_MASK***  
***BF\_T0\_12E1\_GPS\_E3\_T3\_SEL\_LSB***  
***BF\_T0\_DPLL\_START\_DAMPING\_SIZE*** T0\_DPLL\_START\_BW\_DAMPING\_CNFG  
***BF\_T0\_DPLL\_START\_DAMPING\_MASK***  
***BF\_T0\_DPLL\_START\_DAMPING\_LSB***  
***BF\_T0\_DPLL\_START\_BW\_SIZE*** T0\_DPLL\_START\_BW\_DAMPING\_CNFG  
***BF\_T0\_DPLL\_START\_BW\_MASK***  
***BF\_T0\_DPLL\_START\_BW\_LSB***  
***BF\_T0\_DPLL\_ACQ\_DAMPING\_SIZE*** T0\_DPLL\_ACQ\_BW\_DAMPING\_CNFG  
***BF\_T0\_DPLL\_ACQ\_DAMPING\_MASK***  
***BF\_T0\_DPLL\_ACQ\_DAMPING\_LSB***  
***BF\_T0\_DPLL\_ACQ\_BW\_SIZE*** T0\_DPLL\_ACQ\_BW\_DAMPING\_CNFG  
***BF\_T0\_DPLL\_ACQ\_BW\_MASK***  
***BF\_T0\_DPLL\_ACQ\_BW\_LSB***  
***BF\_T0\_DPLL\_LOCKED\_DAMPING\_SIZE*** T0\_DPLL\_LOCKED\_BW\_DAMPING\_CNFG  
***BF\_T0\_DPLL\_LOCKED\_DAMPING\_MASK***  
***BF\_T0\_DPLL\_LOCKED\_DAMPING\_LSB***  
***BF\_T0\_DPLL\_LOCKED\_BW\_SIZE*** T0\_DPLL\_LOCKED\_BW\_DAMPING\_CNFG  
***BF\_T0\_DPLL\_LOCKED\_BW\_MASK***  
***BF\_T0\_DPLL\_LOCKED\_BW\_LSB***  
***BF\_AUTO\_BW\_SEL\_SIZE*** T0\_BW\_OVERSHOOT\_CNFG  
***BF\_AUTO\_BW\_SEL\_MASK***  
***BF\_AUTO\_BW\_SEL\_LSB***  
***BF\_T0\_LIMIT\_SIZE*** T0\_BW\_OVERSHOOT\_CNFG

***BF\_T0\_LIMIT\_MASK***  
***BF\_T0\_LIMIT\_LSB***  
***BF\_COARSE\_PH\_LOS\_LIMT\_EN\_SIZE*** PHASE\_LOSS\_COARSE\_LIMIT\_CNFG  
***BF\_COARSE\_PH\_LOS\_LIMT\_EN\_MASK***  
***BF\_COARSE\_PH\_LOS\_LIMT\_EN\_LSB***  
***BF\_WIDE\_EN\_SIZE*** PHASE\_LOSS\_COARSE\_LIMIT\_CNFG  
***BF\_WIDE\_EN\_MASK***  
***BF\_WIDE\_EN\_LSB***  
***BF\_MULTI\_PH\_APP\_SIZE*** PHASE\_LOSS\_COARSE\_LIMIT\_CNFG  
***BF\_MULTI\_PH\_APP\_MASK***  
***BF\_MULTI\_PH\_APP\_LSB***  
***BF\_MULTI\_PH\_8K\_4K\_2K\_EN\_SIZE*** PHASE\_LOSS\_COARSE\_LIMIT\_CNFG  
***BF\_MULTI\_PH\_8K\_4K\_2K\_EN\_MASK***  
***BF\_MULTI\_PH\_8K\_4K\_2K\_EN\_LSB***  
***BF\_PH\_LOS\_COARSE\_LIMT\_SIZE*** PHASE\_LOSS\_COARSE\_LIMIT\_CNFG  
***BF\_PH\_LOS\_COARSE\_LIMT\_MASK***  
***BF\_PH\_LOS\_COARSE\_LIMT\_LSB***  
***BF\_FINE\_PH\_LOS\_LIMT\_EN\_SIZE*** PHASE\_LOSS\_FINE\_LIMIT\_CNFG  
***BF\_FINE\_PH\_LOS\_LIMT\_EN\_MASK***  
***BF\_FINE\_PH\_LOS\_LIMT\_EN\_LSB***  
***BF\_FAST\_LOS\_SW\_SIZE*** PHASE\_LOSS\_FINE\_LIMIT\_CNFG  
***BF\_FAST\_LOS\_SW\_MASK***  
***BF\_FAST\_LOS\_SW\_LSB***  
***BF\_PH\_LOS\_FINE\_LIMT\_SIZE*** PHASE\_LOSS\_FINE\_LIMIT\_CNFG  
***BF\_PH\_LOS\_FINE\_LIMT\_MASK***  
***BF\_PH\_LOS\_FINE\_LIMT\_LSB***  
***BF\_MAN\_HOLDOVER\_SIZE*** T0\_HOLDOVER\_MODE\_CNFG  
***BF\_MAN\_HOLDOVER\_MASK***  
***BF\_MAN\_HOLDOVER\_LSB***  
***BF\_AUTO\_AVG\_SIZE*** T0\_HOLDOVER\_MODE\_CNFG  
***BF\_AUTO\_AVG\_MASK***  
***BF\_AUTO\_AVG\_LSB***  
***BF\_FAST\_AVG\_SIZE*** T0\_HOLDOVER\_MODE\_CNFG  
***BF\_FAST\_AVG\_MASK***  
***BF\_FAST\_AVG\_LSB***  
***BF\_READ\_AVG\_SIZE*** T0\_HOLDOVER\_MODE\_CNFG  
***BF\_READ\_AVG\_MASK***  
***BF\_READ\_AVG\_LSB***  
***BF\_TEMP\_HOLDOVER\_MODE\_SIZE*** T0\_HOLDOVER\_MODE\_CNFG  
***BF\_TEMP\_HOLDOVER\_MODE\_MASK***  
***BF\_TEMP\_HOLDOVER\_MODE\_LSB***  
***BF\_HOLDOVER\_FREQ\_A\_SIZE*** HOLDOVER\_FREQ\_LOW\_CNFG  
***BF\_HOLDOVER\_FREQ\_A\_MASK***  
***BF\_HOLDOVER\_FREQ\_A\_LSB***  
***BF\_HOLDOVER\_FREQ\_B\_SIZE*** HOLDOVER\_FREQ\_MID\_CNFG

**BF\_HOLD OVER\_FREQ\_B\_MASK**  
**BF\_HOLD OVER\_FREQ\_B\_LSB**  
**BF\_HOLD OVER\_FREQ\_C\_SIZE** HOLD OVER\_FREQ\_HIGH\_CNFG  
**BF\_HOLD OVER\_FREQ\_C\_MASK**  
**BF\_HOLD OVER\_FREQ\_C\_LSB**  
**BF\_T4\_APLL\_PATH\_SIZE** T4\_DPLL\_APLL\_PATH\_CNFG  
**BF\_T4\_APLL\_PATH\_MASK**  
**BF\_T4\_APLL\_PATH\_LSB**  
**BF\_T4\_GSM\_OBSAI\_16E1\_16T1\_SEL\_SIZE** T4\_DPLL\_APLL\_PATH\_CNFG  
**BF\_T4\_GSM\_OBSAI\_16E1\_16T1\_SEL\_MASK**  
**BF\_T4\_GSM\_OBSAI\_16E1\_16T1\_SEL\_LSB**  
**BF\_T4\_12E1\_GPS\_E3\_T3\_SEL\_SIZE** T4\_DPLL\_APLL\_PATH\_CNFG  
**BF\_T4\_12E1\_GPS\_E3\_T3\_SEL\_MASK**  
**BF\_T4\_12E1\_GPS\_E3\_T3\_SEL\_LSB**  
**BF\_T4\_DPLL\_LOCKED\_DAMPING\_SIZE** T4\_DPLL\_LOCKED\_BW\_DAMPING\_CNFG  
**BF\_T4\_DPLL\_LOCKED\_DAMPING\_MASK**  
**BF\_T4\_DPLL\_LOCKED\_DAMPING\_LSB**  
**BF\_T4\_DPLL\_LOCKED\_BW\_SIZE** T4\_DPLL\_LOCKED\_BW\_DAMPING\_CNFG  
**BF\_T4\_DPLL\_LOCKED\_BW\_MASK**  
**BF\_T4\_DPLL\_LOCKED\_BW\_LSB**  
**BF\_CURRENT\_DPLL\_FREQ\_A\_SIZE** CURRENT\_FREQ\_LOW\_STS  
**BF\_CURRENT\_DPLL\_FREQ\_A\_MASK**  
**BF\_CURRENT\_DPLL\_FREQ\_A\_LSB**  
**BF\_CURRENT\_DPLL\_FREQ\_B\_SIZE** CURRENT\_FREQ\_MID\_STS  
**BF\_CURRENT\_DPLL\_FREQ\_B\_MASK**  
**BF\_CURRENT\_DPLL\_FREQ\_B\_LSB**  
**BF\_CURRENT\_DPLL\_FREQ\_C\_SIZE** CURRENT\_FREQ\_HIGH\_STS  
**BF\_CURRENT\_DPLL\_FREQ\_C\_MASK**  
**BF\_CURRENT\_DPLL\_FREQ\_C\_LSB**  
**BF\_FREQ\_LIMT\_PH\_LOS\_SIZE** DPLL\_FREQ\_SOFT\_LIMIT\_CNFG  
**BF\_FREQ\_LIMT\_PH\_LOS\_MASK**  
**BF\_FREQ\_LIMT\_PH\_LOS\_LSB**  
**BF\_DPLL\_FREQ\_SOFT\_LIMT\_SIZE** DPLL\_FREQ\_SOFT\_LIMIT\_CNFG  
**BF\_DPLL\_FREQ\_SOFT\_LIMT\_MASK**  
**BF\_DPLL\_FREQ\_SOFT\_LIMT\_LSB**  
**BF\_DPLL\_FREQ\_HARD\_LIMT\_A\_SIZE** DPLL\_FREQ\_HARD\_LIMIT\_LOW\_CNFG  
**BF\_DPLL\_FREQ\_HARD\_LIMT\_A\_MASK**  
**BF\_DPLL\_FREQ\_HARD\_LIMT\_A\_LSB**  
**BF\_DPLL\_FREQ\_HARD\_LIMT\_B\_SIZE** DPLL\_FREQ\_HARD\_LIMIT\_MID\_CNFG  
**BF\_DPLL\_FREQ\_HARD\_LIMT\_B\_MASK**  
**BF\_DPLL\_FREQ\_HARD\_LIMT\_B\_LSB**  
**BF\_CURRENT\_PH\_DATA\_A\_SIZE** CURRENT\_DPLL\_PHASE\_LOW\_STS  
**BF\_CURRENT\_PH\_DATA\_A\_MASK**  
**BF\_CURRENT\_PH\_DATA\_A\_LSB**  
**BF\_CURRENT\_PH\_DATA\_B\_SIZE** CURRENT\_DPLL\_PHASE\_HIGH\_STS

***BF\_CURRENT\_PH\_DATA\_B\_MASK***  
***BF\_CURRENT\_PH\_DATA\_B\_LSB***  
***BF\_OUT1\_PATH\_SEL\_SIZE*** OUT1\_FREQ\_CNFG  
***BF\_OUT1\_PATH\_SEL\_MASK***  
***BF\_OUT1\_PATH\_SEL\_LSB***  
***BF\_OUT1\_DIVIDER\_SIZE*** OUT1\_FREQ\_CNFG  
***BF\_OUT1\_DIVIDER\_MASK***  
***BF\_OUT1\_DIVIDER\_LSB***  
***BF\_OUT2\_PATH\_SEL\_SIZE*** OUT2\_FREQ\_CNFG  
***BF\_OUT2\_PATH\_SEL\_MASK***  
***BF\_OUT2\_PATH\_SEL\_LSB***  
***BF\_OUT2\_DIVIDER\_SIZE*** OUT2\_FREQ\_CNFG  
***BF\_OUT2\_DIVIDER\_MASK***  
***BF\_OUT2\_DIVIDER\_LSB***  
***BF\_OUT3\_PATH\_SEL\_SIZE*** OUT3\_FREQ\_CNFG  
***BF\_OUT3\_PATH\_SEL\_MASK***  
***BF\_OUT3\_PATH\_SEL\_LSB***  
***BF\_OUT3\_DIVIDER\_SIZE*** OUT3\_FREQ\_CNFG  
***BF\_OUT3\_DIVIDER\_MASK***  
***BF\_OUT3\_DIVIDER\_LSB***  
***BF\_OUT4\_PATH\_SEL\_SIZE*** OUT4\_FREQ\_CNFG  
***BF\_OUT4\_PATH\_SEL\_MASK***  
***BF\_OUT4\_PATH\_SEL\_LSB***  
***BF\_OUT4\_DIVIDER\_SIZE*** OUT4\_FREQ\_CNFG  
***BF\_OUT4\_DIVIDER\_MASK***  
***BF\_OUT4\_DIVIDER\_LSB***  
***BF\_OUT5\_PATH\_SEL\_SIZE*** OUT5\_FREQ\_CNFG  
***BF\_OUT5\_PATH\_SEL\_MASK***  
***BF\_OUT5\_PATH\_SEL\_LSB***  
***BF\_OUT5\_DIVIDER\_SIZE*** OUT5\_FREQ\_CNFG  
***BF\_OUT5\_DIVIDER\_MASK***  
***BF\_OUT5\_DIVIDER\_LSB***  
***BF\_OUT6\_PATH\_SEL\_SIZE*** OUT6\_FREQ\_CNFG  
***BF\_OUT6\_PATH\_SEL\_MASK***  
***BF\_OUT6\_PATH\_SEL\_LSB***  
***BF\_OUT6\_DIVIDER\_SIZE*** OUT6\_FREQ\_CNFG  
***BF\_OUT6\_DIVIDER\_MASK***  
***BF\_OUT6\_DIVIDER\_LSB***  
***BF\_OUT7\_PATH\_SEL\_SIZE*** OUT7\_FREQ\_CNFG  
***BF\_OUT7\_PATH\_SEL\_MASK***  
***BF\_OUT7\_PATH\_SEL\_LSB***  
***BF\_OUT7\_DIVIDER\_SIZE*** OUT7\_FREQ\_CNFG  
***BF\_OUT7\_DIVIDER\_MASK***  
***BF\_OUT7\_DIVIDER\_LSB***  
***BF\_OUT8\_PATH\_SEL\_SIZE*** OUT8\_FREQ\_CNFG

***BF\_OUT8\_PATH\_SEL\_MASK***  
***BF\_OUT8\_PATH\_SEL\_LSB***  
***BF\_OUT8\_EN\_SIZE*** OUT8\_FREQ\_CNFG  
***BF\_OUT8\_EN\_MASK***  
***BF\_OUT8\_EN\_LSB***  
***BF\_T4\_INPUT\_FAIL\_SIZE*** OUT8\_FREQ\_CNFG OUT9\_FREQ\_CNFG  
***BF\_T4\_INPUT\_FAIL\_MASK***  
***BF\_T4\_INPUT\_FAIL\_LSB***  
***BF\_AMI\_OUT\_DUTY\_SIZE*** OUT8\_FREQ\_CNFG  
***BF\_AMI\_OUT\_DUTY\_MASK***  
***BF\_AMI\_OUT\_DUTY\_LSB***  
***BF\_400HZ\_OUT\_SEL\_SIZE*** OUT8\_FREQ\_CNFG  
***BF\_400HZ\_OUT\_SEL\_MASK***  
***BF\_400HZ\_OUT\_SEL\_LSB***  
***BF\_OUT9\_INV\_SIZE*** OUT8\_FREQ\_CNFG  
***BF\_OUT9\_INV\_MASK***  
***BF\_OUT9\_INV\_LSB***  
***BF\_OUT7\_INV\_SIZE*** OUT8\_FREQ\_CNFG  
***BF\_OUT7\_INV\_MASK***  
***BF\_OUT7\_INV\_LSB***  
***BF\_OUT6\_INV\_SIZE*** OUT8\_FREQ\_CNFG  
***BF\_OUT6\_INV\_MASK***  
***BF\_OUT6\_INV\_LSB***  
***BF\_OUT9\_PATH\_SEL\_SIZE*** OUT9\_FREQ\_CNFG  
***BF\_OUT9\_PATH\_SEL\_MASK***  
***BF\_OUT9\_PATH\_SEL\_LSB***  
***BF\_OUT9\_EN\_SIZE*** OUT9\_FREQ\_CNFG  
***BF\_OUT9\_EN\_MASK***  
***BF\_OUT9\_EN\_LSB***  
***BF\_OUT5\_INV\_SIZE*** OUT9\_FREQ\_CNFG  
***BF\_OUT5\_INV\_MASK***  
***BF\_OUT5\_INV\_LSB***  
***BF\_OUT4\_INV\_SIZE*** OUT9\_FREQ\_CNFG  
***BF\_OUT4\_INV\_MASK***  
***BF\_OUT4\_INV\_LSB***  
***BF\_OUT3\_INV\_SIZE*** OUT9\_FREQ\_CNFG  
***BF\_OUT3\_INV\_MASK***  
***BF\_OUT3\_INV\_LSB***  
***BF\_OUT2\_INV\_SIZE*** OUT9\_FREQ\_CNFG  
***BF\_OUT2\_INV\_MASK***  
***BF\_OUT2\_INV\_LSB***  
***BF\_OUT1\_INV\_SIZE*** OUT9\_FREQ\_CNFG  
***BF\_OUT1\_INV\_MASK***  
***BF\_OUT1\_INV\_LSB***  
***BF\_IN\_2K\_4K\_8K\_INV\_SIZE*** FR\_MFR\_SYNC\_CNFG

***BF\_IN\_2K\_4K\_8K\_INV\_MASK***  
***BF\_IN\_2K\_4K\_8K\_INV\_LSB***  
***BF\_8K\_EN\_SIZE*** FR\_MFR\_SYNC\_CNFG  
***BF\_8K\_EN\_MASK***  
***BF\_8K\_EN\_LSB***  
***BF\_2K\_EN\_SIZE*** FR\_MFR\_SYNC\_CNFG  
***BF\_2K\_EN\_MASK***  
***BF\_2K\_EN\_LSB***  
***BF\_2K\_8K\_PUL\_POSITION\_SIZE*** FR\_MFR\_SYNC\_CNFG  
***BF\_2K\_8K\_PUL\_POSITION\_MASK***  
***BF\_2K\_8K\_PUL\_POSITION\_LSB***  
***BF\_8K\_INV\_SIZE*** FR\_MFR\_SYNC\_CNFG  
***BF\_8K\_INV\_MASK***  
***BF\_8K\_INV\_LSB***  
***BF\_8K\_PUL\_SIZE*** FR\_MFR\_SYNC\_CNFG  
***BF\_8K\_PUL\_MASK***  
***BF\_8K\_PUL\_LSB***  
***BF\_2K\_INV\_SIZE*** FR\_MFR\_SYNC\_CNFG  
***BF\_2K\_INV\_MASK***  
***BF\_2K\_INV\_LSB***  
***BF\_2K\_PUL\_SIZE*** FR\_MFR\_SYNC\_CNFG  
***BF\_2K\_PUL\_MASK***  
***BF\_2K\_PUL\_LSB***  
***BF\_IN\_NOISE\_WINDOW\_SIZE*** PHASE\_MON\_CNFG  
***BF\_IN\_NOISE\_WINDOW\_MASK***  
***BF\_IN\_NOISE\_WINDOW\_LSB***  
***BF\_PH\_OFFSET\_A\_SIZE*** PHASE\_OFFSET\_LOW\_CNFG  
***BF\_PH\_OFFSET\_A\_MASK***  
***BF\_PH\_OFFSET\_A\_LSB***  
***BF\_PH\_OFFSET\_EN\_SIZE*** PHASE\_OFFSET\_HIGH\_CNFG  
***BF\_PH\_OFFSET\_EN\_MASK***  
***BF\_PH\_OFFSET\_EN\_LSB***  
***BF\_PH\_OFFSET\_B\_SIZE*** PHASE\_OFFSET\_HIGH\_CNFG  
***BF\_PH\_OFFSET\_B\_MASK***  
***BF\_PH\_OFFSET\_B\_LSB***  
***BF\_SYNC\_BYPASS\_SIZE*** SYNC\_MONITOR\_CNFG  
***BF\_SYNC\_BYPASS\_MASK***  
***BF\_SYNC\_BYPASS\_LSB***  
***BF\_SYNC\_MON\_LIMT\_SIZE*** SYNC\_MONITOR\_CNFG  
***BF\_SYNC\_MON\_LIMT\_MASK***  
***BF\_SYNC\_MON\_LIMT\_LSB***  
***BF\_SYNC\_EDGE\_SIZE*** SYNC\_PHASE\_CNFG  
***BF\_SYNC\_EDGE\_MASK***  
***BF\_SYNC\_EDGE\_LSB***  
***BF\_SYNC\_PH2\_SIZE*** SYNC\_PHASE\_CNFG

***BF\_SYNC\_PH2\_MASK***  
***BF\_SYNC\_PH2\_LSB***  
***BF\_SYNC\_PH1\_SIZE*** SYNC\_PHASE\_CNFG  
***BF\_SYNC\_PH1\_MASK***  
***BF\_SYNC\_PH1\_LSB***  
***BF\_PROTECTION\_DATA\_SIZE*** PROTECTION\_CNFG  
***BF\_PROTECTION\_DATA\_MASK***  
***BF\_PROTECTION\_DATA\_LSB***  
***BF\_MPU\_SEL\_CNFG\_SIZE*** MPU\_SEL\_CNFG  
***BF\_MPU\_SEL\_CNFG\_MASK***  
***BF\_MPU\_SEL\_CNFG\_LSB***  
***BF\_T0\_DFS\_OFF\_3\_SIZE*** DFS\_OFF\_CNFG  
***BF\_T0\_DFS\_OFF\_3\_MASK***  
***BF\_T0\_DFS\_OFF\_3\_LSB***  
***BF\_T0\_DFS\_OFF\_2\_SIZE*** DFS\_OFF\_CNFG  
***BF\_T0\_DFS\_OFF\_2\_MASK***  
***BF\_T0\_DFS\_OFF\_2\_LSB***  
***BF\_T0\_DFS\_OFF\_1\_SIZE*** DFS\_OFF\_CNFG  
***BF\_T0\_DFS\_OFF\_1\_MASK***  
***BF\_T0\_DFS\_OFF\_1\_LSB***  
***BF\_T0\_DFS\_OFF\_0\_SIZE*** DFS\_OFF\_CNFG  
***BF\_T0\_DFS\_OFF\_0\_MASK***  
***BF\_T0\_DFS\_OFF\_0\_LSB***  
***BF\_T4\_DFS\_OFF\_3\_SIZE*** DFS\_OFF\_CNFG  
***BF\_T4\_DFS\_OFF\_3\_MASK***  
***BF\_T4\_DFS\_OFF\_3\_LSB***  
***BF\_T4\_DFS\_OFF\_2\_SIZE*** DFS\_OFF\_CNFG  
***BF\_T4\_DFS\_OFF\_2\_MASK***  
***BF\_T4\_DFS\_OFF\_2\_LSB***  
***BF\_T4\_DFS\_OFF\_1\_SIZE*** DFS\_OFF\_CNFG  
***BF\_T4\_DFS\_OFF\_1\_MASK***  
***BF\_T4\_DFS\_OFF\_1\_LSB***  
***BF\_T4\_DFS\_OFF\_0\_SIZE*** DFS\_OFF\_CNFG  
***BF\_T4\_DFS\_OFF\_0\_MASK***  
***BF\_T4\_DFS\_OFF\_0\_LSB***  
***BF\_PPS\_PHASE\_SIZE*** T0\_PPS\_CNFG T4\_PPS\_CNFG  
***BF\_PPS\_PHASE\_MASK***  
***BF\_PPS\_PHASE\_LSB***  
***BF\_PPS\_PULSE\_SIZE*** T0\_PPS\_CNFG T4\_PPS\_CNFG  
***BF\_PPS\_PULSE\_MASK***  
***BF\_PPS\_PULSE\_LSB***  
***BF\_T0\_PH\_SLOPE\_SIZE*** PH\_SLOPE\_CNFG  
***BF\_T0\_PH\_SLOPE\_MASK***  
***BF\_T0\_PH\_SLOPE\_LSB***  
***BF\_T4\_PH\_SLOPE\_SIZE*** PH\_SLOPE\_CNFG

**BF\_T4\_PH\_SLOPE\_MASK**  
**BF\_T4\_PH\_SLOPE\_LSB**  
**BF\_ICP\_CTRL\_CODE\_SIZE** T0\_ICP\_CTRL\_CNFG T4\_ICP\_CTRL\_CNFG  
**BF\_ICP\_CTRL\_CODE\_MASK**  
**BF\_ICP\_CTRL\_CODE\_LSB**  
**REGMAP\_MAX**

Definition at line 55 of file RegisterDef.h.

#### 4.7.2.2 anonymous enum

Enumerated type listing register offcers and bit field constants.

Enumerator

**REG\_CONTROL** General Control Register  
**REG\_CLK\_SEL** Input Clock Select  
**REG\_PDSEL0\_LSB** Configuration 0 Pre-divider LSB  
**REG\_PDSEL0\_MSB** Configuration 0 Pre-divider MSB  
**REG\_PDSEL1\_LSB** Configuration 1 Pre-divider LSB  
**REG\_PDSEL1\_MSB** Configuration 1 Pre-divider MSB  
**REG\_M0\_LSB** Configuration 0 Feedback divider LSB  
**REG\_M0\_MSB** Configuration 0 Feedback divider MSB  
**REG\_M1\_LSB** Configuration 1 Feedback divider LSB  
**REG\_M1\_MSB** Configuration 1 Feedback divider MSB  
**REG\_ODBSELA0** QA Output Divider Config 0  
**REG\_ODBSELA1** QA Output Divider Config 1  
**REG\_ODBSELB0** QB Output Divider Config 0  
**REG\_ODBSELB1** QB Output Divider Config 1  
**REG\_VCXO\_SEL** VCXO Crystal Select  
**REG\_CONFIG\_QA** Output Configuration QA  
**REG\_CONFIG\_QB** Output Configuration QB  
**BF\_CONFIG\_SIZE** CONTROL  
**BF\_CONFIG\_MASK**  
**BF\_CONFIG\_LSB**  
**BF\_CLK\_SEL\_SIZE** CLK\_SEL  
**BF\_CLK\_SEL\_MASK**  
**BF\_CLK\_SEL\_LSB**  
**BF\_PDSEL0\_LSB\_SIZE** PDSEL0\_LSB  
**BF\_PDSEL0\_LSB\_MASK**  
**BF\_PDSEL0\_LSB\_LSB**  
**BF\_PDSEL0\_MSB\_SIZE** PDSEL0\_MSB  
**BF\_PDSEL0\_MSB\_MASK**  
**BF\_PDSEL0\_MSB\_LSB**  
**BF\_PDSEL1\_LSB\_SIZE** PDSEL1\_LSB  
**BF\_PDSEL1\_LSB\_MASK**  
**BF\_PDSEL1\_LSB\_LSB**

***BF\_PDSEL1\_MSB\_SIZE*** PDSEL1\_MSB  
***BF\_PDSEL1\_MSB\_MASK***  
***BF\_PDSEL1\_MSB\_LSB***  
***BF\_M0\_LSB\_SIZE*** M0\_LSB  
***BF\_M0\_LSB\_MASK***  
***BF\_M0\_LSB\_LSB***  
***BF\_M0\_MSB\_SIZE*** M0\_MSB  
***BF\_M0\_MSB\_MASK***  
***BF\_M0\_MSB\_LSB***  
***BF\_M1\_LSB\_SIZE*** M1\_LSB  
***BF\_M1\_LSB\_MASK***  
***BF\_M1\_LSB\_LSB***  
***BF\_M1\_MSB\_SIZE*** M1\_MSB  
***BF\_M1\_MSB\_MASK***  
***BF\_M1\_MSB\_LSB***  
***BF\_ODBSELA0\_SIZE*** ODBSELA0  
***BF\_ODBSELA0\_MASK***  
***BF\_ODBSELA0\_LSB***  
***BF\_ODBSELA1\_SIZE*** ODBSELA1  
***BF\_ODBSELA1\_MASK***  
***BF\_ODBSELA1\_LSB***  
***BF\_ODBSELB0\_SIZE*** ODBSELB0  
***BF\_ODBSELB0\_MASK***  
***BF\_ODBSELB0\_LSB***  
***BF\_ODBSELB1\_SIZE*** ODBSELB1  
***BF\_ODBSELB1\_MASK***  
***BF\_ODBSELB1\_LSB***  
***BF\_SEL\_DOUBLE\_SIZE*** VCXO\_SEL  
***BF\_SEL\_DOUBLE\_MASK***  
***BF\_SEL\_DOUBLE\_LSB***  
***BF\_MR\_SYNC\_SIZE*** VCXO\_SEL  
***BF\_MR\_SYNC\_MASK***  
***BF\_MR\_SYNC\_LSB***  
***BF\_BYPASS\_SIZE*** VCXO\_SEL  
***BF\_BYPASS\_MASK***  
***BF\_BYPASS\_LSB***  
***BF\_SHUTOFF\_SIZE*** VCXO\_SEL  
***BF\_SHUTOFF\_MASK***  
***BF\_SHUTOFF\_LSB***  
***BF\_VCXO\_SEL\_SIZE*** VCXO\_SEL  
***BF\_VCXO\_SEL\_MASK***  
***BF\_VCXO\_SEL\_LSB***  
***BF\_LVDS\_QA\_SIZE*** CONFIG\_QA  
***BF\_LVDS\_QA\_MASK***  
***BF\_LVDS\_QA\_LSB***

***BF\_OE\_A\_SIZE*** CONFIG\_QA  
***BF\_OE\_A\_MASK***  
***BF\_OE\_A\_LSB***  
***BF\_LVDS\_QB\_SIZE*** CONFIG\_QB  
***BF\_LVDS\_QB\_MASK***  
***BF\_LVDS\_QB\_LSB***  
***BF\_OE\_B\_SIZE*** CONFIG\_QB  
***BF\_OE\_B\_MASK***  
***BF\_OE\_B\_LSB***  
***REGMAP\_APLL\_MAX***

Definition at line 55 of file ApIIRegDef.h.

## 4.8 AplI Function Module

Functions in this group are related to APLL provisioning.

### Data Structures

- struct [T\\_idtAplIConfigPerOutput](#)  
*Structure containing APLL per output configuration.*
- struct [T\\_idtAplIConfig](#)  
*Structure containing APLL full configuration.*
- struct [T\\_idtAplIFreqMapping](#)  
*Type definition for mapping output frequency to APLL configuration.*
- struct [T\\_idtAplIXtal](#)  
*Structure containing APLL crystal id and frequency information.*

### Macros

- #define [IDT\\_GET\\_OUTPUT\\_APLL\\_CONFIG](#)(hdlr, output, outputAplI)  
*Utility macro to translate output port to APLL instance.*
- #define [IDT\\_APLL\\_API\\_LOG\\_HEADER](#)
- #define [IDT\\_APLL\\_API\\_LOG\\_ERR](#)(fmt)
- #define [IDT\\_APLL\\_API\\_LOG\\_INFO](#)(fmt)
- #define [IDT\\_APLL\\_API\\_LOG\\_DEBUG](#)(fmt)
- #define [IDT\\_APLL\\_API\\_TRACE](#)(fmt) OS\_TRACE(fmt)
- #define [INVALID\\_DIVIDER\\_VALUE](#) (0xFFFF)  
*Define constant indicating invalid divider value.*

### Typedefs

- typedef T\_osUInt16 [T\\_idtAplIDividerValue](#)  
*Type define for APLL divider value.*
- typedef T\_osUInt8 [T\\_idtAplIRegAddr](#)  
*Type define for APLL register address.*
- typedef T\_osUInt8 [T\\_idtAplIRegMask](#)  
*Type define for APLL register value mask.*
- typedef T\_osUInt8 [T\\_idtAplIRegVal](#)  
*Type define for APLL register value.*

### Enumerations

- enum [T\\_idtAplIInputFreqType](#) {  
[APLL\\_INPUT\\_FREQ\\_19440KHz](#), [APLL\\_INPUT\\_FREQ\\_25000KHz](#), [APLL\\_INPUT\\_FREQ\\_77760KHz](#), [APLL\\_INPUT\\_FREQ\\_125000KHz](#),  
[APLL\\_INPUT\\_FREQ\\_155520KHz](#), [APLL\\_INPUT\\_FREQ\\_25781\\_DOT\\_25KHz](#), [APLL\\_INPUT\\_FREQ\\_312500KHz](#), [APLL\\_INPUT\\_FREQ\\_128906\\_DOT\\_25KHz](#),  
[APLL\\_INPUT\\_FREQ\\_156250KHz](#), [APLL\\_INPUT\\_FREQ\\_8KHz](#), [APLL\\_INPUT\\_FREQ\\_MAX](#) }  
*Enumerated type listing supported input frequencies.*

- enum `T_idtApIiOutputFreqType` {  
`APLL_OUTPUT_FREQ_24883_DOT_2KHz`, `APLL_OUTPUT_FREQ_25000KHz`, `APLL_OUTPUT_FREQ_25781_DOT_25KHz`, `APLL_OUTPUT_FREQ_77760KHz`,  
`APLL_OUTPUT_FREQ_78125KHz`, `APLL_OUTPUT_FREQ_80566_DOT_40625KHz`, `APLL_OUTPUT_FREQ_124416KHz`, `APLL_OUTPUT_FREQ_125000KHz`,  
`APLL_OUTPUT_FREQ_128906_DOT_25KHz`, `APLL_OUTPUT_FREQ_155520KHz`, `APLL_OUTPUT_FREQ_156250KHz`, `APLL_OUTPUT_FREQ_161132_DOT_8125KHz`,  
`APLL_OUTPUT_FREQ_311040KHz`, `APLL_OUTPUT_FREQ_312500KHz`, `APLL_OUTPUT_FREQ_322265_DOT_625KHz`, `APLL_OUTPUT_FREQ_622080KHz`,  
`APLL_OUTPUT_FREQ_625000KHz`, `APLL_OUTPUT_FREQ_644531_DOT_25KHz`, `APLL_OUTPUT_FREQ_MAX` }  
*Enumerated type listing supported output frequencies.*
- enum `T_idtApIiId` { `APLL_INST_1`, `APLL_INST_2`, `APLL_INST_MAX` }  
*Enumerated type listing APLL instance within device.*
- enum `T_idtApIiAccessType` { `APLL_ACCESS_I2C`, `APLL_ACCESS_SIM`, `APLL_ACCESS_MAX` }  
*Enumerated type listing supported access methods.*
- enum `T_idtApIiConfigSel` { `APLL_CONFIG0 = 0`, `APLL_CONFIG1 = 1`, `APLL_CONFIG_MAX` }  
*Enumerated type listing possible configuration for each APLL.*
- enum `T_idtApIiInputClkSrc` { `APLL_CLK_SEL_EXTERNAL`, `APLL_CLK_SEL_INTERNAL`, `APLL_CLK_SEL_MAX` }  
*Enumerated type listing input ports.*
- enum `T_idtApIiOutput` { `APLL_OUTPUT_QA`, `APLL_OUTPUT_QB`, `APLL_OUTPUT_MAX` }  
*Enumerated type listing output ports.*
- enum `T_idtApIiOutputEn` { `APLL_OUTPUT_DISABLE`, `APLL_OUTPUT_ENABLE`, `APLL_OUTPUT_EN_MAX` }  
*Enumerated type listing output port status (enable/disable)*
- enum `T_idtApIiOutputSigType` { `APLL_OUT_SIG_LVPECL`, `APLL_OUT_SIG_LVDS`, `APLL_OUT_SIG_MAX` }  
*Enumerated type listing output signal type.*
- enum `T_idtApIiOutputSupportedType` { `APLL_OUT_QA_ONLY`, `APLL_OUT_QB_ONLY`, `APLL_OUT_QA_OR_QB`, `APLL_OUT_MAX` }  
*Enumerated type listing frequency supported by the output.*
- enum `T_idtApIiQaDiv` {  
`APLL_QA_ODSEL_DIV_25 = 0`, `APLL_QA_ODSEL_DIV_5 = 1`, `APLL_QA_ODSEL_DIV_4 = 2`, `APLL_QA_ODSEL_DIV_2 = 3`,  
`APLL_QA_ODSEL_DIV_1 = 4`, `APLL_QA_ODSEL_DIV_MAX` }  
*Enumerated type listing output port A divider values.*
- enum `T_idtApIiQbDiv` {  
`APLL_QB_ODSEL_DIV_8 = 0`, `APLL_QB_ODSEL_DIV_5 = 1`, `APLL_QB_ODSEL_DIV_4 = 2`, `APLL_QB_ODSEL_DIV_2 = 3`,  
`APLL_QB_ODSEL_DIV_1 = 4`, `APLL_QB_ODSEL_DIV_MAX` }  
*Enumerated type listing output port B divider values.*
- enum `T_idtApIiFreqDoublersSel` { `APLL_FREQ_SEL_SINGLE = 0`, `APLL_FREQ_SEL_DOUBLE = 1`, `APLL_FREQ_SEL_MAX` }  
*Enumerated type listing feedback frequency doubling values.*
- enum `T_idtApIiXtalId` {  
`APLL_XTAL_1_APLL1`, `APLL_XTAL_2_APLL2`, `APLL_XTAL_3_APLL1`, `APLL_XTAL_4_APLL2`,  
`APLL_XTAL_MAX` }  
*Enumerated type listing crystals.*
- enum `T_idtApIiXtalType` { `APLL_XTAL_FREQ_25000KHz`, `APLL_XTAL_FREQ_25781_DOT_25KHz`, `APLL_XTAL_FREQ_24883_DOT_2KHz`, `APLL_XTAL_FREQ_MAX` }  
*Enumerated type listing support crystal frequencies.*
- enum `T_idtApIiMasterResetSync` { `APLL_OUT_MR_SYNC = 0`, `APLL_IN_MR_SYNC = 1`, `APLL_MR_SYNC_MAX` }

*Enumerated type listing Device master reset sync status.*

- enum `T_idtAplBypass` { `APLL_BYPASS_NORMAL_OP` = 0, `APLL_BYPASS_BYPASS` = 1, `APLL_BYPASS_MAX` }

*Enumerated type listing Apl bypass status/setting.*

- enum `T_idtAplShutoff` { `APLL_SHUTOFF_NORMAL_OP` = 0, `APLL_SHUTOFF_SHUTOFF` = 1, `APLL_SHUTOFF_MAX` }

*Enumerated type listing Apl shutoff status/setting.*

## Functions

- void `IDTApl_SetAplConfigPerOutput` (`T_idtDrvHdlr` hdlr, `T_idtAplId` aplId, `T_idtAplOutput` output, `T_idtAplConfigPerOutput` \*aplConfigPerOutput)  
*Set APLL per output configuration.*
- void `IDTApl_GetAplConfigPerOutput` (`T_idtDrvHdlr` hdlr, `T_idtAplId` aplId, `T_idtAplOutput` output, `T_idtAplConfigPerOutput` \*aplConfigPerOutput)  
*Get APLL per output configuration.*
- void `IDTApl_SetAplConfig` (`T_idtDrvHdlr` hdlr, `T_idtAplId` aplId, `T_idtAplConfig` \*aplConfig)  
*Set APLL full configuration.*
- void `IDTApl_GetAplConfig` (`T_idtDrvHdlr` hdlr, `T_idtAplId` aplId, `T_idtAplConfig` \*aplConfig)  
*Get APLL full configuration.*
- `outputFrequency_t` `IDTApl_AplInput2DpllOutput` (`T_idtAplInputFreqType` aplInputFreq)  
*Translate APLL input frequency to DPLL output frequency.*
- `T_idtAplOutputFreqType` `IDTApl_DpllOutput2AplOutput` (`outputFrequency_t` dpllOutput)  
*Translate DPLL output frequency to APLL output frequency.*
- const `T_idtAplFreqMapping` \* `IDTApl_GetAplFreqTableEntry` (`T_idtDrvHdlr` hdlr, `T_idtOutputAplConfig` outputApl, `outputFrequency_t` nOutFreq)  
*Get APLL frequency configuration table entry based on the output frequency.*
- `T_idtAplConfigSel` `IDTApl_GetNonActiveAplConfig` (`T_idtDrvHdlr` hdlr, `T_idtAplId` aplId)  
*Get non-active APLL configuration.*
- void `IDTApl_Switch2NonActiveAplConfig` (`T_idtDrvHdlr` hdlr, `T_idtAplId` aplId)  
*Switch to non-active APLL configuration.*
- `T_idtAplXtalId` `IDTApl_GetXtalIdFromXtalFreq` (`T_idtDrvHdlr` hdlr, `T_idtAplId` aplId, `T_idtAplXtalType` aplVcxoFreq)  
*Get APLL configured crystal ID from crystal frequency.*
- `T_osBool` `IDTApl_XtalConfigured` (`T_idtDrvHdlr` hdlr, `T_idtAplId` aplId)  
*Checks if there is any XTAL configured for the APLL.*
- `T_osBool` `IDTApl_ValidXtalInput` (const `T_idtDrvDeviceConfig` \*deviceConfig, `T_idtAplXtal` \*xtalList, `T_osUInt8` numberOfXtal)  
*Checks if there is input XTAL information is valid for the device.*
- `T_idtAplXtal` \* `IDTApl_GetSupportedXtal` (const `T_idtDrvDeviceConfig` \*deviceConfig, `T_osUInt8` \*numberOfXtal)  
*Return supported xtal configuration by the device.*
- `T_idtAplConfigSel` `IDTApl_GetConfigSel` (`T_idtDrvHdlr` hdlr, `T_idtAplId` aplInstId)  
*Function to retrieve APLL configuration.*
- void `IDTApl_SetConfigSel` (`T_idtDrvHdlr` hdlr, `T_idtAplId` aplInstId, `T_idtAplConfigSel` aplConfig)  
*Function to provision APLL based on input configuration.*
- `T_idtAplInputClkSrc` `IDTApl_GetInputClkSrc` (`T_idtDrvHdlr` hdlr, `T_idtAplId` aplInstId)  
*Function to retrieve selected APLL input frequency source (external/internal)*
- void `IDTApl_SetInputClkSrc` (`T_idtDrvHdlr` hdlr, `T_idtAplId` aplInstId, `T_idtAplInputClkSrc` inputClk)  
*Function to provision APLL input frequency source (external/internal)*
- `T_idtAplDividerValue` `IDTApl_GetPreDiv` (`T_idtDrvHdlr` hdlr, `T_idtAplId` aplInstId, `T_idtAplConfigSel` aplConfig)

*Function to retrieve pre- (input) divider value.*

- void `IDTApll_GetPreDivRange` (T\_idtDrvHdlr hdlr, T\_idtApllDividerValue \*minDiv, T\_idtApllDividerValue \*maxDiv)

*Function to retrieve pre- (input) divisor range.*

- void `IDTApll_SetPreDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId, T\_idtApllConfigSel apIIConfig, T\_idtApllDividerValue div)

*Function to provision pre- (input) divider value.*

- T\_idtApllDividerValue `IDTApll_GetFeedbackDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId, T\_idtApllConfigSel apIIConfig)

*Function to retrieve feedback (oscillator) divider value.*

- void `IDTApll_GetFeedbackDivRange` (T\_idtDrvHdlr hdlr, T\_idtApllDividerValue \*minDiv, T\_idtApllDividerValue \*maxDiv)

*Function to retrieve feedback (oscillator) divisor range.*

- void `IDTApll_SetFeedbackDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId, T\_idtApllConfigSel apIIConfig, T\_idtApllDividerValue div)

*Function to provision feedback (oscillator) divider value.*

- T\_idtApllQaDiv `IDTApll_GetQaDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId, T\_idtApllConfigSel apIIConfig)

*Function to retrieve output port A divisor.*

- void `IDTApll_SetQaDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId, T\_idtApllConfigSel apIIConfig, T\_idtApllQaDiv div)

*Function to provision output port A divisor.*

- T\_idtApllQaDiv `IDTApll_GetQbDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId, T\_idtApllConfigSel apIIConfig)

*Function to retrieve output port B divisor.*

- void `IDTApll_SetQbDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId, T\_idtApllConfigSel apIIConfig, T\_idtApllQbDiv div)

*Function to provision output port B divisor.*

- T\_idtApllFreqDoublerSel `IDTApll_GetFreqDoublerSel` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId)

*Function to retrieve whether feedback frequency doubling is enabled.*

- void `IDTApll_SetFreqDoublerSel` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId, T\_idtApllFreqDoublerSel doubler)

*Function to enable/disable feedback frequency doubling.*

- T\_idtApllXtalId `IDTApll_GetXtalId` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId)

*Function to retrieve oscillator currently used.*

- void `IDTApll_SetXtalId` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId, T\_idtApllXtalId xtal)

*Function to select oscillator.*

- T\_idtApllOutputSigType `IDTApll_GetOutputSigType` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId, T\_idtApllOutput output)

*Function to retrieve output port signal type.*

- void `IDTApll_SetOutputSigType` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId, T\_idtApllOutput output, T\_idtApllOutputSigType outputSigType)

*Function to provision output port signal type.*

- T\_idtApllOutputEn `IDTApll_GetOutputEnState` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId, T\_idtApllOutput output)

*Function to retrieve output port status (enable/disable)*

- void `IDTApll_SetOutputEnState` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId, T\_idtApllOutput output, T\_idtApllOutputEn enable)

*Function to control (enable/disable) output port.*

- T\_idtApllMasterResetSync `IDTApll_GetMasterResetSync` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId)

*Function to retrieve reset status of device.*

- void `IDTApll_SetMasterResetSync` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId, T\_idtApllMasterResetSync reset)

*Function to reset APLL.*

- T\_idtApllBypass `IDTApll_GetBypass` (T\_idtDrvHdlr hdlr, T\_idtApllId apIIInstId)

*Function to retrieve bypass status of device.*

- void `IDTApl_SetBypass` (`T_idtDrvHdlr` hdlr, `T_idtAplId` apllInstId, `T_idtAplBypass` bypass)

*Function to set APLL bypass setting.*

- `T_idtAplShutoff` `IDTApl_GetShutoff` (`T_idtDrvHdlr` hdlr, `T_idtAplId` apllInstId)

*Function to retrieve shutoff status of device.*

- void `IDTApl_SetShutoff` (`T_idtDrvHdlr` hdlr, `T_idtAplId` apllInstId, `T_idtAplShutoff` shutoff)

*Function to set APLL shutoff setting.*

- const `T_idtAplFreqMapping` \* `IDTApl_GetConfigListByFreq` (`T_idtAplOutputFreqType` apllOutputFreq)

*Function to retrieve output configuration given an output frequency.*

- const `T_idtAplFreqMapping` \* `IDTApl_GetConfigListByXtalType` (`T_idtAplXtalType` apllXtalFreq)

*Function to retrieve output configuration given the oscillator type (frequency)*

- `T_osBool` `IDTApl_NullFreqTableEntry` (const `T_idtAplFreqMapping` \*apllFreqMapping)

*Function to check whether the input APLL frequency mapping is NULL (not configured)*

- const `T_idtAplFreqMapping` \* `IDTApl_GetCurrentApllFreqConfig` (`T_idtAplId` apllId, `T_idtAplOutput` output)

*Function to retrieve the current APLL frequency mapping configuration for the output.*

### 4.8.1 Detailed Description

Functions in this group are related to APLL provisioning.

### 4.8.2 Macro Definition Documentation

#### 4.8.2.1 #define IDT\_APLL\_API\_LOG\_DEBUG( fmt )

**Value:**

```
do { \
    OS_LOGGER(9, (IDT_APLL_API_LOG_HEADER)); \
    OS_LOGGER(9, fmt); \
} while (0)
```

Definition at line 65 of file `ApllLogging.h`.

#### 4.8.2.2 #define IDT\_APLL\_API\_LOG\_ERR( fmt )

**Value:**

```
do { \
    OS_ERROR(fmt); \
} while(0)
```

Definition at line 52 of file `ApllLogging.h`.

#### 4.8.2.3 #define IDT\_APLL\_API\_LOG\_HEADER

**Value:**

```
"[File: %s, Line: %d, Function: %s] ", \
    __FILE__, __LINE__, __FUNCTION__
```

Definition at line 47 of file `ApllLogging.h`.

**4.8.2.4 #define IDT\_APLL\_API\_LOG\_INFO( *fmt* )****Value:**

```
do { \
    OS_LOGGER(5, (IDT_APLL_API_LOG_HEADER)); \
    OS_LOGGER(5, fmt); \
} while(0)
```

Definition at line 58 of file ApIILogging.h.

**4.8.2.5 #define IDT\_APLL\_API\_TRACE( *fmt* ) OS\_TRACE(*fmt*)**

Definition at line 72 of file ApIILogging.h.

**4.8.2.6 #define IDT\_GET\_OUTPUT\_APLL\_CONFIG( *hdlr*, *output*, *outputApII* )****Value:**

```
do {
    OS_ASSERT((output) <= IDT_DRV_MAX_OUTPUT); \
    OS_ASSERT((output) > 0); \
    outputApII = hdlr->deviceConfig_m->outPutApIIConfig[output]; \
} while(0)
```

Utility macro to translate output port to APLL instance.

**Parameters**

|                   |                             |
|-------------------|-----------------------------|
| <i>hdlr</i>       | Device driver handle        |
| <i>output</i>     | External output port number |
| <i>outputApII</i> | APLL instance               |

Definition at line 60 of file ApII.h.

**4.8.2.7 #define INVALID\_DIVIDER\_VALUE (0xFFFF)**

Define constant indicating invalid divider value.

Definition at line 49 of file ApIITypeDef.h.

**4.8.3 Typedef Documentation****4.8.3.1 typedef T\_osUInt16 T\_idtApIIDividerValue**

Type define for APLL divider value.

Definition at line 224 of file ApIITypeDef.h.

**4.8.3.2 typedef T\_osUInt8 T\_idtApIIRegAddr**

Type define for APLL register address.

Definition at line 229 of file ApIITypeDef.h.

#### 4.8.3.3 typedef T\_osUInt8 T\_idtApIRegMask

Type define for APLL register value mask.

Definition at line 234 of file ApITypeDef.h.

#### 4.8.3.4 typedef T\_osUInt8 T\_idtApIRegVal

Type define for APLL register value.

Definition at line 239 of file ApITypeDef.h.

### 4.8.4 Enumeration Type Documentation

#### 4.8.4.1 enum T\_idtApIAccessType

Enumerated type listing supported access methods.

Enumerator

**APLL\_ACCESS\_I2C** Device access through I2C

**APLL\_ACCESS\_SIM** Simulated device

**APLL\_ACCESS\_MAX**

Definition at line 64 of file ApITypeDef.h.

#### 4.8.4.2 enum T\_idtApIBypass

Enumerated type listing ApI bypass status/setting.

Enumerator

**APLL\_BYPASS\_NORMAL\_OP** ApI in normal operation

**APLL\_BYPASS\_BYPASS** ApI bypass VCXO and connect selected clock input to output A and B

**APLL\_BYPASS\_MAX**

Definition at line 204 of file ApITypeDef.h.

#### 4.8.4.3 enum T\_idtApIConfigSel

Enumerated type listing possible configuration for each APLL.

Enumerator

**APLL\_CONFIG0** Configuration #1

**APLL\_CONFIG1** Configuration #2

**APLL\_CONFIG\_MAX**

Definition at line 74 of file ApITypeDef.h.

## 4.8.4.4 enum T\_idtApIIFreqDoublerSel

Enumerated type listing feedback frequency doubling values.

Enumerator

***APLL\_FREQ\_SEL\_SINGLE*** Feedback frequency not doubled  
***APLL\_FREQ\_SEL\_DOUBLE*** Feedback frequency doubled  
***APLL\_FREQ\_SEL\_MAX***

Definition at line 161 of file ApIITypeDef.h.

## 4.8.4.5 enum T\_idtApIIId

Enumerated type listing APLL instance within device.

Enumerator

***APLL\_INST\_1*** APLL #1  
***APLL\_INST\_2*** APLL #2  
***APLL\_INST\_MAX***

Definition at line 54 of file ApIITypeDef.h.

## 4.8.4.6 enum T\_idtApIIInputClkSrc

Enumerated type listing input ports.

Enumerator

***APLL\_CLK\_SEL\_EXTERNAL*** External clock source  
***APLL\_CLK\_SEL\_INTERNAL*** Internal clock source  
***APLL\_CLK\_SEL\_MAX***

Definition at line 84 of file ApIITypeDef.h.

## 4.8.4.7 enum T\_idtApIIInputFreqType

Enumerated type listing supported input frequencies.

Enumerator

***APLL\_INPUT\_FREQ\_19440KHz*** 19.4 MHz  
***APLL\_INPUT\_FREQ\_25000KHz*** 25.0 MHz  
***APLL\_INPUT\_FREQ\_77760KHz*** 77.76 MHz  
***APLL\_INPUT\_FREQ\_125000KHz*** 125 MHz  
***APLL\_INPUT\_FREQ\_155520KHz*** 155.52 MHz  
***APLL\_INPUT\_FREQ\_25781\_DOT\_25KHz*** 25.78125 MHz  
***APLL\_INPUT\_FREQ\_312500KHz*** 312.5 MHz  
***APLL\_INPUT\_FREQ\_128906\_DOT\_25KHz*** 128.90625 MHz  
***APLL\_INPUT\_FREQ\_156250KHz*** 156.25 MHz  
***APLL\_INPUT\_FREQ\_8KHz*** 8 KHz  
***APLL\_INPUT\_FREQ\_MAX***

Definition at line 56 of file ApIIFreqProfile.h.

#### 4.8.4.8 enum T\_idtApIIMasterResetSync

Enumerated type listing Device master reset sync status.

Enumerator

***APLL\_OUT\_MR\_SYNC*** Device out of master reset sync (normal operation)  
***APLL\_IN\_MR\_SYNC*** Device in master reset sync  
***APLL\_MR\_SYNC\_MAX***

Definition at line 194 of file ApIITypeDef.h.

#### 4.8.4.9 enum T\_idtApIIOutput

Enumerated type listing output ports.

Enumerator

***APLL\_OUTPUT\_QA*** Output port A  
***APLL\_OUTPUT\_QB*** Output port B  
***APLL\_OUTPUT\_MAX***

Definition at line 94 of file ApIITypeDef.h.

#### 4.8.4.10 enum T\_idtApIIOutputEn

Enumerated type listing output port status (enable/disable)

Enumerator

***APLL\_OUTPUT\_DISABLE*** Disable output port  
***APLL\_OUTPUT\_ENABLE*** Enable output port  
***APLL\_OUTPUT\_EN\_MAX***

Definition at line 104 of file ApIITypeDef.h.

#### 4.8.4.11 enum T\_idtApIIOutputFreqType

Enumerated type listing supported output frequencies.

Enumerator

***APLL\_OUTPUT\_FREQ\_24883\_DOT\_2KHz*** 24.8832 MHz  
***APLL\_OUTPUT\_FREQ\_25000KHz*** 25.0 MHz  
***APLL\_OUTPUT\_FREQ\_25781\_DOT\_25KHz*** 25.78125 MHz  
***APLL\_OUTPUT\_FREQ\_77760KHz*** 77.76 MHz  
***APLL\_OUTPUT\_FREQ\_78125KHz*** 78.125 MHz  
***APLL\_OUTPUT\_FREQ\_80566\_DOT\_40625KHz*** 80.56640625 MHz  
***APLL\_OUTPUT\_FREQ\_124416KHz*** 124.416 MHz  
***APLL\_OUTPUT\_FREQ\_125000KHz*** 125.0 MHz  
***APLL\_OUTPUT\_FREQ\_128906\_DOT\_25KHz*** 128.90625 MHz  
***APLL\_OUTPUT\_FREQ\_155520KHz*** 155.52 MHz

***APLL\_OUTPUT\_FREQ\_156250KHz*** 156.250 MHz  
***APLL\_OUTPUT\_FREQ\_161132\_DOT\_8125KHz*** 161.1328125 MHz  
***APLL\_OUTPUT\_FREQ\_311040KHz*** 311.040 MHz  
***APLL\_OUTPUT\_FREQ\_312500KHz*** 312.50 MHz  
***APLL\_OUTPUT\_FREQ\_322265\_DOT\_625KHz*** 322.265625 MHz  
***APLL\_OUTPUT\_FREQ\_622080KHz*** 622.080 MHz  
***APLL\_OUTPUT\_FREQ\_625000KHz*** 625.0 MHz  
***APLL\_OUTPUT\_FREQ\_644531\_DOT\_25KHz*** 644.53125 MHz  
***APLL\_OUTPUT\_FREQ\_MAX***

Definition at line 74 of file ApIIFreqProfile.h.

#### 4.8.4.12 enum T\_idtApIIOutputSigType

Enumerated type listing output signal type.

Enumerator

***APLL\_OUT\_SIG\_LVPECL*** LVPECL output signal type  
***APLL\_OUT\_SIG\_LVDS*** LVDS output signal type  
***APLL\_OUT\_SIG\_MAX***

Definition at line 114 of file ApIITypeDef.h.

#### 4.8.4.13 enum T\_idtApIIOutputSupportedType

Enumerated type listing frequency supported by the output.

Enumerator

***APLL\_OUT\_QA\_ONLY*** Frequency only supported by output port A  
***APLL\_OUT\_QB\_ONLY*** Frequency only supported by output port B  
***APLL\_OUT\_QA\_OR\_QB*** Frequency supported by both output port A and B  
***APLL\_OUT\_MAX***

Definition at line 124 of file ApIITypeDef.h.

#### 4.8.4.14 enum T\_idtApIIQaDiv

Enumerated type listing output port A divider values.

Enumerator

***APLL\_QA\_ODSEL\_DIV\_25*** Output A divider 25  
***APLL\_QA\_ODSEL\_DIV\_5*** Output A divider 5  
***APLL\_QA\_ODSEL\_DIV\_4*** Output A divider 4  
***APLL\_QA\_ODSEL\_DIV\_2*** Output A divider 2  
***APLL\_QA\_ODSEL\_DIV\_1*** Output A divider 1  
***APLL\_QA\_ODSEL\_DIV\_MAX***

Definition at line 135 of file ApIITypeDef.h.

## 4.8.4.15 enum T\_idtApIIQbDiv

Enumerated type listing output port B divider values.

Enumerator

***APLL\_QB\_ODSEL\_DIV\_8*** Output B divider 8  
***APLL\_QB\_ODSEL\_DIV\_5*** Output B divider 5  
***APLL\_QB\_ODSEL\_DIV\_4*** Output B divider 4  
***APLL\_QB\_ODSEL\_DIV\_2*** Output B divider 2  
***APLL\_QB\_ODSEL\_DIV\_1*** Output B divider 1  
***APLL\_QB\_ODSEL\_DIV\_MAX***

Definition at line 148 of file ApIITypeDef.h.

## 4.8.4.16 enum T\_idtApIIShutoff

Enumerated type listing ApII shutoff status/setting.

Enumerator

***APLL\_SHUTOFF\_NORMAL\_OP*** ApII in normal operation  
***APLL\_SHUTOFF\_SHUTOFF*** VCXO current is shut off and Outputs are squelched  
***APLL\_SHUTOFF\_MAX***

Definition at line 214 of file ApIITypeDef.h.

## 4.8.4.17 enum T\_idtApIIXtalId

Enumerated type listing crystals.

Enumerator

***APLL\_XTAL\_1\_APPL1*** XTAL1 connects to APPL1  
***APLL\_XTAL\_2\_APPL2*** XTAL2 connects to APPL2  
***APLL\_XTAL\_3\_APPL1*** XTAL3 connects to APPL1  
***APLL\_XTAL\_4\_APPL2*** XTAL4 connects to APPL2  
***APLL\_XTAL\_MAX***

Definition at line 171 of file ApIITypeDef.h.

## 4.8.4.18 enum T\_idtApIIXtalType

Enumerated type listing support crystal frequencies.

Enumerator

***APLL\_XTAL\_FREQ\_25000KHz*** Ethernet (25 MHz)  
***APLL\_XTAL\_FREQ\_25781\_DOT\_25KHz*** Ethernet FEC (25.78125 MHz)  
***APLL\_XTAL\_FREQ\_24883\_DOT\_2KHz*** SONET/SDH (24.8832 MHz)  
***APLL\_XTAL\_FREQ\_MAX***

Definition at line 183 of file ApIITypeDef.h.

### 4.8.5 Function Documentation

#### 4.8.5.1 `outputFrequency_t` IDTAplI\_AplIInput2DplIOutput ( `T_idtAplIInputFreqType` *apIIInputFreq* )

Translate APLL input frequency to DPLL output frequency.

##### Parameters

|                      |                      |
|----------------------|----------------------|
| <i>apIIInputFreq</i> | APLL input frequency |
|----------------------|----------------------|

##### Returns

DPLL output frequency

Definition at line 474 of file ApII.c.

#### 4.8.5.2 `T_idtAplIOutputFreqType` IDTAplI\_DplIOutput2AplIOutput ( `outputFrequency_t` *dplIOutput* )

Translate DPLL output frequency to APLL output frequency.

##### Parameters

|                   |                       |
|-------------------|-----------------------|
| <i>dplIOutput</i> | DPLL output frequency |
|-------------------|-----------------------|

##### Returns

APLL output frequency

Definition at line 533 of file ApII.c.

#### 4.8.5.3 `void` IDTAplI\_GetAplIConfig ( `T_idtDrvHdlr` *hdlr*, `T_idtAplIIId` *apIIInstId*, `T_idtAplIConfig *` *apIIConfig* )

Get APLL full configuration.

##### Parameters

|                   |                       |
|-------------------|-----------------------|
| <i>hdlr</i>       | Device driver handler |
| <i>apIIInstId</i> | APLL instance         |
| <i>apIIConfig</i> | APLL configuration    |

Definition at line 416 of file ApII.c.

#### 4.8.5.4 `void` IDTAplI\_GetAplIConfigPerOutput ( `T_idtDrvHdlr` *hdlr*, `T_idtAplIIId` *apIIInstId*, `T_idtAplIOutput` *output*, `T_idtAplIConfigPerOutput *` *apIIConfigPerOutput* )

Get APLL per output configuration.

##### Parameters

|                             |                             |
|-----------------------------|-----------------------------|
| <i>hdlr</i>                 | Device driver handler       |
| <i>apIIInstId</i>           | APLL instance               |
| <i>output</i>               | Output port                 |
| <i>apIIConfigPer-Output</i> | APLL per port configuration |

Definition at line 318 of file ApII.c.

**4.8.5.5** `const T_idtApplFreqMapping* IDTApl_GetApplFreqTableEntry ( T_idtDrvHdlr hdlr, T_idtOutputApplConfig outputAppl, outputFrequency_t nOutFreq )`

Get APLL frequency configuration table entry based on the output frequency.

#### Parameters

|                   |                           |
|-------------------|---------------------------|
| <i>hdlr</i>       | Device driver handler     |
| <i>outputAppl</i> | Output APLL configuration |
| <i>nOutFreq</i>   | Output frequency          |

#### Returns

APLL frequency configuration

Definition at line 627 of file Appl.c.

**4.8.5.6** `T_idtApplBypass IDTApl_GetBypass ( T_idtDrvHdlr hdlr, T_idtApplId applInstId )`

Function to retrieve bypass status of device.

#### Parameters

|                   |                                                                                                                          |
|-------------------|--------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                    |
| <i>applInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul> |

#### Returns

Bypass status

- APLL\_BYPASS\_NORMAL\_OP - Normal operation
- APLL\_BYPASS\_BYPASS - Bypass VCXO, connect input to output

Definition at line 1359 of file ApplApi.c.

**4.8.5.7** `const T_idtApplFreqMapping* IDTApl_GetConfigListByFreq ( T_idtApplOutputFreqType applOutputFreq )`

Function to retrieve output configuration given an output frequency.

#### Parameters

|                       |                  |
|-----------------------|------------------|
| <i>applOutputFreq</i> | Output frequency |
|-----------------------|------------------|

#### Returns

Output configuration

Definition at line 545 of file ApplFreqProfile.c.

**4.8.5.8** `const T_idtApplFreqMapping* IDTApl_GetConfigListByXtalType ( T_idtApplXtalType applVcxoFreq )`

Function to retrieve output configuration given the oscillator type (frequency)

## Parameters

|                     |                             |
|---------------------|-----------------------------|
| <i>apllVcxoFreq</i> | Oscillator type (frequency) |
|---------------------|-----------------------------|

## Returns

Output configuration

Definition at line 564 of file ApIIFreqProfile.c.

#### 4.8.5.9 T\_idtApIIConfigSel IDTAplI\_GetConfigSel ( T\_idtDrvHdlr *hdlr*, T\_idtApIIId *apllInstId* )

Function to retrieve APLL configuration.

## Parameters

|                   |                                                                                                                          |
|-------------------|--------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                    |
| <i>apllInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul> |

## Returns

APLL configuration

- APLL\_CONFIG0 - configuration 0
- APLL\_CONFIG1 - configuration 1

Definition at line 219 of file ApIIApi.c.

#### 4.8.5.10 const T\_idtApIIFreqMapping\* IDTAplI\_GetCurrentApIIFreqConfig ( T\_idtApIIId *apllId*, T\_idtApIIOutput *output* )

Function to retrieve the current APLL frequency mapping configuration for the output.

## Parameters

|               |               |
|---------------|---------------|
| <i>apllId</i> | APLL instance |
| <i>output</i> | APLL output   |

## Returns

APLL frequency mapping configuration

Definition at line 604 of file ApIIFreqProfile.c.

#### 4.8.5.11 T\_idtApIIDividerValue IDTAplI\_GetFeedbackDiv ( T\_idtDrvHdlr *hdlr*, T\_idtApIIId *apllInstId*, T\_idtApIIConfigSel *apllConfig* )

Function to retrieve feedback (oscillator) divider value.

## Parameters

|                   |                                                                                                                                                 |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                           |
| <i>apllInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                        |
| <i>apllConfig</i> | APLL configuration <ul style="list-style-type: none"> <li>• APLL_CONFIG0 - configuration 0</li> <li>• APLL_CONFIG1 - configuration 1</li> </ul> |

## Returns

Divisor value

Definition at line 525 of file ApIApi.c.

**4.8.5.12** void IDTApll.GetFeedbackDivRange ( T\_idtDrvHdlr *hdlr*, T\_idtApllDividerValue \* *minDiv*, T\_idtApllDividerValue \* *maxDiv* )

Function to retrieve feedback (oscillator) divisor range.

## Parameters

|               |                       |
|---------------|-----------------------|
| <i>hdlr</i>   | Device driver handler |
| <i>minDiv</i> | minimum divisor       |
| <i>maxDiv</i> | maximum divisor       |

Definition at line 502 of file ApIApi.c.

**4.8.5.13** T\_idtApllFreqDoublerSel IDTApll.GetFreqDoublerSel ( T\_idtDrvHdlr *hdlr*, T\_idtApllId *apllInstId* )

Function to retrieve whether feedback frequency doubling is enabled.

## Parameters

|                   |                                                                                                                          |
|-------------------|--------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                    |
| <i>apllInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul> |

## Returns

Feedback frequency doubling value

- APLL\_FREQ\_SEL\_SINGLE - Feedback frequency not doubled
- APLL\_FREQ\_SEL\_DOUBLE - Feedback frequency doubled

Definition at line 890 of file ApIApi.c.

**4.8.5.14** T\_idtApllInputClkSrc IDTApll.GetInputClkSrc ( T\_idtDrvHdlr *hdlr*, T\_idtApllId *apllInstId* )

Function to retrieve selected APLL input frequency source (external/internal)

## Parameters

|                   |                                                                                                                          |
|-------------------|--------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                    |
| <i>apIIInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul> |

## Returns

Input clock source for APLL

- APLL\_CLK\_SEL\_EXTERNAL - external clock source
- APLL\_CLK\_SEL\_INTERNAL - internal clock source

Definition at line 287 of file ApIIApi.c.

#### 4.8.5.15 T\_idtApII MasterResetSync IDTAplI\_GetMasterResetSync ( T\_idtDrvHdlr *hdlr*, T\_idtApIIId *apIIInstId* )

Function to retrieve reset status of device.

## Parameters

|                   |                                                                                                                          |
|-------------------|--------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                    |
| <i>apIIInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul> |

## Returns

Master reset sync

- APLL\_OUT\_MR\_SYNC - Normal operation
- APLL\_IN\_MR\_SYNC - In master sync reset

Definition at line 1291 of file ApIIApi.c.

#### 4.8.5.16 T\_idtApII ConfigSel IDTAplI\_GetNonActiveApIIConfig ( T\_idtDrvHdlr *hdlr*, T\_idtApIIId *apIIInst* )

Get non-active APLL configuration.

## Parameters

|                 |                       |
|-----------------|-----------------------|
| <i>hdlr</i>     | Device driver handler |
| <i>apIIInst</i> | APLL instance         |

**Returns**

Non-active APLL configuration

Definition at line 706 of file ApI.c.

#### 4.8.5.17 **T\_idtApIOutputEn** IDTAplI.GetOutputEnState ( *T\_idtDrvHdlr hdlr*, *T\_idtApIIId apIInstId*, *T\_idtApIOutput output* )

Function to retrieve output port status (enable/disable)

**Parameters**

|                  |                                                                                                                                          |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>      | Device driver handler                                                                                                                    |
| <i>apIInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                 |
| <i>output</i>    | Output port <ul style="list-style-type: none"> <li>• APLL_OUTPUT_QA - Output port A</li> <li>• APLL_OUTPUT_QB - Output port B</li> </ul> |

**Returns**

Enable /Disable output port

- APLL\_OUTPUT\_DISABLE - Disable output
- APLL\_OUTPUT\_ENABLE - Enable output

Definition at line 1146 of file ApIApi.c.

#### 4.8.5.18 **T\_idtApIOutputSigType** IDTAplI.GetOutputSigType ( *T\_idtDrvHdlr hdlr*, *T\_idtApIIId apIInstId*, *T\_idtApIOutput output* )

Function to retrieve output port signal type.

**Parameters**

|                  |                                                                                                                                          |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>      | Device driver handler                                                                                                                    |
| <i>apIInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                 |
| <i>output</i>    | Output port <ul style="list-style-type: none"> <li>• APLL_OUTPUT_QA - Output port A</li> <li>• APLL_OUTPUT_QB - Output port B</li> </ul> |

**Returns**

output signal type

- APLL\_OUT\_SIG\_LVPECL - Output LVPECL signal
- APLL\_OUT\_SIG\_LVDS - Output LVDS signal

Definition at line 1035 of file ApIIApi.c.

#### 4.8.5.19 T\_idtApIIDividerValue IDTAplI\_GetPreDiv ( T\_idtDrvHdlr *hdlr*, T\_idtApIIId *apIIInstId*, T\_idtApIIConfigSel *apIIConfig* )

Function to retrieve pre- (input) divider value.

##### Parameters

|                   |                                                                                                                                                 |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                           |
| <i>apIIInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                        |
| <i>apIIConfig</i> | APLL configuration <ul style="list-style-type: none"> <li>• APLL_CONFIG0 - configuration 0</li> <li>• APLL_CONFIG1 - configuration 1</li> </ul> |

##### Returns

Divisor value

Definition at line 356 of file ApIIApi.c.

#### 4.8.5.20 void IDTAplI\_GetPreDivRange ( T\_idtDrvHdlr *hdlr*, T\_idtApIIDividerValue \* *minDiv*, T\_idtApIIDividerValue \* *maxDiv* )

Function to retrieve pre- (input) divisor range.

##### Parameters

|               |                       |
|---------------|-----------------------|
| <i>hdlr</i>   | Device driver handler |
| <i>minDiv</i> | minimum divisor       |
| <i>maxDiv</i> | maximum divisor       |

##### Note

Divisor "1" is also a valid divisor

Definition at line 415 of file ApIIApi.c.

#### 4.8.5.21 T\_idtApIIQaDiv IDTAplI\_GetQaDiv ( T\_idtDrvHdlr *hdlr*, T\_idtApIIId *apIIInstId*, T\_idtApIIConfigSel *config* )

Function to retrieve output port A divisor.

##### Parameters

|                   |                                                                                                                          |
|-------------------|--------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                    |
| <i>apIIInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul> |

|               |                                                                                                                                                 |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>config</i> | APLL configuration <ul style="list-style-type: none"> <li>• APLL_CONFIG0 - configuration 0</li> <li>• APLL_CONFIG1 - configuration 1</li> </ul> |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------|

#### Returns

Output divider assigned to output port A

- APLL\_QA\_ODSEL\_DIV\_25 - divide by 25
- APLL\_QA\_ODSEL\_DIV\_5 - divide by 5
- APLL\_QA\_ODSEL\_DIV\_4 - divide by 4
- APLL\_QA\_ODSEL\_DIV\_2 - divide by 2
- APLL\_QA\_ODSEL\_DIV\_1 - divide by 1

Definition at line 662 of file AplApi.c.

#### 4.8.5.22 T\_idtAplQaDiv IDTApl\_GetQbDiv ( T\_idtDrvHdlr *hdlr*, T\_idtAplId *apllInstId*, T\_idtAplConfigSel *config* )

Function to retrieve output port B divisor.

#### Parameters

|                   |                                                                                                                                                 |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                           |
| <i>apllInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                        |
| <i>config</i>     | APLL configuration <ul style="list-style-type: none"> <li>• APLL_CONFIG0 - configuration 0</li> <li>• APLL_CONFIG1 - configuration 1</li> </ul> |

#### Returns

Output divider assigned to output port B

- APLL\_QB\_ODSEL\_DIV\_8 - divide by 8
- APLL\_QB\_ODSEL\_DIV\_5 - divide by 5
- APLL\_QB\_ODSEL\_DIV\_4 - divide by 4
- APLL\_QB\_ODSEL\_DIV\_2 - divide by 2
- APLL\_QB\_ODSEL\_DIV\_1 - divide by 1

Definition at line 779 of file AplApi.c.

#### 4.8.5.23 T\_idtAplShutoff IDTApl\_GetShutoff ( T\_idtDrvHdlr *hdlr*, T\_idtAplId *apllInstId* )

Function to retrieve shutoff status of device.

## Parameters

|                   |                                                                                                                          |
|-------------------|--------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                    |
| <i>apllInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul> |

## Returns

Shutoff status

- APLL\_SHUTOFF\_NORMAL\_OP - Normal operation
- APLL\_SHUTOFF\_SHUTOFF - VCXO shutoff

Definition at line 1427 of file ApIIApi.c.

#### 4.8.5.24 T\_idtApIIxTxl\* IDTApll\_GetSupportedXtal ( const T\_idtDrvDeviceConfig \* deviceConfig, T\_osUInt8 \* numberOfXtal )

Return supported xtal configuration by the device.

## Parameters

|                     |                                                   |
|---------------------|---------------------------------------------------|
| <i>deviceConfig</i> | APLL device configuration                         |
| <i>numberOfXtal</i> | Number of xtal configuration in the returned list |

## Returns

The list of supported XTAL configuration

Caller needs to free the list after finishing using it.

Definition at line 899 of file ApII.c.

#### 4.8.5.25 T\_idtApIIxTxlId IDTApll\_GetXtalId ( T\_idtDrvHdlr hdlr, T\_idtApIIId apIIInstId )

Function to retrieve oscillator currently used.

## Parameters

|                   |                                                                                                                          |
|-------------------|--------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                    |
| <i>apIIInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul> |

## Returns

current oscillator used

- APLL\_XTAL\_1\_APLL1 - XTAL1 for APLL1
- APLL\_XTAL\_2\_APLL2 - XTAL2 for APLL2
- APLL\_XTAL\_3\_APLL1 - XTAL3 for APLL1
- APLL\_XTAL\_4\_APLL2 - XTAL4 for APLL2

Definition at line 960 of file ApIIApi.c.

#### 4.8.5.26 **T\_idtApplXtalId** IDTApl\_GetXtalIdFromXtalFreq ( T\_idtDrvHdlr *hdlr*, T\_idtApplId *apllInst*, T\_idtApplXtalType *apllVcxoFreq* )

Get APLL configured crystal ID from crystal frequency.

##### Parameters

|                     |                       |
|---------------------|-----------------------|
| <i>hdlr</i>         | Device driver handler |
| <i>apllInst</i>     | APLL instance         |
| <i>apllVcxoFreq</i> | Crytal frequency      |

##### Returns

Crytal ID

Definition at line 753 of file Appl.c.

#### 4.8.5.27 **T\_osBool** IDTApl\_NullFreqTableEntry ( const T\_idtApplFreqMapping \* *apllFreqConfig* )

Function to check whether the input APLL frequency mapping is NULL (not configured)

##### Parameters

|                       |                                      |
|-----------------------|--------------------------------------|
| <i>apllFreqConfig</i> | APLL frequency mapping configuration |
|-----------------------|--------------------------------------|

##### Returns

E\_osTrue - not configured, E\_osFalse - configured

Definition at line 583 of file ApplFreqProfile.c.

#### 4.8.5.28 **void** IDTApl\_SetApplConfig ( T\_idtDrvHdlr *hdlr*, T\_idtApplId *apllInstId*, T\_idtApplConfig \* *apllConfig* )

Set APLL full configuration.

##### Parameters

|                   |                       |
|-------------------|-----------------------|
| <i>hdlr</i>       | Device driver handler |
| <i>apllInstId</i> | APLL instance         |
| <i>apllConfig</i> | APLL configuration    |

Definition at line 357 of file Appl.c.

#### 4.8.5.29 **void** IDTApl\_SetApplConfigPerOutput ( T\_idtDrvHdlr *hdlr*, T\_idtApplId *apllInstId*, T\_idtApplOutput *output*, T\_idtApplConfigPerOutput \* *apllConfigPerOutput* )

Set APLL per output configuration.

##### Parameters

|                             |                             |
|-----------------------------|-----------------------------|
| <i>hdlr</i>                 | Device driver handler       |
| <i>apllInstId</i>           | APLL instance               |
| <i>output</i>               | Output port                 |
| <i>apllConfigPer-Output</i> | APLL per port configuration |

Definition at line 272 of file ApII.c.

4.8.5.30 void IDTAplI.SetBypass ( T\_idtDrvHdlr *hdlr*, T\_idtApIIId *apIIInstId*, T\_idtApIIBypass *bypass* )

Function to set APLL bypass setting.

#### Parameters

|                   |                                                                                                                                                                                       |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                                                                 |
| <i>apIIInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                                                              |
| <i>bypass</i>     | Value to set bypass <ul style="list-style-type: none"> <li>• APLL_BYPASS_NORMAL_OP - Normal operation</li> <li>• APLL_BYPASS_BYPASS - Bypass VCXO, connect input to output</li> </ul> |

Definition at line 1322 of file ApIIApi.c.

4.8.5.31 void IDTAplI.SetConfigSel ( T\_idtDrvHdlr *hdlr*, T\_idtApIIId *apIIInstId*, T\_idtApIIConfigSel *apIIConfig* )

Function to provision APLL based on input configuration.

#### Parameters

|                   |                                                                                                                                                 |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                           |
| <i>apIIInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                        |
| <i>apIIConfig</i> | APLL configuration <ul style="list-style-type: none"> <li>• APLL_CONFIG0 - configuration 0</li> <li>• APLL_CONFIG1 - configuration 1</li> </ul> |

Definition at line 250 of file ApIIApi.c.

4.8.5.32 void IDTAplI.SetFeedbackDiv ( T\_idtDrvHdlr *hdlr*, T\_idtApIIId *apIIInstId*, T\_idtApIIConfigSel *apIIConfig*, T\_idtApIIDividerValue *div* )

Function to provision feedback (oscillator) divider value.

## Parameters

|                   |                                                                                                                                                 |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                           |
| <i>apllInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                        |
| <i>apllConfig</i> | APLL configuration <ul style="list-style-type: none"> <li>• APLL_CONFIG0 - configuration 0</li> <li>• APLL_CONFIG1 - configuration 1</li> </ul> |
| <i>div</i>        | Divisor value                                                                                                                                   |

Definition at line 588 of file ApIApi.c.

4.8.5.33 void IDTAplI\_SetFreqDoublerSel ( T\_idtDrvHdlr *hdlr*, T\_idtAplIId *apllInstId*, T\_idtAplIFreqDoublerSel *doubler* )

Function to enable/disable feedback frequency doubling.

## Parameters

|                   |                                                                                                                                                                                                          |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                                                                                    |
| <i>apllInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                                                                                 |
| <i>doubler</i>    | Feedback frequency doubling value <ul style="list-style-type: none"> <li>• APLL_FREQ_SEL_SINGLE - Feedback frequency not doubled</li> <li>• APLL_FREQ_SEL_DOUBLE - Feedback frequency doubled</li> </ul> |

Definition at line 921 of file ApIApi.c.

4.8.5.34 void IDTAplI\_SetInputClkSrc ( T\_idtDrvHdlr *hdlr*, T\_idtAplIId *apllInstId*, T\_idtAplIInputClkSrc *inputClk* )

Function to provision APLL input frequency source (external/internal)

## Parameters

|                   |                                                                                                                                                                                        |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                                                                  |
| <i>apllInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                                                               |
| <i>inputClk</i>   | Input clock source for APLL <ul style="list-style-type: none"> <li>• APLL_CLK_SEL_EXTERNAL - external clock source</li> <li>• APLL_CLK_SEL_INTERNAL - internal clock source</li> </ul> |

Definition at line 318 of file ApIApi.c.

4.8.5.35 void IDTAplI\_SetMasterResetSync ( T\_idtDrvHdlr *hdlr*, T\_idtAplIId *apIiInstId*, T\_idtAplIMasterResetSync *reset* )

Function to reset APLL.

#### Parameters

|                   |                                                                                                                                                                          |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                                                    |
| <i>apIiInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                                                 |
| <i>reset</i>      | Value to set master reset sync <ul style="list-style-type: none"> <li>• APLL_OUT_MR_SYNC - Normal operation</li> <li>• APLL_IN_MR_SYNC - In master sync reset</li> </ul> |

Definition at line 1254 of file AplIapi.c.

4.8.5.36 void IDTAplI\_SetOutputEnState ( T\_idtDrvHdlr *hdlr*, T\_idtAplIId *apIiInstId*, T\_idtAplIOutput *output*, T\_idtAplIOutputEn *enable* )

Function to control (enable/disable) output port.

#### Parameters

|                   |                                                                                                                                                               |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                                         |
| <i>apIiInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                                      |
| <i>output</i>     | Output port <ul style="list-style-type: none"> <li>• APLL_OUTPUT_QA - Output port A</li> <li>• APLL_OUTPUT_QB - Output port B</li> </ul>                      |
| <i>enable</i>     | Enable /Disable output <ul style="list-style-type: none"> <li>• APLL_OUTPUT_DISABLE - Disable output</li> <li>• APLL_OUTPUT_ENABLE - Enable output</li> </ul> |

Definition at line 1199 of file AplIapi.c.

4.8.5.37 void IDTAplI\_SetOutputSigType ( T\_idtDrvHdlr *hdlr*, T\_idtAplIId *apIiInstId*, T\_idtAplIOutput *output*, T\_idtAplIOutputSigType *outputSigType* )

Function to provision output port signal type.

#### Parameters

|                   |                                                                                                                          |
|-------------------|--------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                    |
| <i>apIiInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul> |

|                      |                                                                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>output</i>        | Output port <ul style="list-style-type: none"> <li>• APLL_OUTPUT_QA - Output port A</li> <li>• APLL_OUTPUT_QB - Output port B</li> </ul>                            |
| <i>outputSigType</i> | output signal type <ul style="list-style-type: none"> <li>• APLL_OUT_SIG_LVPECL - Output LVPECL signal</li> <li>• APLL_OUT_SIG_LVDS - Output LVDS signal</li> </ul> |

Definition at line 1088 of file AplApi.c.

**4.8.5.38** void IDTApll.SetPreDiv ( T\_idtDrvHdlr *hdlr*, T\_idtApllId *apllInstId*, T\_idtApllConfigSel *apllConfig*, T\_idtApllDividerValue *div* )

Function to provision pre- (input) divider value.

#### Parameters

|                   |                                                                                                                                                 |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                           |
| <i>apllInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                        |
| <i>apllConfig</i> | APLL configuration <ul style="list-style-type: none"> <li>• APLL_CONFIG0 - configuration 0</li> <li>• APLL_CONFIG1 - configuration 1</li> </ul> |
| <i>div</i>        | Divisor value                                                                                                                                   |

Definition at line 438 of file AplApi.c.

**4.8.5.39** void IDTApll.SetQaDiv ( T\_idtDrvHdlr *hdlr*, T\_idtApllId *apllInstId*, T\_idtApllConfigSel *config*, T\_idtApllQaDiv *div* )

Function to provision output port A divisor.

## Parameters

|                   |                                                                                                                                                                                                                                                                                           |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                                                                                                                                                                     |
| <i>apIiInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                                                                                                                                                                  |
| <i>config</i>     | APLL configuration <ul style="list-style-type: none"> <li>• APLL_CONFIG0 - configuration 0</li> <li>• APLL_CONFIG1 - configuration 1</li> </ul>                                                                                                                                           |
| <i>div</i>        | Divisor value <ul style="list-style-type: none"> <li>• APLL_QA_ODSEL_DIV_25 - divide by 25</li> <li>• APLL_QA_ODSEL_DIV_5 - divide by 5</li> <li>• APLL_QA_ODSEL_DIV_4 - divide by 4</li> <li>• APLL_QA_ODSEL_DIV_2 - divide by 2</li> <li>• APLL_QA_ODSEL_DIV_1 - divide by 1</li> </ul> |

Definition at line 718 of file ApIiApi.c.

4.8.5.40 void IDTAplI\_SetQbDiv ( T\_idtDrvHdlr *hdlr*, T\_idtApIId *apIiInstId*, T\_idtApIConfigSel *config*, T\_idtApIqBDiv *div* )

Function to provision output port B divisor.

## Parameters

|                   |                                                                                                                                                                                                                                                                                         |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdlr</i>       | Device driver handler                                                                                                                                                                                                                                                                   |
| <i>apIiInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                                                                                                                                                                |
| <i>config</i>     | APLL configuration <ul style="list-style-type: none"> <li>• APLL_CONFIG0 - configuration 0</li> <li>• APLL_CONFIG1 - configuration 1</li> </ul>                                                                                                                                         |
| <i>div</i>        | Divisor value <ul style="list-style-type: none"> <li>• APLL_QB_ODSEL_DIV_8 - divide by 8</li> <li>• APLL_QB_ODSEL_DIV_5 - divide by 5</li> <li>• APLL_QB_ODSEL_DIV_4 - divide by 4</li> <li>• APLL_QB_ODSEL_DIV_2 - divide by 2</li> <li>• APLL_QB_ODSEL_DIV_1 - divide by 1</li> </ul> |

Definition at line 835 of file ApIiApi.c.

4.8.5.41 void IDTAplI\_SetShutoff ( T\_idtDrvHdlr *hdlr*, T\_idtApIId *apIiInstId*, T\_idtApIShutoff *shutoff* )

Function to set APLL shutoff setting.

## Parameters

|                   |                                                                                                                                                                   |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdr</i>        | Device driver handler                                                                                                                                             |
| <i>apllInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                                          |
| <i>shutoff</i>    | Value to set shutoff <ul style="list-style-type: none"> <li>• APLL_SHUTOFF_NORMAL_OP - Normal operation</li> <li>• APLL_SHUTOFF_SHUTOFF - VCXO shutoff</li> </ul> |

Definition at line 1390 of file AplApi.c.

4.8.5.42 void IDTApl\_SetXtalId ( T\_idtDrvHdr *hdr*, T\_idtAplId *apllInstId*, T\_idtAplXtalId *xtal* )

Function to select oscillator.

## Parameters

|                   |                                                                                                                                                                                                                                                           |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hdr</i>        | Device driver handler                                                                                                                                                                                                                                     |
| <i>apllInstId</i> | APLL instance <ul style="list-style-type: none"> <li>• APLL_INST_1 - APLL #1</li> <li>• APLL_INST_2 - APLL #2</li> </ul>                                                                                                                                  |
| <i>xtal</i>       | Oscillator to select <ul style="list-style-type: none"> <li>• APLL_XTAL_1_APLL1 - XTAL1 for APLL1</li> <li>• APLL_XTAL_2_APLL2 - XTAL2 for APLL2</li> <li>• APLL_XTAL_3_APLL1 - XTAL3 for APLL1</li> <li>• APLL_XTAL_4_APLL2 - XTAL4 for APLL2</li> </ul> |

Definition at line 991 of file AplApi.c.

4.8.5.43 void IDTApl\_Switch2NonActiveApllConfig ( T\_idtDrvHdr *hdr*, T\_idtAplId *apllId* )

Switch to non-active APLL configuration.

## Parameters

|               |                       |
|---------------|-----------------------|
| <i>hdr</i>    | Device driver handler |
| <i>apllId</i> | APLL instance         |

Definition at line 733 of file Apl.c.

4.8.5.44 T\_osBool IDTApl\_ValidXtalInput ( const T\_idtDrvDeviceConfig \* *deviceConfig*, T\_idtAplXtal \* *xtalList*, T\_osUInt8 *numberOfXtal* )

Checks if there is input XTAL information is valid for the device.

## Parameters

|                     |                                         |
|---------------------|-----------------------------------------|
| <i>deviceConfig</i> | APLL device configuration               |
| <i>xtalList</i>     | List of XTAL information for the device |
| <i>numberOfXtal</i> | Number of XTAL in the xtalList          |

**Returns**

TRUE, if input XTAL information is valid; FALSE, otherwise

Definition at line 834 of file ApII.c.

**4.8.5.45 T\_osBool IDTApII\_XtalConfigured ( T\_idtDrvHdlr *hdlr*, T\_idtApIIId *apIIId* )**

Checks if there is any XTAL configured for the APLL.

**Parameters**

|               |                       |
|---------------|-----------------------|
| <i>hdlr</i>   | Device driver handler |
| <i>apIIId</i> | APLL instance         |

**Returns**

TRUE, if there is at least one XTAL configured; FALSE, otherwise

Definition at line 790 of file ApII.c.



## Chapter 5

# Data Structure Documentation

### 5.1 bucketReg\_t Struct Reference

Structure used to store leaky bucket register offsets.

#### Data Fields

- T\_uint8 [upperThreshReg\\_m](#)
- T\_uint8 [lowerThreshReg\\_m](#)
- T\_uint8 [bucketSizeReg\\_m](#)
- T\_uint8 [decayRateReg\\_m](#)

#### 5.1.1 Detailed Description

Structure used to store leaky bucket register offsets.

Definition at line 47 of file Input.c.

#### 5.1.2 Field Documentation

##### 5.1.2.1 T\_uint8 bucketSizeReg\_m

Bucket size register offset

Definition at line 51 of file Input.c.

##### 5.1.2.2 T\_uint8 decayRateReg\_m

Decay rate register offset

Definition at line 52 of file Input.c.

##### 5.1.2.3 T\_uint8 lowerThreshReg\_m

Lower threshold register offset

Definition at line 50 of file Input.c.

#### 5.1.2.4 T\_osUInt8 upperThreshReg\_m

Upper threshold register offset

Definition at line 49 of file Input.c.

The documentation for this struct was generated from the following file:

- [common/src/Input.c](#)

## 5.2 T\_idtApIIdConfig::fbDiv\_s Struct Reference

```
#include <ApI.h>
```

### Data Fields

- [T\\_idtApIIdDividerValue fbDivConfig0](#)
- [T\\_idtApIIdDividerValue fbDivConfig1](#)

#### 5.2.1 Detailed Description

Definition at line 95 of file ApI.h.

#### 5.2.2 Field Documentation

##### 5.2.2.1 T\_idtApIIdDividerValue fbDivConfig0

APLL feedback divider value for configuration 0

Definition at line 97 of file ApI.h.

##### 5.2.2.2 T\_idtApIIdDividerValue fbDivConfig1

APLL feedback divider value for configuration 1

Definition at line 98 of file ApI.h.

The documentation for this struct was generated from the following file:

- [common/include/ApI.h](#)

## 5.3 globalArgs\_t Struct Reference

Structure to define program command line arguments.

### Data Fields

- [T\\_osUInt32 modes\\_m](#)
- [char filename\\_ma \[SAMPLE\\_MAX\\_FILENAME\]](#)
- [T\\_osUInt32 inputFreq\\_m \[IDT\\_DRV\\_MAX\\_INPUT\]](#)
- [outputFrequency\\_t outFreq\\_m \[IDT\\_DRV\\_MAX\\_OUTPUT\]](#)
- [T\\_idtDpllType dpll\\_m \[IDT\\_DRV\\_MAX\\_OUTPUT\]](#)

- T\_osUInt8 [isSim\\_m](#)
- T\_osUInt8 [dcoMode\\_m](#)
- long [offset\\_m](#)
- T\_osUInt8 [numberOfXtal](#)
- T\_idtApllXtal \* [xtalList](#)
- T\_DpllProfile [dpllProfile\\_ma](#) [DPLL\_INSTANCE\_MAX]
- T\_osUInt8 [sampleConfig\\_m](#)
- T\_osUInt8 [regOffset\\_m](#)
- T\_osUInt8 [regValue\\_m](#)

### 5.3.1 Detailed Description

Structure to define program command line arguments.

Definition at line 127 of file main.c.

### 5.3.2 Field Documentation

#### 5.3.2.1 T\_osUInt8 dcoMode\_m

Enable/Disable

DCO mode

Definition at line 143 of file main.c.

#### 5.3.2.2 T\_idtDpllType dpll\_m[IDT\_DRV\_MAX\_OUTPUT]

Select DPLL T0 / T4

Definition at line 138 of file main.c.

#### 5.3.2.3 T\_DpllProfile dpllProfile\_ma[DPLL\_INSTANCE\_MAX]

Dpll Profile

Definition at line 148 of file main.c.

#### 5.3.2.4 char filename\_ma[SAMPLE\_MAX\_FILENAME]

File name for

load/store

Definition at line 130 of file main.c.

#### 5.3.2.5 T\_osUInt32 inputFreq\_m[IDT\_DRV\_MAX\_INPUT]

Input frequency (kHz)

Definition at line 132 of file main.c.

#### 5.3.2.6 T\_osUInt8 isSim\_m

Switch to

simulated mode

Definition at line 141 of file main.c.

#### 5.3.2.7 T\_osUInt32 modes\_m

Operation modes

for program

Definition at line 128 of file main.c.

#### 5.3.2.8 T\_osUInt8 numberOfXtal

Number APLL external Xtals

Definition at line 146 of file main.c.

#### 5.3.2.9 long offset\_m

DCO Offset

Definition at line 145 of file main.c.

#### 5.3.2.10 outputFrequency\_t outFreq\_m[IDT\_DRV\_MAX\_OUTPUT]

Output frequency (enum)

Definition at line 135 of file main.c.

#### 5.3.2.11 T\_osUInt8 regOffset\_m

register offset

Definition at line 151 of file main.c.

#### 5.3.2.12 T\_osUInt8 regValue\_m

register value

Definition at line 152 of file main.c.

#### 5.3.2.13 T\_osUInt8 sampleConfig\_m

Sample configuration

Definition at line 150 of file main.c.

#### 5.3.2.14 T\_idtApIIXtal\* xtalList

APLL external Xtal list

Definition at line 147 of file main.c.

The documentation for this struct was generated from the following file:

- 82V3910/sample/src/main.c

## 5.4 T\_idtApIConfig::outputConfig\_s Struct Reference

```
#include <ApI.h>
```

### Data Structures

- struct [qaConfig\\_s](#)
- struct [qbConfig\\_s](#)

### Data Fields

- struct  
[T\\_idtApIConfig::outputConfig\\_s::qaConfig\\_s](#) qaConfig
- struct  
[T\\_idtApIConfig::outputConfig\\_s::qbConfig\\_s](#) qbConfig

#### 5.4.1 Detailed Description

Definition at line 118 of file ApI.h.

#### 5.4.2 Field Documentation

5.4.2.1 struct [T\\_idtApIConfig::outputConfig\\_s::qaConfig\\_s](#) qaConfig

5.4.2.2 struct [T\\_idtApIConfig::outputConfig\\_s::qbConfig\\_s](#) qbConfig

The documentation for this struct was generated from the following file:

- common/include/ApI.h

## 5.5 T\_idtApIConfig::outputDiv\_s Struct Reference

```
#include <ApI.h>
```

### Data Structures

- struct [qaDiv\\_s](#)
- struct [qbDiv\\_s](#)

### Data Fields

- struct  
[T\\_idtApIConfig::outputDiv\\_s::qaDiv\\_s](#) qaDiv
- struct  
[T\\_idtApIConfig::outputDiv\\_s::qbDiv\\_s](#) qbDiv

### 5.5.1 Detailed Description

Definition at line 100 of file ApI.h.

### 5.5.2 Field Documentation

5.5.2.1 struct T\_idtApIConfig::outputDiv\_s::qaDiv\_s qaDiv

5.5.2.2 struct T\_idtApIConfig::outputDiv\_s::qbDiv\_s qbDiv

The documentation for this struct was generated from the following file:

- common/include/ApI.h

## 5.6 pathSelectorConfig\_t Struct Reference

Structure containing select configuration. Used to translate between frequency and output path configuration.

```
#include <outputFreq_priv.h>
```

### Data Fields

- [pathSelectors\\_t selector\\_m](#)
- int [value\\_m](#)
- T\_osUInt8 [outputDivider\\_m](#)
- T\_osUInt8 [outputPathSelector\\_m](#)
- T\_osUInt8 [instance\\_m](#)

### 5.6.1 Detailed Description

Structure containing select configuration. Used to translate between frequency and output path configuration.

Definition at line 69 of file outputFreq\_priv.h.

### 5.6.2 Field Documentation

5.6.2.1 T\_osUInt8 instance\_m

instance of this configuration

Definition at line 74 of file outputFreq\_priv.h.

5.6.2.2 T\_osUInt8 outputDivider\_m

Output divider value

Definition at line 72 of file outputFreq\_priv.h.

5.6.2.3 T\_osUInt8 outputPathSelector\_m

Output path selector

Definition at line 73 of file outputFreq\_priv.h.

#### 5.6.2.4 pathSelectors\_t selector\_m

Path selector

Definition at line 70 of file outputFreq\_priv.h.

#### 5.6.2.5 int value\_m

Path selector value

Definition at line 71 of file outputFreq\_priv.h.

The documentation for this struct was generated from the following file:

- [common/hlapi/include/outputFreq\\_priv.h](#)

## 5.7 pathSelectorIndex\_t Struct Reference

Structure to index to path selector configs.

```
#include <outputFreq_priv.h>
```

### Data Fields

- int [start\\_m](#)
- int [size\\_m](#)

### 5.7.1 Detailed Description

Structure to index to path selector configs.

Definition at line 78 of file outputFreq\_priv.h.

### 5.7.2 Field Documentation

#### 5.7.2.1 int size\_m

Number of entries

Definition at line 80 of file outputFreq\_priv.h.

#### 5.7.2.2 int start\_m

Start index

Definition at line 79 of file outputFreq\_priv.h.

The documentation for this struct was generated from the following file:

- [common/hlapi/include/outputFreq\\_priv.h](#)

## 5.8 T\_idtApplConfig::preDiv\_s Struct Reference

```
#include <Appl.h>
```

## Data Fields

- [T\\_idtApIldividerValue preDivConfig0](#)
- [T\\_idtApIldividerValue preDivConfig1](#)

### 5.8.1 Detailed Description

Definition at line 90 of file ApI.h.

### 5.8.2 Field Documentation

#### 5.8.2.1 T\_idtApIldividerValue preDivConfig0

APLL pre-divider value for configuration 0

Definition at line 92 of file ApI.h.

#### 5.8.2.2 T\_idtApIldividerValue preDivConfig1

APLL pre-divider value for configuration 1

Definition at line 93 of file ApI.h.

The documentation for this struct was generated from the following file:

- common/include/[ApI.h](#)

## 5.9 T\_idtApIConfig::outputConfig\_s::qaConfig\_s Struct Reference

```
#include <ApI.h>
```

## Data Fields

- [T\\_idtApIOutputSigType outputSigType](#)
- [T\\_idtApIOutputEn enOutput](#)

### 5.9.1 Detailed Description

Definition at line 120 of file ApI.h.

### 5.9.2 Field Documentation

#### 5.9.2.1 T\_idtApIOutputEn enOutput

APLL QA output enable

Definition at line 123 of file ApI.h.

### 5.9.2.2 T\_idtApIOutputSigType outputSigType

APLL QA output signal type

Definition at line 122 of file ApI.h.

The documentation for this struct was generated from the following file:

- common/include/[ApI.h](#)

## 5.10 T\_idtApIConfig::outputDiv\_s::qaDiv\_s Struct Reference

```
#include <ApI.h>
```

### Data Fields

- [T\\_idtApIDividerValue outputDivConfig0](#)
- [T\\_idtApIDividerValue outputDivConfig1](#)

### 5.10.1 Detailed Description

Definition at line 102 of file ApI.h.

### 5.10.2 Field Documentation

#### 5.10.2.1 T\_idtApIDividerValue outputDivConfig0

APLL QA output divider value for configuration 0

Definition at line 104 of file ApI.h.

#### 5.10.2.2 T\_idtApIDividerValue outputDivConfig1

APLL QB output divider value for configuration 1

Definition at line 105 of file ApI.h.

The documentation for this struct was generated from the following file:

- common/include/[ApI.h](#)

## 5.11 T\_idtApIConfig::outputConfig\_s::qbConfig\_s Struct Reference

```
#include <ApI.h>
```

### Data Fields

- [T\\_idtApIOutputSigType outputSigType](#)
- [T\\_idtApIOutputEn enOutput](#)

### 5.11.1 Detailed Description

Definition at line 125 of file ApI.h.

### 5.11.2 Field Documentation

#### 5.11.2.1 T\_idtApIOutputEn enOutput

APLL QB output enable

Definition at line 128 of file ApI.h.

#### 5.11.2.2 T\_idtApIOutputSigType outputSigType

APLL QB output signal type

Definition at line 127 of file ApI.h.

The documentation for this struct was generated from the following file:

- common/include/[ApI.h](#)

## 5.12 T\_idtApIConfig::outputDiv\_s::qbDiv\_s Struct Reference

```
#include <ApI.h>
```

### Data Fields

- [T\\_idtApIDividerValue outputDivConfig0](#)
- [T\\_idtApIDividerValue outputDivConfig1](#)

### 5.12.1 Detailed Description

Definition at line 107 of file ApI.h.

### 5.12.2 Field Documentation

#### 5.12.2.1 T\_idtApIDividerValue outputDivConfig0

APLL QB output divider value for configuration 0

Definition at line 109 of file ApI.h.

#### 5.12.2.2 T\_idtApIDividerValue outputDivConfig1

APLL QB output divider value for configuration 1

Definition at line 110 of file ApI.h.

The documentation for this struct was generated from the following file:

- common/include/[ApI.h](#)

## 5.13 T\_apIOutputFrequencyEntry Struct Reference

Structure used to translate APLL configuration and output port frequency.

## Data Fields

- [T\\_idtApIDividerValue apIldivVal\\_m](#)
- [outputFrequency\\_t frequencies\\_m \[APLL\\_XTAL\\_FREQ\\_MAX\]](#)

### 5.13.1 Detailed Description

Structure used to translate APLL configuration and output port frequency.

Definition at line 67 of file [outputFreqString.c](#).

### 5.13.2 Field Documentation

#### 5.13.2.1 T\_idtApIDividerValue apIldivVal\_m

Divider value

Definition at line 68 of file [outputFreqString.c](#).

#### 5.13.2.2 outputFrequency\_t frequencies\_m[APLL\_XTAL\_FREQ\_MAX]

Definition at line 71 of file [outputFreqString.c](#).

The documentation for this struct was generated from the following file:

- [common/hlapi/src/outputFreqString.c](#)

## 5.14 T\_idtApIConfig Struct Reference

Structure containing APLL full configuration.

```
#include <ApI1.h>
```

## Data Structures

- struct [fbDiv\\_s](#)
- struct [outputConfig\\_s](#)
- struct [outputDiv\\_s](#)
- struct [preDiv\\_s](#)

## Data Fields

- [T\\_idtApIConfigSel apICfg](#)
- [T\\_idtApIInputClkSrc inputClk](#)
- struct [T\\_idtApIConfig::preDiv\\_s](#) preDiv
- struct [T\\_idtApIConfig::fbDiv\\_s](#) fbDiv
- struct [T\\_idtApIConfig::outputDiv\\_s](#) outputDiv
- [T\\_idtApIFreqDoublerSel dbleSel](#)
- [T\\_idtApIMasterResetSync mrSync](#)
- [T\\_idtApIBypass byPass](#)
- [T\\_idtApIShutoff shutOff](#)
- [T\\_idtApIXtalId xtal](#)
- struct [T\\_idtApIConfig::outputConfig\\_s](#) outputConfig

### 5.14.1 Detailed Description

Structure containing APLL full configuration.

Definition at line 86 of file Apll.h.

### 5.14.2 Field Documentation

#### 5.14.2.1 T\_idtApIIConfigSel apIICfg

APLL stored register configuration selection

Definition at line 88 of file Apll.h.

#### 5.14.2.2 T\_idtApIIBypass byPass

APLL bypass configuration

Definition at line 115 of file Apll.h.

#### 5.14.2.3 T\_idtApIIFreqDoublerSel dbleSel

APLL frequency doubler in feedback path

Definition at line 113 of file Apll.h.

#### 5.14.2.4 struct T\_idtApIIConfig::fbDiv\_s fbDiv

#### 5.14.2.5 T\_idtApIInputClkSrc inputClk

APLL sourced clock selection

Definition at line 89 of file Apll.h.

#### 5.14.2.6 T\_idtApIIMasterResetSync mrSync

APLL Master reset sync

Definition at line 114 of file Apll.h.

#### 5.14.2.7 struct T\_idtApIIConfig::outputConfig\_s outputConfig

#### 5.14.2.8 struct T\_idtApIIConfig::outputDiv\_s outputDiv

#### 5.14.2.9 struct T\_idtApIIConfig::preDiv\_s preDiv

#### 5.14.2.10 T\_idtApIIShutoff shutOff

APLL shut off mode

Definition at line 116 of file Apll.h.

#### 5.14.2.11 T\_idtApIIXtalId xtal

APLL crystal selection

Definition at line 117 of file ApI.h.

The documentation for this struct was generated from the following file:

- common/include/ApI.h

## 5.15 T\_idtApIConfigPerOutput Struct Reference

Structure containing APLL per output configuration.

```
#include <ApI.h>
```

### Data Fields

- T\_idtApIConfigSel apIcFg
- T\_idtApIInputClkSrc inputClk
- T\_idtApIDividerValue preDiv
- T\_idtApIDividerValue fbDiv
- T\_idtApIDividerValue outputDiv
- T\_idtApIFreqDoublerSel dbleSel
- T\_idtApIXtalId xtal
- T\_idtApIOutputSigType sigType
- T\_idtApIOutputEn enOutput

### 5.15.1 Detailed Description

Structure containing APLL per output configuration.

Definition at line 70 of file ApI.h.

### 5.15.2 Field Documentation

#### 5.15.2.1 T\_idtApIConfigSel apIcFg

APLL stored register configuration selection

Definition at line 72 of file ApI.h.

#### 5.15.2.2 T\_idtApIFreqDoublerSel dbleSel

APLL frequency doubler in feedback path

Definition at line 77 of file ApI.h.

#### 5.15.2.3 T\_idtApIOutputEn enOutput

APLL output enable

Definition at line 80 of file ApI.h.

#### 5.15.2.4 T\_idtApIDividerValue fbDiv

APLL feedback divider value

Definition at line 75 of file ApI.h.

#### 5.15.2.5 T\_idtApIInputClkSrc inputClk

APLL sourced clock selection

Definition at line 73 of file ApI.h.

#### 5.15.2.6 T\_idtApIDividerValue outputDiv

APLL output divider value

Definition at line 76 of file ApI.h.

#### 5.15.2.7 T\_idtApIDividerValue preDiv

APLL pre-divider value

Definition at line 74 of file ApI.h.

#### 5.15.2.8 T\_idtApIOutputSigType sigType

APLL output signal configuration

Definition at line 79 of file ApI.h.

#### 5.15.2.9 T\_idtApIXtalId xtal

APLL crystal selection

Definition at line 78 of file ApI.h.

The documentation for this struct was generated from the following file:

- common/include/[ApI.h](#)

## 5.16 T\_idtApIFreqMapping Struct Reference

Type definition for mapping output frequency to APLL configuration.

```
#include <ApIIFreqProfile.h>
```

### Data Fields

- [T\\_idtApIInputFreqType](#) apIInputFreq
- [T\\_idtApIDividerValue](#) apIPreDivider
- [T\\_idtApIDividerValue](#) apIFbDivider
- [T\\_idtApIXtalType](#) apIXtalFreq
- [T\\_idtApIDividerValue](#) apIOutputDivider
- [T\\_idtApIOutputFreqType](#) apIOutputFreq
- [T\\_idtApIOutputSupportedType](#) apIOutputs

### 5.16.1 Detailed Description

Type definition for mapping output frequency to APLL configuration.

Definition at line 100 of file ApIIFreqProfile.h.

### 5.16.2 Field Documentation

#### 5.16.2.1 T\_idtApIIDividerValue apIIFbDivider

Feedback (oscillator) divider

Definition at line 104 of file ApIIFreqProfile.h.

#### 5.16.2.2 T\_idtApIInputFreqType apIInputFreq

Input frequency

Definition at line 102 of file ApIIFreqProfile.h.

#### 5.16.2.3 T\_idtApIIDividerValue apIOutputDivider

Output divider

Definition at line 106 of file ApIIFreqProfile.h.

#### 5.16.2.4 T\_idtApIOutputFreqType apIOutputFreq

Output frequency type

Definition at line 107 of file ApIIFreqProfile.h.

#### 5.16.2.5 T\_idtApIOutputSupportedType apIOutputs

Output supported (A or B or both)

Definition at line 108 of file ApIIFreqProfile.h.

#### 5.16.2.6 T\_idtApIIDividerValue apIPreDivider

Pre- (input) divider

Definition at line 103 of file ApIIFreqProfile.h.

#### 5.16.2.7 T\_idtApIIXtalType apIIXtalFreq

Oscillator type

Definition at line 105 of file ApIIFreqProfile.h.

The documentation for this struct was generated from the following file:

- [apI/include/ApIIFreqProfile.h](#)

## 5.17 T\_idtApIIXtal Struct Reference

Structure containing APLL crystal id and frequency information.

```
#include <ApITypeDef.h>
```

## Data Fields

- [T\\_idtApIIXtalId apIIXtalId](#)
- [T\\_idtApIIXtalType apIIXtalFreq](#)

### 5.17.1 Detailed Description

Structure containing APLL crystal id and frequency information.

Definition at line 244 of file ApIITypeDef.h.

### 5.17.2 Field Documentation

#### 5.17.2.1 T\_idtApIIXtalType apIIXtalFreq

APLL crystal frequency

Definition at line 247 of file ApIITypeDef.h.

#### 5.17.2.2 T\_idtApIIXtalId apIIXtalId

APLL crystal ID

Definition at line 246 of file ApIITypeDef.h.

The documentation for this struct was generated from the following file:

- [apIi/include/ApIITypeDef.h](#)

## 5.18 T\_idtApIIXtalList Struct Reference

Wrapper structure containing APLL crystal id and frequency information.

```
#include <access.h>
```

## Data Fields

- int [xtal1ApIi1](#)
- [T\\_idtApIIXtalType xtal1ApIi1Freq](#)
- int [xtal3ApIi1](#)
- [T\\_idtApIIXtalType xtal3ApIi1Freq](#)
- int [xtal2ApIi2](#)
- [T\\_idtApIIXtalType xtal2ApIi2Freq](#)
- int [xtal4ApIi2](#)
- [T\\_idtApIIXtalType xtal4ApIi2Freq](#)

### 5.18.1 Detailed Description

Wrapper structure containing APLL crystal id and frequency information.

This structure is used for TCL extension library

Definition at line 50 of file access.h.

## 5.18.2 Field Documentation

### 5.18.2.1 int xtal1Apl1

APLL1 XTAL1 flag

Definition at line 52 of file access.h.

### 5.18.2.2 T\_idtAplIXtalType xtal1Apl1Freq

APLL1 XTAL1 Frequency

Definition at line 53 of file access.h.

### 5.18.2.3 int xtal2Apl2

APLL2 XTAL2 flag

Definition at line 56 of file access.h.

### 5.18.2.4 T\_idtAplIXtalType xtal2Apl2Freq

APLL2 XTAL2 Frequency

Definition at line 57 of file access.h.

### 5.18.2.5 int xtal3Apl1

APLL1 XTAL3 flag

Definition at line 54 of file access.h.

### 5.18.2.6 T\_idtAplIXtalType xtal3Apl1Freq

APLL1 XTAL3 Frequency

Definition at line 55 of file access.h.

### 5.18.2.7 int xtal4Apl2

APLL2 XTAL4 flag

Definition at line 58 of file access.h.

### 5.18.2.8 T\_idtAplIXtalType xtal4Apl2Freq

APLL2 XTAL4 Frequency

Definition at line 59 of file access.h.

The documentation for this struct was generated from the following file:

- 82V3910/sample/include/[access.h](#)

## 5.19 T\_idtDrvAccess Struct Reference

Initialization structure detailing device access information for device driver.

```
#include <Define.h>
```

### Data Fields

- [T\\_idtAccessInitFunc initFunc\\_m](#)
- [T\\_idtAccessDeInitFunc deinitFunc\\_m](#)
- [T\\_idtReadFunc readFunc\\_m](#)
- [T\\_idtWriteFunc writeFunc\\_m](#)
- void \* [userData\\_m](#)

### 5.19.1 Detailed Description

Initialization structure detailing device access information for device driver.

Definition at line 193 of file Define.h.

### 5.19.2 Field Documentation

#### 5.19.2.1 T\_idtAccessDeInitFunc deinitFunc\_m

Access deinitialization

Definition at line 196 of file Define.h.

#### 5.19.2.2 T\_idtAccessInitFunc initFunc\_m

Access initialization

Definition at line 195 of file Define.h.

#### 5.19.2.3 T\_idtReadFunc readFunc\_m

Read function

Definition at line 197 of file Define.h.

#### 5.19.2.4 void\* userData\_m

Extra user data

Definition at line 199 of file Define.h.

#### 5.19.2.5 T\_idtWriteFunc writeFunc\_m

Write function

Definition at line 198 of file Define.h.

The documentation for this struct was generated from the following file:

- common/include/[Define.h](#)

## 5.20 T\_idtDrvDeviceConfig Struct Reference

Device type/variant information.

```
#include <Define.h>
```

### Data Fields

- T\_osUInt8 [inputExt2IntXlate\\_ma](#) [IDT\_DRV\_MAX\_INPUT+1]
- T\_osUInt8 [inSyncXlate\\_ma](#) [2]
- T\_osUInt8 [outputExt2IntXlate\\_ma](#) [IDT\_DRV\_MAX\_OUTPUT+1]
- T\_idtOutputApplConfig [outPutApplConfig](#) [IDT\_DRV\_MAX\_OUTPUT+1]
- T\_osUInt8 [outSyntXlate\\_ma](#) [2]
- T\_osUInt8 [numberOfSonetGigEthPath\\_m](#)
- T\_osUInt8 [numberOfAppl\\_m](#)
- T\_osUInt8 [populatedAppl\\_ma](#) [APLL\_INST\_MAX]
- T\_osUInt8 [pllExt2IntXlate\\_ma](#) [DPLL\_INSTANCE\_MAX]
- T\_osUInt8 [pllInt2ExtXlate\\_ma](#) [DPLL\_INSTANCE\_MAX]
- T\_idtDpllType [t0PllMode\\_m](#)
- T\_osUInt8 [dcoModeSupport\\_ma](#) [DPLL\_INSTANCE\_MAX]
- T\_osUInt8 [phaseSlopeLimiting\\_ma](#) [DPLL\_INSTANCE\_MAX]
- T\_osUInt8 [maxNumXtalPerAppl\\_m](#)
- T\_osUInt8 [supportedXtalId\\_ma](#) [APLL\_XTAL\_MAX]
- T\_osUInt8 [supportedXtalFreq\\_ma](#) [APLL\_XTAL\_MAX][APLL\_XTAL\_FREQ\_MAX]
- T\_idtInputFreqValid [inputFreqValidFunc\\_m](#)
- T\_idtOutputFreqValid [outputFreqValidFunc\\_m](#)

### 5.20.1 Detailed Description

Device type/variant information.

This data structure contains device variant information.

Definition at line 219 of file Define.h.

### 5.20.2 Field Documentation

#### 5.20.2.1 T\_osUInt8 dcoModeSupport\_ma[DPLL\_INSTANCE\_MAX]

DPLL support DCO mode

Definition at line 232 of file Define.h.

#### 5.20.2.2 T\_osUInt8 inputExt2IntXlate\_ma[IDT\_DRV\_MAX\_INPUT+1]

Input translation table (external to internal)

Definition at line 221 of file Define.h.

#### 5.20.2.3 T\_idtInputFreqValid inputFreqValidFunc\_m

Input frequency validation function

Definition at line 237 of file Define.h.

**5.20.2.4 T\_osUInt8 inSyncXlate\_ma[2]**

Input sync translation table

Definition at line 222 of file Define.h.

**5.20.2.5 T\_osUInt8 maxNumXtalPerAppl\_m**

Maximum number of XTALs per APLL

Definition at line 234 of file Define.h.

**5.20.2.6 T\_osUInt8 numberOfAppl\_m**

Number of APLLs

Definition at line 227 of file Define.h.

**5.20.2.7 T\_osUInt8 numberOfSonetGigEthPath\_m**

Number of Sonet GigE path

Definition at line 226 of file Define.h.

**5.20.2.8 T\_idtOutputApplConfig outPutApplConfig[IDT\_DRV\_MAX\_OUTPUT+1]**

Output APLL configuration

Definition at line 224 of file Define.h.

**5.20.2.9 T\_osUInt8 outputExt2IntXlate\_ma[IDT\_DRV\_MAX\_OUTPUT+1]**

Output translation table (external to internal)

Definition at line 223 of file Define.h.

**5.20.2.10 T\_idtOuputFreqValid outputFreqValidFunc\_m**

Output frequency validation function

Definition at line 238 of file Define.h.

**5.20.2.11 T\_osUInt8 outSyntXlate\_ma[2]**

Output sync translation table

Definition at line 225 of file Define.h.

**5.20.2.12 T\_osUInt8 phaseSlopeLimiting\_ma[DPLL\_INSTANCE\_MAX]**

DPLL support phase slope limiting

Definition at line 233 of file Define.h.

## 5.20.2.13 T\_osUInt8 pllExt2IntXlate\_ma[DPLL\_INSTANCE\_MAX]

DPLL translation table (external to internal)

Definition at line 229 of file Define.h.

## 5.20.2.14 T\_osUInt8 pllInt2ExtXlate\_ma[DPLL\_INSTANCE\_MAX]

DPLL translation table (internal to external)

Definition at line 230 of file Define.h.

## 5.20.2.15 T\_osUInt8 populatedAppl\_ma[APLL\_INST\_MAX]

Populated APLL(s)

Definition at line 228 of file Define.h.

## 5.20.2.16 T\_osUInt8 supportedXtalFreq\_ma[APLL\_XTAL\_MAX][APLL\_XTAL\_FREQ\_MAX]

Supported XTAL frequency

Definition at line 236 of file Define.h.

## 5.20.2.17 T\_osUInt8 supportedXtalId\_ma[APLL\_XTAL\_MAX]

Supported XTAL id

Definition at line 235 of file Define.h.

## 5.20.2.18 T\_idtDpllType t0PllMode\_m

If T0 exists, T0 mode

Definition at line 231 of file Define.h.

The documentation for this struct was generated from the following file:

- common/include/[Define.h](#)

## 5.21 T\_idtDrvHdlr Struct Reference

Device driver handler.

```
#include <Define.h>
```

## Data Fields

- const char \* [name\\_m](#)
- [T\\_idtDeviceType](#) [deviceType\\_m](#)
- [T\\_idtDrvAccess](#) [drvAccess\\_m](#)
- const [T\\_idtDrvDeviceConfig](#) \* [deviceConfig\\_m](#)
- [T\\_osUInt8](#) [numberOfApplXtal\\_m](#)
- [T\\_idtApplXtal](#) [xtalList](#) [[IDT\\_DRV\\_MAX\\_APLL\\_XTAL](#)]
- struct [S\\_IDTPathConfiguration](#) \* [currOutPathCfg\\_mp](#)

- void \* [dpllI2CTransData\\_mp](#)
- void \* [apllI2CTransData\\_mp](#)
- [T\\_idtI2CAddrTransFunc](#) [i2cAddrTransFunc\\_m](#)
- [T\\_osUInt8](#) [i2cAddr](#)

### 5.21.1 Detailed Description

Device driver handler.

All device driver functions have driver handlers as an input parameter. This handler allows the device driver to support an unlimited number devices.

Definition at line 255 of file Define.h.

### 5.21.2 Field Documentation

#### 5.21.2.1 void\* [apllI2CTransData\\_mp](#)

Helper data for translating I2C address mapping to apll register space

Definition at line 265 of file Define.h.

#### 5.21.2.2 struct [S\\_IDTPathConfiguration](#)\* [currOutPathCfg\\_mp](#)

Current output path configuration - used by high level API code. Basic driver doesn't require this.

Definition at line 263 of file Define.h.

#### 5.21.2.3 const [T\\_idtDrvDeviceConfig](#)\* [deviceConfig\\_m](#)

Device configuration

Definition at line 260 of file Define.h.

#### 5.21.2.4 [T\\_idtDeviceType](#) [deviceType\\_m](#)

Device Type

Definition at line 258 of file Define.h.

#### 5.21.2.5 void\* [dpllI2CTransData\\_mp](#)

Helper data for translating I2C address mapping to dpll register space

Definition at line 264 of file Define.h.

#### 5.21.2.6 [T\\_idtDrvAccess](#) [drvAccess\\_m](#)

Device access functions

Definition at line 259 of file Define.h.

#### 5.21.2.7 [T\\_osUInt8](#) [i2cAddr](#)

7-bit unshifted i2c slave address for PLL

Definition at line 267 of file Define.h.

#### 5.21.2.8 T\_idtI2CaddrTransFunc i2cAddrTransFunc\_m

Function to translate I2C address

Definition at line 266 of file Define.h.

#### 5.21.2.9 const char\* name\_m

Name of driver instance

Definition at line 257 of file Define.h.

#### 5.21.2.10 T\_osUInt8 numberOfApllXtal\_m

Number of APLL XTALs

Definition at line 261 of file Define.h.

#### 5.21.2.11 T\_idtApllXtal xtalList[IDT\_DRV\_MAX\_APLL\_XTAL]

List of APLL XTALs

Definition at line 262 of file Define.h.

The documentation for this struct was generated from the following file:

- common/include/[Define.h](#)

## 5.22 T\_IDTFreq2bfVal Struct Reference

Structure to contain translation information for frequency to bitfield value.

### Data Fields

- T\_osUInt32 [freq\\_m](#)
- [T\\_idtInputFrequencyBitfieldValue bfVal\\_m](#)

### 5.22.1 Detailed Description

Structure to contain translation information for frequency to bitfield value.

Definition at line 50 of file inputFreq.c.

### 5.22.2 Field Documentation

#### 5.22.2.1 T\_idtInputFrequencyBitfieldValue bfVal\_m

Bitfield value

Definition at line 52 of file inputFreq.c.

#### 5.22.2.2 T\_osUInt32 freq\_m

Frequency (in Hz)

Definition at line 51 of file inputFreq.c.

The documentation for this struct was generated from the following file:

- [common/hlapi/src/inputFreq.c](#)

### 5.23 T\_idtHfDivEntry Struct Reference

Structure to use as high frequency (HF) divider information.

#### Data Fields

- [T\\_osUInt8 divValue\\_m](#)
- [T\\_idtHighFrequencyDivisor hfDivValue\\_m](#)

#### 5.23.1 Detailed Description

Structure to use as high frequency (HF) divider information.

Definition at line 58 of file inputFreq.c.

#### 5.23.2 Field Documentation

##### 5.23.2.1 T\_osUInt8 divValue\_m

Divisor value

Definition at line 59 of file inputFreq.c.

##### 5.23.2.2 T\_idtHighFrequencyDivisor hfDivValue\_m

```
HF divisor bit field
```

value

Definition at line 60 of file inputFreq.c.

The documentation for this struct was generated from the following file:

- [common/hlapi/src/inputFreq.c](#)

### 5.24 T\_idtI2CtransData Struct Reference

I2C address translation information used internally by the driver.

```
#include <General.h>
```

#### Data Fields

- [T\\_osUInt8 bitLocation](#)
- [T\\_osUInt8 bitValue](#)

### 5.24.1 Detailed Description

I2C address translation information used internally by the driver.

Definition at line 78 of file General.h.

### 5.24.2 Field Documentation

#### 5.24.2.1 T\_osUInt8 bitLocation

Definition at line 80 of file General.h.

#### 5.24.2.2 T\_osUInt8 bitValue

Definition at line 81 of file General.h.

The documentation for this struct was generated from the following file:

- common/include/[General.h](#)

## 5.25 T\_idtOutputApIConfig Struct Reference

Structure containing APLL inforatmion.

```
#include <Define.h>
```

### Data Fields

- [T\\_idtApIId apIInst](#)
- [T\\_osUInt8 apIInput](#)
- [T\\_idtApIOutput apIOutput](#)
- [T\\_osUInt8 extOutput](#)

### 5.25.1 Detailed Description

Structure containing APLL inforatmion.

Definition at line 205 of file Define.h.

### 5.25.2 Field Documentation

#### 5.25.2.1 T\_osUInt8 apIInput

APLL input port number

Definition at line 208 of file Define.h.

#### 5.25.2.2 T\_idtApIId apIInst

APLL instanace ID

Definition at line 207 of file Define.h.

### 5.25.2.3 T\_idtApIOutput apIOutput

APLL output port number

Definition at line 209 of file Define.h.

### 5.25.2.4 T\_osUInt8 extOutput

External output port number

Definition at line 210 of file Define.h.

The documentation for this struct was generated from the following file:

- common/include/[Define.h](#)

## 5.26 T\_IDTPathConfiguration Struct Reference

Structure containing output path configuration.

```
#include <Dpll.h>
```

### Data Fields

- [networkType\\_t networkType\\_m](#)
- [T\\_idtDpllOutputSelectorA T0SelA\\_m](#)
- [T\\_idtDpllOutputSelectorB T0SelB\\_m](#)
- [T\\_idtDpllOutputSelectorA T4SelA\\_m](#)
- [T\\_idtDpllOutputSelectorB T4SelB\\_m](#)
- [T\\_idtSonetGigEthOutput sonetGigEthSel\\_ma \[SonetGigEthSel\\_NUMMAX\]](#)

### 5.26.1 Detailed Description

Structure containing output path configuration.

Definition at line 114 of file Dpll.h.

### 5.26.2 Field Documentation

#### 5.26.2.1 networkType\_t networkType\_m

Network type

Definition at line 115 of file Dpll.h.

#### 5.26.2.2 T\_idtSonetGigEthOutput sonetGigEthSel\_ma[SonetGigEthSel\_NUMMAX]

Sonet/GigE Output Path Selector

Definition at line 122 of file Dpll.h.

#### 5.26.2.3 T\_idtDpllOutputSelectorA T0SelA\_m

T0 Output Path Selector A

Definition at line 116 of file Dpll.h.

#### 5.26.2.4 T\_idtDpllOutputSelectorB T0SelB\_m

T0 Output Path Selector B

Definition at line 117 of file Dpll.h.

#### 5.26.2.5 T\_idtDpllOutputSelectorA T4SelA\_m

T4 Output Path Selector A

Definition at line 118 of file Dpll.h.

#### 5.26.2.6 T\_idtDpllOutputSelectorB T4SelB\_m

T4 Output Path Selector B

Definition at line 119 of file Dpll.h.

The documentation for this struct was generated from the following file:

- [common/include/Dpll.h](#)

## 5.27 T\_idtSonetGigEthBitfield2PathConfig Struct Reference

Structure Translate from Sonet/GigE bitfield to path configuration.

### Data Fields

- [T\\_idtDpllInstance inputDpll\\_m](#)
- [T\\_idtSonetGigEthOutput outputFreq\\_m](#)

### 5.27.1 Detailed Description

Structure Translate from Sonet/GigE bitfield to path configuration.

Definition at line 51 of file DpllCnfg.c.

### 5.27.2 Field Documentation

#### 5.27.2.1 T\_idtDpllInstance inputDpll\_m

input DPLL type

Definition at line 53 of file DpllCnfg.c.

#### 5.27.2.2 T\_idtSonetGigEthOutput outputFreq\_m

output frequency

Definition at line 54 of file DpllCnfg.c.

The documentation for this struct was generated from the following file:

- [common/src/DpllCnfg.c](#)

## 5.28 T\_SampleConfigEntry Struct Reference

Sample configuration information.

### Data Fields

- [T\\_SampleConfigFunc configFunction\\_m](#)
- [T\\_SampleConfigDescrFunc configDescription\\_m](#)

### 5.28.1 Detailed Description

Sample configuration information.

Definition at line 75 of file main.c.

### 5.28.2 Field Documentation

#### 5.28.2.1 T\_SampleConfigDescrFunc configDescription\_m

Function to output description

Definition at line 77 of file main.c.

#### 5.28.2.2 T\_SampleConfigFunc configFunction\_m

Function to configure

Definition at line 76 of file main.c.

The documentation for this struct was generated from the following file:

- 82V3910/sample/src/[main.c](#)

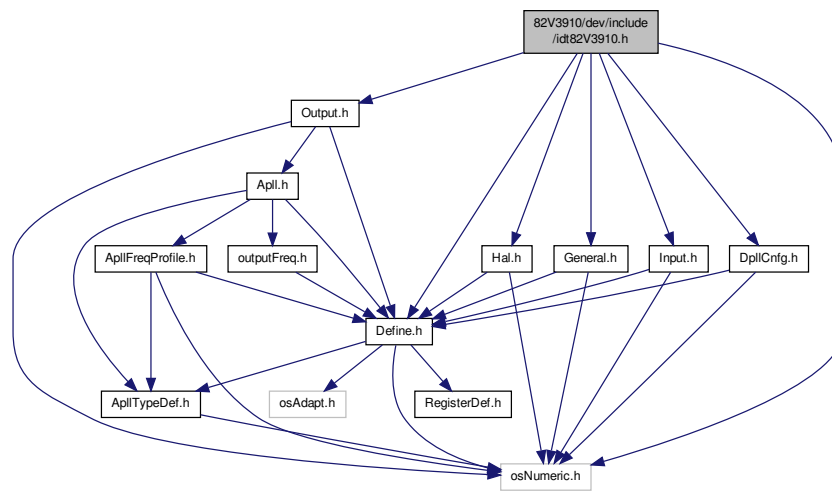
## Chapter 6

# File Documentation

### 6.1 82V3910/dev/include/idt82V3910.h File Reference

```
#include "General.h"  
#include "Define.h"  
#include "Input.h"  
#include "Output.h"  
#include "DpllCnfg.h"  
#include "Hal.h"  
#include "osNumeric.h"
```

Include dependency graph for idt82V3910.h:



### Functions

- void [IDTGeneral\\_InitDevice](#) (T\_idtDrvHdlr hdlr)

*Initialize the device to its power on defaults.*

- T\_idtDrvHdlr [IDTGeneral\\_InitDriver](#) (const char \*name, T\_idtDrvAccess drvAccess, T\_idtDeviceType deviceType, T\_idtAplXtal \*xtalList, T\_osUInt8 numberOfXtal, T\_osUInt8 i2cAddr, int useHighLevelApi)

*Initialize device driver.*

### 6.1.1 Function Documentation

#### 6.1.1.1 void IDTGeneral\_InitDevice ( T\_idtDrvHdlr *hdlr* )

Initialize the device to its power on defaults.

This function is optionally called. This function is required only after power on to initialize device registers to know states. For "warm" restarts skip this function.

##### Parameters

|             |                      |
|-------------|----------------------|
| <i>hdlr</i> | Device driver handle |
|-------------|----------------------|

Definition at line 101 of file 82V3910.c.

#### 6.1.1.2 T\_idtDrvHdlr IDTGeneral\_InitDriver ( const char \* *name*, T\_idtDrvAccess *drvAccess*, T\_idtDeviceType *deviceType*, T\_idtApplXtal \* *xtalList*, T\_osUInt8 *numberOfXtal*, T\_osUInt8 *i2cAddr*, int *useHighLevelApi* )

Initialize device driver.

This function creates a driver handler used as reference to a specific device. It does not perform any device access.

The *isHighLevelApi* determines whether device driver initializes high level APIs.

##### Parameters

|                  |                              |
|------------------|------------------------------|
| <i>name</i>      | Name of device (ex. "myDev") |
| <i>drvAccess</i> | Device access data.          |

##### See Also

*drvAccess\_t* structure information

##### Parameters

|                        |                                                                                                             |
|------------------------|-------------------------------------------------------------------------------------------------------------|
| <i>deviceType</i>      | What supported device                                                                                       |
| <i>xtalList</i>        | A list of crystal(s) connected to APLL(s), set this parameter to NULL if there is no XTAL connected to APLL |
| <i>numberOfXtal</i>    | Number of crystal(s) in the <i>xtalList</i> , set this parameter to 0 if <i>xtalList</i> is NULL            |
| <i>i2cAddr</i>         | 7-bit unmodified i2c slave address                                                                          |
| <i>useHighLevelApi</i> | 0-Don't use high level API, 1-high level API enabled.                                                       |

##### Returns

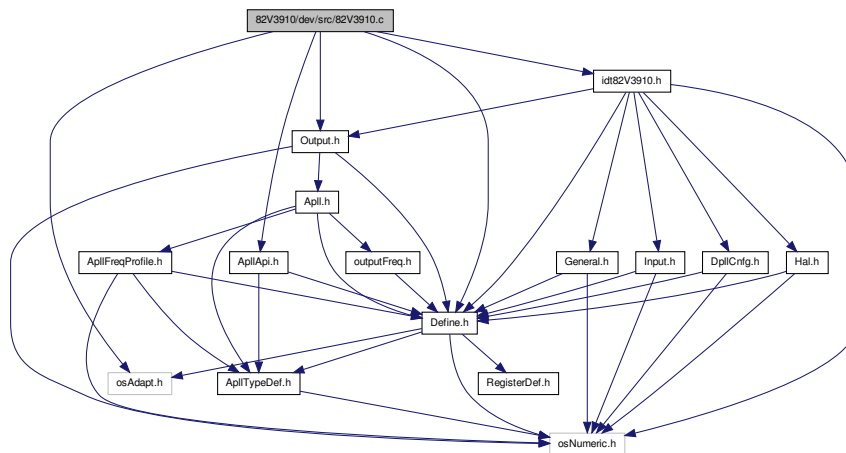
Device driver handler (used in all APIs)

Definition at line 167 of file 82V3910.c.

## 6.2 82V3910/dev/src/82V3910.c File Reference

```
#include "osAdapt.h"
#include "idt82V3910.h"
#include "Define.h"
#include "ApplApi.h"
#include "Output.h"
```

Include dependency graph for 82V3910.c:



## Functions

- void [IDTGeneral\\_InitDevice](#) ([T\\_idtDrvHdlr](#) hdlr)  
*Initialize the device to its power on defaults.*
- [T\\_idtDrvHdlr](#) [IDTGeneral\\_InitDriver](#) (const char \*name, [T\\_idtDrvAccess](#) drvAccess, [T\\_idtDeviceType](#) deviceType, [T\\_idtAplIXtal](#) \*xtalList, [T\\_osUInt8](#) numberOfXtal, [T\\_osUInt8](#) i2cAddr, int useHighLevelApi)  
*Initialize device driver.*

## Variables

- const [T\\_idtDrvDeviceConfig](#) [idt82V3910Config](#)  
*Specific device profile.*

### 6.2.1 Function Documentation

#### 6.2.1.1 void IDTGeneral\_InitDevice ( T\_idtDrvHdlr hdlr )

Initialize the device to its power on defaults.

This function is optionally called. This function is required only after power on to initialize device registers to know states. For "warm" restarts skip this function.

#### Parameters

|             |                      |
|-------------|----------------------|
| <i>hdlr</i> | Device driver handle |
|-------------|----------------------|

Definition at line 101 of file 82V3910.c.

#### 6.2.1.2 T\_idtDrvHdlr IDTGeneral\_InitDriver ( const char \* name, T\_idtDrvAccess drvAccess, T\_idtDeviceType deviceType, T\_idtAplIXtal \* xtalList, T\_osUInt8 numberOfXtal, T\_osUInt8 i2cAddr, int useHighLevelApi )

Initialize device driver.

This function creates a driver handler used as reference to a specific device. It does not perform any device access.

The `isHighLevelApi` determines whether device driver initializes high level APIs.

#### Parameters

|                  |                              |
|------------------|------------------------------|
| <i>name</i>      | Name of device (ex. "myDev") |
| <i>drvAccess</i> | Device access data.          |

#### See Also

`drvAccess_t` structure information

#### Parameters

|                        |                                                                                                             |
|------------------------|-------------------------------------------------------------------------------------------------------------|
| <i>deviceType</i>      | What supported device                                                                                       |
| <i>xtalList</i>        | A list of crystal(s) connected to APLL(s), set this parameter to NULL if there is no XTAL connected to APLL |
| <i>numberOfXtal</i>    | Number of crystal(s) in the xtalList, set this parameter to 0 if xtalList is NULL                           |
| <i>i2cAddr</i>         | 7-bit unmodified i2c slave address                                                                          |
| <i>useHighLevelApi</i> | 0-Don't use high level API, 1-high level API enabled.                                                       |

#### Returns

Device driver handler (used in all APIs)

Definition at line 167 of file 82V3910.c.

## 6.2.2 Variable Documentation

### 6.2.2.1 `const T_idtDrvDeviceConfig idt82V3910Config`

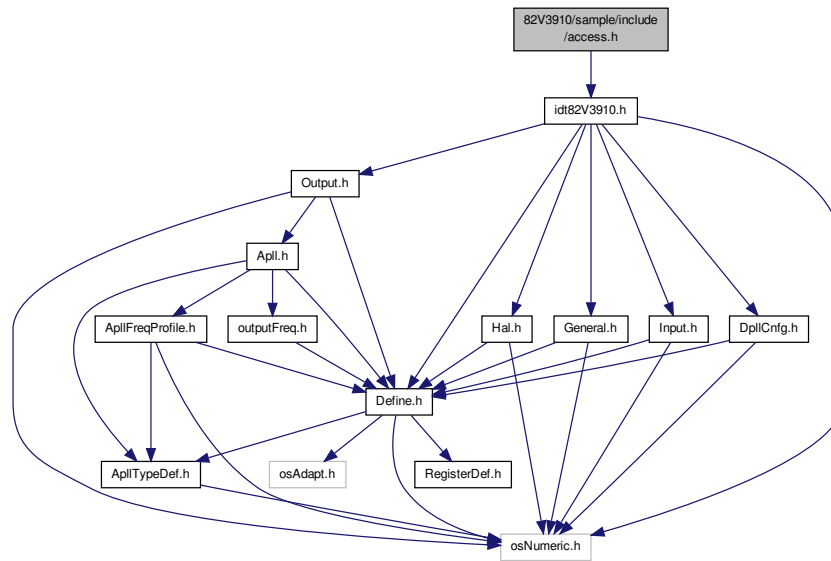
Specific device profile.

Definition at line 50 of file 82V3910.c.

## 6.3 82V3910/sample/include/access.h File Reference

```
#include "idt82V3910.h"
```

Include dependency graph for access.h:



## Data Structures

- struct [T\\_idtAplXtalList](#)

*Wrapper structure containing APLL crystal id and frequency information.*

## Functions

- [T\\_idtDrvHdlr IDTSample\\_InitDriverI2c](#) ([T\\_idtAplXtalList](#) \*vcxoList, [T\\_osUInt8](#) coreI2CAddr)  
*IDTSample\_InitDriverI2c Utility function to initialize device driver for I2C access.*
- [T\\_idtDrvHdlr IDTSample\\_InitDriverSimulator](#) ([T\\_idtAplXtalList](#) \*vcxoList, [T\\_osUInt8](#) coreI2CAddr)  
*Utility function to initialize device driver for simulator access.*
- [void IDTSample\\_ResetXtalList](#) ([T\\_idtAplXtalList](#) \*vcxoList)  
*Utility function to reset APLL XTAL list to unconfigured value.*
- [void IDTSample\\_SetLogVerbosity](#) ([int](#) level)  
*Utility function to change log verbosity level access.*
- [int IDTSample\\_GetLogVerbosity](#) ([void](#))  
*Utility function to get current log verbosity level.*

### 6.3.1 Function Documentation

#### 6.3.1.1 [int IDTSample\\_GetLogVerbosity](#) ( [void](#) )

Utility function to get current log verbosity level.

This function is used for TCL extension library

Definition at line 326 of file access.c.

### 6.3.1.2 T\_idtDrvHdlr IDTSample\_InitDriverI2c ( T\_idtApIIXtalList \* *xtalList*, T\_osUInt8 *coreI2CAAddr* )

IDTSample\_InitDriverI2c Utility function to initialize device driver for I2C access.

This function is used for TCL extension library

#### Parameters

|                     |                    |
|---------------------|--------------------|
| <i>xtalList</i>     | List of XTAL       |
| <i>coreI2CAAddr</i> | Device I2C address |

#### Returns

Device driver handle

Definition at line 161 of file access.c.

### 6.3.1.3 T\_idtDrvHdlr IDTSample\_InitDriverSimulator ( T\_idtApIIXtalList \* *xtalList*, T\_osUInt8 *coreI2CAAddr* )

Utility function to initialize device driver for simulator access.

This function is used for TCL extension library

#### Parameters

|                     |                    |
|---------------------|--------------------|
| <i>xtalList</i>     | List of XTAL       |
| <i>coreI2CAAddr</i> | Device I2C address |

#### Returns

Device driver handle

Definition at line 229 of file access.c.

### 6.3.1.4 void IDTSample\_ResetXtalList ( T\_idtApIIXtalList \* *xtalList* )

Utility function to reset APLL XTAL list to unconfigured value.

This function is used for TCL extension library

#### Parameters

|                 |              |
|-----------------|--------------|
| <i>xtalList</i> | List of XTAL |
|-----------------|--------------|

Definition at line 296 of file access.c.

### 6.3.1.5 void IDTSample\_SetLogVerbosity ( int *level* )

Utility function to change log verbosity level access.

This function is used for TCL extension library

#### Parameters

|              |                     |
|--------------|---------------------|
| <i>level</i> | Log verbosity level |
|--------------|---------------------|

Definition at line 316 of file access.c.

## 6.4 82V3910/sample/include/loadstore.h File Reference

### Functions

- void [IDTSample\\_SaveRgfFile](#) (T\_idtDrvHdlr hdlr, char \*filename)  
*Function to output register contents of a device to standard IO or to a file.*
- void [IDTSample\\_LoadRgfFile](#) (T\_idtDrvHdlr hdlr, char \*filename)  
*Function to read in a file and program device registers.*

### 6.4.1 Function Documentation

#### 6.4.1.1 void IDTSample\_LoadRgfFile ( T\_idtDrvHdlr hdlr, char \* filename )

Function to read in a file and program device registers.

##### Parameters

|                 |                                                 |
|-----------------|-------------------------------------------------|
| <i>hdlr</i>     | Device driver handler                           |
| <i>filename</i> | Input file name to load register settings from. |

Definition at line 124 of file loadstore.c.

#### 6.4.1.2 void IDTSample\_SaveRgfFile ( T\_idtDrvHdlr hdlr, char \* filename )

Function to output register contents of a device to standard IO or to a file.

##### Parameters

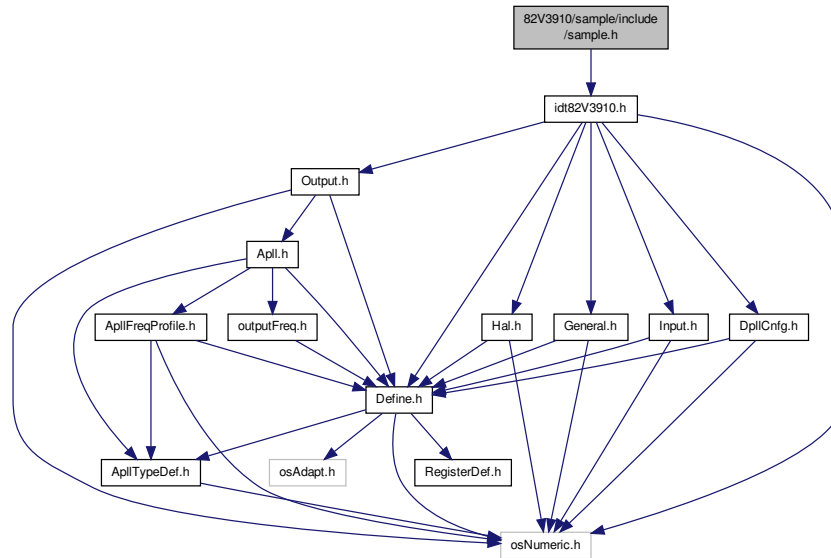
|                 |                                        |
|-----------------|----------------------------------------|
| <i>hdlr</i>     | Device driver handler                  |
| <i>filename</i> | Output file name, NULL for standard IO |

Definition at line 94 of file loadstore.c.

## 6.5 82V3910/sample/include/sample.h File Reference

```
#include "idt82V3910.h"
```

Include dependency graph for sample.h:



### Functions

- void [IDTSample\\_ConfigureSampleConfig1](#) (T\_idtDrvHdlr hdlr)  
Sample configuration 1.
- void [IDTSample\\_ConfigureSampleConfig2](#) (T\_idtDrvHdlr hdlr)  
Sample configuration 2.
- void [IDTSample\\_ConfigureSampleConfig3](#) (T\_idtDrvHdlr hdlr)  
Sample configuration 3.
- void [IDTSample\\_ConfigureSampleConfig4](#) (T\_idtDrvHdlr hdlr)  
Sample configuration 4.
- void [IDTSample\\_ConfigureSampleConfig5](#) (T\_idtDrvHdlr hdlr)  
Sample configuration 5.
- void [IDTSample\\_Configure1Pps](#) (T\_idtDrvHdlr hdlr)  
Sample configuration 1PPS.
- void [IDTSample\\_Configure1PpsHoldover](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for 1PPS holdover issue.
- void [IDTSample\\_ConfigureDcoMode](#) (T\_idtDrvHdlr hdlr)  
Function to configure device for DCO mode.
- void [IDTSample\\_SetCombinedDcoOffset](#) (T\_idtDrvHdlr hdlr, long pptOffset)  
Function to set combined DCO offset value. It uses both DCO (holdover) and nominal offsets.
- long [IDTSample\\_GetCombinedDcoOffset](#) (T\_idtDrvHdlr hdlr)  
Function to get combined DCO offset value. It uses both DCO (holdover) and nominal offsets.

### 6.5.1 Function Documentation

#### 6.5.1.1 void IDTSample\_Configure1Pps ( T\_idtDrvHdlr *hdlr* )

Sample configuration 1PPS.

```
IN3 (1PPS) -> T0 -> OUT3 (1PPS)
```

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 420 of file sample.c.

#### 6.5.1.2 void IDTSample\_Configure1PpsHoldover ( T\_idtDrvHdlr *hdlr* )

Sample configuration for 1PPS holdover issue.

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 449 of file sample.c.

#### 6.5.1.3 void IDTSample\_ConfigureDcoMode ( T\_idtDrvHdlr *hdlr* )

Function to configure device for DCO mode.

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 480 of file sample.c.

#### 6.5.1.4 void IDTSample\_ConfigureSampleConfig1 ( T\_idtDrvHdlr *hdlr* )

Sample configuration 1.

```
IN3 (25 MHz) -> T0 +-> OUT1 (25 MHz)
               +-> OUT6 (156.25 MHz)
```

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 46 of file sample.c.

#### 6.5.1.5 void IDTSample\_ConfigureSampleConfig2 ( T\_idtDrvHdlr *hdlr* )

Sample configuration 2.

```
IN3 (25 MHz) -> T0 +-> OUT1 (25 MHz)
               +-> OUT3 (8 kHz)
               +-> OUT5 (125 MHz)
```

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 98 of file sample.c.

**6.5.1.6 void IDTSample\_ConfigureSampleConfig3 ( T\_idtDrvHdlr *hdlr* )**

Sample configuration 3.

```
IN3 (25 MHz) -> T0 +-> OUT1 (25 MHz)
                +-> OUT3 (10 MHz)
                +-> OUT5 (1 PPS)
```

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 154 of file sample.c.

**6.5.1.7 void IDTSample\_ConfigureSampleConfig4 ( T\_idtDrvHdlr *hdlr* )**

Sample configuration 4.

```
Inputs: IN3 (10MHz), IN4 (1.544MHz), IN5 (25MHz), IN6 (8KHz), IN7 (8KHz),
        IN8 (156.25MHz), IN9 (156.25MHz), IN10 (25MHz), IN11 (25MHz)
Outputs: OUT1 (1.544MHz), OUT3 (10MHz), OUT4 (25MHz), OUT7 (156.25MHz),
        OUT10 (156.25MHz), OUT11 (156.25MHz)
T0 Profile: G.8262 Option 2
T4 Profile: Default
XTAL1/2: 25MHz
```

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 226 of file sample.c.

**6.5.1.8 void IDTSample\_ConfigureSampleConfig5 ( T\_idtDrvHdlr *hdlr* )**

Sample configuration 5.

```
Carbon copy of IDTSample_ConfigureSampleConfig4,
except there is no input and DPLL is in free-run
mode (default).
```

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 339 of file sample.c.

**6.5.1.9 long IDTSample\_GetCombinedDcoOffset ( T\_idtDrvHdlr *hdlr* )**

Function to get combined DCO offset value. It uses both DCO (holdover) and nominal offsets.

## Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

## Returns

Parts per trillion offset

Definition at line 542 of file sample.c.

#### 6.5.1.10 void IDTSample\_SetCombinedDcoOffset ( T\_idtDrvHdlr *hdlr*, long *pptOffset* )

Function to set combined DCO offset value. It uses both DCO (holdover) and nominal offsets.

## Parameters

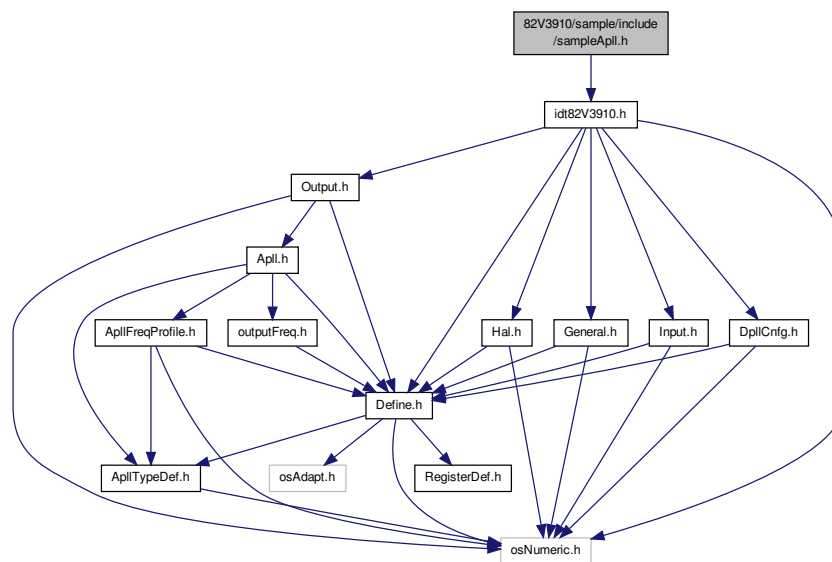
|                  |                           |
|------------------|---------------------------|
| <i>hdlr</i>      | Device driver handler     |
| <i>pptOffset</i> | Parts per trillion offset |

Definition at line 492 of file sample.c.

## 6.6 82V3910/sample/include/sampleAppl.h File Reference

```
#include "idt82V3910.h"
```

Include dependency graph for sampleAppl.h:



## Functions

- void [IDTSample\\_ConfigureSampleConfigOutput6\\_156\\_dot\\_25MHz](#) (T\_idtDrvHdlr *hdlr*)  
Sample configuration for 156.25MHz output at port 6.
- void [IDTSample\\_ConfigureSampleConfigOutput7\\_77\\_dot\\_76MHz](#) (T\_idtDrvHdlr *hdlr*)  
Sample configuration for 77.76 MHz output at port 7.

- void [IDTSample\\_ConfigureSampleConfigureOutput10\\_25\\_MHz](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for 25MHz output at port 10.
- void [IDTSample\\_SyncE\\_WAN](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for SyncE WAN application.
- void [IDTSample\\_Sonet](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for SONET application.
- void [IDTSample\\_DualMode](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for dual mode application.
- void [IDTSample\\_SyncE\\_Sonet](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for SyncE SONET application.
- void [IDTSample\\_SyncE\\_LAN](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for SyncE LAN application.
- void [IDTSample\\_GPS\\_1PPS](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for GPS 1PPS application.
- void [IDTSample\\_GPS](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for GPS application.

## 6.6.1 Function Documentation

### 6.6.1.1 void IDTSample\_ConfigureSampleConfigOutput6\_156\_dot\_25MHz ( T\_idtDrvHdlr hdlr )

Sample configuration for 156.25MHz output at port 6.

```
IN3 (25 MHz) -> T0 +-> OUT6 (Ethernet, 156.25 MHz)
```

#### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 47 of file sampleAppl.c.

### 6.6.1.2 void IDTSample\_ConfigureSampleConfigureOutput10\_25\_MHz ( T\_idtDrvHdlr hdlr )

Sample configuration for 25MHz output at port 10.

```
IN3 (25 MHz) -> T0 +-> OUT10 (Ethernet, 25 MHz)
```

#### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 125 of file sampleAppl.c.

### 6.6.1.3 void IDTSample\_ConfigureSampleConfigureOutput7\_77\_dot\_76MHz ( T\_idtDrvHdlr hdlr )

Sample configuration for 77.76 MHz output at port 7.

```
IN3 (25 MHz) -> T0 +-> OUT7 (SONET, 77.76 MHz)
```

#### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 86 of file sampleAppl.c.

#### 6.6.1.4 void IDTSample\_DualMode ( T\_idtDrvHdlr *hdlr* )

Sample configuration for dual mode application.

```
Inputs: IN3 (19.44MHz), IN5 (155.52MHz), IN14 (1.544MHz)
T0 Profile: ST3 SONET
T4 Profile: default
Select XTAL3: done automatically by IDTH1Api_ConfigureOutput function
              based on the output frequency. To manually select XTAL,
              call SetJaXtalId function.
Outputs: OUT1 (19.44MHz), OUT6 (155.52MHz), OUT7 (77.76MHz), OUT9 (1.544MHz)
T0: IN3, IN5, IN14 -> OUT1, OUT6, OUT7 (IN14 is highest priority)
T4: IN3, IN5 -> OUT9
APLL1 XTAL1 at 25MHz & XTAL3 at 24.8832MHz
```

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 410 of file sampleAppl.c.

#### 6.6.1.5 void IDTSample\_GPS ( T\_idtDrvHdlr *hdlr* )

Sample configuration for GPS application.

```
Inputs: IN3 (10MHz), IN4 (10MHz), EX_SYNC1 (1PPS), EX_SYNC2 (1PPS)
T0 Profile: BITS / SSU (Wideband) - default
Outputs: OUT1 (25MHz), OUT2 (19.44MHz), OUT3 (10MHz), OUT4 (1PPS),
          OUT6 (25MHz), OUT7 (156.25MHz), OUT9 (1.544MHz), OUT10 (155.52MHz),
          OUT11 (77.76MHz), FRSYNC_8K_1PPS (1PPS), MFRSYNC_2K_1PPS (1PPS)
APLL1 XTAL1 = 25MHz & APLL2 XTAL2 = 24.8832MHz
```

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 1013 of file sampleAppl.c.

#### 6.6.1.6 void IDTSample\_GPS\_1PPS ( T\_idtDrvHdlr *hdlr* )

Sample configuration for GPS 1PPS application.

```
Inputs: IN3 (1PPS), IN5 (1PPS)
T0 Profile: GPS (15mHz)
Outputs: OUT1 (25MHz), OUT2 (19.44MHz), OUT3 (10MHz), OUT4 (1PPS),
          OUT6 (25MHz), OUT7 (156.25MHz), OUT9 (2.048MHz),
          OUT10 (155.52MHz), OUT11 (77.76MHz)
APLL1 XTAL1 at 25MHz & APLL2 XTAL2 at 24.8832MHz
```

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 835 of file sampleAppl.c.

### 6.6.1.7 void IDTSample\_Sonet ( T\_idtDrvHdlr hdlr )

Sample configuration for SONET application.

```
Inputs: IN3 (19.44MHz), IN5 (155.52MHz), IN14 (1.544MHz)
T0 Profile: ST3 SONET
T4 Profile: default
Outputs: OUT1 (19.44MHz), OUT9 (1.544MHz), OUT10 (155.52MHz)
T0: IN3, IN5, IN14 -> OUT1, OUT10 (IN14 is highest priority)
T4: IN3, IN5 -> OUT9
APLL2 XTAL2 at 24.8832MHz
```

#### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 294 of file sampleAppl.c.

### 6.6.1.8 void IDTSample\_SyncE\_LAN ( T\_idtDrvHdlr hdlr )

Sample configuration for SyncE LAN application.

```
Inputs: IN3 (25MHz), IN14 (2.048MHz)
T0 Profile: G.8262 Option 2
T4 Profile: BITS / SSU (Wideband)
Outputs: OUT1 (25MHz), OUT10 (25.78125), OUT11 (161.1328125MHz), OUT9 (2.048MHz)
T0: IN3, IN14 -> OUT1, OUT10, OUT11 (IN14 is highest priority)
T4: IN3 -> OUT9
APLL2 XTAL4 at 25.78125MHz
```

#### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 720 of file sampleAppl.c.

### 6.6.1.9 void IDTSample\_SyncE\_Sonet ( T\_idtDrvHdlr hdlr )

Sample configuration for SyncE SONET application.

```
Inputs: IN3 (25MHz), IN5 (156.25MHz), IN6 (155.52MHz), IN7 (19.44MHz), IN14 (1.544MHz)
T0 Profile: G.8262 Option 2
T4 Profile: BITS / SSU (Wideband) - default
Outputs: OUT1 (25MHz), OUT2 (19.44MHz), OUT6 (25MHz), OUT7 (156.25MHz),
        OUT9 (1.544MHz), OUT10 (155.52MHz), OUT11 (77.76MHz)
T0: IN3, IN5, IN6, IN7, IN14 -> OUT1, OUT2, OUT6, OUT7, OUT10, OUT11 (IN14 is highest priority)
T4: IN3, IN5, IN6, IN7 -> OUT9
APLL1 XTAL1 at 25MHz & APLL2 XTAL2 at 24.8832MHz
```

#### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 534 of file sampleAppl.c.

### 6.6.1.10 void IDTSample\_SyncE\_WAN ( T\_idtDrvHdlr hdlr )

Sample configuration for SyncE WAN application.

Inputs: IN3 (25MHz), IN5 (156.25MHz), IN14 (1.544MHz)  
 T0 Profile: G.8262 Option 2  
 T4 Profile: Default  
 Outputs: OUT1 (25MHz), OUT6 (25MHz), OUT7 (156.25MHz), OUT9 (1.544MHz)  
 T0: IN3, IN5, IN14 -> OUT1, OUT10, OUT11 (IN14 has highest priority)  
 T4: IN3, IN5 -> OUT9  
 APLL1 XTAL1 at 25MHz

#### Parameters

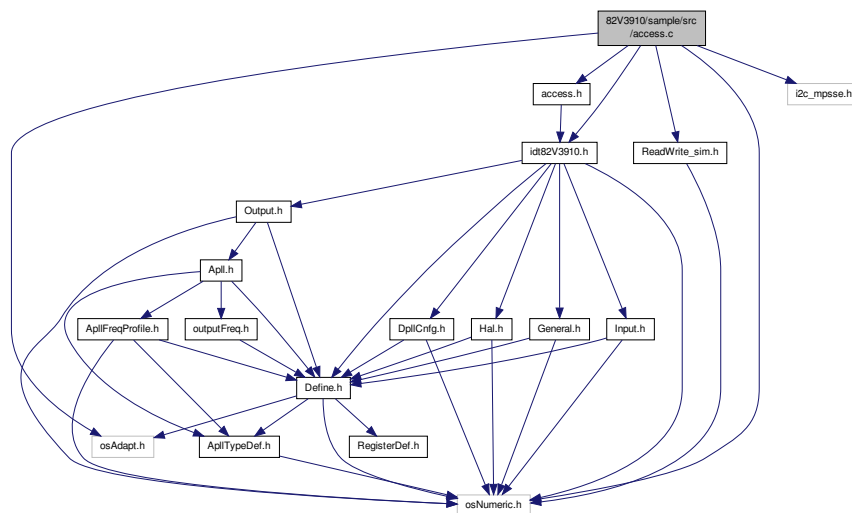
|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 170 of file sampleAppl.c.

## 6.7 82V3910/sample/src/access.c File Reference

```
#include "osNumeric.h"
#include "osAdapt.h"
#include "idt82V3910.h"
#include "access.h"
#include "ReadWrite_sim.h"
#include "i2c_mpsse.h"
```

Include dependency graph for access.c:



#### Functions

- void [Init\\_SimWrapper](#) (void \*usrData)  
Function to wrap both common and APLL HAL initialization functions.
- void [DeInit\\_SimWrapper](#) (void \*usrData)  
Function to wrap both common and APLL HAL deinitialization functions.
- T\_osUInt8 [ReadByte\\_SimWrapper](#) (void \*usrData, T\_osUInt8 devAddr, T\_osUInt8 nAddr)  
Function to wrap both common and APLL HAL read register functions.
- void [WriteByte\\_SimWrapper](#) (void \*usrData, T\_osUInt8 devAddr, T\_osUInt8 nAddr, T\_osUInt8 nData)  
Function to wrap both common and APLL HAL write register functions.
- T\_idtDrvHdlr [IDTSample\\_InitDriverI2c](#) (T\_idtApplXtalList \*xtalList, T\_osUInt8 coreI2CAddr)

*IDTSample\_InitDriverI2c* Utility function to initialize device driver for I2C access.

- T\_idtDrvHdlr [IDTSample\\_InitDriverSimulator](#) (T\_idtApIIXtalList \*xtalList, T\_osUInt8 coreI2CAAddr)

*Utility function to initialize device driver for simulator access.*

- void [IDTSample\\_ResetXtalList](#) (T\_idtApIIXtalList \*xtalList)

*Utility function to reset APLL XTAL list to unconfigured value.*

- void [IDTSample\\_SetLogVerbosity](#) (int level)

*Utility function to change log verbosity level access.*

- int [IDTSample\\_GetLogVerbosity](#) (void)

*Utility function to get current log verbosity level.*

## Variables

- const [T\\_idtDrvAccess I2cAccessMethods](#)
- const [T\\_idtDrvAccess SimAccessMethods](#)

## 6.7.1 Function Documentation

### 6.7.1.1 void Delnit\_SimWrapper ( void \* usrData )

Function to wrap both common and APLL HAL deinitialization functions.

#### Parameters

|                |                                                 |
|----------------|-------------------------------------------------|
| <i>usrData</i> | User data optionally used for de-initialization |
|----------------|-------------------------------------------------|

Definition at line 116 of file access.c.

### 6.7.1.2 int IDTSample\_GetLogVerbosity ( void )

Utility function to get current log verbosity level.

This function is used for TCL extension library

Definition at line 326 of file access.c.

### 6.7.1.3 T\_idtDrvHdlr IDTSample\_InitDriverI2c ( T\_idtApIIXtalList \* xtalList, T\_osUInt8 coreI2CAAddr )

*IDTSample\_InitDriverI2c* Utility function to initialize device driver for I2C access.

This function is used for TCL extension library

#### Parameters

|                     |                    |
|---------------------|--------------------|
| <i>xtalList</i>     | List of XTAL       |
| <i>coreI2CAAddr</i> | Device I2C address |

#### Returns

Device driver handle

Definition at line 161 of file access.c.

### 6.7.1.4 T\_idtDrvHdlr IDTSample\_InitDriverSimulator ( T\_idtApIIXtalList \* xtalList, T\_osUInt8 coreI2CAAddr )

Utility function to initialize device driver for simulator access.

This function is used for TCL extension library

#### Parameters

|                    |                    |
|--------------------|--------------------|
| <i>xtalList</i>    | List of XTAL       |
| <i>coreI2CAddr</i> | Device I2C address |

#### Returns

Device driver handle

Definition at line 229 of file access.c.

#### 6.7.1.5 void IDTSample\_ResetXtalList ( T\_idtAplIXtalList \* *xtalList* )

Utility function to reset APLL XTAL list to unconfigured value.

This function is used for TCL extension library

#### Parameters

|                 |              |
|-----------------|--------------|
| <i>xtalList</i> | List of XTAL |
|-----------------|--------------|

Definition at line 296 of file access.c.

#### 6.7.1.6 void IDTSample\_SetLogVerbosity ( int *level* )

Utility function to change log verbosity level access.

This function is used for TCL extension library

#### Parameters

|              |                     |
|--------------|---------------------|
| <i>level</i> | Log verbosity level |
|--------------|---------------------|

Definition at line 316 of file access.c.

#### 6.7.1.7 void Init\_SimWrapper ( void \* *usrData* )

Function to wrap both common and APLL HAL initialization functions.

#### Parameters

|                |                                               |
|----------------|-----------------------------------------------|
| <i>usrData</i> | User data optionally used for initialization. |
|----------------|-----------------------------------------------|

Definition at line 106 of file access.c.

#### 6.7.1.8 T\_osUInt8 ReadByte\_SimWrapper ( void \* *usrData*, T\_osUInt8 *devAddr*, T\_osUInt8 *nAddr* )

Function to wrap both common and APLL HAL read register functions.

#### Parameters

|                |                                                 |
|----------------|-------------------------------------------------|
| <i>usrData</i> | User data optionally used for reading registers |
| <i>devAddr</i> | Device Address (I2C addr)                       |
| <i>nAddr</i>   | Register offset                                 |

**Returns**

Read register

Definition at line 129 of file access.c.

**6.7.1.9** void WriteByte\_SimWrapper ( void \* *usrData*, T\_osUInt8 *devAddr*, T\_osUInt8 *nAddr*, T\_osUInt8 *nData* )

Function to wrap both common and APLL HAL write register functions.

**Parameters**

|                |                                                 |
|----------------|-------------------------------------------------|
| <i>usrData</i> | User data optionally used for writing registers |
| <i>devAddr</i> | Device Address (I2C addr)                       |
| <i>nAddr</i>   | Register offset                                 |
| <i>nData</i>   | Register value to write                         |

Definition at line 144 of file access.c.

**6.7.2 Variable Documentation**

**6.7.2.1** const T\_idtDrvAccess I2cAccessMethods

**Initial value:**

```
= {
    i2c_init,
    i2c_deinit,
    i2c_read_byte,
    i2c_write_byte,
    NULL
}
```

Definition at line 79 of file access.c.

**6.7.2.2** const T\_idtDrvAccess SimAccessMethods

**Initial value:**

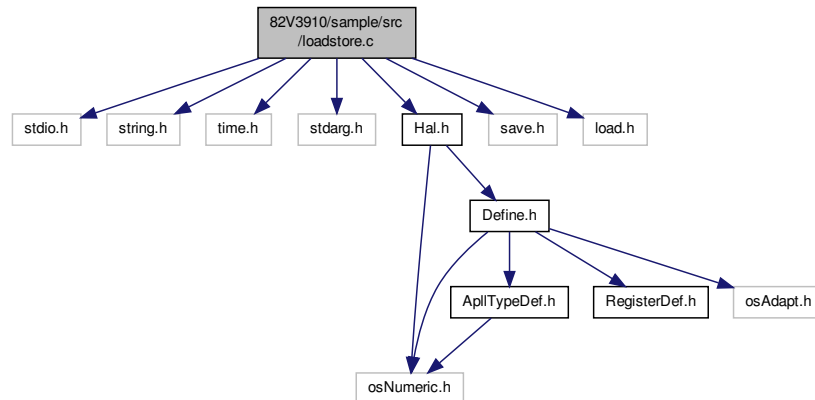
```
= {
    Init_SimWrapper,
    DeInit_SimWrapper,
    ReadByte_SimWrapper,
    WriteByte_SimWrapper,
    NULL
}
```

Definition at line 89 of file access.c.

**6.8 82V3910/sample/src/loadstore.c File Reference**

```
#include <stdio.h>
#include <string.h>
#include <time.h>
#include <stdarg.h>
#include "Hal.h"
#include "save.h"
#include "load.h"
```

Include dependency graph for loadstore.c:



## Functions

- void [IDTSample\\_SaveRgfFile](#) (T\_idtDrvHdlr hdlr, char \*filename)  
Function to output register contents of a device to standard IO or to a file.
- void [IDTSample\\_LoadRgfFile](#) (T\_idtDrvHdlr hdlr, char \*filename)  
Function to read in a file and program device registers.

### 6.8.1 Function Documentation

#### 6.8.1.1 void IDTSample\_LoadRgfFile ( T\_idtDrvHdlr hdlr, char \* filename )

Function to read in a file and program device registers.

##### Parameters

|                 |                                                 |
|-----------------|-------------------------------------------------|
| <i>hdlr</i>     | Device driver handler                           |
| <i>filename</i> | Input file name to load register settings from. |

Definition at line 124 of file loadstore.c.

#### 6.8.1.2 void IDTSample\_SaveRgfFile ( T\_idtDrvHdlr hdlr, char \* filename )

Function to output register contents of a device to standard IO or to a file.

##### Parameters

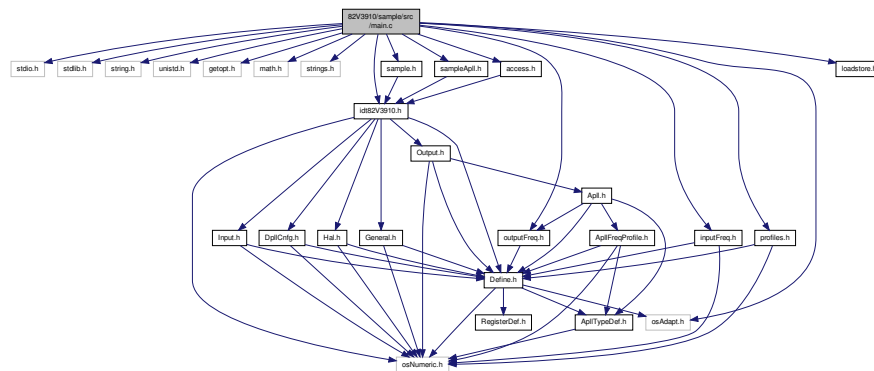
|                 |                                        |
|-----------------|----------------------------------------|
| <i>hdlr</i>     | Device driver handler                  |
| <i>filename</i> | Output file name, NULL for standard IO |

Definition at line 94 of file loadstore.c.

## 6.9 82V3910/sample/src/main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <getopt.h>
#include <math.h>
#include <strings.h>
#include "idt82V3910.h"
#include "osAdapt.h"
#include "sample.h"
#include "sampleAppl.h"
#include "inputFreq.h"
#include "outputFreq.h"
#include "profiles.h"
#include "loadstore.h"
#include "access.h"
```

Include dependency graph for main.c:



## Data Structures

- struct [T\\_SampleConfigEntry](#)  
Sample configuration information.
- struct [globalArgs\\_t](#)  
Structure to define program command line arguments.

## Macros

- #define [MODE\\_None](#) (0)
- #define [MODE\\_Load](#) (1<<0)
- #define [MODE\\_Save](#) (1<<1)
- #define [MODE\\_Input](#) (1<<2)
- #define [MODE\\_Output](#) (1<<3)
- #define [MODE\\_DpiIProfile](#) (1<<4)
- #define [MODE\\_Sample](#) (1<<5)
- #define [MODE\\_Read](#) (1<<6)
- #define [MODE\\_Write](#) (1<<7)
- #define [SAMPLE\\_MAX\\_FILENAME](#) 50
- #define [no\\_argument](#) 0

- #define `required_argument` 1
- #define `optional_argument` 2

## Typedefs

- typedef void(\* `T_DpllProfileFunc` )(T\_idtDrvHdlr hdlr, `T_idtDpllInstance` nDpll)  
*Function typedef for DPLL profile provisioning.*
- typedef void(\* `T_SampleConfigFunc` )(T\_idtDrvHdlr hdlr)  
*Function typedef for sample configurations.*
- typedef void(\* `T_SampleConfigDescrFunc` )(void)  
*Function typedef for sample configuration descriptions.*

## Enumerations

- enum `T_DpllProfile` {  
    `DpllProfile_GR253`, `DpllProfile_GR1244`, `DpllProfile_SMC`, `DpllProfile_G8262Option1`,  
    `DpllProfile_G8262Option2`, `DpllProfile_G813Option1`, `DpllProfile_G813Option2`, `DpllProfile_1PPS`,  
    `DpllProfile_MAX` }  
*List of possible DPLL profiles.*

## Functions

- void `IDTSample_DumpRegistersRgf` (char \*filename)
- void `printHelp` (void)
- int `processCmdLineArguments` (int argc, char \*argv[], `globalArgs_t` \*gblArgs)
- int `main` (int argc, char \*argv[])

## Variables

- `globalArgs_t` `globalArgs`
- int `simDebugFlag`
- const char \* `IDTHIApi_Frequency2String_c` [`OutFreq_MAX`]

### 6.9.1 Macro Definition Documentation

#### 6.9.1.1 #define `MODE_DpllProfile` (1<<4)

Provision DPLL profile mode

Definition at line 111 of file main.c.

#### 6.9.1.2 #define `MODE_Input` (1<<2)

Provision input port mode

Definition at line 105 of file main.c.

#### 6.9.1.3 #define `MODE_Load` (1<<0)

Load from file mode

Definition at line 99 of file main.c.

**6.9.1.4 #define MODE\_None (0)**

No-op mode

Definition at line 96 of file main.c.

**6.9.1.5 #define MODE\_Output (1<<3)**

Provision output port mode

Definition at line 108 of file main.c.

**6.9.1.6 #define MODE\_Read (1<<6)**

Read register mode

Definition at line 117 of file main.c.

**6.9.1.7 #define MODE\_Sample (1<<5)**

Sample configuration mode

Definition at line 114 of file main.c.

**6.9.1.8 #define MODE\_Save (1<<1)**

Save to file mode

Definition at line 102 of file main.c.

**6.9.1.9 #define MODE\_Write (1<<7)**

Read register mode

Definition at line 120 of file main.c.

**6.9.1.10 #define no\_argument 0**

Definition at line 155 of file main.c.

**6.9.1.11 #define optional\_argument 2**

Definition at line 157 of file main.c.

**6.9.1.12 #define required\_argument 1**

Definition at line 156 of file main.c.

**6.9.1.13 #define SAMPLE\_MAX\_FILENAME 50**

Max filename size

Definition at line 123 of file main.c.

## 6.9.2 Typedef Documentation

### 6.9.2.1 typedef void(\* T\_DpllProfileFunc)(T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)

Function typedef for DPLL profile provisioning.

Definition at line 59 of file main.c.

### 6.9.2.2 typedef void(\* T\_SampleConfigDescrFunc)(void)

Function typedef for sample configuration descriptions.

Definition at line 70 of file main.c.

### 6.9.2.3 typedef void(\* T\_SampleConfigFunc)(T\_idtDrvHdlr hdlr)

Function typedef for sample configurations.

Definition at line 65 of file main.c.

## 6.9.3 Enumeration Type Documentation

### 6.9.3.1 enum T\_DpllProfile

List of possible DPLL profiles.

Enumerator

***DpllProfile\_GR253*** GR-253 (SONET) Stratum 3 DPLL Profile

***DpllProfile\_GR1244*** GR-1244 Stratum 3 DPLL Profile

***DpllProfile\_SMC*** SMC DPLL Profile

***DpllProfile\_G8262Option1*** G-8262 Option 1 DPLL Profile

***DpllProfile\_G8262Option2*** G-8262 Option 2 DPLL Profile

***DpllProfile\_G813Option1*** G-813 Option 1 DPLL Profile

***DpllProfile\_G813Option2*** G-813 Option 2 DPLL Profile

***DpllProfile\_1PPS*** 1 PPS DPLL Profile

***DpllProfile\_MAX***

Definition at line 83 of file main.c.

## 6.9.4 Function Documentation

### 6.9.4.1 void IDTSample\_DumpRegistersRgf ( char \* filename )

### 6.9.4.2 int main ( int argc, char \* argv[] )

Definition at line 641 of file main.c.

### 6.9.4.3 void printHelp ( void )

Definition at line 257 of file main.c.

#### 6.9.4.4 `int processCmdLineArguments ( int argc, char * argv[], globalArgs_t * gblArgs )`

Definition at line 441 of file main.c.

### 6.9.5 Variable Documentation

#### 6.9.5.1 `globalArgs_t globalArgs`

Global command line argument variable

Definition at line 167 of file main.c.

#### 6.9.5.2 `const char* IDTHIApi_Frequency2String_c[OutFreq_MAX]`

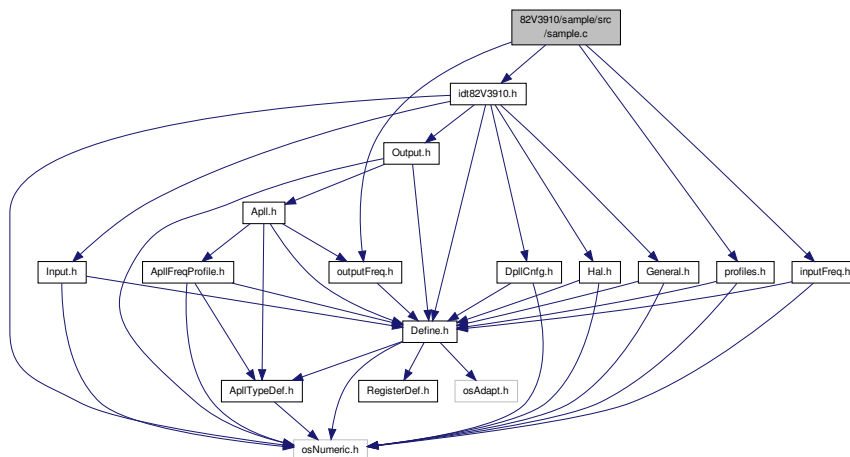
Definition at line 77 of file outputFreqString.c.

#### 6.9.5.3 `int simDebugFlag`

Definition at line 48 of file ReadWrite\_sim.c.

## 6.10 82V3910/sample/src/sample.c File Reference

```
#include "idt82V3910.h"
#include "profiles.h"
#include "inputFreq.h"
#include "outputFreq.h"
Include dependency graph for sample.c:
```



## Functions

- void [IDTSample\\_ConfigureSampleConfig1](#) (T\_idtDrvHdlr hdlr)  
Sample configuration 1.
- void [IDTSample\\_ConfigureSampleConfig2](#) (T\_idtDrvHdlr hdlr)  
Sample configuration 2.

- void [IDTSample\\_ConfigureSampleConfig3](#) (T\_idtDrvHdlr hdlr)  
*Sample configuration 3.*
- void [IDTSample\\_ConfigureSampleConfig4](#) (T\_idtDrvHdlr hdlr)  
*Sample configuration 4.*
- void [IDTSample\\_ConfigureSampleConfig5](#) (T\_idtDrvHdlr hdlr)  
*Sample configuration 5.*
- void [IDTSample\\_Configure1Pps](#) (T\_idtDrvHdlr hdlr)  
*Sample configuration 1PPS.*
- void [IDTSample\\_Configure1PpsHoldover](#) (T\_idtDrvHdlr hdlr)  
*Sample configuration for 1PPS holdover issue.*
- void [IDTSample\\_ConfigureDcoMode](#) (T\_idtDrvHdlr hdlr)  
*Function to configure device for DCO mode.*
- void [IDTSample\\_SetCombinedDcoOffset](#) (T\_idtDrvHdlr hdlr, long pptOffset)  
*Function to set combined DCO offset value. It uses both DCO (holdover) and nominal offsets.*
- long [IDTSample\\_GetCombinedDcoOffset](#) (T\_idtDrvHdlr hdlr)  
*Function to get combined DCO offset value. It uses both DCO (holdover) and nominal offsets.*

## 6.10.1 Function Documentation

### 6.10.1.1 void IDTSample\_Configure1Pps ( T\_idtDrvHdlr hdlr )

Sample configuration 1PPS.

IN3 (1PPS) -> T0 -> OUT3 (1PPS)

#### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 420 of file sample.c.

### 6.10.1.2 void IDTSample\_Configure1PpsHoldover ( T\_idtDrvHdlr hdlr )

Sample configuration for 1PPS holdover issue.

#### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 449 of file sample.c.

### 6.10.1.3 void IDTSample\_ConfigureDcoMode ( T\_idtDrvHdlr hdlr )

Function to configure device for DCO mode.

#### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 480 of file sample.c.

#### 6.10.1.4 void IDTSample\_ConfigureSampleConfig1 ( T\_idtDrvHdlr *hdlr* )

Sample configuration 1.

```
IN3 (25 MHz) -> T0 +-> OUT1 (25 MHz)
               +-> OUT6 (156.25 MHz)
```

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 46 of file sample.c.

#### 6.10.1.5 void IDTSample\_ConfigureSampleConfig2 ( T\_idtDrvHdlr *hdlr* )

Sample configuration 2.

```
IN3 (25 MHz) -> T0 +-> OUT1 (25 MHz)
               +-> OUT3 (8 kHz)
               +-> OUT5 (125 MHz)
```

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 98 of file sample.c.

#### 6.10.1.6 void IDTSample\_ConfigureSampleConfig3 ( T\_idtDrvHdlr *hdlr* )

Sample configuration 3.

```
IN3 (25 MHz) -> T0 +-> OUT1 (25 MHz)
               +-> OUT3 (10 MHz)
               +-> OUT5 (1 PPS)
```

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 154 of file sample.c.

#### 6.10.1.7 void IDTSample\_ConfigureSampleConfig4 ( T\_idtDrvHdlr *hdlr* )

Sample configuration 4.

```
Inputs: IN3 (10MHz), IN4 (1.544MHz), IN5 (25MHz), IN6 (8KHz), IN7 (8KHz),
        IN8 (156.25MHz), IN9 (156.25MHz), IN10 (25MHz), IN11 (25MHz)
Outputs: OUT1 (1.544MHz), OUT3 (10MHz), OUT4 (25MHz), OUT7 (156.25MHz),
        OUT10 (156.25MHz), OUT11 (156.25MHz)
T0 Profile: G.8262 Option 2
T4 Profile: Default
XTAL1/2: 25MHz
```

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 226 of file sample.c.

#### 6.10.1.8 void IDTSample\_ConfigureSampleConfig5 ( T\_idtDrvHdlr *hdlr* )

Sample configuration 5.

```
Carbon copy of IDTSample_ConfigureSampleConfig4,
except there is no input and DPLL is in free-run
mode (default).
```

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 339 of file sample.c.

#### 6.10.1.9 long IDTSample\_GetCombinedDcoOffset ( T\_idtDrvHdlr *hdlr* )

Function to get combined DCO offset value. It uses both DCO (holdover) and nominal offsets.

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

##### Returns

Parts per trillion offset

Definition at line 542 of file sample.c.

#### 6.10.1.10 void IDTSample\_SetCombinedDcoOffset ( T\_idtDrvHdlr *hdlr*, long *pptOffset* )

Function to set combined DCO offset value. It uses both DCO (holdover) and nominal offsets.

##### Parameters

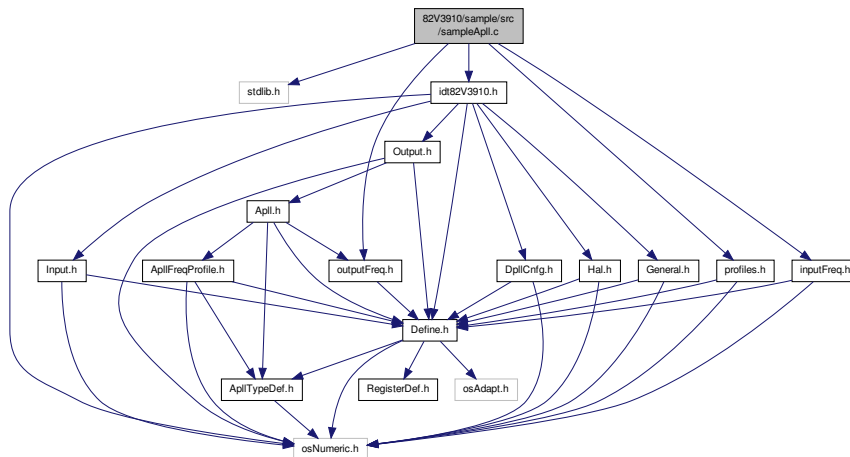
|                  |                           |
|------------------|---------------------------|
| <i>hdlr</i>      | Device driver handler     |
| <i>pptOffset</i> | Parts per trillion offset |

Definition at line 492 of file sample.c.

## 6.11 82V3910/sample/src/sampleAppl.c File Reference

```
#include <stdlib.h>
#include "idt82V3910.h"
#include "profiles.h"
#include "inputFreq.h"
#include "outputFreq.h"
```

Include dependency graph for sampleApl.c:



## Functions

- void [IDTSample\\_ConfigureSampleConfigOutput6\\_156\\_dot\\_25MHz](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for 156.25MHz output at port 6.
- void [IDTSample\\_ConfigureSampleConfigureOutput7\\_77\\_dot\\_76MHz](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for 77.76 MHz output at port 7.
- void [IDTSample\\_ConfigureSampleConfigureOutput10\\_25\\_MHz](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for 25MHz output at port 10.
- void [IDTSample\\_SyncE\\_WAN](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for SyncE WAN application.
- void [IDTSample\\_Sonet](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for SONET application.
- void [IDTSample\\_DualMode](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for dual mode application.
- void [IDTSample\\_SyncE\\_Sonet](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for SyncE SONET application.
- void [IDTSample\\_SyncE\\_LAN](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for SyncE LAN application.
- void [IDTSample\\_GPS\\_1PPS](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for GPS 1PPS application.
- void [IDTSample\\_GPS](#) (T\_idtDrvHdlr hdlr)  
Sample configuration for GPS application.

### 6.11.1 Function Documentation

#### 6.11.1.1 void IDTSample\_ConfigureSampleConfigOutput6\_156\_dot\_25MHz ( T\_idtDrvHdlr hdlr )

Sample configuration for 156.25MHz output at port 6.

```
IN3 (25 MHz) -> T0 +-> OUT6 (Ethernet, 156.25 MHz)
```

#### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 47 of file sampleAppl.c.

#### 6.11.1.2 void IDTSample\_ConfigureSampleConfigureOutput10\_25\_MHz ( T\_idtDrvHdlr *hdlr* )

Sample configuration for 25MHz output at port 10.

```
IN3 (25 MHz) -> T0 +-> OUT10 (Ethernet, 25 MHz)
```

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 125 of file sampleAppl.c.

#### 6.11.1.3 void IDTSample\_ConfigureSampleConfigureOutput7\_77\_dot\_76MHz ( T\_idtDrvHdlr *hdlr* )

Sample configuration for 77.76 MHz output at port 7.

```
IN3 (25 MHz) -> T0 +-> OUT7 (SONET, 77.76 MHz)
```

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 86 of file sampleAppl.c.

#### 6.11.1.4 void IDTSample\_DualMode ( T\_idtDrvHdlr *hdlr* )

Sample configuration for dual mode application.

```
Inputs: IN3 (19.44MHz), IN5 (155.52MHz), IN14 (1.544MHz)
T0 Profile: ST3 SONET
T4 Profile: default
Select XTAL3: done automatically by IDTHlApi_ConfigureOutput function
              based on the output frequency. To manually select XTAL,
              call SetJaXtalId function.
Outputs: OUT1 (19.44MHz), OUT6 (155.52MHz), OUT7 (77.76MHz), OUT9 (1.544MHz)
T0: IN3, IN5, IN14 -> OUT1, OUT6, OUT7 (IN14 is highest priority)
T4: IN3, IN5 -> OUT9
APLL1 XTAL1 at 25MHz & XTAL3 at 24.8832MHz
```

##### Parameters

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 410 of file sampleAppl.c.

#### 6.11.1.5 void IDTSample\_GPS ( T\_idtDrvHdlr *hdlr* )

Sample configuration for GPS application.

```
Inputs: IN3 (10MHz), IN4 (10MHz), EX_SYNC1 (1PPS), EX_SYNC2 (1PPS)
```

```
T0 Profile: BITS / SSU (Wideband) - default
Outputs: OUT1 (25MHz), OUT2 (19.44MHz), OUT3 (10MHz), OUT4 (1PPS),
         OUT6 (25MHz), OUT7 (156.25MHz), OUT9 (1.544MHz), OUT10 (155.52MHz),
         OUT11 (77.76MHz), FRSYNC_8K_1PPS (1PPS), MFRSYNC_2K_1PPS (1PPS)
APLL1 XTAL1 = 25MHz & APLL2 XTAL2 = 24.8832MHz
```

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 1013 of file sampleAppl.c.

**6.11.1.6 void IDTSample\_GPS\_1PPS ( T\_idtDrvHdlr *hdlr* )**

Sample configuration for GPS 1PPS application.

```
Inputs: IN3 (1PPS), IN5 (1PPS)
T0 Profile: GPS (15mHz)
Outputs: OUT1 (25MHz), OUT2 (19.44MHz), OUT3 (10MHz), OUT4 (1PPS),
         OUT6 (25MHz), OUT7 (156.25MHz), OUT9 (2.048MHz),
         OUT10 (155.52MHz), OUT11 (77.76MHz)
APLL1 XTAL1 at 25MHz & APLL2 XTAL2 at 24.8832MHz
```

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 835 of file sampleAppl.c.

**6.11.1.7 void IDTSample\_Sonet ( T\_idtDrvHdlr *hdlr* )**

Sample configuration for SONET application.

```
Inputs: IN3 (19.44MHz), IN5 (155.52MHz), IN14 (1.544MHz)
T0 Profile: ST3 SONET
T4 Profile: default
Outputs: OUT1 (19.44MHz), OUT9 (1.544MHz), OUT10 (155.52MHz)
T0: IN3, IN5, IN14 -> OUT1, OUT10 (IN14 is highest priority)
T4: IN3, IN5 -> OUT9
APLL2 XTAL2 at 24.8832MHz
```

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 294 of file sampleAppl.c.

**6.11.1.8 void IDTSample\_SyncE\_LAN ( T\_idtDrvHdlr *hdlr* )**

Sample configuration for SyncE LAN application.

```
Inputs: IN3 (25MHz), IN14 (2.048MHz)
T0 Profile: G.8262 Option 2
T4 Profile: BITS / SSU (Wideband)
Outputs: OUT1 (25MHz), OUT10 (25.78125), OUT11 (161.1328125MHz), OUT9 (2.048MHz)
T0: IN3, IN14 -> OUT1, OUT10, OUT11 (IN14 is highest priority)
T4: IN3 -> OUT9
APLL2 XTAL4 at 25.78125MHz
```

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 720 of file sampleApll.c.

**6.11.1.9 void IDTSample\_SyncE\_Sonet ( T\_idtDrvHdlr *hdlr* )**

Sample configuration for SyncE SONET application.

```
Inputs: IN3 (25MHz) , IN5 (156.25MHz), IN6 (155.52MHz), IN7 (19.44MHz), IN14 (1.544MHz)
T0 Profile: G.8262 Option 2
T4 Profile: BITS / SSU (Wideband) - default
Outputs: OUT1 (25MHz), OUT2 (19.44MHz), OUT6 (25MHz), OUT7 (156.25MHz),
         OUT9 (1.544MHz), OUT10 (155.52MHz), OUT11 (77.76MHz)
T0: IN3, IN5, IN6, IN7, IN14 -> OUT1, OUT2, OUT6, OUT7, OUT10, OUT11 (IN14 is highest priority)
T4: IN3, IN5, IN6, IN7 -> OUT9
APLL1 XTAL1 at 25MHz & APLL2 XTAL2 at 24.8832MHz
```

**Parameters**

|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 534 of file sampleApll.c.

**6.11.1.10 void IDTSample\_SyncE\_WAN ( T\_idtDrvHdlr *hdlr* )**

Sample configuration for SyncE WAN application.

```
Inputs: IN3 (25MHz), IN5 (156.25MHz), IN14 (1.544MHz)
T0 Profile: G.8262 Option 2
T4 Profile: Default
Outputs: OUT1 (25MHz), OUT6 (25MHz), OUT7 (156.25MHz), OUT9 (1.544MHz)
T0: IN3, IN5, IN14 -> OUT1, OUT10, OUT11 (IN14 has highest priority)
T4: IN3, IN5 -> OUT9
APLL1 XTAL1 at 25MHz
```

**Parameters**

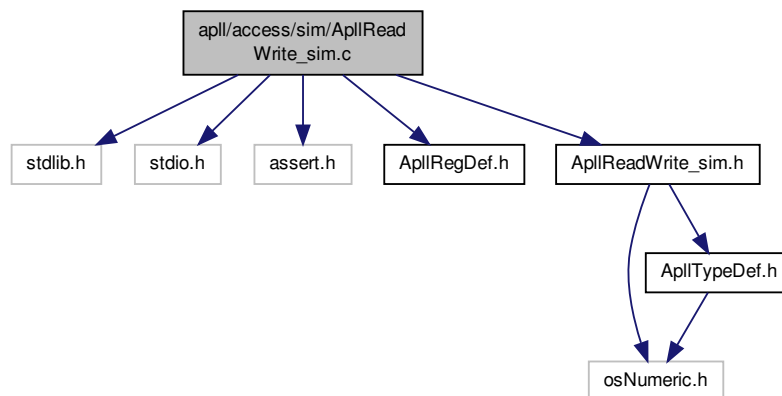
|             |                       |
|-------------|-----------------------|
| <i>hdlr</i> | Device driver handler |
|-------------|-----------------------|

Definition at line 170 of file sampleApll.c.

**6.12 apll/access/sim/ApllReadWrite\_sim.c File Reference**

```
#include <stdlib.h>
#include <stdio.h>
#include <assert.h>
#include "ApllRegDef.h"
#include "ApllReadWrite_sim.h"
```

Include dependency graph for `ApIIRedWrite_sim.c`:



## Macros

- `#define SIM_REG_MAP_SIZE (0x11)`

## Functions

- `void ApIIRedWrite_Sim (void *usrData)`
- `void ApIIRedWrite_Sim (void *usrData)`
- `T_osUInt8 ApIIRedWrite_Sim (void *usrData, T_osUInt8 deviceAddr, T_idtApIIRegAddr nAddr)`
- `void ApIIRedWrite_Sim (void *usrData, T_osUInt8 deviceAddr, T_idtApIIRegAddr nAddr, T_idtApIIRegVal data)`

## Variables

- `T_idtApIIRegVal simulatedRegMapApIIR0 [SIM_REG_MAP_SIZE]`
- `T_idtApIIRegVal simulatedRegMapApIIR1 [SIM_REG_MAP_SIZE]`

### 6.12.1 Macro Definition Documentation

#### 6.12.1.1 `#define SIM_REG_MAP_SIZE (0x11)`

Definition at line 41 of file `ApIIRedWrite_sim.c`.

### 6.12.2 Function Documentation

#### 6.12.2.1 `void ApIIRedWrite_Sim ( void * usrData )`

Definition at line 62 of file `ApIIRedWrite_sim.c`.

#### 6.12.2.2 `void ApIIRedWrite_Sim ( void * usrData )`

Definition at line 57 of file `ApIIRedWrite_sim.c`.

### 6.12.2.3 T\_osUInt8 ApllRead\_Sim ( void \* *usrData*, T\_osUInt8 *deviceAddr*, T\_idtApllRegAddr *nAddr* )

Definition at line 67 of file ApllReadWrite\_sim.c.

### 6.12.2.4 void ApllWrite\_Sim ( void \* *usrData*, T\_osUInt8 *deviceAddr*, T\_idtApllRegAddr *nAddr*, T\_idtApllRegVal *data* )

Definition at line 82 of file ApllReadWrite\_sim.c.

## 6.12.3 Variable Documentation

### 6.12.3.1 T\_idtApllRegVal simulatedRegMapApll0[SIM\_REG\_MAP\_SIZE]

**Initial value:**

```
= {
    0x00, 0x01, 0x0c, 0x35, 0x0c,
    0x35, 0x0c, 0x35, 0x0c, 0x35,
    0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00
}
```

Definition at line 43 of file ApllReadWrite\_sim.c.

### 6.12.3.2 T\_idtApllRegVal simulatedRegMapApll1[SIM\_REG\_MAP\_SIZE]

**Initial value:**

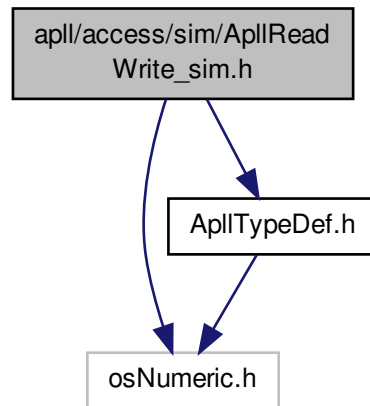
```
= {
    0x00, 0x01, 0x0c, 0x35, 0x0c,
    0x35, 0x0c, 0x35, 0x0c, 0x35,
    0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00
}
```

Definition at line 50 of file ApllReadWrite\_sim.c.

## 6.13 apll/access/sim/ApllReadWrite\_sim.h File Reference

```
#include "osNumeric.h"
#include "ApllTypeDef.h"
```

Include dependency graph for `ApllReadWrite_sim.h`:



## Functions

- `T_osUInt8 ApIlRead_Sim` (void \*usrData, T\_osUInt8 deviceAddr, T\_idtApIlRegAddr nAddr)
- void `ApIlWrite_Sim` (void \*usrData, T\_osUInt8 deviceAddr, T\_idtApIlRegAddr nAddr, T\_idtApIlRegVal data)
- void `ApIlInit_Sim` (void \*userData)
- void `ApIlDeInit_Sim` (void \*userData)

### 6.13.1 Function Documentation

#### 6.13.1.1 void ApIlDeInit\_Sim ( void \* userData )

Definition at line 62 of file `ApIlReadWrite_sim.c`.

#### 6.13.1.2 void ApIlInit\_Sim ( void \* userData )

Definition at line 57 of file `ApIlReadWrite_sim.c`.

#### 6.13.1.3 T\_osUInt8 ApIlRead\_Sim ( void \* usrData, T\_osUInt8 deviceAddr, T\_idtApIlRegAddr nAddr )

Definition at line 67 of file `ApIlReadWrite_sim.c`.

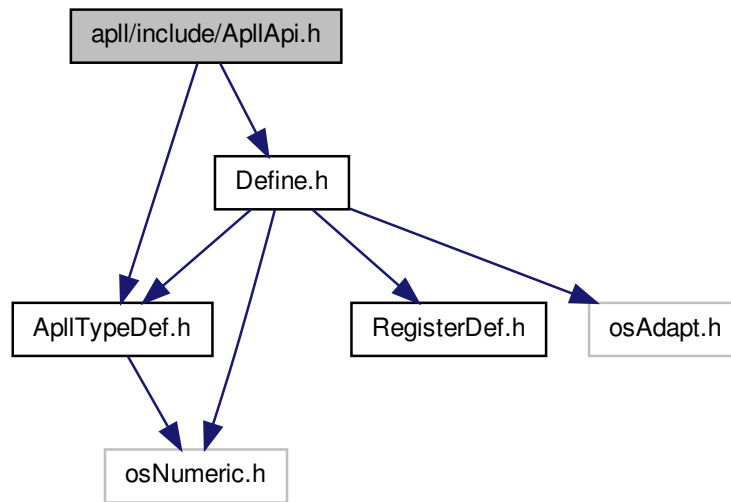
#### 6.13.1.4 void ApIlWrite\_Sim ( void \* usrData, T\_osUInt8 deviceAddr, T\_idtApIlRegAddr nAddr, T\_idtApIlRegVal data )

Definition at line 82 of file `ApIlReadWrite_sim.c`.

## 6.14 apIl/include/ApIlApi.h File Reference

```
#include "Define.h"
#include "ApIlTypeDef.h"
```

Include dependency graph for ApllApi.h:



## Functions

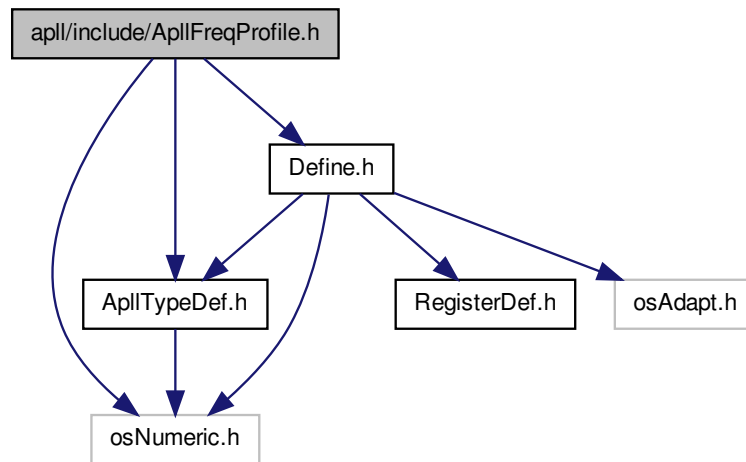
- [T\\_idtApllConfigSel](#) [IDTApll\\_GetConfigSel](#) (T\_idtDrvHdlr hdlr, [T\\_idtApllId](#) appllInstId)  
Function to retrieve APLL configuration.
- void [IDTApll\\_SetConfigSel](#) (T\_idtDrvHdlr hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllConfigSel](#) appllConfig)  
Function to provision APLL based on input configuration.
- [T\\_idtApllInputClkSrc](#) [IDTApll\\_GetInputClkSrc](#) (T\_idtDrvHdlr hdlr, [T\\_idtApllId](#) appllInstId)  
Function to retrieve selected APLL input frequency source (external/internal)
- void [IDTApll\\_SetInputClkSrc](#) (T\_idtDrvHdlr hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllInputClkSrc](#) inputClk)  
Function to provision APLL input frequency source (external/internal)
- [T\\_idtApllDividerValue](#) [IDTApll\\_GetPreDiv](#) (T\_idtDrvHdlr hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllConfigSel](#) appllConfig)  
Function to retrieve pre- (input) divider value.
- void [IDTApll\\_GetPreDivRange](#) (T\_idtDrvHdlr hdlr, [T\\_idtApllDividerValue](#) \*minDiv, [T\\_idtApllDividerValue](#) \*maxDiv)  
Function to retrieve pre- (input) divisor range.
- void [IDTApll\\_SetPreDiv](#) (T\_idtDrvHdlr hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllConfigSel](#) appllConfig, [T\\_idtApllDividerValue](#) div)  
Function to provision pre- (input) divider value.
- [T\\_idtApllDividerValue](#) [IDTApll\\_GetFeedbackDiv](#) (T\_idtDrvHdlr hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllConfigSel](#) appllConfig)  
Function to retrieve feedback (oscillator) divider value.
- void [IDTApll\\_GetFeedbackDivRange](#) (T\_idtDrvHdlr hdlr, [T\\_idtApllDividerValue](#) \*minDiv, [T\\_idtApllDividerValue](#) \*maxDiv)  
Function to retrieve feedback (oscillator) divisor range.
- void [IDTApll\\_SetFeedbackDiv](#) (T\_idtDrvHdlr hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllConfigSel](#) appllConfig, [T\\_idtApllDividerValue](#) div)  
Function to provision feedback (oscillator) divider value.

- [T\\_idtApplQaDiv IDTApl\\_GetQaDiv](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId, [T\\_idtApplConfigSel](#) apIConfig)  
*Function to retrieve output port A divisor.*
- void [IDTApl\\_SetQaDiv](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId, [T\\_idtApplConfigSel](#) apIConfig, [T\\_idtApplQaDiv](#) div)  
*Function to provision output port A divisor.*
- [T\\_idtApplQaDiv IDTApl\\_GetQbDiv](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId, [T\\_idtApplConfigSel](#) apIConfig)  
*Function to retrieve output port B divisor.*
- void [IDTApl\\_SetQbDiv](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId, [T\\_idtApplConfigSel](#) apIConfig, [T\\_idtApplQbDiv](#) div)  
*Function to provision output port B divisor.*
- [T\\_idtApplFreqDoublerSel IDTApl\\_GetFreqDoublerSel](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId)  
*Function to retrieve whether feedback frequency doubling is enabled.*
- void [IDTApl\\_SetFreqDoublerSel](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId, [T\\_idtApplFreqDoublerSel](#) doubler)  
*Function to enable/disable feedback frequency doubling.*
- [T\\_idtApplXtalId IDTApl\\_GetXtalId](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId)  
*Function to retrieve oscillator currently used.*
- void [IDTApl\\_SetXtalId](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId, [T\\_idtApplXtalId](#) xtal)  
*Function to select oscillator.*
- [T\\_idtApplOutputSigType IDTApl\\_GetOutputSigType](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId, [T\\_idtApplOutput](#) output)  
*Function to retrieve output port signal type.*
- void [IDTApl\\_SetOutputSigType](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId, [T\\_idtApplOutput](#) output, [T\\_idtApplOutputSigType](#) outputSigType)  
*Function to provision output port signal type.*
- [T\\_idtApplOutputEn IDTApl\\_GetOutputEnState](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId, [T\\_idtApplOutput](#) output)  
*Function to retrieve output port status (enable/disable)*
- void [IDTApl\\_SetOutputEnState](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId, [T\\_idtApplOutput](#) output, [T\\_idtApplOutputEn](#) enable)  
*Function to control (enable/disable) output port.*
- [T\\_idtApplMasterResetSync IDTApl\\_GetMasterResetSync](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId)  
*Function to retrieve reset status of device.*
- void [IDTApl\\_SetMasterResetSync](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId, [T\\_idtApplMasterResetSync](#) reset)  
*Function to reset APLL.*
- [T\\_idtApplBypass IDTApl\\_GetBypass](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId)  
*Function to retrieve bypass status of device.*
- void [IDTApl\\_SetBypass](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId, [T\\_idtApplBypass](#) bypass)  
*Function to set APLL bypass setting.*
- [T\\_idtApplShutoff IDTApl\\_GetShutoff](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId)  
*Function to retrieve shutoff status of device.*
- void [IDTApl\\_SetShutoff](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApplId](#) apIInstId, [T\\_idtApplShutoff](#) shutoff)  
*Function to set APLL shutoff setting.*

## 6.15 apI/include/ApIFreqProfile.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "ApITypeDef.h"
```

Include dependency graph for ApllFreqProfile.h:



## Data Structures

- struct [T\\_idtApllFreqMapping](#)  
Type definition for mapping output frequency to APLL configuration.

## Enumerations

- enum [T\\_idtApllInputFreqType](#) {  
APLL\_INPUT\_FREQ\_19440KHz, APLL\_INPUT\_FREQ\_25000KHz, APLL\_INPUT\_FREQ\_77760KHz, APLL\_INPUT\_FREQ\_125000KHz,  
APLL\_INPUT\_FREQ\_155520KHz, APLL\_INPUT\_FREQ\_25781\_DOT\_25KHz, APLL\_INPUT\_FREQ\_312500KHz, APLL\_INPUT\_FREQ\_128906\_DOT\_25KHz,  
APLL\_INPUT\_FREQ\_156250KHz, APLL\_INPUT\_FREQ\_8KHz, APLL\_INPUT\_FREQ\_MAX }  
Enumerated type listing supported input frequencies.
- enum [T\\_idtApllOutputFreqType](#) {  
APLL\_OUTPUT\_FREQ\_24883\_DOT\_2KHz, APLL\_OUTPUT\_FREQ\_25000KHz, APLL\_OUTPUT\_FREQ\_25781\_DOT\_25KHz, APLL\_OUTPUT\_FREQ\_77760KHz,  
APLL\_OUTPUT\_FREQ\_78125KHz, APLL\_OUTPUT\_FREQ\_80566\_DOT\_40625KHz, APLL\_OUTPUT\_FREQ\_124416KHz, APLL\_OUTPUT\_FREQ\_125000KHz,  
APLL\_OUTPUT\_FREQ\_128906\_DOT\_25KHz, APLL\_OUTPUT\_FREQ\_155520KHz, APLL\_OUTPUT\_FREQ\_156250KHz, APLL\_OUTPUT\_FREQ\_161132\_DOT\_8125KHz,  
APLL\_OUTPUT\_FREQ\_311040KHz, APLL\_OUTPUT\_FREQ\_312500KHz, APLL\_OUTPUT\_FREQ\_322265\_DOT\_625KHz, APLL\_OUTPUT\_FREQ\_622080KHz,  
APLL\_OUTPUT\_FREQ\_625000KHz, APLL\_OUTPUT\_FREQ\_644531\_DOT\_25KHz, APLL\_OUTPUT\_FREQ\_MAX }  
Enumerated type listing supported output frequencies.

## Functions

- const [T\\_idtApllFreqMapping](#) \* [IDTApll\\_GetConfigListByFreq](#) ([T\\_idtApllOutputFreqType](#) apllOutputFreq)  
Function to retrieve output configuration given an output frequency.

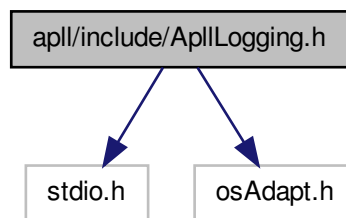
- const `T_idtApIIFreqMapping` \* `IDTAplI_GetConfigListByXtalType` (`T_idtApIIXtalType` `apIIXtalFreq`)  
Function to retrieve output configuration given the oscillator type (frequency)
- `T_osBool` `IDTAplI_NullFreqTableEntry` (const `T_idtApIIFreqMapping` \*`apIIFreqMapping`)  
Function to check whether the input APLL frequency mapping is NULL (not configured)
- const `T_idtApIIFreqMapping` \* `IDTAplI_GetCurrentApIIFreqConfig` (`T_idtApIIId` `apIIId`, `T_idtApIIOutput` `output`)  
Function to retrieve the current APLL frequency mapping configuration for the output.

## 6.16 apI//include/ApIILogging.h File Reference

```
#include <stdio.h>
```

```
#include "osAdapt.h"
```

Include dependency graph for ApIILogging.h:



### Macros

- `#define` `IDT_APLL_API_LOG_HEADER`
- `#define` `IDT_APLL_API_LOG_ERR`(fmt)
- `#define` `IDT_APLL_API_LOG_INFO`(fmt)
- `#define` `IDT_APLL_API_LOG_DEBUG`(fmt)
- `#define` `IDT_APLL_API_TRACE`(fmt) `OS_TRACE`(fmt)

## 6.17 apI//include/ApIIRegDef.h File Reference

### Enumerations

- enum {  
`REG_CONTROL` = 0x00, `REG_CLK_SEL` = 0x01, `REG_PDSEL0_LSB` = 0x02, `REG_PDSEL0_MSB` = 0x03,  
`REG_PDSEL1_LSB` = 0x04, `REG_PDSEL1_MSB` = 0x05, `REG_M0_LSB` = 0x06, `REG_M0_MSB` = 0x07,  
`REG_M1_LSB` = 0x08, `REG_M1_MSB` = 0x09, `REG_ODBSELA0` = 0x0A, `REG_ODBSELA1` = 0x0B,  
`REG_ODBSELB0` = 0x0C, `REG_ODBSELB1` = 0x0D, `REG_VCXO_SEL` = 0x0E, `REG_CONFIG_QA` = 0x0F,  
`REG_CONFIG_QB` = 0x10, `BF_CONFIG_SIZE` = 1, `BF_CONFIG_MASK` = 0x02, `BF_CONFIG_LSB` = 1,  
`BF_CLK_SEL_SIZE` = 1, `BF_CLK_SEL_MASK` = 0x01, `BF_CLK_SEL_LSB` = 0, `BF_PDSEL0_LSB_SIZE` =  
8,  
`BF_PDSEL0_LSB_MASK` = 0xFF, `BF_PDSEL0_LSB_LSB` = 0, `BF_PDSEL0_MSB_SIZE` = 7, `BF_PDSEL0-`  
`_MSB_MASK` = 0x7F,  
`BF_PDSEL0_MSB_LSB` = 0, `BF_PDSEL1_LSB_SIZE` = 8, `BF_PDSEL1_LSB_MASK` = 0xFF, `BF_PDSEL1-`

```

_LSB_LSB = 0,
BF_PDSEL1_MSB_SIZE = 7, BF_PDSEL1_MSB_MASK = 0x7F, BF_PDSEL1_MSB_LSB = 0, BF_M0_LSB_SIZE = 8,
BF_M0_LSB_MASK = 0xFF, BF_M0_LSB_LSB = 0, BF_M0_MSB_SIZE = 7, BF_M0_MSB_MASK = 0x7F,
BF_M0_MSB_LSB = 0, BF_M1_LSB_SIZE = 8, BF_M1_LSB_MASK = 0xFF, BF_M1_LSB_LSB = 0,
BF_M1_MSB_SIZE = 7, BF_M1_MSB_MASK = 0x7F, BF_M1_MSB_LSB = 0, BF_ODBSELA0_SIZE = 3,
BF_ODBSELA0_MASK = 0x07, BF_ODBSELA0_LSB = 0, BF_ODBSELA1_SIZE = 3, BF_ODBSELA1_MASK = 0x07,
BF_ODBSELA1_LSB = 0, BF_ODBSELB0_SIZE = 3, BF_ODBSELB0_MASK = 0x07, BF_ODBSELB0_LSB = 0,
BF_ODBSELB1_SIZE = 3, BF_ODBSELB1_MASK = 0x07, BF_ODBSELB1_LSB = 0, BF_SEL_DOUBLE_SIZE = 1,
BF_SEL_DOUBLE_MASK = 0x80, BF_SEL_DOUBLE_LSB = 7, BF_MR_SYNC_SIZE = 1, BF_MR_SYNC_MASK = 0x40,
BF_MR_SYNC_LSB = 6, BF_BYPASS_SIZE = 1, BF_BYPASS_MASK = 0x20, BF_BYPASS_LSB = 5,
BF_SHUTOFF_SIZE = 1, BF_SHUTOFF_MASK = 0x10, BF_SHUTOFF_LSB = 4, BF_VC XO_SEL_SIZE = 1,
BF_VC XO_SEL_MASK = 0x01, BF_VC XO_SEL_LSB = 0, BF_LVDS_QA_SIZE = 1, BF_LVDS_QA_MASK = 0x02,
BF_LVDS_QA_LSB = 1, BF_OE_A_SIZE = 1, BF_OE_A_MASK = 0x01, BF_OE_A_LSB = 0,
BF_LVDS_QB_SIZE = 1, BF_LVDS_QB_MASK = 0x02, BF_LVDS_QB_LSB = 1, BF_OE_B_SIZE = 1,
BF_OE_B_MASK = 0x01, BF_OE_B_LSB = 0, REGMAP_APLL_MAX }

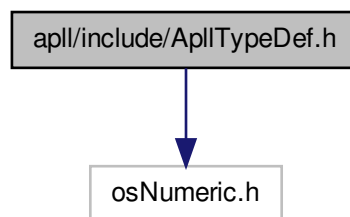
```

*Enumerated type listing register offsets and bit field constants.*

## 6.18 apll/include/ApllTypeDef.h File Reference

```
#include "osNumeric.h"
```

Include dependency graph for ApllTypeDef.h:



### Data Structures

- struct [T\\_idtApllXtal](#)

*Structure containing APLL crystal id and frequency information.*

### Macros

- #define [INVALID\\_DIVIDER\\_VALUE](#) (0xFFFF)

*Define constant indicating invalid divider value.*

## Typedefs

- typedef T\_osUInt16 [T\\_idtApllDividerValue](#)  
*Type define for APLL divider value.*
- typedef T\_osUInt8 [T\\_idtApllRegAddr](#)  
*Type define for APLL register address.*
- typedef T\_osUInt8 [T\\_idtApllRegMask](#)  
*Type define for APLL register value mask.*
- typedef T\_osUInt8 [T\\_idtApllRegVal](#)  
*Type define for APLL register value.*

## Enumerations

- enum [T\\_idtApllId](#) { [APLL\\_INST\\_1](#), [APLL\\_INST\\_2](#), [APLL\\_INST\\_MAX](#) }  
*Enumerated type listing APLL instance within device.*
- enum [T\\_idtApllAccessType](#) { [APLL\\_ACCESS\\_I2C](#), [APLL\\_ACCESS\\_SIM](#), [APLL\\_ACCESS\\_MAX](#) }  
*Enumerated type listing supported access methods.*
- enum [T\\_idtApllConfigSel](#) { [APLL\\_CONFIG0](#) = 0, [APLL\\_CONFIG1](#) = 1, [APLL\\_CONFIG\\_MAX](#) }  
*Enumerated type listing possible configuration for each APLL.*
- enum [T\\_idtApllInputClkSrc](#) { [APLL\\_CLK\\_SEL\\_EXTERNAL](#), [APLL\\_CLK\\_SEL\\_INTERNAL](#), [APLL\\_CLK\\_SEL\\_MAX](#) }  
*Enumerated type listing input ports.*
- enum [T\\_idtApllOutput](#) { [APLL\\_OUTPUT\\_QA](#), [APLL\\_OUTPUT\\_QB](#), [APLL\\_OUTPUT\\_MAX](#) }  
*Enumerated type listing output ports.*
- enum [T\\_idtApllOutputEn](#) { [APLL\\_OUTPUT\\_DISABLE](#), [APLL\\_OUTPUT\\_ENABLE](#), [APLL\\_OUTPUT\\_EN\\_MAX](#) }  
*Enumerated type listing output port status (enable/disable)*
- enum [T\\_idtApllOutputSigType](#) { [APLL\\_OUT\\_SIG\\_LVPECL](#), [APLL\\_OUT\\_SIG\\_LVDS](#), [APLL\\_OUT\\_SIG\\_MAX](#) }  
*Enumerated type listing output signal type.*
- enum [T\\_idtApllOutputSupportedType](#) { [APLL\\_OUT\\_QA\\_ONLY](#), [APLL\\_OUT\\_QB\\_ONLY](#), [APLL\\_OUT\\_QA\\_OR\\_QB](#), [APLL\\_OUT\\_MAX](#) }  
*Enumerated type listing frequency supported by the output.*
- enum [T\\_idtApllQaDiv](#) {  
[APLL\\_QA\\_ODSEL\\_DIV\\_25](#) = 0, [APLL\\_QA\\_ODSEL\\_DIV\\_5](#) = 1, [APLL\\_QA\\_ODSEL\\_DIV\\_4](#) = 2, [APLL\\_QA\\_ODSEL\\_DIV\\_2](#) = 3,  
[APLL\\_QA\\_ODSEL\\_DIV\\_1](#) = 4, [APLL\\_QA\\_ODSEL\\_DIV\\_MAX](#) }  
*Enumerated type listing output port A divider values.*
- enum [T\\_idtApllQbDiv](#) {  
[APLL\\_QB\\_ODSEL\\_DIV\\_8](#) = 0, [APLL\\_QB\\_ODSEL\\_DIV\\_5](#) = 1, [APLL\\_QB\\_ODSEL\\_DIV\\_4](#) = 2, [APLL\\_QB\\_ODSEL\\_DIV\\_2](#) = 3,  
[APLL\\_QB\\_ODSEL\\_DIV\\_1](#) = 4, [APLL\\_QB\\_ODSEL\\_DIV\\_MAX](#) }  
*Enumerated type listing output port B divider values.*
- enum [T\\_idtApllFreqDoublerSel](#) { [APLL\\_FREQ\\_SEL\\_SINGLE](#) = 0, [APLL\\_FREQ\\_SEL\\_DOUBLE](#) = 1, [APLL\\_FREQ\\_SEL\\_MAX](#) }  
*Enumerated type listing feedback frequency doubling values.*
- enum [T\\_idtApllXtalId](#) {  
[APLL\\_XTAL\\_1\\_APLL1](#), [APLL\\_XTAL\\_2\\_APLL2](#), [APLL\\_XTAL\\_3\\_APLL1](#), [APLL\\_XTAL\\_4\\_APLL2](#),  
[APLL\\_XTAL\\_MAX](#) }  
*Enumerated type listing crystals.*
- enum [T\\_idtApllXtalType](#) { [APLL\\_XTAL\\_FREQ\\_25000KHz](#), [APLL\\_XTAL\\_FREQ\\_25781\\_DOT\\_25KHz](#), [APLL\\_XTAL\\_FREQ\\_24883\\_DOT\\_2KHz](#), [APLL\\_XTAL\\_FREQ\\_MAX](#) }

Enumerated type listing support crystal frequencies.

- enum `T_idtApllMasterResetSync` { `APLL_OUT_MR_SYNC` = 0, `APLL_IN_MR_SYNC` = 1, `APLL_MR_SYNC_MAX` }

Enumerated type listing Device master reset sync status.

- enum `T_idtApllBypass` { `APLL_BYPASS_NORMAL_OP` = 0, `APLL_BYPASS_BYPASS` = 1, `APLL_BYPASS_MAX` }

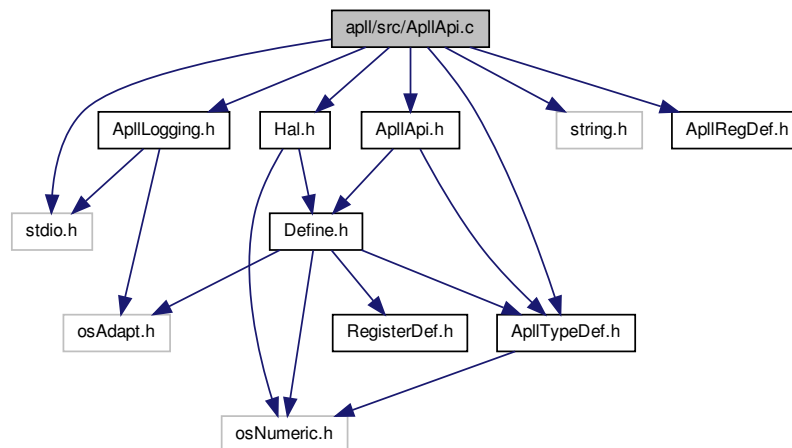
Enumerated type listing Apll bypass status/setting.

- enum `T_idtApllShutoff` { `APLL_SHUTOFF_NORMAL_OP` = 0, `APLL_SHUTOFF_SHUTOFF` = 1, `APLL_SHUTOFF_MAX` }

Enumerated type listing Apll shutoff status/setting.

## 6.19 apll/src/ApllApi.c File Reference

```
#include <stdio.h>
#include <string.h>
#include "Hal.h"
#include "ApllRegDef.h"
#include "ApllTypeDef.h"
#include "ApllApi.h"
#include "ApllLogging.h"
Include dependency graph for ApllApi.c:
```



## Macros

- #define `APLL_DIVIDER_LSB(val)` `((val)&0x00FF)`
- #define `APLL_DIVIDER_MSB(val)` `((val)&0x7F00)>>8)`
- #define `APLL_DIVIDER_VAL(lsb, msb)` `((msb)<<8|(lsb))`
- #define `APLL_PRE_DIV_LSB(val)` `APLL_DIVIDER_LSB(val)`
- #define `APLL_PRE_DIV_MSB(val)` `APLL_DIVIDER_MSB(val)`
- #define `APLL_PRE_DIV_VAL(lsb, msb)` `APLL_DIVIDER_VAL(lsb, msb)`
- #define `APLL_FB_DIV_LSB(val)` `APLL_DIVIDER_LSB(val)`
- #define `APLL_FB_DIV_MSB(val)` `APLL_DIVIDER_MSB(val)`
- #define `APLL_FB_DIV_VAL(lsb, msb)` `APLL_DIVIDER_VAL(lsb, msb)`
- #define `APLL_PRE_DIV_MIN` `(0x0004)`

- #define `APLL_PRE_DIV_MAX` (0x7FFF)
- #define `APLL_FB_DIV_MIN` (0x0005)
- #define `APLL_FB_DIV_MAX` (0x7FFF)
- #define `APLL1_REG_HIGH_BIT` (0)
- #define `APLL2_REG_HIGH_BIT` (1)
- #define `APLL_REG_HIGH_BIT_OFFSET` (7)
- #define `APLL_REG_OFFSET`(apllInstId, base)  
Utility macro to translate APLL register address.
- #define `APLL1_REG_VALID_OFFSET`(offset) ((offset >= `REG_CONTROL`) && (offset <= `REG_CONFIG_QB`))  
Utility macro to sanity check APLL #1 register address.
- #define `APLL2_REG_VALID_OFFSET`(offset)  
Utility macro to translate APLL #2 register address.
- #define `INVALID_XTAL_SEL_VALUE` (0xFF)

## Functions

- `T_idtApllConfigSel IDTApll_GetConfigSel` (T\_idtDrvHdlr hdlr, T\_idtApllId apllInstId)  
Function to retrieve APLL configuration.
- void `IDTApll_SetConfigSel` (T\_idtDrvHdlr hdlr, T\_idtApllId apllInstId, T\_idtApllConfigSel apllConfig)  
Function to provision APLL based on input configuration.
- `T_idtApllInputClkSrc IDTApll_GetInputClkSrc` (T\_idtDrvHdlr hdlr, T\_idtApllId apllInstId)  
Function to retrieve selected APLL input frequency source (external/internal)
- void `IDTApll_SetInputClkSrc` (T\_idtDrvHdlr hdlr, T\_idtApllId apllInstId, T\_idtApllInputClkSrc inputClk)  
Function to provision APLL input frequency source (external/internal)
- `T_idtApllDividerValue IDTApll_GetPreDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apllInstId, T\_idtApllConfigSel apllConfig)  
Function to retrieve pre- (input) divider value.
- void `IDTApll_GetPreDivRange` (T\_idtDrvHdlr hdlr, T\_idtApllDividerValue \*minDiv, T\_idtApllDividerValue \*maxDiv)  
Function to retrieve pre- (input) divisor range.
- void `IDTApll_SetPreDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apllInstId, T\_idtApllConfigSel apllConfig, T\_idtApllDividerValue div)  
Function to provision pre- (input) divider value.
- void `IDTApll_GetFeedbackDivRange` (T\_idtDrvHdlr hdlr, T\_idtApllDividerValue \*minDiv, T\_idtApllDividerValue \*maxDiv)  
Function to retrieve feedback (oscillator) divisor range.
- `T_idtApllDividerValue IDTApll_GetFeedbackDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apllInstId, T\_idtApllConfigSel apllConfig)  
Function to retrieve feedback (oscillator) divider value.
- void `IDTApll_SetFeedbackDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apllInstId, T\_idtApllConfigSel apllConfig, T\_idtApllDividerValue div)  
Function to provision feedback (oscillator) divider value.
- `T_idtApllQaDiv IDTApll_GetQaDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apllInstId, T\_idtApllConfigSel config)  
Function to retrieve output port A divisor.
- void `IDTApll_SetQaDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apllInstId, T\_idtApllConfigSel config, T\_idtApllQaDiv div)  
Function to provision output port A divisor.
- `T_idtApllQaDiv IDTApll_GetQbDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apllInstId, T\_idtApllConfigSel config)  
Function to retrieve output port B divisor.
- void `IDTApll_SetQbDiv` (T\_idtDrvHdlr hdlr, T\_idtApllId apllInstId, T\_idtApllConfigSel config, T\_idtApllQbDiv div)

- Function to provision output port B divisor.*
- [T\\_idtApllFreqDoublerSel](#) [IDTApll\\_GetFreqDoublerSel](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApllId](#) appllInstId)
- Function to retrieve whether feedback frequency doubling is enabled.*
- void [IDTApll\\_SetFreqDoublerSel](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllFreqDoublerSel](#) doubler)
- Function to enable/disable feedback frequency doubling.*
- [T\\_idtApllXtalId](#) [IDTApll\\_GetXtalId](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApllId](#) appllInstId)
- Function to retrieve oscillator currently used.*
- void [IDTApll\\_SetXtalId](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllXtalId](#) xtal)
- Function to select oscillator.*
- [T\\_idtApllOutputSigType](#) [IDTApll\\_GetOutputSigType](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllOutput](#) output)
- Function to retrieve output port signal type.*
- void [IDTApll\\_SetOutputSigType](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllOutput](#) output, [T\\_idtApllOutputSigType](#) outputSigType)
- Function to provision output port signal type.*
- [T\\_idtApllOutputEn](#) [IDTApll\\_GetOutputEnState](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllOutput](#) output)
- Function to retrieve output port status (enable/disable)*
- void [IDTApll\\_SetOutputEnState](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllOutput](#) output, [T\\_idtApllOutputEn](#) enable)
- Function to control (enable/disable) output port.*
- void [IDTApll\\_SetMasterResetSync](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllMasterResetSync](#) reset)
- Function to reset APLL.*
- [T\\_idtApllMasterResetSync](#) [IDTApll\\_GetMasterResetSync](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApllId](#) appllInstId)
- Function to retrieve reset status of device.*
- void [IDTApll\\_SetBypass](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllBypass](#) bypass)
- Function to set APLL bypass setting.*
- [T\\_idtApllBypass](#) [IDTApll\\_GetBypass](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApllId](#) appllInstId)
- Function to retrieve bypass status of device.*
- void [IDTApll\\_SetShutoff](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApllId](#) appllInstId, [T\\_idtApllShutoff](#) shutoff)
- Function to set APLL shutoff setting.*
- [T\\_idtApllShutoff](#) [IDTApll\\_GetShutoff](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApllId](#) appllInstId)
- Function to retrieve shutoff status of device.*

## 6.19.1 Macro Definition Documentation

### 6.19.1.1 #define APLL1\_REG\_HIGH\_BIT (0)

Definition at line 61 of file ApllApi.c.

### 6.19.1.2 #define APLL1\_REG\_VALID\_OFFSET( offset ) ((offset >= REG\_CONTROL) && (offset <= REG\_CONFIG\_QB))

Utility macro to sanity check APLL #1 register address.

#### Parameters

|               |                       |
|---------------|-----------------------|
| <i>offset</i> | APLL register address |
|---------------|-----------------------|

Definition at line 78 of file ApllApi.c.

**6.19.1.3 #define APLL2\_REG\_HIGH\_BIT (1)**

Definition at line 62 of file ApIApi.c.

**6.19.1.4 #define APLL2\_REG\_VALID\_OFFSET( offset )****Value:**

```
(offset >= APLL_REG_OFFSET(APLL_INST_2, REG_CONTROL)) && \
  (offset <= APLL_REG_OFFSET(APLL_INST_2, \
    REG_CONFIG_QB))
```

Utility macro to translate APLL #2 register address.

**Parameters**

|               |                            |
|---------------|----------------------------|
| <i>offset</i> | APLL register base address |
|---------------|----------------------------|

Definition at line 85 of file ApIApi.c.

**6.19.1.5 #define APLL\_DIVIDER\_LSB( val ) ((val)&0x00FF)**

Definition at line 45 of file ApIApi.c.

**6.19.1.6 #define APLL\_DIVIDER\_MSB( val ) (((val)&0x7F00)>>8)**

Definition at line 46 of file ApIApi.c.

**6.19.1.7 #define APLL\_DIVIDER\_VAL( lsb, msb ) ((msb)<<8|(lsb))**

Definition at line 47 of file ApIApi.c.

**6.19.1.8 #define APLL\_FB\_DIV\_LSB( val ) APLL\_DIVIDER\_LSB(val)**

Definition at line 52 of file ApIApi.c.

**6.19.1.9 #define APLL\_FB\_DIV\_MAX (0x7FFF)**

Definition at line 59 of file ApIApi.c.

**6.19.1.10 #define APLL\_FB\_DIV\_MIN (0x0005)**

Definition at line 58 of file ApIApi.c.

**6.19.1.11 #define APLL\_FB\_DIV\_MSB( val ) APLL\_DIVIDER\_MSB(val)**

Definition at line 53 of file ApIApi.c.

**6.19.1.12 #define APLL\_FB\_DIV\_VAL( lsb, msb ) APLL\_DIVIDER\_VAL(lsb, msb)**

Definition at line 54 of file ApIApi.c.

6.19.1.13 `#define APLL_PRE_DIV_LSB( val ) APLL_DIVIDER_LSB(val)`

Definition at line 49 of file ApllApi.c.

6.19.1.14 `#define APLL_PRE_DIV_MAX (0x7FFF)`

Definition at line 57 of file ApllApi.c.

6.19.1.15 `#define APLL_PRE_DIV_MIN (0x0004)`

Definition at line 56 of file ApllApi.c.

6.19.1.16 `#define APLL_PRE_DIV_MSB( val ) APLL_DIVIDER_MSB(val)`

Definition at line 50 of file ApllApi.c.

6.19.1.17 `#define APLL_PRE_DIV_VAL( lsb, msb ) APLL_DIVIDER_VAL(lsb, msb)`

Definition at line 51 of file ApllApi.c.

6.19.1.18 `#define APLL_REG_HIGH_BIT_OFFSET (7)`

Definition at line 63 of file ApllApi.c.

6.19.1.19 `#define APLL_REG_OFFSET( apllInstId, base )`

#### Value:

```
(apllInstId == APLL_INST_1) ? (base) : \
    (base | (APLL2_REG_HIGH_BIT << APLL_REG_HIGH_BIT_OFFSET))
```

Utility macro to translate APLL register address.

#### Parameters

|                   |                            |
|-------------------|----------------------------|
| <i>apllInstId</i> | APLL instance              |
| <i>base</i>       | APLL register base address |

Definition at line 70 of file ApllApi.c.

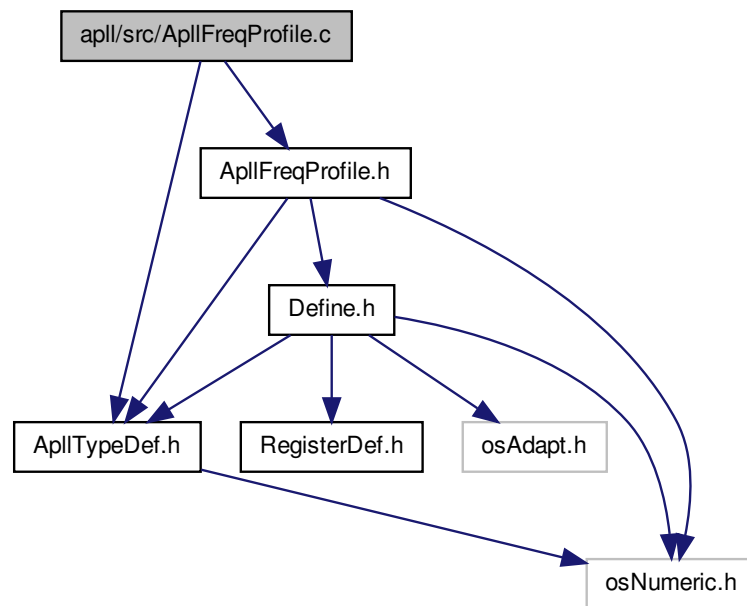
6.19.1.20 `#define INVALID_XTAL_SEL_VALUE (0xFF)`

Definition at line 89 of file ApllApi.c.

## 6.20 apll/src/ApllFreqProfile.c File Reference

```
#include "ApllTypeDef.h"
#include "ApllFreqProfile.h"
```

Include dependency graph for AplIFreqProfile.c:



## Macros

- #define [APLL\\_PRE\\_DIV\\_ETH](#) (19525)
- #define [APLL\\_PRE\\_DIV\\_SONET](#) (9775)
- #define [APLL\\_PRE\\_DIV\\_ETH\\_FEC](#) (15625)
- #define [APLL\\_FB\\_DIV\\_ETH](#) (3124)
- #define [APLL\\_FB\\_DIV\\_SONET](#) (3128)
- #define [APLL\\_FB\\_DIV\\_ETH\\_FEC](#) (3125)
- #define [APLL\\_OUTPUT\\_DIV\\_1](#) (1)
- #define [APLL\\_OUTPUT\\_DIV\\_2](#) (2)
- #define [APLL\\_OUTPUT\\_DIV\\_4](#) (4)
- #define [APLL\\_OUTPUT\\_DIV\\_5](#) (5)
- #define [APLL\\_OUTPUT\\_DIV\\_8](#) (8)
- #define [APLL\\_OUTPUT\\_DIV\\_25](#) (25)
- #define [MAX\\_CONFIG\\_COMBO\\_PER\\_FREQ](#) (1)
- #define [MAX\\_CONFIG\\_COMBO\\_PER\\_VC\\_XO\\_TYPE](#) (6)
- #define [NULL\\_APLL\\_FREQ\\_TABLE\\_ENTRY](#)

*Utility marco for initializing NULL APLL configuration entry.*

## Functions

- const [T\\_idtAplIFreqMapping](#) \* [IDTAplI\\_GetConfigListByFreq](#) ([T\\_idtAplIOutputFreqType](#) aplIOutputFreq)  
*Function to retrieve output configuration given an output frequency.*
- const [T\\_idtAplIFreqMapping](#) \* [IDTAplI\\_GetConfigListByXtalType](#) ([T\\_idtAplIXtalType](#) aplIVcxoFreq)  
*Function to retrieve output configuration given the oscillator type (frequency)*
- [T\\_osBool](#) [IDTAplI\\_NullFreqTableEntry](#) (const [T\\_idtAplIFreqMapping](#) \*aplIFreqConfig)

*Function to check whether the input APLL frequency mapping is NULL (not configured)*

- const `T_idtApIIFreqMapping` \* `IDTApII_GetCurrentApIIFreqConfig` (`T_idtApIIId` apIIId, `T_idtApIIOutput` output)

*Function to retrieve the current APLL frequency mapping configuration for the output.*

## 6.20.1 Macro Definition Documentation

### 6.20.1.1 `#define APLL_FB_DIV_ETH (3124)`

Definition at line 45 of file `ApIIFreqProfile.c`.

### 6.20.1.2 `#define APLL_FB_DIV_ETH_FEC (3125)`

Definition at line 47 of file `ApIIFreqProfile.c`.

### 6.20.1.3 `#define APLL_FB_DIV_SONET (3128)`

Definition at line 46 of file `ApIIFreqProfile.c`.

### 6.20.1.4 `#define APLL_OUTPUT_DIV_1 (1)`

Definition at line 50 of file `ApIIFreqProfile.c`.

### 6.20.1.5 `#define APLL_OUTPUT_DIV_2 (2)`

Definition at line 51 of file `ApIIFreqProfile.c`.

### 6.20.1.6 `#define APLL_OUTPUT_DIV_25 (25)`

Definition at line 55 of file `ApIIFreqProfile.c`.

### 6.20.1.7 `#define APLL_OUTPUT_DIV_4 (4)`

Definition at line 52 of file `ApIIFreqProfile.c`.

### 6.20.1.8 `#define APLL_OUTPUT_DIV_5 (5)`

Definition at line 53 of file `ApIIFreqProfile.c`.

### 6.20.1.9 `#define APLL_OUTPUT_DIV_8 (8)`

Definition at line 54 of file `ApIIFreqProfile.c`.

### 6.20.1.10 `#define APLL_PRE_DIV_ETH (19525)`

Definition at line 40 of file `ApIIFreqProfile.c`.

### 6.20.1.11 `#define APLL_PRE_DIV_ETH_FEC (15625)`

Definition at line 42 of file `ApIIFreqProfile.c`.

**6.20.1.12 #define APLL\_PRE\_DIV\_SONET (9775)**

Definition at line 41 of file ApIIFreqProfile.c.

**6.20.1.13 #define MAX\_CONFIG\_COMBO\_PER\_FREQ (1)**

Definition at line 57 of file ApIIFreqProfile.c.

**6.20.1.14 #define MAX\_CONFIG\_COMBO\_PER\_VC XO\_TYPE (6)**

Definition at line 58 of file ApIIFreqProfile.c.

**6.20.1.15 #define NULL\_APLL\_FREQ\_TABLE\_ENTRY**

**Value:**

```
{
    APLL_INPUT_FREQ_MAX,      \
    0,                        \
    0,                        \
    APLL_XTAL_FREQ_MAX,      \
    0,                        \
    APLL_OUTPUT_FREQ_MAX,    \
    APLL_OUT_MAX              \
}
```

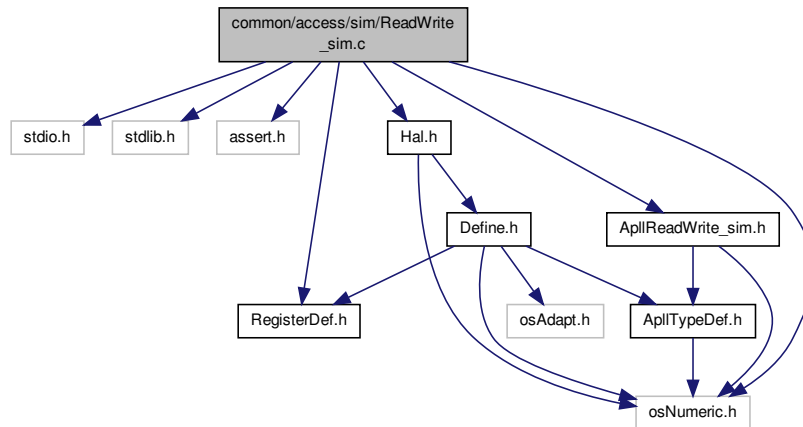
Utility marco for initializing NULL APLL configuration entry.

Definition at line 63 of file ApIIFreqProfile.c.

**6.21 common/access/sim/ReadWrite\_sim.c File Reference**

```
#include <stdio.h>
#include <stdlib.h>
#include <assert.h>
#include "RegisterDef.h"
#include "osNumeric.h"
#include "Hal.h"
#include "ApIIFreqProfile.h"
```

Include dependency graph for ReadWrite\_sim.c:



## Macros

- `#define DRV_MEMMAP_SIZE 0x80`
- `#define NONE_APLL_REG_SPACE_FLAG 0x01`

## Functions

- void `Init_Sim` (void \*usrData)
- void `DeInit_Sim` (void \*usrData)
- T\_osUInt8 `RDByte_Sim` (void \*usrData, T\_osUInt8 devAddr, T\_osUInt8 nAddr)
- void `WRByte_Sim` (void \*usrData, T\_osUInt8 devAddr, T\_osUInt8 nAddr, T\_osUInt8 nData)

## Variables

- int `simDebugFlag` = 0
- T\_osUInt8 `simulatedMemMap` [2][`DRV_MEMMAP_SIZE`]
- T\_osUInt8 `dplOverlayRegisters` []
- T\_osUInt8 `page1Registers` []

### 6.21.1 Macro Definition Documentation

#### 6.21.1.1 `#define DRV_MEMMAP_SIZE 0x80`

Definition at line 45 of file ReadWrite\_sim.c.

#### 6.21.1.2 `#define NONE_APLL_REG_SPACE_FLAG 0x01`

Definition at line 46 of file ReadWrite\_sim.c.

## 6.21.2 Function Documentation

### 6.21.2.1 void Delnit\_Sim ( void \* *usrData* )

Definition at line 162 of file ReadWrite\_sim.c.

### 6.21.2.2 void Init\_Sim ( void \* *usrData* )

Definition at line 157 of file ReadWrite\_sim.c.

### 6.21.2.3 T\_uint8 RDByte\_Sim ( void \* *usrData*, T\_uint8 *devAddr*, T\_uint8 *nAddr* )

Definition at line 167 of file ReadWrite\_sim.c.

### 6.21.2.4 void WRByte\_Sim ( void \* *usrData*, T\_uint8 *devAddr*, T\_uint8 *nAddr*, T\_uint8 *nData* )

Definition at line 222 of file ReadWrite\_sim.c.

## 6.21.3 Variable Documentation

### 6.21.3.1 T\_uint8 dpllOverlayRegisters[]

**Initial value:**

```
= {
    REG_IN1_IN2_SEL_PRIORITY_CNFG,
    REG_IN3_IN4_SEL_PRIORITY_CNFG,
    REG_IN5_IN6_SEL_PRIORITY_CNFG,
    REG_IN7_IN8_SEL_PRIORITY_CNFG,
    REG_IN9_IN10_SEL_PRIORITY_CNFG,
    REG_IN11_IN12_SEL_PRIORITY_CNFG,
    REG_IN13_IN14_SEL_PRIORITY_CNFG,
    REG_PRIORITY_TABLE1_STS,
    REG_PRIORITY_TABLE2_STS,
    REG_PHASE_LOSS_COARSE_LIMIT_CNFG,
    REG_PHASE_LOSS_FINE_LIMIT_CNFG,
    REG_HOLDOVER_FREQ_LOW_CNFG,
    REG_HOLDOVER_FREQ_MID_CNFG,
    REG_HOLDOVER_FREQ_HIGH_CNFG,
    REG_CURRENT_FREQ_LOW_STS,
    REG_CURRENT_FREQ_MID_STS,
    REG_CURRENT_FREQ_HIGH_STS,
    REG_CURRENT_DPLL_PHASE_LOW_STS,
    REG_CURRENT_DPLL_PHASE_HIGH_STS
}
```

Definition at line 90 of file ReadWrite\_sim.c.

### 6.21.3.2 T\_uint8 page1Registers[]

**Initial value:**

```
= {
    REG_DFS_OFF_CNFG,
    REG_T0_PPS_CNFG,
    REG_PH_SLOPE_CNFG,
    REG_T0_ICP_CTRL_CNFG,
    REG_T4_ICP_CTRL_CNFG,
    REG_T4_PPS_CNFG
}
```

Definition at line 139 of file ReadWrite\_sim.c.

## 6.21.3.3 int simDebugFlag = 0

Definition at line 48 of file ReadWrite\_sim.c.

## 6.21.3.4 T\_osUInt8 simulatedMemMap[2][DRV\_MEMMAP\_SIZE]

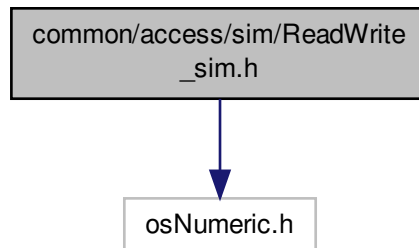
Reset values for device

Definition at line 51 of file ReadWrite\_sim.c.

## 6.22 common/access/sim/ReadWrite\_sim.h File Reference

```
#include "osNumeric.h"
```

Include dependency graph for ReadWrite\_sim.h:



### Functions

- void [Init\\_Sim](#) (void \*usrData)
- void [DelInit\\_Sim](#) (void \*usrData)
- T\_osUInt8 [RDByte\\_Sim](#) (void \*usrData, T\_osUInt8 devAddr, T\_osUInt8 nAddr)
- void [WRByte\\_Sim](#) (void \*usrData, T\_osUInt8 devAddr, T\_osUInt8 nAddr, T\_osUInt8 nData)

### 6.22.1 Function Documentation

6.22.1.1 void DelInit\_Sim ( void \* *usrData* )

Definition at line 162 of file ReadWrite\_sim.c.

6.22.1.2 void Init\_Sim ( void \* *usrData* )

Definition at line 157 of file ReadWrite\_sim.c.

6.22.1.3 T\_osUInt8 RDByte\_Sim ( void \* *usrData*, T\_osUInt8 *devAddr*, T\_osUInt8 *nAddr* )

Definition at line 167 of file ReadWrite\_sim.c.

6.22.1.4 void WRByte\_Sim ( void \* usrData, T\_osUInt8 devAddr, T\_osUInt8 nAddr, T\_osUInt8 nData )

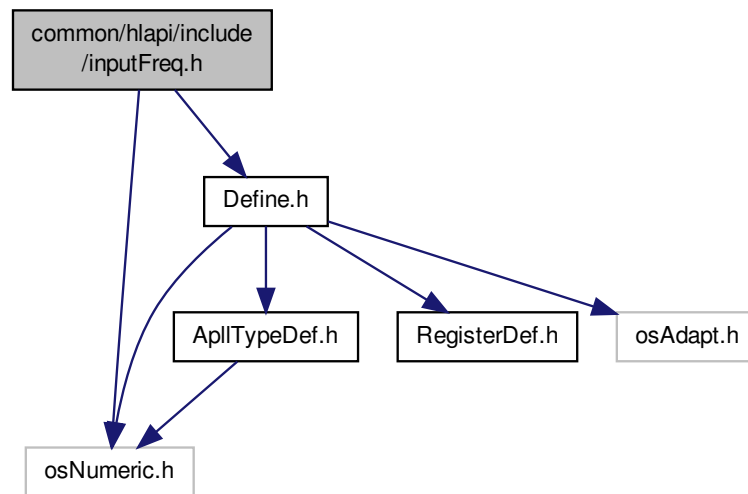
Definition at line 222 of file ReadWrite\_sim.c.

## 6.23 common/hlapi/include/inputFreq.h File Reference

```
#include "osNumeric.h"
```

```
#include "Define.h"
```

Include dependency graph for inputFreq.h:



## Functions

- void [IDTHIApi\\_ConfigureInput1PPS](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Function to provision input port for 1PPS (1 Hz)*
- void [IDTHIApi\\_ConfigureInputAmi](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, [T\\_idtAmiInputFrequencyBitFieldValue](#) amiSignal)  
*Function to provision input AMI port.*
- void [IDTHIApi\\_ConfigureInputFrequency](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, T\_osUInt32 nFrequency)  
*Function to provision input port.*
- void [IDTHIApi\\_DisableAllInputPriorities](#) (T\_idtDrvHdlr hdlr)  
*Function to disable all inputs priorities.*

### 6.23.1 Function Documentation

6.23.1.1 void IDTHIApi\_ConfigureInput1PPS ( T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo )

Function to provision input port for 1PPS (1 Hz)

Provision input for 1PPS (1 Hz)

## Parameters

|                 |                   |
|-----------------|-------------------|
| <i>hdlr</i>     | driver handler    |
| <i>nInputNo</i> | input port number |

Definition at line 180 of file inputFreq.c.

**6.23.1.2** void IDTHIApi\_ConfigureInputAmi ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo*, T\_idtAmiInputFrequencyBitFieldValue *amiSignal* )

Function to provision input AMI port.

Provision AMI input

## Parameters

|                  |                                                              |
|------------------|--------------------------------------------------------------|
| <i>hdlr</i>      | driver handler                                               |
| <i>nInputNo</i>  | input port number                                            |
| <i>amiSignal</i> | Selected AMI signal /see T_idtAmiInputFrequencyBitFieldValue |

Definition at line 222 of file inputFreq.c.

**6.23.1.3** void IDTHIApi\_ConfigureInputFrequency ( T\_idtDrvHdlr *hdlr*, T\_osUInt8 *nInputNo*, T\_osUInt32 *nFrequency* )

Function to provision input port.

Provision input port specify input frequency

## Parameters

|                   |                          |
|-------------------|--------------------------|
| <i>hdlr</i>       | driver handler           |
| <i>nInputNo</i>   | input port number        |
| <i>nFrequency</i> | input frequency (in kHz) |

Definition at line 357 of file inputFreq.c.

**6.23.1.4** void IDTHIApi\_DisableAllInputPriorities ( T\_idtDrvHdlr *hdlr* )

Function to disable all inputs priorities.

Provision disable all input priorities

This is different to enabling inputs. Input priorities are used by DPLLs to select input ports to lock to. Disable input priorities tells DPLLs to ignore all inputs.

Use this function when a custom priority is desired. This functions will clear all input priorities first. Input priorities can then be set by calling IDTInput\_SetPriority function.

## Parameters

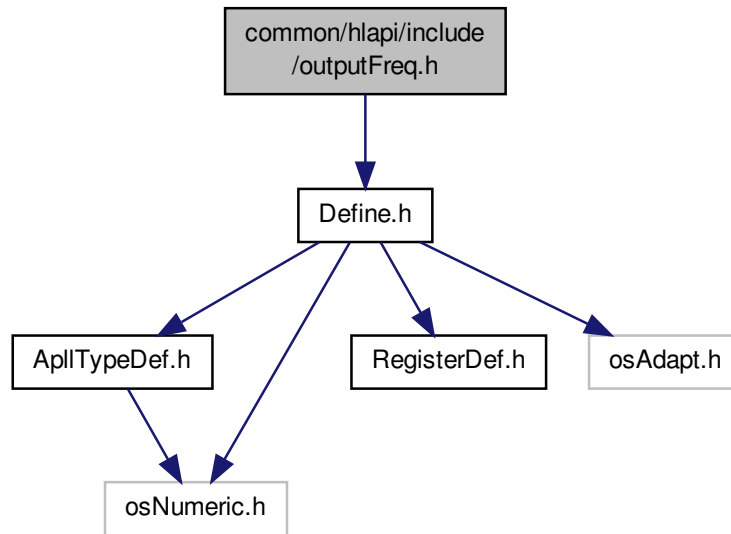
|             |                      |
|-------------|----------------------|
| <i>hdlr</i> | Device driver handle |
|-------------|----------------------|

Definition at line 470 of file inputFreq.c.

## 6.24 common/hlapi/include/outputFreq.h File Reference

```
#include "Define.h"
```

Include dependency graph for outputFreq.h:



## Enumerations

- enum `outputFrequency_t` {  
`OutFreq_Invalid`, `OutFreq_64k`, `OutFreq_8k`, `OutFreq_2k`,  
`OutFreq_400`, `OutFreq_1`, `OutFreq_24576k`, `OutFreq_12288k`,  
`OutFreq_6144k`, `OutFreq_4096k`, `OutFreq_2048k`, `OutFreq_32768k`,  
`OutFreq_16384k`, `OutFreq_8192k`, `OutFreq_24704k`, `OutFreq_12352k`,  
`OutFreq_6176k`, `OutFreq_3088k`, `OutFreq_1544k`, `OutFreq_34368k`,  
`OutFreq_44736k`, `OutFreq_26000k`, `OutFreq_13000k`, `OutFreq_30720k`,  
`OutFreq_15360k`, `OutFreq_7680k`, `OutFreq_3840k`, `OutFreq_37056k`,  
`OutFreq_18528k`, `OutFreq_9264k`, `OutFreq_4632k`, `OutFreq_40000k`,  
`OutFreq_20000k`, `OutFreq_10000k`, `OutFreq_5000k`, `OutFreq_622080k`,  
`OutFreq_625000k`, `OutFreq_625000k_66_64`, `OutFreq_311040k`, `OutFreq_312500k`,  
`OutFreq_312500k_66_64`, `OutFreq_155520k`, `OutFreq_156250k`, `OutFreq_156250k_66_64`,  
`OutFreq_77760k`, `OutFreq_51840k`, `OutFreq_38880k`, `OutFreq_25920k`,  
`OutFreq_125000k`, `OutFreq_125000k_66_64`, `OutFreq_19440k`, `OutFreq_25000k`,  
`OutFreq_25000k_66_64`, `OutFreq_6480k`, `OutFreq_24883_DOT_2k`, `OutFreq_78125k`,  
`OutFreq_78125k_66_64`, `OutFreq_124416k`, `OutFreq_MAX` }

*Enumerated type listing output frequencies.*

- enum `T_sonetGigeMode` { `sonetGigeMode_Matching`, `sonetGigeMode_AllFromDpll1`, `sonetGigeMode_AllFromDpll2`, `sonetGigeMode_MAX` }

*Enumerated type listing sonetGige modes.*

## Functions

- int `IDTHIApi_ConfigureOutput` (`T_idtDrvHdlr` hdlr, `T_osUInt8` nOutN, `outputFrequency_t` nOutFreq, `T_idtDpll-Instance` dpllInstance, `T_sonetGigeMode` sonetGigeMode, `T_idtApplConfigSel` applConfSel, `T_idtApplOutput-SigType` applOutputSigType, `T_idtApplInputClkSrc` applClkSource)

*Function to provision output port.*

- void [IDTHIApi\\_DisableAllOutputs](#) (T\_idtDrvHdlr hdlr)  
*Function to disable all outputs.*
- void [IDTHIApi\\_ClearProvisioning](#) (T\_idtDrvHdlr hdlr)  
*Function to clear saved provisioning used by high level API.*
- void [IDTHIApi\\_PrintOutput](#) (T\_idtDrvHdlr hdlr)
- void [IDTHIApi\\_PrintInput](#) (T\_idtDrvHdlr hdlr)

## 6.24.1 Enumeration Type Documentation

### 6.24.1.1 enum outputFrequency\_t

Enumerated type listing output frequencies.

Enumerator

**OutFreq\_Invalid** Output frequency invalid

**OutFreq\_64k** Output frequency is 64 kHz

**OutFreq\_8k** Output frequency is 8 kHz

**OutFreq\_2k** Output frequency is 2 kHz

**OutFreq\_400** Output frequency is 400 Hz

**OutFreq\_1** Output frequency is 1 Hz (1 PPS)

**OutFreq\_24576k** Output frequency is 24.576 MHz

**OutFreq\_12288k** Output frequency is 12.288 MHz

**OutFreq\_6144k** Output frequency is 6.144 MHz

**OutFreq\_4096k** Output frequency is 4.096 MHz

**OutFreq\_2048k** Output frequency is 2.048 MHz

**OutFreq\_32768k** Output frequency is 32.768 MHz

**OutFreq\_16384k** Output frequency is 16.384MHz

**OutFreq\_8192k** Output frequency is 8.192 MHz

**OutFreq\_24704k** Output frequency is 24.704 MHz

**OutFreq\_12352k** Output frequency is 12.352 MHz

**OutFreq\_6176k** Output frequency is 6.176 MHz

**OutFreq\_3088k** Output frequency is 3.088 MHz

**OutFreq\_1544k** Output frequency is 1.544 MHz

**OutFreq\_34368k** Output frequency is 34.368 MHz

**OutFreq\_44736k** Output frequency is 44.736 MHz

**OutFreq\_26000k** Output frequency is 26 MHz

**OutFreq\_13000k** Output frequency is 13 MHz

**OutFreq\_30720k** Output frequency is 30.72 MHz

**OutFreq\_15360k** Output frequency is 15.36 MHz

**OutFreq\_7680k** Output frequency is 7.68 MHz

**OutFreq\_3840k** Output frequency is 3.84 MHz

**OutFreq\_37056k** Output frequency is 37.056 MHz

**OutFreq\_18528k** Output frequency is 18.528 MHz

**OutFreq\_9264k** Output frequency is 9.264 MHz

**OutFreq\_4632k** Output frequency is 4.632 MHz

**OutFreq\_40000k** Output frequency is 40 MHz

**OutFreq\_20000k** Output frequency is 20 MHz  
**OutFreq\_10000k** Output frequency is 10 MHz  
**OutFreq\_5000k** Output frequency is 5 MHz  
**OutFreq\_622080k** Output frequency is 622.08 MHz  
**OutFreq\_625000k** Output frequency is 625 MHz  
**OutFreq\_625000k\_66\_64** Output frequency is  $625 \times (66/64)$  MHz  
**OutFreq\_311040k** Output frequency is 311.04 MHz  
**OutFreq\_312500k** Output frequency is 312.5 MHz  
**OutFreq\_312500k\_66\_64** Output frequency is  $312.5 \times (66/64)$  MHz  
**OutFreq\_155520k** Output frequency is 155.52 MHz  
**OutFreq\_156250k** Output frequency is 156.25 MHz  
**OutFreq\_156250k\_66\_64** Output frequency is  $156.25 \times (66/64)$  MHz  
**OutFreq\_77760k** Output frequency is 77.76 MHz  
**OutFreq\_51840k** Output frequency is 51.84 MHz  
**OutFreq\_38880k** Output frequency is 38.88 MHz  
**OutFreq\_25920k** Output frequency is 25.92 MHz  
**OutFreq\_125000k** Output frequency is 125 MHz  
**OutFreq\_125000k\_66\_64** Output frequency is  $125 \times (66/64)$  MHz  
**OutFreq\_19440k** Output frequency is 19.44 MHz  
**OutFreq\_25000k** Output frequency is 25 MHz  
**OutFreq\_25000k\_66\_64** Output frequency is  $25 \times (66/64)$  MHz  
**OutFreq\_6480k** Output frequency is 6.48 MHz  
**OutFreq\_24883\_DOT\_2k** Output frequency is 24.8832 MHz  
**OutFreq\_78125k** Output frequency is 78.125 MHz  
**OutFreq\_78125k\_66\_64** Output frequency is 80.56640625 MHz  
**OutFreq\_124416k** Output frequency is 124.416 MHz  
**OutFreq\_MAX**

Definition at line 46 of file outputFreq.h.

#### 6.24.1.2 enum T\_sonetGigeMode

Enumerated type listing sonetGige modes.

Enumerator

**sonetGigeMode\_Matching** DPLL #1 to APLL #1 and DPLL #2 to APLL #2  
**sonetGigeMode\_AllFromDpll1** DPLL #1 to APLL #1 and DPLL #1 to APLL #2  
**sonetGigeMode\_AllFromDpll2** DPLL #2 to APLL #1 and DPLL #2 to APLL #2  
**sonetGigeMode\_MAX**

Definition at line 113 of file outputFreq.h.

### 6.24.2 Function Documentation

#### 6.24.2.1 void IDTHIApi\_ClearProvisioning ( T\_idtDrvHdlr hdlr )

Function to clear saved provisioning used by high level API.

Parameters

|             |                      |
|-------------|----------------------|
| <i>hdlr</i> | Device driver handle |
|-------------|----------------------|

Definition at line 1464 of file outputFreq.c.

6.24.2.2 `int IDTHIApi_ConfigureOutput ( T_idtDrvHdlr hdlr, T_osUInt8 nOutN, outputFrequency_t nOutFreq, T_idtDpllInstance dpllInstance, T_sonetGigeMode sonetGigeMode, T_idtApllConfigSel apIConfSel, T_idtApllOutputSigType apIOutputSigType, T_idtApllInputClkSrc apIClkSource )`

Function to provision output port.

Provision output port specifying output frequency and DPLL to use

#### Parameters

|                      |                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------|
| <i>hdlr</i>          | driver handler                                                                                   |
| <i>nOutN</i>         | Output port (1-11)                                                                               |
| <i>nOutFreq</i>      | Output frequency /see outputFrequency_t for list of supported frequencies                        |
| <i>dpllInstance</i>  | Dpll Instance (DPLL_INSTANCE_1, DPLL_INSTANCE_2, specify DPLL_INSTANCE_MAX for no specific DPLL) |
| <i>sonetGigeMode</i> | sonetGige mux configuration.                                                                     |

#### See Also

[T\\_sonetGigeMode](#) This parameter and dpllInstance determine what output path(s) software may use to generate specified output frequency. The following table illustrates the available paths,

| dpllInstance                      | sonetGigeMode | Avail. Paths                                      |
|-----------------------------------|---------------|---------------------------------------------------|
| <a href="#">DPLL_INSTANCE_1</a>   | Matching      | DPLL#1->SonetGigEth#1 only                        |
| <a href="#">DPLL_INSTANCE_2</a>   | Matching      | DPLL#2->SonetGigEth#2 only                        |
| <a href="#">DPLL_INSTANCE_MAX</a> | Matching      | DPLL#1->SonetGigEth#1 or<br>DPLL#2->SonetGigEth#2 |
| <a href="#">DPLL_INSTANCE_1</a>   | AllFromDpll1  | DPLL#1->SonetGigEth#1 only                        |
| <a href="#">DPLL_INSTANCE_2</a>   | AllFromDpll1  | DPLL#1->SonetGigEth#2 only                        |
| <a href="#">DPLL_INSTANCE_MAX</a> | AllFromDpll1  | DPLL#1->SonetGigEth#1 or<br>DPLL#1->SonetGigEth#2 |
| <a href="#">DPLL_INSTANCE_1</a>   | AllFromDpll2  | DPLL#2->SonetGigEth#1 only                        |
| <a href="#">DPLL_INSTANCE_2</a>   | AllFromDpll2  | DPLL#2->SonetGigEth#2 only                        |
| <a href="#">DPLL_INSTANCE_MAX</a> | AllFromDpll2  | DPLL#2->SonetGigEth#1 or<br>DPLL#2->SonetGigEth#2 |

#### Parameters

|                          |                                                                                                                                                     |
|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>apIConfSel</i>        | Selects the APLL configuration, set to APLL_CONFIG_MAX if output port does not go through APLL, OR if you want to program backup APLL configuration |
| <i>apIOutputSig-Type</i> | APLL output signal type, set to APLL_OUT_SIG_MAX if output port does not go through APLL                                                            |

|                      |                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------|
| <i>apllClkSource</i> | APLL clock source (internal or external), set to APLL_CLK_SEL_MAX if output port does not go through APLL |
|----------------------|-----------------------------------------------------------------------------------------------------------|

#### Returns

0-OK, 1-Error, unable to configure output path mux

Definition at line 1261 of file outputFreq.c.

#### 6.24.2.3 void IDTHIApi\_DisableAllOutputs ( T\_idtDrvHdlr *hdlr* )

Function to disable all outputs.

#### Parameters

|             |                |
|-------------|----------------|
| <i>hdlr</i> | driver handler |
|-------------|----------------|

Definition at line 1443 of file outputFreq.c.

#### 6.24.2.4 void IDTHIApi\_PrintInput ( T\_idtDrvHdlr *hdlr* )

Definition at line 948 of file outputFreqString.c.

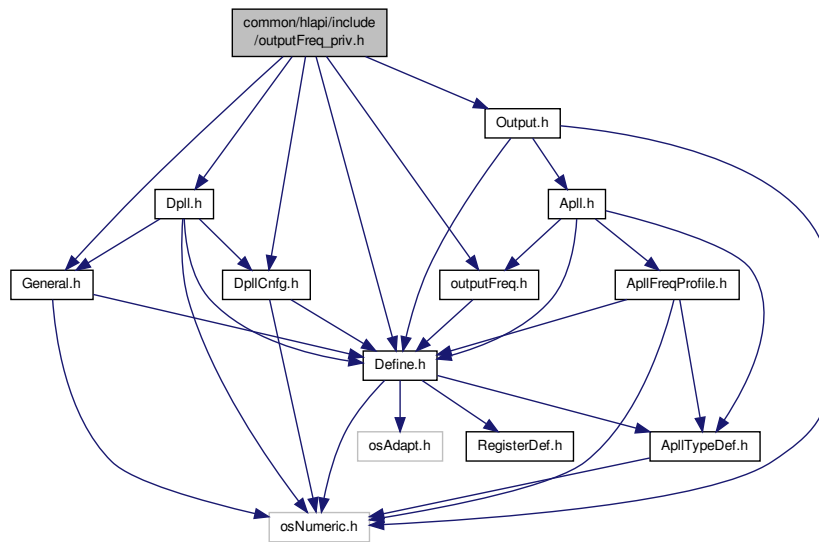
#### 6.24.2.5 void IDTHIApi\_PrintOutput ( T\_idtDrvHdlr *hdlr* )

Definition at line 1011 of file outputFreqString.c.

## 6.25 common/hlapi/include/outputFreq\_priv.h File Reference

```
#include "Define.h"
#include "General.h"
#include "DpllCnfg.h"
#include "Output.h"
#include "Dpll.h"
#include "outputFreq.h"
```

Include dependency graph for outputFreq\_priv.h:



## Data Structures

- struct [pathSelectorConfig\\_t](#)  
Structure containing select configuration. Used to translate between frequency and output path configuration.
- struct [pathSelectorIndex\\_t](#)  
Structure to index to path selector configs.

## Macros

- #define [SET\\_BIT](#)(array, bit, set) array[bit] = set  
set a bit within a bit array
- #define [CHECK\\_BIT](#)(array, bit) (array[bit] != OutFreq\_Invalid)  
check a bit within a bit array

## Typedefs

- typedef int [validFreq\\_t](#) [OutFreq\_MAX]  
Array type for storing frequency usage.

## Enumerations

- enum [pathSelectors\\_t](#) {  
Selector\_Network, Selector\_T0DpllSelA, Selector\_T0DpllSelB, Selector\_T4DpllSelA,  
Selector\_T4DpllSelB, Selector\_SonetGigEth, Selector\_Any, Selector\_MAX }  
Enumerated type listing output path selectors.

## Functions

- void `IDTHIApi_ConvertFromFreqListToFreqArray` (`outputFrequency_t` frequencies[], `validFreq_t` validFrequencies)
- Convert from a frequency list to frequency array.*
- void `IDTHIApi_ApplyFrequencies` (const `outputFrequency_t` \*frequencies, `validFreq_t` validFrequencies, int value)
- Function to set status of frequency in frequency array.*
- int `IDTHIApi_IsFreqArrayInFreqList` (const `outputFrequency_t` \*frequencies, `validFreq_t` validFrequencies)
- int `IDTHIApi_IsFrequencyInFreqList` (`outputFrequency_t` frequency, const `outputFrequency_t` validFrequencies[])
- Function to verify that frequency is within frequency list.*
- void `IDTHIApi_PrintAvailFrequencies` (`validFreq_t` frequencyArray)
- Function to print frequency array in string format.*
- void `IDTHIApi_PrintFrequencyList` (const `outputFrequency_t` frequencies[])
- Function to print frequency list in string format.*
- void `IDTHIApi_PrintConfiguration` (`T_IDTPathConfiguration` pathConfig)
- Function to print output path configuration to a string format.*
- void `IDTHIApi_PrintOption` (`pathSelectorConfig_t` option, `T_osUInt8` printPort)
- Function to print output path configuration option to string format.*
- void `IDTHIApi_PrintSelectedOutputPath` (`T_outputPathType` nPath)
- Function to display selected output port path.*
- void `IDTHIApi_PrintOutputHwConfig` (`networkType_t` network, `T_osUInt8` nOutN, `T_osUInt8` internalPort, `T_osUInt8` isEnabled, `T_osUInt8` hasApI, `T_idtApIldividerValue` apIldivVal, `T_idtApIXtalType` xtalType, `T_osUInt8` outDiv, `T_outputPathType` outputPath, `T_idtDpllInstance` dpllInstOrSonetGigEthInst, `T_idtDpllInstance` sonetGigEthInput, `T_idtSonetGigEthOutput` sonetGigEthMode, `T_idtDpllOutputSelectorA` dpllSelA, `T_idtDpllOutputSelectorB` dpllSelB)

## Variables

- const char \* `IDTHIApi_Frequency2String_c` [`OutFreq_MAX`]

### 6.25.1 Macro Definition Documentation

#### 6.25.1.1 `#define CHECK_BIT( array, bit ) (array[bit] != OutFreq_Invalid)`

check a bit within a bit array

Definition at line 89 of file `outputFreq_priv.h`.

#### 6.25.1.2 `#define SET_BIT( array, bit, set ) array[bit] = set`

set a bit within a bit array

Definition at line 85 of file `outputFreq_priv.h`.

### 6.25.2 Typedef Documentation

#### 6.25.2.1 `typedef int validFreq_t[OutFreq_MAX]`

Array type for storing frequency usage.

Definition at line 104 of file `outputFreq_priv.h`.

### 6.25.3 Enumeration Type Documentation

#### 6.25.3.1 enum pathSelectors\_t

Enumerated type listing output path selectors.

Enumerator

**Selector\_Network** Network selector (SDH/SONET)  
**Selector\_T0DpllSelA** T0 Dpll selector A (16E1/16T1/GSM/OBSAI)  
**Selector\_T0DpllSelB** T0 Dpll selector B (12E1/GPS/E3/T3)  
**Selector\_T4DpllSelA** T4 Dpll selector A (16E1/16T1/GSM/GPS)  
**Selector\_T4DpllSelB** T4 Dpll selector B (12E1/24T1/E3/T3)  
**Selector\_SonetGigEth** APLL selector (T0 622/T4 622/T0 625/ T4 625/T0 625(66/64)/ T4 625(66/64))  
**Selector\_Any** For 77.76 / Eth always available frequencies  
**Selector\_MAX**

Definition at line 53 of file outputFreq\_priv.h.

### 6.25.4 Function Documentation

6.25.4.1 void IDTHIApi.ApplyFrequencies ( const outputFrequency\_t \* frequencies, validFreq\_t validFrequencies, int value )

Function to set status of frequency in frequency array.

Parameters

|                         |                                   |
|-------------------------|-----------------------------------|
| <i>frequencies</i>      | Constant frequency list           |
| <i>validFrequencies</i> |                                   |
| <i>value</i>            | Status to set for frequency array |

Definition at line 79 of file outputFreqUtil.c.

6.25.4.2 void IDTHIApi.ConvertFromFreqListToFreqArray ( outputFrequency\_t frequencies[], validFreq\_t validFrequencies )

Convert from a frequency list to frequency array.

Parameters

|                         |                           |
|-------------------------|---------------------------|
| <i>frequencies</i>      | Frequency list to convert |
| <i>validFrequencies</i> | Output frequency array    |

Definition at line 57 of file outputFreqUtil.c.

6.25.4.3 int IDTHIApi.IsFreqArrayInFreqList ( const outputFrequency\_t \* frequencies, validFreq\_t validFrequencies )

6.25.4.4 int IDTHIApi.IsFrequencyInFreqList ( outputFrequency\_t frequency, const outputFrequency\_t validFrequencies[] )

Function to verify that frequency is within frequency list.

## Parameters

|                         |                                 |
|-------------------------|---------------------------------|
| <i>frequency</i>        | Frequency to check              |
| <i>validFrequencies</i> | Frequency list to check against |

## Returns

1-Frequency is in list, 0-Frequency is not in list

Definition at line 124 of file outputFreqUtil.c.

#### 6.25.4.5 void IDTHIApi\_PrintAvailFrequencies ( validFreq\_t frequencyArray )

Function to print frequency array in string format.

## Parameters

|                       |                 |
|-----------------------|-----------------|
| <i>frequencyArray</i> | Frequency array |
|-----------------------|-----------------|

Definition at line 317 of file outputFreqString.c.

#### 6.25.4.6 void IDTHIApi\_PrintConfiguration ( T\_IDTPathConfiguration pathConfig )

Function to print output path configuration to a string format.

## Parameters

|                   |                    |
|-------------------|--------------------|
| <i>pathConfig</i> | Output path config |
|-------------------|--------------------|

Definition at line 358 of file outputFreqString.c.

#### 6.25.4.7 void IDTHIApi\_PrintFrequencyList ( const outputFrequency\_t frequencies[] )

Function to print frequency list in string format.

## Parameters

|                    |                |
|--------------------|----------------|
| <i>frequencies</i> | Frequency list |
|--------------------|----------------|

Definition at line 339 of file outputFreqString.c.

#### 6.25.4.8 void IDTHIApi\_PrintOption ( pathSelectorConfig\_t option, T\_osUInt8 printPort )

Function to print output path configuration option to string format.

## Parameters

|                  |                                                             |
|------------------|-------------------------------------------------------------|
| <i>option</i>    | Output path configuration option.                           |
| <i>printPort</i> | Flag to indicate whether port information is also displayed |

Definition at line 387 of file outputFreqString.c.

6.25.4.9 void IDTHIApi\_PrintOutputHwConfig ( networkType\_t network, T\_osUInt8 nOutN, T\_osUInt8 internalPort, T\_osUInt8 isEnabled, T\_osUInt8 hasAplI, T\_idtAplIDividerValue aplIDivVal, T\_idtAplIXtalType xtalType, T\_osUInt8 outDiv, T\_outputPathType outPath, T\_idtDpllInstance dpllInstOrSonetGigEthInst, T\_idtDpllInstance sonetGigEthInput, T\_idtSonetGigEthOutput sonetGigEthMode, T\_idtDpllOutputSelectorA dpllSelA, T\_idtDpllOutputSelectorB dpllSelB )

Definition at line 581 of file outputFreqString.c.

6.25.4.10 void IDTHIApi\_PrintSelectedOutputPath ( T\_outputPathType nPath )

Function to display selected output port path.

#### Parameters

|              |                  |
|--------------|------------------|
| <i>nPath</i> | output port path |
|--------------|------------------|

Definition at line 443 of file outputFreqString.c.

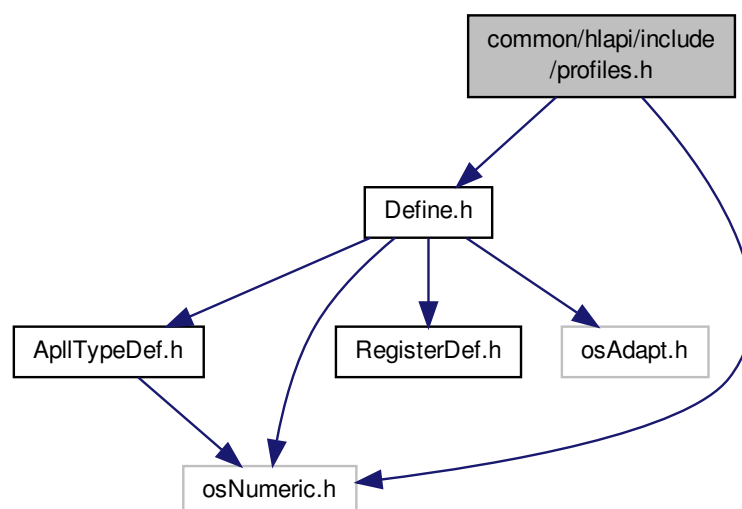
## 6.25.5 Variable Documentation

6.25.5.1 const char\* IDTHIApi\_Frequency2String\_c[OutFreq\_MAX]

Definition at line 77 of file outputFreqString.c.

## 6.26 common/hlapi/include/profiles.h File Reference

```
#include "Define.h"
#include "osNumeric.h"
Include dependency graph for profiles.h:
```



## Functions

- void [IDTHIApi\\_gr253SonetStratum3](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*DPLL configuration for GR-253 (SONET) Stratum 3.*
- void [IDTHIApi\\_gr1244Stratum3](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*DPLL configuration for GR-1244 Stratum 3.*
- void [IDTHIApi\\_smc](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*DPLL configuration for SMC.*
- void [IDTHIApi\\_g8262EecOption1](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*DPLL configuration for G-8262 Option 1.*
- void [IDTHIApi\\_g8262EecOption2](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*DPLL configuration for G-8262 Option 2.*
- void [IDTHIApi\\_g813Option1](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*DPLL configuration for G-813 Option 1.*
- void [IDTHIApi\\_g813Option2](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*DPLL configuration for G-813 Option 2.*
- void [IDTHIApi\\_pps](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*DPLL configuration for 1 pulse per second (PPS)*
- void [IDTHIApi\\_wideband](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*DPLL configuration for wideband.*

### 6.26.1 Function Documentation

#### 6.26.1.1 void IDTHIApi\_g813Option1 ( T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll )

DPLL configuration for G-813 Option 1.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 396 of file profiles.c.

#### 6.26.1.2 void IDTHIApi\_g813Option2 ( T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll )

DPLL configuration for G-813 Option 2.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 408 of file profiles.c.

#### 6.26.1.3 void IDTHIApi\_g8262EecOption1 ( T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll )

DPLL configuration for G-8262 Option 1.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 258 of file profiles.c.

6.26.1.4 void IDTHIApi\_g8262EecOption2 ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

DPLL configuration for G-8262 Option 2.

Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 327 of file profiles.c.

6.26.1.5 void IDTHIApi\_gr1244Stratum3 ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

DPLL configuration for GR-1244 Stratum 3.

Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 120 of file profiles.c.

6.26.1.6 void IDTHIApi\_gr253SonetStratum3 ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

DPLL configuration for GR-253 (SONET) Stratum 3.

Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 51 of file profiles.c.

6.26.1.7 void IDTHIApi\_pps ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

DPLL configuration for 1 pulse per second (PPS)

Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 421 of file profiles.c.

6.26.1.8 void IDTHIApi\_smc ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

DPLL configuration for SMC.

Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 189 of file profiles.c.

6.26.1.9 void IDTHIApi\_wideband ( T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll )

DPLL configuration for wideband.

#### Parameters

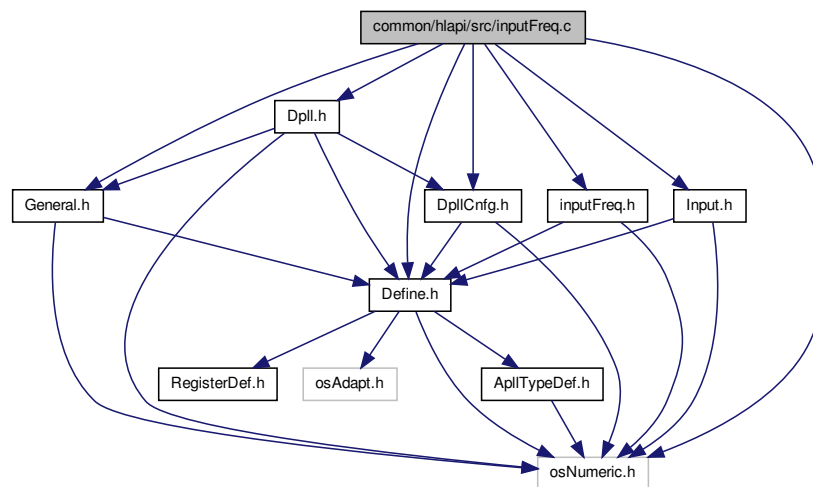
|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 523 of file profiles.c.

## 6.27 common/hlapi/src/inputFreq.c File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "Input.h"
#include "General.h"
#include "DpllCnfg.h"
#include "Dpll.h"
#include "inputFreq.h"
```

Include dependency graph for inputFreq.c:



### Data Structures

- struct [T\\_IDTFreq2bfVal](#)  
Structure to contain translation information for frequency to bitfield value.
- struct [T\\_idtHfDivEntry](#)  
Structure to use as high frequency (HF) divider information.

### Functions

- void [IDTHIApi\\_ConfigureInput1PPS](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)

*Function to provision input port for 1PPS (1 Hz)*

- void [IDTHIApi\\_ConfigureInputAmi](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, [T\\_idtAmiInputFrequencyBitField-Value](#) amiSignal)

*Function to provision input AMI port.*

- void [IDTHIApi\\_ConfigureInputFrequency](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, T\_osUInt32 nFrequency)

*Function to provision input port.*

- void [IDTHIApi\\_DisableAllInputPriorities](#) (T\_idtDrvHdlr hdlr)

*Function to disable all inputs priorities.*

## 6.27.1 Function Documentation

### 6.27.1.1 void IDTHIApi\_ConfigureInput1PPS ( T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo )

Function to provision input port for 1PPS (1 Hz)

#### Parameters

|                 |                   |
|-----------------|-------------------|
| <i>hdlr</i>     | driver handler    |
| <i>nInputNo</i> | input port number |

Definition at line 180 of file inputFreq.c.

### 6.27.1.2 void IDTHIApi\_ConfigureInputAmi ( T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, T\_idtAmiInputFrequencyBitField-Value amiSignal )

Function to provision input AMI port.

#### Parameters

|                  |                                                                              |
|------------------|------------------------------------------------------------------------------|
| <i>hdlr</i>      | driver handler                                                               |
| <i>nInputNo</i>  | input port number                                                            |
| <i>amiSignal</i> | Selected AMI signal /see <a href="#">T_idtAmiInputFrequencyBitFieldValue</a> |

Definition at line 222 of file inputFreq.c.

### 6.27.1.3 void IDTHIApi\_ConfigureInputFrequency ( T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, T\_osUInt32 nFrequency )

Function to provision input port.

#### Parameters

|                   |                          |
|-------------------|--------------------------|
| <i>hdlr</i>       | driver handler           |
| <i>nInputNo</i>   | input port number        |
| <i>nFrequency</i> | input frequency (in kHz) |

Definition at line 357 of file inputFreq.c.

### 6.27.1.4 void IDTHIApi\_DisableAllInputPriorities ( T\_idtDrvHdlr hdlr )

Function to disable all inputs priorities.

This is different to enabling inputs. Input priorities are used by DPLLs to select input ports to lock to. Disable input priorities tells DPLLs to ignore all inputs.

Use this function when a custom priority is desired. This functions will clear all input priorities first. Input priorities

can then be set by calling IDTInput\_SetPriority function.

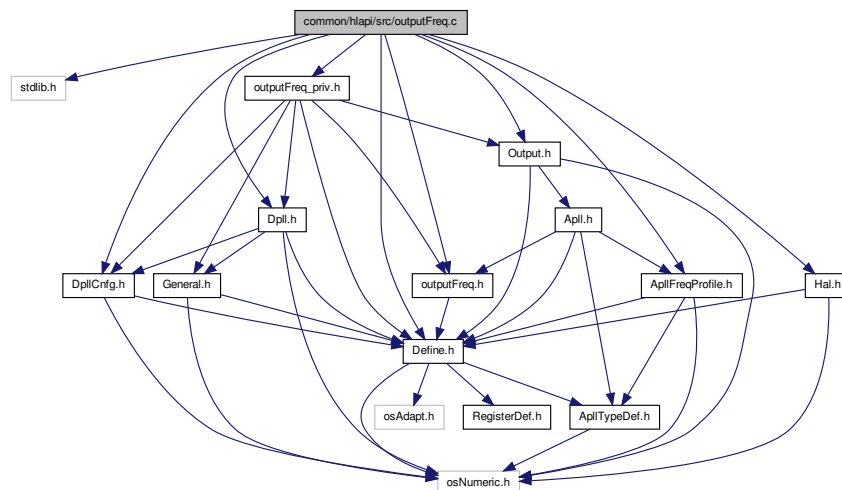
#### Parameters

|             |                      |
|-------------|----------------------|
| <i>hdlr</i> | Device driver handle |
|-------------|----------------------|

Definition at line 470 of file inputFreq.c.

## 6.28 common/hlapi/src/outputFreq.c File Reference

```
#include <stdlib.h>
#include "Define.h"
#include "Dpll.h"
#include "Hal.h"
#include "Output.h"
#include "DpllCnfg.h"
#include "outputFreq.h"
#include "outputFreq_priv.h"
#include "ApllFreqProfile.h"
Include dependency graph for outputFreq.c:
```



#### Macros

- `#define MAX_SUPPORTED_FREQUENCIES 9`
- `#define OUTMUX_ENTRY(sel, selval, div, path, inst) { Selector_##sel, selval, div, OutputPath_##path, inst }`

*Macro to reduce size of output path configuration entries. Used to reduce size of table so that it fits inside 80 columns.*

#### Functions

- `void IDTHIApi_RetrieveOutputPathMuxFromHardware (T_idtDrvHdlr hdlr, T_IDTPathConfiguration *path-Config_p)`

*Function to retrieve output path configuration from hardware.*

- int [IDTHIApi\\_ConfigureOutput](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN, [outputFrequency\\_t](#) nOutFreq, T\_idtDpll-Instance dppllInstance, T\_sonetGigeMode sonetGigeMode, T\_idtApllConfigSel apllConfSel, T\_idtApllOutputSigType apllOutputSigType, T\_idtApllInputClkSrc apllClkSource)

*Function to provision output port.*

- void [IDTHIApi\\_DisableAllOutputs](#) (T\_idtDrvHdlr hdlr)

*Function to disable all outputs.*

- void [IDTHIApi\\_ClearProvisioning](#) (T\_idtDrvHdlr hdlr)

*Function to clear saved provisioning used by high level API.*

## 6.28.1 Macro Definition Documentation

### 6.28.1.1 #define MAX\_SUPPORTED\_FREQUENCIES 9

Definition at line 53 of file outputFreq.c.

### 6.28.1.2 #define OUTMUX\_ENTRY( sel, selval, div, path, inst ) { Selector\_##sel, selval, div, OutputPath\_##path, inst }

Macro to reduce size of output path configuration entries. Used to reduce size of table so that it fits inside 80 columns.

Definition at line 61 of file outputFreq.c.

## 6.28.2 Function Documentation

### 6.28.2.1 void IDTHIApi\_ClearProvisioning ( T\_idtDrvHdlr hdlr )

Function to clear saved provisioning used by high level API.

#### Parameters

|             |                      |
|-------------|----------------------|
| <i>hdlr</i> | Device driver handle |
|-------------|----------------------|

Definition at line 1464 of file outputFreq.c.

### 6.28.2.2 int IDTHIApi\_ConfigureOutput ( T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN, [outputFrequency\\_t](#) nOutFreq, T\_idtDpllInstance dppllInstance, T\_sonetGigeMode sonetGigeMode, T\_idtApllConfigSel apllConfSel, T\_idtApllOutputSigType apllOutputSigType, T\_idtApllInputClkSrc apllClkSource )

Function to provision output port.

#### Parameters

|                      |                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------|
| <i>hdlr</i>          | driver handler                                                                                   |
| <i>nOutN</i>         | Output port (1-11)                                                                               |
| <i>nOutFreq</i>      | Output frequency /see <a href="#">outputFrequency_t</a> for list of supported frequencies        |
| <i>dppllInstance</i> | Dpll Instance (DPLL_INSTANCE_1, DPLL_INSTANCE_2, specify DPLL_INSTANCE_MAX for no specific DPLL) |
| <i>sonetGigeMode</i> | sonetGige mux configuration.                                                                     |

#### See Also

[T\\_sonetGigeMode](#) This parameter and dppllInstance determine what output path(s) software may use to generate specified output frequency. The following table illustrates the available paths,

| dppllInstance     | sonetGigeMode | Avail. Paths                                      |
|-------------------|---------------|---------------------------------------------------|
| DPLL_INSTANCE_1   | Matching      | DPLL#1->SonetGigEth#1 only                        |
| DPLL_INSTANCE_2   | Matching      | DPLL#2->SonetGigEth#2 only                        |
| DPLL_INSTANCE_MAX | Matching      | DPLL#1->SonetGigEth#1 or<br>DPLL#2->SonetGigEth#2 |
| DPLL_INSTANCE_1   | AllFromDpll1  | DPLL#1->SonetGigEth#1 only                        |
| DPLL_INSTANCE_2   | AllFromDpll1  | DPLL#1->SonetGigEth#2 only                        |
| DPLL_INSTANCE_MAX | AllFromDpll1  | DPLL#1->SonetGigEth#1 or<br>DPLL#1->SonetGigEth#2 |
| DPLL_INSTANCE_1   | AllFromDpll2  | DPLL#2->SonetGigEth#1 only                        |
| DPLL_INSTANCE_2   | AllFromDpll2  | DPLL#2->SonetGigEth#2 only                        |
| DPLL_INSTANCE_MAX | AllFromDpll2  | DPLL#2->SonetGigEth#1 or<br>DPLL#2->SonetGigEth#2 |

## Parameters

|                           |                                                                                                                                                            |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>apllConfSel</i>        | Selects the APLL configuration, set to APLL_CONFIG_MAX if output port does not go through APLL, <i>OR</i> if you want to program backup APLL configuration |
| <i>apllOutputSig-Type</i> | APLL output signal type, set to APLL_OUT_SIG_MAX if output port does not go through APLL                                                                   |
| <i>apllClkSource</i>      | APLL clock source (internal or external), set to APLL_CLK_SEL_MAX if output port does not go through APLL                                                  |

## Returns

0-OK, 1-Error, unable to configure output path mux

Definition at line 1261 of file outputFreq.c.

### 6.28.2.3 void IDTHIApi\_DisableAllOutputs ( T\_idtDrvHdlr hdlr )

Function to disable all outputs.

## Parameters

|             |                |
|-------------|----------------|
| <i>hdlr</i> | driver handler |
|-------------|----------------|

Definition at line 1443 of file outputFreq.c.

### 6.28.2.4 void IDTHIApi\_RetrieveOutputPathMuxFromHardware ( T\_idtDrvHdlr hdlr, T\_IDTPathConfiguration \* pathConfig\_p )

Function to retrieve output path configuration from hardware.

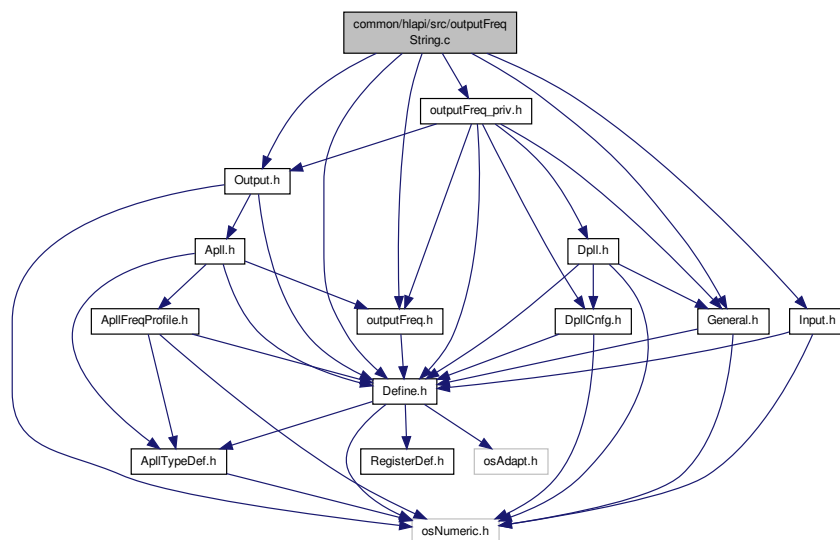
## Parameters

|                     |                           |
|---------------------|---------------------------|
| <i>hdlr</i>         | Device driver handler     |
| <i>pathConfig_p</i> | Output path configuration |

Definition at line 282 of file outputFreq.c.

## 6.29 common/hlapi/src/outputFreqString.c File Reference

```
#include "Define.h"
#include "Output.h"
#include "Input.h"
#include "General.h"
#include "outputFreq.h"
#include "outputFreq_priv.h"
Include dependency graph for outputFreqString.c:
```



## Data Structures

- struct [T\\_aplOutputFrequencyEntry](#)

Structure used to translate APLL configuration and output port frequency.

## Enumerations

- enum [T\\_outFreqLookupIdx](#) {  
[outFreqLookupIdx\\_Dpll77p76](#), [outFreqLookupIdx\\_Dpll12E1](#), [outFreqLookupIdx\\_Dpll16E1](#), [outFreqLookupIdx\\_Dpll16T1](#),  
[outFreqLookupIdx\\_DpllE3](#), [outFreqLookupIdx\\_DpllT3](#), [outFreqLookupIdx\\_DpllGsm](#), [outFreqLookupIdx\\_DpllObsai](#),  
[outFreqLookupIdx\\_DpllEth](#), [outFreqLookupIdx\\_Dpll24T1](#), [outFreqLookupIdx\\_DpllGps](#), [outFreqLookupIdx\\_SonetGigEthSonet](#),  
[outFreqLookupIdx\\_SonetGigEthEth](#), [outFreqLookupIdx\\_SonetGigEthFec](#), [outFreqLookupIdx\\_MAX](#) }

## Functions

- void [IDTHIApi\\_PrintAvailFrequencies](#) ([validFreq\\_t](#) frequencyArray)  
*Function to print frequency array in string format.*
- void [IDTHIApi\\_PrintFrequencyList](#) (const [outputFrequency\\_t](#) frequencies[])  
*Function to print frequency list in string format.*
- void [IDTHIApi\\_PrintConfiguration](#) ([T\\_IDTPathConfiguration](#) pathConfig)  
*Function to print output path configuration to a string format.*
- void [IDTHIApi\\_PrintOption](#) ([pathSelectorConfig\\_t](#) option, [T\\_osUInt8](#) printPort)  
*Function to print output path configuration option to string format.*
- void [IDTHIApi\\_PrintSelectedOutputPath](#) ([T\\_outputPathType](#) nPath)  
*Function to display selected output port path.*
- void [IDTHIApi\\_PrintOutputHwConfig](#) ([networkType\\_t](#) network, [T\\_osUInt8](#) nOutN, [T\\_osUInt8](#) internalPort, [T\\_osUInt8](#) isEnabled, [T\\_osUInt8](#) hasApI, [T\\_idtApIDividerValue](#) apIldivVal, [T\\_idtApIXtalType](#) xtalType, [T\\_osUInt8](#) outDiv, [T\\_outputPathType](#) outputPath, [T\\_idtDpllInstance](#) dpllInstOrSonetGigEthInst, [T\\_idtDpllInstance](#) sonetGigEthInput, [T\\_idtSonetGigEthOutput](#) sonetGigEMode, [T\\_idtDpllOutputSelectorA](#) dpllSelA, [T\\_idtDpllOutputSelectorB](#) dpllSelB)
- void [IDTHIApi\\_PrintInputHwConfig](#) ([T\\_osUInt8](#) nInputN, [T\\_osUInt8](#) internalPort, [networkType\\_t](#) networkType, [T\\_osUInt8](#) isEnabled, [T\\_idtHighFrequencyDivisor](#) nUsage, [T\\_osUInt16](#) nFactor, [T\\_idtInputDividerMux](#) nDivider, [T\\_idtInputFrequencyBitfieldValue](#) nFrequency, [T\\_osUInt8](#) numberOfDpll, [T\\_osUInt8](#) priorities[])
- void [IDTHIApi\\_PrintInput](#) ([T\\_idtDrvHdlr](#) hdlr)
- void [IDTHIApi\\_PrintOutput](#) ([T\\_idtDrvHdlr](#) hdlr)

## Variables

- const char \* [IDTHIApi\\_Frequency2String\\_c](#) [[OutFreq\\_MAX](#)]

## 6.29.1 Enumeration Type Documentation

### 6.29.1.1 enum [T\\_outFreqLookupIdx](#)

#### Enumerator

**[outFreqLookupIdx\\_Dpll77p76](#)** DPLL 77.76MHz  
**[outFreqLookupIdx\\_Dpll12E1](#)** DPLL 12E1  
**[outFreqLookupIdx\\_Dpll16E1](#)** DPLL 16E1  
**[outFreqLookupIdx\\_Dpll16T1](#)** DPLL 16T1  
**[outFreqLookupIdx\\_DpllE3](#)** DPLL E3  
**[outFreqLookupIdx\\_DpllT3](#)** DPLL T3  
**[outFreqLookupIdx\\_DpllGsm](#)** DPLL GSM  
**[outFreqLookupIdx\\_DpllObsai](#)** DPLL OBSAI  
**[outFreqLookupIdx\\_DpllEth](#)** DPLL 24T1  
**[outFreqLookupIdx\\_Dpll24T1](#)** DPLL 24T1  
**[outFreqLookupIdx\\_DpllGps](#)** DPLL GPS  
**[outFreqLookupIdx\\_SonetGigEthSonet](#)** SONET/GIGE SONET  
**[outFreqLookupIdx\\_SonetGigEthEth](#)** SONET/GIGE ETH  
**[outFreqLookupIdx\\_SonetGigEthFec](#)** SONET/GIGE FEC  
**[outFreqLookupIdx\\_MAX](#)**

Definition at line 45 of file [outputFreqString.c](#).

## 6.29.2 Function Documentation

### 6.29.2.1 void IDTHIApi\_PrintAvailFrequencies ( validFreq\_t frequencyArray )

Function to print frequency array in string format.

#### Parameters

|                       |                 |
|-----------------------|-----------------|
| <i>frequencyArray</i> | Frequency array |
|-----------------------|-----------------|

Definition at line 317 of file outputFreqString.c.

### 6.29.2.2 void IDTHIApi\_PrintConfiguration ( T\_IDTPathConfiguration pathConfig )

Function to print output path configuration to a string format.

#### Parameters

|                   |                    |
|-------------------|--------------------|
| <i>pathConfig</i> | Output path config |
|-------------------|--------------------|

Definition at line 358 of file outputFreqString.c.

### 6.29.2.3 void IDTHIApi\_PrintFrequencyList ( const outputFrequency\_t frequencies[] )

Function to print frequency list in string format.

#### Parameters

|                    |                |
|--------------------|----------------|
| <i>frequencies</i> | Frequency list |
|--------------------|----------------|

Definition at line 339 of file outputFreqString.c.

### 6.29.2.4 void IDTHIApi\_PrintInput ( T\_idtDrvHdlr hdlr )

Definition at line 948 of file outputFreqString.c.

### 6.29.2.5 void IDTHIApi\_PrintInputHwConfig ( T\_osUInt8 nInputN, T\_osUInt8 internalPort, networkType\_t networkType, T\_osUInt8 isEnabled, T\_idtHighFrequencyDivisor nUsage, T\_osUInt16 nFactor, T\_idtInputDividerMux nDivider, T\_idtInputFrequencyBitfieldValue nFrequency, T\_osUInt8 numberOfDpll, T\_osUInt8 priorities[] )

Definition at line 773 of file outputFreqString.c.

### 6.29.2.6 void IDTHIApi\_PrintOption ( pathSelectorConfig\_t option, T\_osUInt8 printPort )

Function to print output path configuration option to string format.

#### Parameters

|                  |                                                             |
|------------------|-------------------------------------------------------------|
| <i>option</i>    | Output path configuration option.                           |
| <i>printPort</i> | Flag to indicate whether port information is also displayed |

Definition at line 387 of file outputFreqString.c.

#### 6.29.2.7 void IDTHIApi\_PrintOutput ( T\_idtDrvHdlr *hdlr* )

Definition at line 1011 of file outputFreqString.c.

#### 6.29.2.8 void IDTHIApi\_PrintOutputHwConfig ( networkType\_t *network*, T\_osUInt8 *nOutN*, T\_osUInt8 *internalPort*, T\_osUInt8 *isEnabled*, T\_osUInt8 *hasAppl*, T\_idtApIIDividerValue *apIIDivVal*, T\_idtApIIXtalType *xtalType*, T\_osUInt8 *outDiv*, T\_outputPathType *outPath*, T\_idtDpIIInstance *dpIIInstOrSonetGigEthInst*, T\_idtDpIIInstance *sonetGigEthInput*, T\_idtSonetGigEthOutput *sonetGigEthMode*, T\_idtDpIIOutputSelectorA *dpIISelA*, T\_idtDpIIOutputSelectorB *dpIISelB* )

Definition at line 581 of file outputFreqString.c.

#### 6.29.2.9 void IDTHIApi\_PrintSelectedOutputPath ( T\_outputPathType *nPath* )

Function to display selected output port path.

##### Parameters

|              |                  |
|--------------|------------------|
| <i>nPath</i> | output port path |
|--------------|------------------|

Definition at line 443 of file outputFreqString.c.

### 6.29.3 Variable Documentation

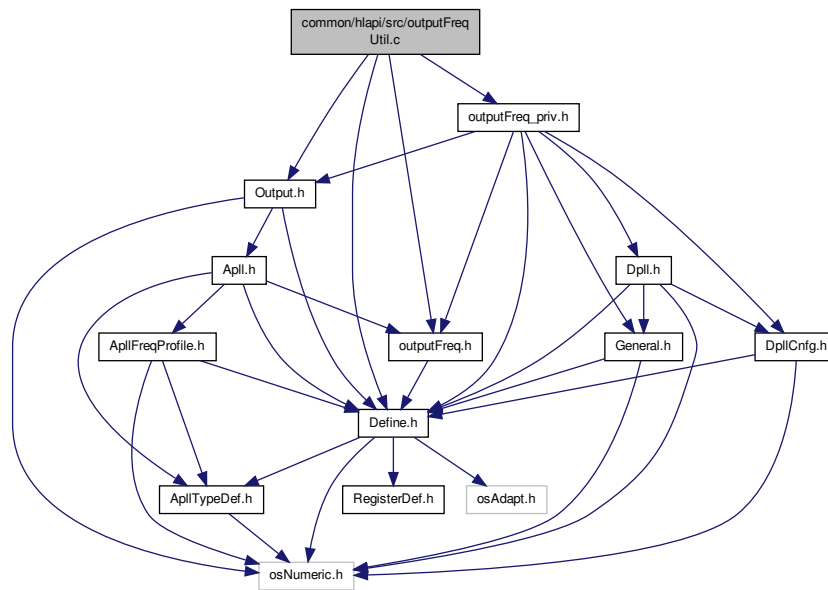
#### 6.29.3.1 const char\* IDTHIApi\_Frequency2String\_c[OutFreq\_MAX]

Definition at line 77 of file outputFreqString.c.

## 6.30 common/hlapi/src/outputFreqUtil.c File Reference

```
#include "Define.h"
#include "Output.h"
#include "outputFreq.h"
#include "outputFreq_priv.h"
```

Include dependency graph for outputFreqUtil.c:



## Functions

- void [IDTHIApi\\_ConvertFromFreqListToFreqArray](#) ([outputFrequency\\_t](#) frequencies[], [validFreq\\_t](#) validFrequencies)  
*Convert from a frequency list to frequency array.*
- void [IDTHIApi\\_ApplyFrequencies](#) (const [outputFrequency\\_t](#) \*frequencies, [validFreq\\_t](#) validFrequencies, int value)  
*Function to set status of frequency in frequency array.*
- int [IDTHIApi\\_IsFreqArrayInFreqList](#) (const [outputFrequency\\_t](#) frequencies[], [validFreq\\_t](#) validFrequencies)  
*Function to compare group of frequencies with used frequency list.*
- int [IDTHIApi\\_IsFrequencyInFreqList](#) ([outputFrequency\\_t](#) frequency, const [outputFrequency\\_t](#) validFrequencies[])  
*Function to verify that frequency is within frequency list.*

### 6.30.1 Function Documentation

6.30.1.1 void [IDTHIApi\\_ApplyFrequencies](#) ( const [outputFrequency\\_t](#) \* frequencies, [validFreq\\_t](#) validFrequencies, int value )

Function to set status of frequency in frequency array.

#### Parameters

|                         |                                   |
|-------------------------|-----------------------------------|
| <i>frequencies</i>      | Constant frequency list           |
| <i>validFrequencies</i> |                                   |
| <i>value</i>            | Status to set for frequency array |

Definition at line 79 of file outputFreqUtil.c.

6.30.1.2 void IDTHIApi.ConvertFromFreqListToFreqArray ( outputFrequency\_t *frequencies*[], validFreq\_t *validFrequencies* )

Convert from a frequency list to frequency array.

#### Parameters

|                         |                           |
|-------------------------|---------------------------|
| <i>frequencies</i>      | Frequency list to convert |
| <i>validFrequencies</i> | Output frequency array    |

Definition at line 57 of file outputFreqUtil.c.

6.30.1.3 int IDTHIApi.IsFreqArrayInFreqList ( const outputFrequency\_t *frequencies*[], validFreq\_t *validFrequencies* )

Function to compare group of frequencies with used frequency list.

#### Parameters

|                         |                                      |
|-------------------------|--------------------------------------|
| <i>frequencies</i>      | List to frequencies to check for     |
| <i>validFrequencies</i> | List of frequencies to check against |

#### Returns

0-Not in frequency list, 1-All in frequency list

Definition at line 100 of file outputFreqUtil.c.

6.30.1.4 int IDTHIApi.IsFrequencyInFreqList ( outputFrequency\_t *frequency*, const outputFrequency\_t *validFrequencies*[] )

Function to verify that frequency is within frequency list.

#### Parameters

|                         |                                 |
|-------------------------|---------------------------------|
| <i>frequency</i>        | Frequency to check              |
| <i>validFrequencies</i> | Frequency list to check against |

#### Returns

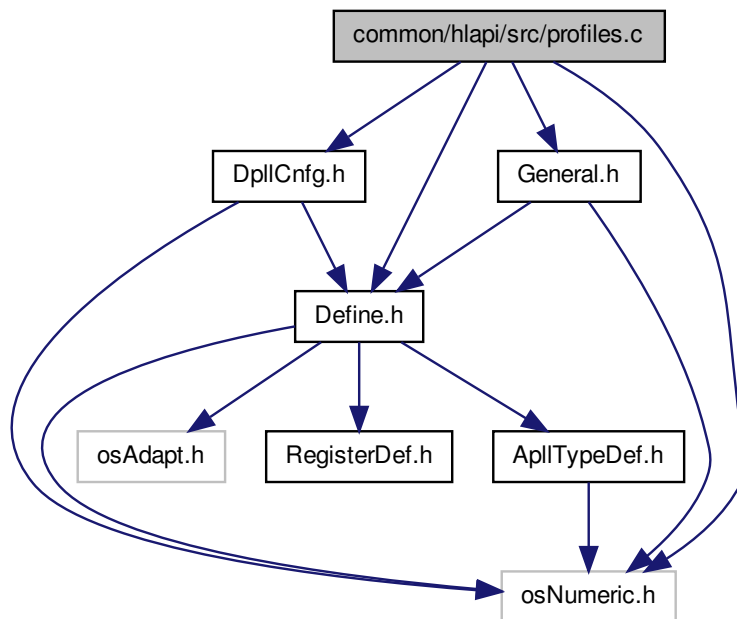
1-Frequency is in list, 0-Frequency is not in list

Definition at line 124 of file outputFreqUtil.c.

## 6.31 common/hlapi/src/profiles.c File Reference

```
#include "Define.h"
#include "osNumeric.h"
#include "DpllCnfg.h"
#include "General.h"
```

Include dependency graph for profiles.c:



## Functions

- void [IDTHIApi\\_gr253SonetStratum3](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
DPLL configuration for GR-253 (SONET) Stratum 3.
- void [IDTHIApi\\_gr1244Stratum3](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
DPLL configuration for GR-1244 Stratum 3.
- void [IDTHIApi\\_smc](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
DPLL configuration for SMC.
- void [IDTHIApi\\_g8262EecOption1](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
DPLL configuration for G-8262 Option 1.
- void [IDTHIApi\\_g8262EecOption2](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
DPLL configuration for G-8262 Option 2.
- void [IDTHIApi\\_g813Option1](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
DPLL configuration for G-813 Option 1.
- void [IDTHIApi\\_g813Option2](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
DPLL configuration for G-813 Option 2.
- void [IDTHIApi\\_pps](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
DPLL configuration for 1 pulse per second (PPS)
- void [IDTHIApi\\_wideband](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
DPLL configuration for wideband.

### 6.31.1 Detailed Description

This sample file contains DPLL configurations used to conform to various standards.

Definition in file [profiles.c](#).

### 6.31.2 Function Documentation

#### 6.31.2.1 void IDTHIApi\_g813Option1 ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

DPLL configuration for G-813 Option 1.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 396 of file profiles.c.

#### 6.31.2.2 void IDTHIApi\_g813Option2 ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

DPLL configuration for G-813 Option 2.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 408 of file profiles.c.

#### 6.31.2.3 void IDTHIApi\_g8262EecOption1 ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

DPLL configuration for G-8262 Option 1.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 258 of file profiles.c.

#### 6.31.2.4 void IDTHIApi\_g8262EecOption2 ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

DPLL configuration for G-8262 Option 2.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 327 of file profiles.c.

#### 6.31.2.5 void IDTHIApi\_gr1244Stratum3 ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )

DPLL configuration for GR-1244 Stratum 3.

##### Parameters

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 120 of file profiles.c.

**6.31.2.6 void IDTHIApi\_gr253SonetStratum3 ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )**

DPLL configuration for GR-253 (SONET) Stratum 3.

**Parameters**

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 51 of file profiles.c.

**6.31.2.7 void IDTHIApi\_pps ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )**

DPLL configuration for 1 pulse per second (PPS)

**Parameters**

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 421 of file profiles.c.

**6.31.2.8 void IDTHIApi\_smc ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )**

DPLL configuration for SMC.

**Parameters**

|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 189 of file profiles.c.

**6.31.2.9 void IDTHIApi\_wideband ( T\_idtDrvHdlr *hdlr*, T\_idtDpllInstance *nDpll* )**

DPLL configuration for wideband.

**Parameters**

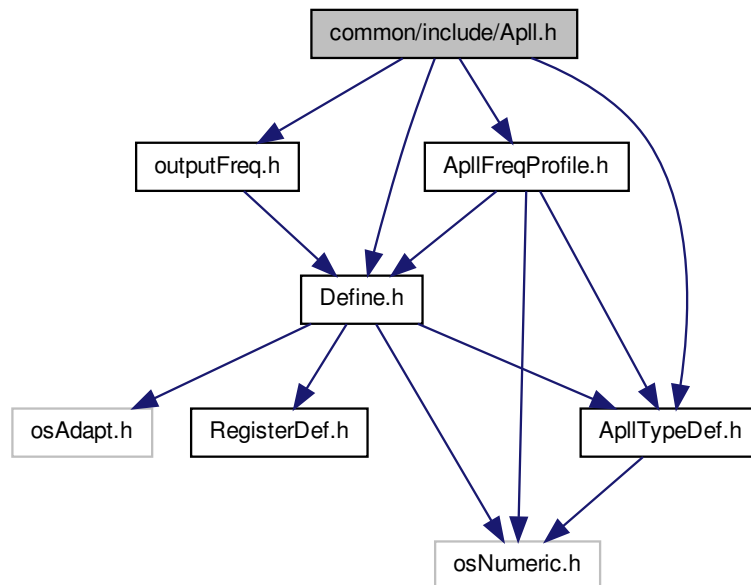
|              |                       |
|--------------|-----------------------|
| <i>hdlr</i>  | Device driver handler |
| <i>nDpll</i> | DPLL instance         |

Definition at line 523 of file profiles.c.

**6.32 common/include/Apll.h File Reference**

```
#include "Define.h"
#include "outputFreq.h"
#include "ApllTypeDef.h"
#include "ApllFreqProfile.h"
```

Include dependency graph for ApI.h:



## Data Structures

- struct [T\\_idtApIConfigPerOutput](#)  
*Structure containing APLL per output configuration.*
- struct [T\\_idtApIConfig](#)  
*Structure containing APLL full configuration.*
- struct [T\\_idtApIConfig::preDiv\\_s](#)
- struct [T\\_idtApIConfig::fbDiv\\_s](#)
- struct [T\\_idtApIConfig::outputDiv\\_s](#)
- struct [T\\_idtApIConfig::outputDiv\\_s::qaDiv\\_s](#)
- struct [T\\_idtApIConfig::outputDiv\\_s::qbDiv\\_s](#)
- struct [T\\_idtApIConfig::outputConfig\\_s](#)
- struct [T\\_idtApIConfig::outputConfig\\_s::qaConfig\\_s](#)
- struct [T\\_idtApIConfig::outputConfig\\_s::qbConfig\\_s](#)

## Macros

- `#define IDT_GET_OUTPUT_APLL_CONFIG(hdlr, output, outputApI)`  
*Utility macro to translate output port to APLL instance.*

## Functions

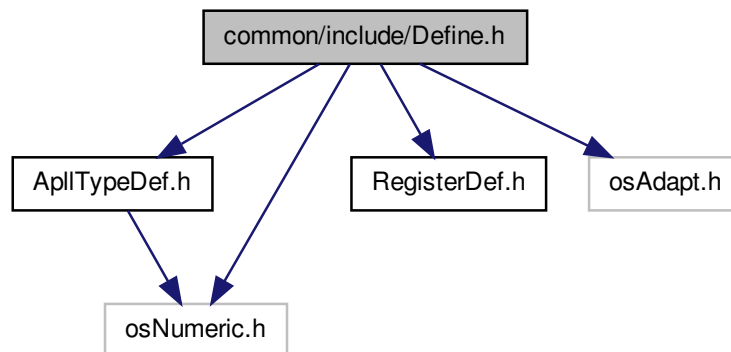
- void [IDTApI\\_SetApIConfigPerOutput](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_idtApIIId](#) apIIId, [T\\_idtApIOutput](#) output, [T\\_idtApIConfigPerOutput](#) \*apIConfigPerOutput)  
*Set APLL per output configuration.*

- void `IDTApll_GetApllConfigPerOutput` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllId, `T_idtApllOutput` output, `T_idtApllConfigPerOutput` \*apllConfigPerOutput)  
*Get APLL per output configuration.*
- void `IDTApll_SetApllConfig` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllId, `T_idtApllConfig` \*apllConfig)  
*Set APLL full configuration.*
- void `IDTApll_GetApllConfig` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllId, `T_idtApllConfig` \*apllConfig)  
*Get APLL full configuration.*
- `outputFrequency_t` `IDTApll_ApllInput2DpllOutput` (`T_idtApllInputFreqType` apllInputFreq)  
*Translate APLL input frequency to DPLL output frequency.*
- `T_idtApllOutputFreqType` `IDTApll_DpllOutput2ApllOutput` (`outputFrequency_t` dpllOutput)  
*Translate DPLL output frequency to APLL output frequency.*
- const `T_idtApllFreqMapping` \* `IDTApll_GetApllFreqTableEntry` (`T_idtDrvHdlr` hdlr, `T_idtOutputApllConfig` outputApll, `outputFrequency_t` nOutFreq)  
*Get APLL frequency configuration table entry based on the output frequency.*
- `T_idtApllConfigSel` `IDTApll_GetNonActiveApllConfig` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllId)  
*Get non-active APLL configuration.*
- void `IDTApll_Switch2NonActiveApllConfig` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllId)  
*Switch to non-active APLL configuration.*
- `T_idtApllXtalId` `IDTApll_GetXtalIdFromXtalFreq` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllId, `T_idtApllXtalType` apllVcxoFreq)  
*Get APLL configured crystal ID from crystal frequency.*
- `T_osBool` `IDTApll_XtalConfigured` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllId)  
*Checks if there is any XTAL configured for the APLL.*
- `T_osBool` `IDTApll_ValidXtalInput` (const `T_idtDrvDeviceConfig` \*deviceConfig, `T_idtApllXtal` \*xtalList, `T_osUInt8` numberOfXtal)  
*Checks if there is input XTAL information is valid for the device.*
- `T_idtApllXtal` \* `IDTApll_GetSupportedXtal` (const `T_idtDrvDeviceConfig` \*deviceConfig, `T_osUInt8` \*numberOfXtal)  
*Return supported xtal configuration by the device.*

## 6.33 common/include/Define.h File Reference

```
#include "ApllTypeDef.h"
#include "RegisterDef.h"
#include "osAdapt.h"
#include "osNumeric.h"
```

Include dependency graph for Define.h:



## Data Structures

- struct [T\\_idtDrvAccess](#)  
*Initialization structure detailing device access information for device driver.*
- struct [T\\_idtOutputApIITConfig](#)  
*Structure containing APLL information.*
- struct [T\\_idtDrvDeviceConfig](#)  
*Device type/variant information.*
- struct [T\\_idtDrvHdlr](#)  
*Device driver handler.*

## Macros

- `#define MPU_I2C_MODE 0x00`
- `#define MPU_SERIAL_MODE 0x01`
- `#define NULL_APLL_CONFIG`  
*Utility macro to set APLL configuration to NULL for the output path that does not have APLL.*
- `#define APLL_CONFIGURED(apIITConfig)`  
*Utility macro to check if APLL is configured.*
- `#define IDT_DRV_MAX_INPUT 14`  
*Define maximum number of supported inputs.*
- `#define IDT_DRV_MAX_OUTPUT 11`  
*Define maximum number of supported outputs.*
- `#define IDT_DRV_MAX_APLL_XTAL 4`  
*Define maximum number of APLL XTALs per device.*
- `#define IDT_DRV_MAX_XTAL_PER_APLL 2`  
*Define maximum number of XTALs per APLL.*

## Typedefs

- typedef T\_osUInt8(\* [T\\_idtReadFunc](#) )(void \*usrData, T\_osUInt8 deviceAddr, T\_osUInt8 addrOff)  
*Function type for reading from i2c device.*
- typedef void(\* [T\\_idtWriteFunc](#) )(void \*usrData, T\_osUInt8 deviceAddr, T\_osUInt8 addrOff, T\_osUInt8 data)  
*Function type to writing to i2c device.*
- typedef void(\* [T\\_idtAccessInitFunc](#) )(void \*userData)  
*Function type to initialize device access.*
- typedef void(\* [T\\_idtAccessDeInitFunc](#) )(void \*userData)  
*Function type to deintialize device access.*
- typedef T\_osUInt8(\* [T\\_idtI2CAddrTransFunc](#) )(T\_osUInt8 i2cAddr, void \*transData)  
*Function type to translate i2c address used internally by the driver.*
- typedef T\_osBool(\* [T\\_idtInputFreqValid](#) )(T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, T\_osUInt32 nFrequency, void \*auxData)  
*Function type to sanity check the input frequency.*
- typedef T\_osBool(\* [T\\_idtOuputFreqValid](#) )(T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN, T\_osUInt32 nOutFreq, void \*auxData)  
*Function type to sanity check the output frequency.*

## Enumerations

- enum [T\\_idtDpllType](#) { [Dpll\\_T0](#) = 0, [Dpll\\_T4](#) = 1, [Dpll\\_MAX](#) }  
*Enumerated type for selecting DPLLs (T0 or T4)*
- enum [T\\_idtDeviceType](#) {  
[IDT\\_82V3910](#), [IDT\\_82V3911](#), [IDT\\_82V33930](#), [IDT\\_8V89316](#),  
[IDT\\_8V89317](#), [IDT\\_DEVICE\\_MAX](#) }  
*Enumerated type listing supported device(s)*
- enum [T\\_idtDpllInstance](#) { [DPLL\\_INSTANCE\\_1](#), [DPLL\\_INSTANCE\\_2](#), [DPLL\\_INSTANCE\\_MAX](#) }  
*Define maximum number of supported DPLLs.*

### 6.33.1 Macro Definition Documentation

#### 6.33.1.1 #define APLL\_CONFIGURED( *apllConfig* )

##### Value:

```
((apllConfig.apllInst != APLL\_INST\_MAX) && \
 (apllConfig.apllInput) && \
 (apllConfig.apllOutput != APLL\_OUTPUT\_MAX) && \
 (apllConfig.extOutput)) \
? 1 : 0
```

Utility macro to check if APLL is configured.

##### Parameters

|                   |                    |
|-------------------|--------------------|
| <i>apllConfig</i> | APLL configuration |
|-------------------|--------------------|

Definition at line 81 of file Define.h.

#### 6.33.1.2 #define IDT\_DRV\_MAX\_APLL\_XTAL 4

Define maximum number of APLL XTALs per device.

Definition at line 110 of file Define.h.

**6.33.1.3 #define IDT\_DRV\_MAX\_INPUT 14**

Define maximum number of supported inputs.

Definition at line 104 of file Define.h.

**6.33.1.4 #define IDT\_DRV\_MAX\_OUTPUT 11**

Define maximum number of supported outputs.

Definition at line 107 of file Define.h.

**6.33.1.5 #define IDT\_DRV\_MAX\_XTAL\_PER\_APLL 2**

Define maximum number of XTALs per APLL.

Definition at line 113 of file Define.h.

**6.33.1.6 #define MPU\_I2C\_MODE 0x00**

Definition at line 62 of file Define.h.

**6.33.1.7 #define MPU\_SERIAL\_MODE 0x01**

Definition at line 63 of file Define.h.

**6.33.1.8 #define NULL\_APLL\_CONFIG****Value:**

```
{
    APLL_INST_MAX,
    0,
    APLL_OUTPUT_MAX,
    0,
}
```

Utility macro to set APLL configuration to NULL for the output path that does not have APLL.

Definition at line 69 of file Define.h.

**6.33.2 Typedef Documentation****6.33.2.1 typedef void(\* T\_idtAccessDeInitFunc)(void \*userData)**

Function type to deinitialize device access.

Optional binding for device access

**Parameters**

|                 |                                                        |
|-----------------|--------------------------------------------------------|
| <i>userData</i> | optional user data passed to deinitialization function |
|-----------------|--------------------------------------------------------|

Definition at line 156 of file Define.h.

**6.33.2.2 typedef void(\* T\_idtAccessInitFunc)(void \*userData)**

Function type to initialize device access.

Optional binding for device access

**Parameters**

|                 |                                                      |
|-----------------|------------------------------------------------------|
| <i>userData</i> | optional user data passed to initialization function |
|-----------------|------------------------------------------------------|

Definition at line 147 of file Define.h.

**6.33.2.3 typedef T\_osUInt8(\* T\_idtI2CAddrTransFunc)(T\_osUInt8 i2cAddr, void \*transData)**

Function type to translate i2c address used internally by the driver.

**Parameters**

|                  |                                                   |
|------------------|---------------------------------------------------|
| <i>i2cAddr</i>   | 7-bit unshifted i2c slave address for the WAN PLL |
| <i>transData</i> | Helper data for translating                       |

**Returns**

Translated i2c address used internally by the device driver

Definition at line 164 of file Define.h.

**6.33.2.4 typedef T\_osBool(\* T\_idtInputFreqValid)(T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, T\_osUInt32 nFrequency, void \*auxData)**

Function type to sanity check the input frequency.

**Parameters**

|                   |                           |
|-------------------|---------------------------|
| <i>hdlr</i>       | driver handler            |
| <i>nInputNo</i>   | input port number         |
| <i>nFrequency</i> | input frequency (in Hz)   |
| <i>auxData</i>    | auxiliary data (optional) |

**Returns**

E\_osTrue - nFrequency is supported, E\_osFalse - nFrequency is not supported

Definition at line 175 of file Define.h.

**6.33.2.5 typedef T\_osBool(\* T\_idtOutputFreqValid)(T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN, T\_osUInt32 nOutFreq, void \*auxData)**

Function type to sanity check the output frequency.

**Parameters**

|                 |                                                                           |
|-----------------|---------------------------------------------------------------------------|
| <i>hdlr</i>     | driver handler                                                            |
| <i>nOutN</i>    | output port number                                                        |
| <i>nOutFreq</i> | Output frequency /see outputFrequency_t for list of supported frequencies |
| <i>auxData</i>  | auxiliary data (optional)                                                 |

**Returns**

E\_osTrue - nOutFreq is supported, E\_osFalse - nOutFreq is not supported

Definition at line 187 of file Define.h.

**6.33.2.6 typedef T\_osUInt8(\* T\_idtReadFunc)(void \*usrData, T\_osUInt8 deviceAddr, T\_osUInt8 addrOff)**

Function type for reading from i2c device.

**Parameters**

|                   |                                                               |
|-------------------|---------------------------------------------------------------|
| <i>usrData</i>    | optional data passed to write function                        |
| <i>deviceAddr</i> | device address for bus that requires slave address (e.g. i2c) |
| <i>addrOff</i>    | address offset                                                |

**Returns**

read data

Definition at line 129 of file Define.h.

**6.33.2.7 typedef void(\* T\_idtWriteFunc)(void \*usrData, T\_osUInt8 deviceAddr, T\_osUInt8 addrOff, T\_osUInt8 data)**

Function type to writing to i2c device.

**Parameters**

|                   |                                                |
|-------------------|------------------------------------------------|
| <i>usrData</i>    | optional user data passed to read function     |
| <i>deviceAddr</i> | for bus that requires slave address (e.g. i2c) |
| <i>addrOff</i>    | address offset                                 |
| <i>data</i>       | data to write                                  |

Definition at line 138 of file Define.h.

**6.33.3 Enumeration Type Documentation****6.33.3.1 enum T\_idtDeviceType**

Enumerated type listing supported device(s)

**Enumerator**

**IDT\_82V3910** 82V3910  
**IDT\_82V3911** 82V3911  
**IDT\_82V33930** 82V33930  
**IDT\_8V89316** 8V89316  
**IDT\_8V89317** 8V89317  
**IDT\_DEVICE\_MAX**

Definition at line 93 of file Define.h.

**6.33.3.2 enum T\_idtDpllInstance**

Define maximum number of supported DPLLs.

## Enumerator

***DPLL\_INSTANCE\_1*** DPLL instance #1  
***DPLL\_INSTANCE\_2*** DPLL instance #2  
***DPLL\_INSTANCE\_MAX***

Definition at line 116 of file Define.h.

## 6.33.3.3 enum T\_idtDpllType

Enumerated type for selecting DPLLs (T0 or T4)

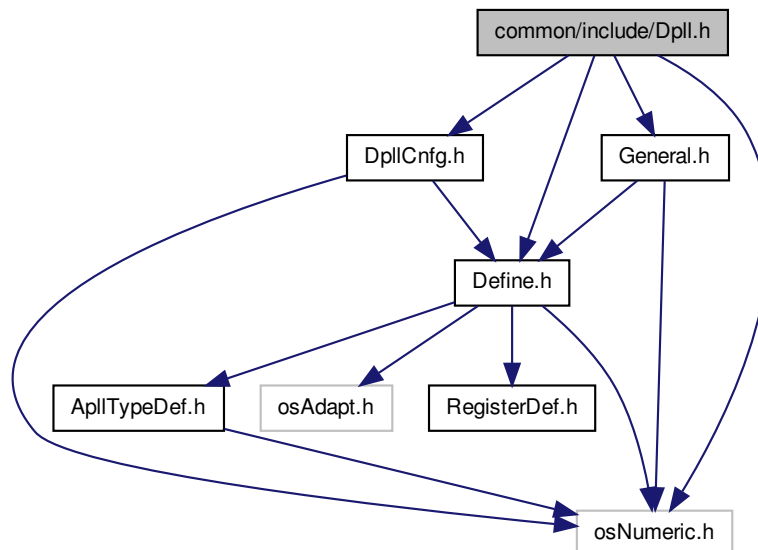
## Enumerator

***Dpll\_T0*** Select T0 DPLL  
***Dpll\_T4*** Select T4 DPLL  
***Dpll\_MAX***

Definition at line 53 of file Define.h.

## 6.34 common/include/Dpll.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "DpllCnfg.h"
#include "General.h"
Include dependency graph for Dpll.h:
```



## Data Structures

- struct [T\\_IDTPathConfiguration](#)

*Structure containing output path configuration.*

## Macros

- `#define SonetGigEthSel_NUMMAX 2`

*Constant defining number of instances of SonetGigEth selectors.*

## Enumerations

- enum `T_idtT0DpllSelectorA` {  
`T0DpllSelA_16E1`, `T0DpllSelA_16T1`, `T0DpllSelA_GSM`, `T0DpllSelA_OBSAI`,  
`T0DpllSelA_MAX` }

*Enumerated type listing T0 DPLL selector A options.*

- enum `T_idtT0DpllSelectorB` {  
`T0DpllSelB_12E1`, `T0DpllSelB_GPS`, `T0DpllSelB_E3`, `T0DpllSelB_T3`,  
`T0DpllSelB_MAX` }

*Enumerated type listing T0 DPLL selector B options.*

- enum `T_idtT4DpllSelectorA` {  
`T4DpllSelA_16E1`, `T4DpllSelA_16T1`, `T4DpllSelA_GSM`, `T4DpllSelA_GPS`,  
`T4DpllSelA_MAX` }

*Enumerated type listing T4 DPLL selector A options.*

- enum `T_idtT4DpllSelectorB` {  
`T4DpllSelB_12E1`, `T4DpllSelB_24T1`, `T4DpllSelB_E3`, `T4DpllSelB_T3`,  
`T4DpllSelB_MAX` }

*Enumerated type listing T4 DPLL selector B options.*

- enum `T_idtSonetGigEthSelector` {  
`SonetGigEthSel_T0DpllSonet` = 0, `SonetGigEthSel_T4DpllSonet` = 4, `SonetGigEthSel_T0Dpll10GbE` = 8,  
`SonetGigEthSel_T0Dpll10GbE_6664` = 9,  
`SonetGigEthSel_T4Dpll10GbE` = 12, `SonetGigEthSel_T4Dpll10GbE_6664` = 13, `SonetGigEthSel_MAX` }

*Enumerated type listing Sonet/GigE path selection options.*

## Variables

- const `T_IDTPathConfiguration IDTDpll_pathConfigurationNULL_c`

*Output path configuration no path constant.*

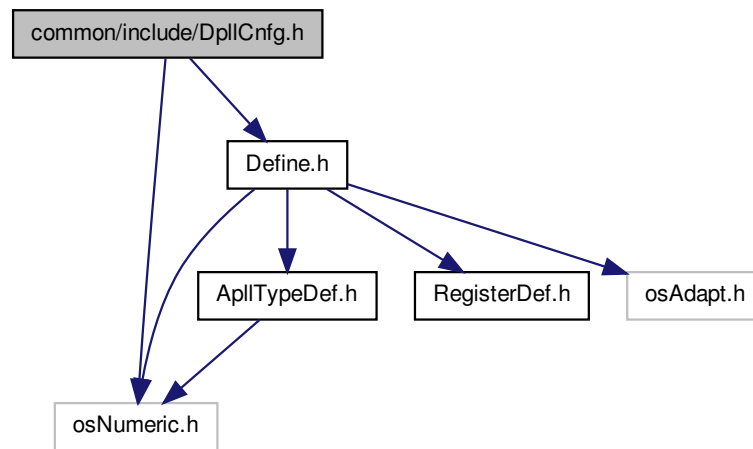
- const `T_idtSonetGigEthSelector IDTDpll_sonetGigEthPathConfig2Bitfield [Dpll_MAX][SonetGigEthOutput - MAX]`

*Table to translate from Sonet/GigE path configuration to bitfield value.*

## 6.35 common/include/DpllCnfg.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
```

Include dependency graph for DpllCnfg.h:



## Macros

- `#define DCO_PPT2DCOUNTS 11`  
*Conversion from parts per trillion to register unit value for DCO offset.*
- `#define DCO_MAX_PPT 92274677`  
*Max DCO value.*
- `#define DCO_MIN_PPT -92274688`  
*Max DCO value.*
- `#define IDT_TRANSLATE_EXT_TO_INT_DPLL(hdlr, ext, internal)`  
*Utility macro to check if DPLL instance is supported and to translated from DPLL instance to DPLL type.*

## Enumerations

- `enum T_idtFreqMonFactor {`  
`FreqMonFactor_0p0032 = 0, FreqMonFactor_0p0064 = 1, FreqMonFactor_0p0127 = 2, FreqMonFactor_0p0257 = 3,`  
`FreqMonFactor_0p0514 = 4, FreqMonFactor_0p1030 = 5, FreqMonFactor_0p2060 = 6, FreqMonFactor_0p4120 = 7,`  
`FreqMonFactor_0p8230 = 8, FreqMonFactor_1p6460 = 9, FreqMonFactor_3p2920 = 10, FreqMonFactor_3p8100 = 11,`  
`FreqMonFactor_4p6000 = 12, FreqMonFactor_MAX }`  
*Enumerated list indicating frequency monitoring factor.*
- `enum T_idtBandwidthLimitStage { dpllBandwidthLimitStart, dpllBandwidthLimitAcquire, dpllBandwidthLimitLocked, dpllBandwidthLimit_MAX }`  
*Enumerated list used for selecting bandwidth limiting stages.*
- `enum T_idtDampingFactor {`  
`DampingFactor_1p2 = 1, DampingFactor_2p5 = 2, DampingFactor_5 = 3, DampingFactor_10 = 4,`  
`DampingFactor_20 = 5, DampingFactor_MAX }`  
*Enumerated list showing options for bandwidth damping factor.*

- enum `T_idtDpllBandwidth` {  
`dpllBandwidth_0p5mHz` = 0, `dpllBandwidth_1mHz` = 1, `dpllBandwidth_2mHz` = 2, `dpllBandwidth_4mHz` = 3,  
`dpllBandwidth_8mHz` = 4, `dpllBandwidth_15mHz` = 5, `dpllBandwidth_30mHz` = 6, `dpllBandwidth_60mHz` = 7,  
`dpllBandwidth_0p1Hz` = 8, `dpllBandwidth_0p3Hz` = 9, `dpllBandwidth_0p6Hz` = 10, `dpllBandwidth_1p2Hz` = 11,  
`dpllBandwidth_2p5Hz` = 12, `dpllBandwidth_4Hz` = 13, `dpllBandwidth_8Hz` = 14, `dpllBandwidth_18Hz` = 15,  
`dpllBandwidth_35Hz` = 16, `dpllBandwidth_70Hz` = 17, `dpllBandwidth_560Hz` = 18, `dpllBandwidth_MAX` }  
*List of options for DPLL bandwidth.*
- enum `T_idtPhaseSlope` {  
`phaseSlope_gr1244st3` = 0, `phaseSlope_gr1244st3e` = 1, `phaseSlope_gr813` = 2, `phaseSlope_noLimit` = 3,  
`phaseSlope_MAX` }  
*Enumerated type listing phase slope.*
- enum `T_idtDpllOperMode` {  
`Automatic_Mode` = 0, `Force_FreeRun_Mode` = 1, `Force_Holdover_Mode` = 2, `Force_Locked_Mode` = 4,  
`Force_Pre_Locked2_Mode` = 5, `Force_Pre_Locked_Mode` = 6, `Force_Lost_Phase_Mode` = 7, `Dco_Mode` = 8,  
`DpllOperMode_MAX` }  
*Enumerated type for DPLL operation mode.*
- enum `T_idtCoarsePhaseLimit` {  
`coarsePhaseLimit_1` = 0, `coarsePhaseLimit_3` = 1, `coarsePhaseLimit_7` = 2, `coarsePhaseLimit_15` = 3,  
`coarsePhaseLimit_31` = 4, `coarsePhaseLimit_63` = 5, `coarsePhaseLimit_127` = 6, `coarsePhaseLimit_255` = 7,  
`coarsePhaseLimit_511` = 8, `coarsePhaseLimit_1023` = 9, `coarsePhaseLimit_MAX` }  
*Enumerated type listing coarse phase limits.*
- enum `T_idtFinePhaseLimit` {  
`finePhaseLimit_45_90degree` = 1, `finePhaseLimit_90_180degree` = 2, `finePhaseLimit_180_360degree` = 3,  
`finePhaseLimit_20_25nanosec` = 4,  
`finePhaseLimit_60_65nanosec` = 5, `finePhaseLimit_120_125nanosec` = 6, `finePhaseLimit_950_955nanosec` = 7,  
`finePhaseLimit_MAX` }  
*Enumerated type listing fine phase limits.*
- enum `T_idtDpllHoldover` {  
`Holdover_Instantaneous` = 0, `Holdover_Slow_Average` = 2, `Holdover_Fast_Average` = 3, `Holdover_Manual` = 4,  
`Holdover_MAX` }  
*Enumerated type listing holdover modes.*
- enum `T_idtTemp_holdover` {  
`Temp_Same_As_Full_Holdover` = 0, `Temp_Holdover_Instantaneous` = 1, `Temp_Holdover_Fast_Average` = 2,  
`Temp_Holdover_Slow_Average` = 3,  
`Temp_Holdover_MAX` }  
*Enumerated type listing temp-holdover modes.*
- enum `T_idtOperDpllMode` {  
`OperDpllMode_FreeRun` = 1, `OperDpllMode_HoldOver` = 2, `OperDpllMode_Locked` = 4, `OperDpllMode_Pre-Locked2` = 5,  
`OperDpllMode_PreLocked` = 6, `OperDpllMode_LostPhase` = 7 }  
*Enumerated type listing DPLL operational modes.*
- enum `T_idtSonetGigEthOutput` { `SonetGigEthOutput_Sonet`, `SonetGigEthOutput_10GbE`, `SonetGigEthOutput_10GbE_6664`, `SonetGigEthOutput_MAX` }  
*Enumerated type listing possible Sonet/GigE output frequencies.*
- enum `T_idtDpllOutputSelectorA` {  
`DPLLOutputSelA_16E1`, `DPLLOutputSelA_16T1`, `DPLLOutputSelA_GSM`, `DPLLOutputSelA_OBSAI`,  
`DPLLOutputSelA_GPS`, `DPLLOutputSelA_MAX` }  
*Enumerated type listing DPLL selector A paths.*
- enum `T_idtDpllOutputSelectorB` {  
`DPLLOutputSelB_12E1`, `DPLLOutputSelB_GPS`, `DPLLOutputSelB_E3`, `DPLLOutputSelB_T3`,  
`DPLLOutputSelB_24T1`, `DPLLOutputSelB_MAX` }

*Enumerated type listing DPLL selector B paths.*

- enum `T_idtDpllPpsPhasePostion` { `PpsPhasePostion_0degree`, `PpsPhasePostion_50ns`, `PpsPhasePostion_100ns`, `PpsPhasePostion_MAX` }

*Enumerated type listing PPS phase options.*

- enum `T_idtDpllPpsPulseWidth` { `PpsPulseWidth_0pt5sec`, `PpsPulseWidth_10ns`, `PpsPulseWidth_200ns`, `PpsPulseWidth_400ns`, `PpsPulseWidth_800ns`, `PpsPulseWidth_1us`, `PpsPulseWidth_20us`, `PpsPulseWidth_50us`, `PpsPulseWidth_100us`, `PpsPulseWidth_200us`, `PpsPulseWidth_1pt2ms`, `PpsPulseWidth_800us`, `PpsPulseWidth_MAX` }

*Enumerated type listing PPS pulse width.*

## Functions

- `T_idtDpllType IDTDpll_GetTypeOfDpll` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`)  
*Function to determine type of DPLL give a DPLL instance.*
- `void IDTDpll_SetAutoSelInput` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`, `T_osUInt8` `mode`)  
*Set the DPLL as automatic reference clock selection.*
- `T_osUInt8 IDTDpll_GetAutoSelInput` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`)  
*Get automatic input selection status.*
- `void IDTDpll_SetForceSelInput` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`, `T_osUInt8` `nInputNo`)  
*Set the DPLL to force to lock to a specified reference clock.*
- `T_osUInt8 IDTDpll_GetForceSelInput` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`)  
*Set the DPLL to force to lock to a specified reference clock.*
- `T_osUInt8 IDTDpll_Get1stPriorityInput` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`)  
*Get input number of a valid clock with the highest priority.*
- `T_osUInt8 IDTDpll_Get2ndPriorityInput` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`)  
*Get input number of a valid clock with the second highest priority.*
- `T_osUInt8 IDTDpll_Get3rdPriorityInput` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`)  
*Get input number of a valid clock with the third highest priority.*
- `T_osUInt8 IDTDpll_GetCurrentSelectInput` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`)  
*Get the current selected input clock.*
- `void IDTDpll_SetDpllOperMod` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`, `T_idtDpllOperMode` `nMode`)  
*Set DPLL working at the specified operating mode.*
- `T_idtDpllOperMode IDTDpll_GetDpllOperMod` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`)  
*Set DPLL working at the specified operating mode.*
- `T_osUInt8 IDTDpll_IsDpllLocked` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`)  
*Inquire if the DPLL is in lock state.*
- `void IDTDpll_SetBandWidthDamping` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`, `T_idtBandwidthLimitStage` `stage`, `T_idtDpllBandwidth` `nBandWidth`, `T_idtDampingFactor` `nDamping`)  
*Set the bandwidth and damping factor used for bandwidth limiting.*
- `void IDTDpll_GetBandWidthDamping` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`, `T_idtBandwidthLimitStage` `stage`, `T_idtDpllBandwidth` `*nBandWidth_p`, `T_idtDampingFactor` `*nDamping_p`)  
*Get configured bandwidth and damping factor for bandwidth limiting.*
- `void IDTDpll_SetAutoSelBandWidth` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`, `T_osUInt8` `nEnable`)  
*Set device to select a suitable bandwidth and damping factor automatically.*
- `T_osUInt8 IDTDpll_GetAutoSelBandWidth` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`)  
*Get auto bandwidth/damping factor status automatically.*
- `void IDTDpll_SetDpllFrozen` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`, `T_osUInt8` `nEnable`)  
*Set DPLL integral path value frozen.*
- `T_osUInt8 IDTDpll_GetDpllFrozen` (`T_idtDrvHdlr` `hdlr`, `T_idtDpllInstance` `nDpll`)  
*Get DPLL integral path value frozen.*

- void [IDTDpll\\_SetPhaseCoarseDetector](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, T\_osUInt8 nEnable, T\_osUInt8 nWide, T\_osUInt8 nMultiPhase, T\_osUInt8 nMultiPh8kEn)
 

*Set the coarse phase lose detector enable.*
- void [IDTDpll\\_GetPhaseCoarseDetector](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, T\_osUInt8 \*nEnable\_p, T\_osUInt8 \*nWide\_p, T\_osUInt8 \*nMultiPhase\_p, T\_osUInt8 \*nMultiPh8kEn\_p)
 

*Set the coarse phase lose detector enable.*
- void [IDTDpll\\_SetPhaseCoarseLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtCoarsePhaseLimit](#) nLimit)
 

*Set the limitation of the coarse phase lose detector.*
- [T\\_idtCoarsePhaseLimit](#) [IDTDpll\\_GetPhaseCoarseLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
 

*Get the limitation of the coarse phase lose detector.*
- void [IDTDpll\\_SetPhaseFineDetectorEnable](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, T\_osUInt8 nEnable)
 

*Set the fine phase lose detector enable.*
- T\_osUInt8 [IDTDpll\\_GetPhaseFineDetectorEnable](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
 

*Get the fine phase lose detector enable.*
- void [IDTDpll\\_SetPhaseFineLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtFinePhaseLimit](#) nLimit)
 

*Set the limitation of the fine phase lose detector.*
- [T\\_idtFinePhaseLimit](#) [IDTDpll\\_GetPhaseFineLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
 

*Set the limitation of the fine phase lose detector.*
- void [IDTDpll\\_SetFastLossSwitch](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)
 

*Set reference clock switch if a fast loss occurs.*
- T\_osUInt8 [IDTDpll\\_GetFastLossSwitch](#) (T\_idtDrvHdlr hdlr)
 

*Get reference clock switch if a fast loss occurs.*
- void [IDTDpll\\_SetHoldoverMode](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtDpllHoldover](#) nMode, T\_osUInt8 nReadMode)
 

*Set calculation mode of holdover frequency and read back mode.*
- void [IDTDpll\\_GetHoldoverMode](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtDpllHoldover](#) \*nMode\_p, T\_osUInt8 \*nReadMode\_p)
 

*Get calculation mode of holdover frequency and read back mode.*
- void [IDTDpll\\_SetTempHoldoverMode](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtTemp\\_holdover](#) nMode)
 

*Set calculation mode of temporary holdover frequency.*
- [T\\_idtTemp\\_holdover](#) [IDTDpll\\_GetTempHoldoverMode](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
 

*Set calculation mode of temporary holdover frequency.*
- long [IDTDpll\\_GetHoldoverFreq](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
 

*Get holdover frequency offset (in parts per trillion). In DCO mode, this function returns current DCO offset (in parts per trillion).*
- void [IDTDpll\\_SetHoldoverFreq](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, long pptOffset)
 

*Set DCO frequency to device. Note this function should only be called when DCO mode is enabled.*
- long [IDTDpll\\_GetCurrentDpllFreqOffset](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
 

*Get current frequency offset of DPLL in parts per trillion (PPT)*
- void [IDTDpll\\_SetDpllSoftAlarmLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, T\_osUInt8 nLimit)
 

*Set the limitation of Soft alarm of DPLL.*
- T\_osUInt8 [IDTDpll\\_GetDpllSoftAlarmLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
 

*Get the limitation of Soft alarm of DPLL.*
- void [IDTDpll\\_SetDpllHardAlarmLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, T\_osUInt16 nLimit)
 

*Set the limitation of Hard alarm of DPLL.*
- T\_osUInt16 [IDTDpll\\_GetDpllHardAlarmLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)
 

*Get the limitation of Hard alarm of DPLL.*
- void [IDTDpll\\_SetDpllHardAlarmEnable](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)
 

*Set DPLL in unlocked state when a Hard alarm raised.*

- T\_osUInt8 [IDTDpll\\_GetDpllHardAlarmEnable](#) (T\_idtDrvHdlr hdlr)  
*Get DPLL in unlocked state when a Hard alarm raised.*
- T\_osUInt16 [IDTDpll\\_GetCurrentPhaseError](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get the current phase error of DPLL.*
- void [IDTDpll\\_SetSonetGigEthOutputPath](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 instance, T\_idtDpllInstance inputDpll, T\_idtSonetGigEthOutput outputFreq)  
*Set Sonet/GigE path configuration.*
- void [IDTDpll\\_GetSonetGigEthOutputPath](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 instance, T\_idtDpllInstance \*inputDpll\_p, T\_idtSonetGigEthOutput \*outputFreq\_p)  
*Retrieve Sonet/GigE path configuration.*
- void [IDTDpll\\_SetDpllOutputPathSelectorA](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtDpllOutputSelectorA path)  
*Set DPLL output path selector A.*
- T\_idtDpllOutputSelectorA [IDTDpll\\_GetDpllOutputPathSelectorA](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get DPLL output path selector A.*
- void [IDTDpll\\_SetDpllOutputPathSelectorB](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtDpllOutputSelectorB path)  
*Set DPLL output path selector B.*
- T\_idtDpllOutputSelectorB [IDTDpll\\_GetDpllOutputPathSelectorB](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get DPLL output path selector B.*
- void [IDTDpll\\_SetDpllEthPathEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nDisable)  
*Set ETH Enable/Disable.*
- T\_osUInt8 [IDTDpll\\_GetDpllEthPathEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get ETH Enable/Disable.*
- void [IDTDpll\\_SetDpllSelBPathEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nDisable)  
*Set DPLL output path selector B Enable/Disable.*
- T\_osUInt8 [IDTDpll\\_GetDpllSelBPathEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get DPLL output path selector B Enable/Disable.*
- void [IDTDpll\\_SetDpll16E116T1Enable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nDisable)  
*Set 16E1/16T1 Enable/Disable.*
- T\_osUInt8 [IDTDpll\\_GetDpll16E116T1Enable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get 16E1/16T1 Enable/Disable.*
- void [IDTDpll\\_SetDpllSelAPathEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nDisable)  
*Set DPLL output path selector A Enable/Disable.*
- T\_osUInt8 [IDTDpll\\_GetDpllSelAPathEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Set DPLL output path selector A Enable/Disable.*
- void [IDTDpll\\_SetFrequencyMonitoringFactor](#) (T\_idtDrvHdlr hdlr, T\_idtFreqMonFactor freqMonFactor)  
*Function to set the frequency monitoring factor. Hard and soft alarm thresholds use this factor as their base unit.*
- T\_idtFreqMonFactor [IDTDpll\\_GetFrequencyMonitoringFactor](#) (T\_idtDrvHdlr hdlr)  
*Function to get the frequency monitoring factor. Hard and soft alarm thresholds use this factor as their base unit.*
- void [IDTDpll\\_SetHardAlarmThreshold](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 accepting, T\_osUInt8 rejecting)  
*Set the frequency hard alarm accepting thresholds of reference clock monitoring.*
- void [IDTDpll\\_GetHardAlarmThreshold](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 \*accepting\_p, T\_osUInt8 \*rejecting\_p)  
*Get the frequency hard alarm accepting thresholds of reference clock monitoring.*
- void [IDTDpll\\_SetSoftAlarmThreshold](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 accepting, T\_osUInt8 rejecting)  
*Set the frequency soft alarm accepting thresholds of reference clock monitoring.*
- void [IDTDpll\\_GetSoftAlarmThreshold](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 \*accepting\_p, T\_osUInt8 \*rejecting\_p)  
*Get the frequency soft alarm accepting thresholds of reference clock monitoring.*
- void [IDTDpll\\_SetLcpCtrl](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nlcp)

*Set charge pump control.*

- T\_osUInt8 IDTDpll\_GetIcpCtrl (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)

*Get charge pump control.*

- void IDTDpll\_SetPhaseSlope (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtPhaseSlope nPhaseSlope)

*Set DPLL phase slope for DPLLs.*

- T\_idtPhaseSlope IDTDpll\_GetPhaseSlope (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)

*Get DPLL phase slope for DPLLs.*

- void IDTDpll\_SetPpsFastLock (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnableFreq, T\_osUInt8 nEnablePhase)

*Enable/Disable 1 PPS fast lock.*

- void IDTDpll\_GetPpsFastLock (T\_idtDrvHdlr hdlr, T\_osUInt8 \*nEnableFreq\_p, T\_osUInt8 \*nEnablePhase\_p)

*Get enable/disable 1 PPS fast lock status.*

- void IDTDpll\_SetPhaseTransient (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)

*Set the configuration when phase transient occurs.*

- T\_osUInt8 IDTDpll\_GetPhaseTransient (T\_idtDrvHdlr hdlr)

*Get the configuration when phase transient occurs.*

- void IDTDpll\_SetHitlessSwitchingFeature (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable, T\_osUInt8 nFrozen)

*Set the configuration of hitless switching feature.*

- void IDTDpll\_GetHitlessSwitchingFeature (T\_idtDrvHdlr hdlr, T\_osUInt8 \*nEnable\_p, T\_osUInt8 \*nFrozen\_p)

*Get the configuration of hitless switching feature.*

- void IDTDpll\_SetPhaseOffset (T\_idtDrvHdlr hdlr, T\_osUInt16 nOffset)

*Set the phase offset value.*

- T\_osUInt16 IDTDpll\_GetPhaseOffset (T\_idtDrvHdlr hdlr)

*Get the phase offset value.*

- void IDTDpll\_SetPpsPhase (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtDpllPpsPhasePostion nPhase, T\_idtDpllPpsPulseWidth nPulse)

*Set PPS phase and pulse for DPLL.*

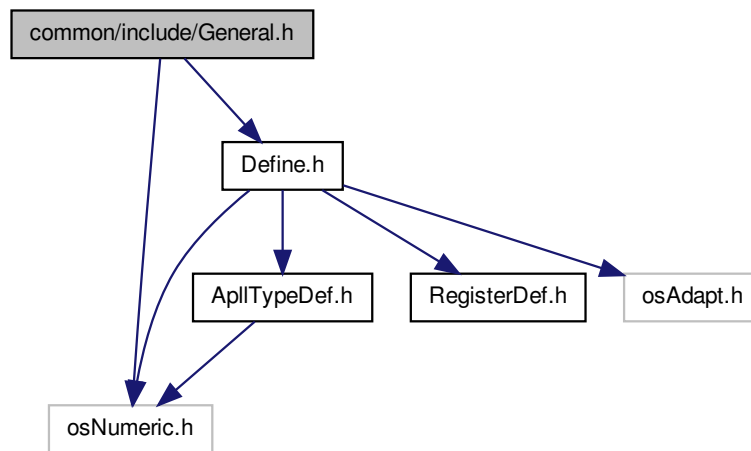
- void IDTDpll\_GetPpsPhase (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtDpllPpsPhasePostion \*nPhase\_p, T\_idtDpllPpsPulseWidth \*nPulse\_p)

*Get PPS phase and pulse for DPLL.*

## 6.36 common/include/General.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
```

Include dependency graph for General.h:



## Data Structures

- struct [T\\_idtl2CtransData](#)  
*I2C address translation information used internally by the driver.*

## Macros

- `#define Activate_Edge_Rising 0`
- `#define Activate_Edge_Falling 1`
- `#define INT_Active_Low 0`
- `#define INT_Active_High 1`
- `#define NOMINAL_PPT_TO_2COMPL(ppt) (T_osUInt32)((((ppt)*10)/884)`  
*Conversion from parts per trillion to register unit value for nominal (oscillator) offset.*
- `#define NOMINAL_2COMPL_TO_PPT(twoCompl) (long)((((twoCompl)*884)/10)`  
*Conversion from parts per trillion to register unit value for nominal (oscillator) offset.*
- `#define NOMINAL_MAX_PPT 94893`  
*Maximum oscillator offset value.*
- `#define NOMINAL_MIN_PPT -94893`  
*Minimum oscillator offset value.*

## Enumerations

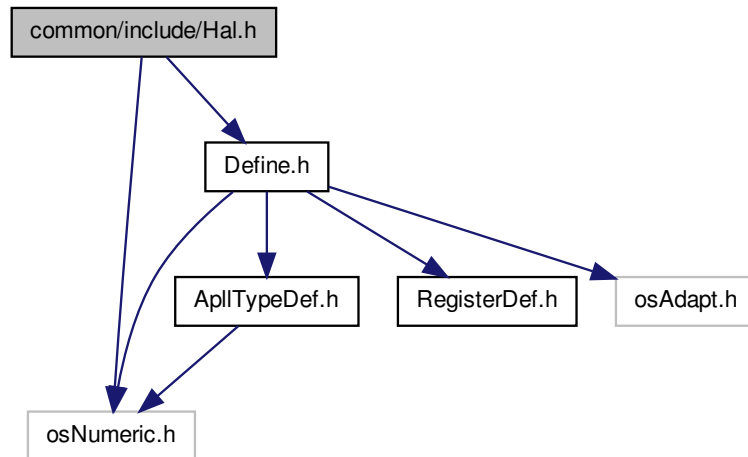
- enum [networkType\\_t](#) { [Network\\_SDH](#) = 0, [Network\\_SONET](#) = 1, [Network\\_MAX](#) }
  - enum [phaseTimeoutMultiplier\\_t](#) { [phaseTimeoutMultiplier\\_2](#) = 0, [phaseTimeoutMultiplier\\_4](#) = 1, [phaseTimeoutMultiplier\\_8](#) = 2, [phaseTimeoutMultiplier\\_16](#) = 3, [phaseTimeoutMultiplier\\_MAX](#) }
- Enumerated type listing phase alarm timeout multipliers.*

## Functions

- T\_osUInt16 [IDTGeneral\\_GetDeviceID](#) (T\_idtDrvHdlr hdlr)  
*Get ID number of the device.*
- void [IDTGeneral\\_SetOscFreqOffset](#) (T\_idtDrvHdlr hdlr, long pptOffset)  
*Set compensation to the oscillator offset.*
- long [IDTGeneral\\_GetOscFreqOffset](#) (T\_idtDrvHdlr hdlr)  
*Get compensated oscillator offset.*
- void [IDTGeneral\\_SetNetworkType](#) (T\_idtDrvHdlr hdlr, [networkType\\_t](#) nType)  
*Set the network type, SDH or SONET.*
- [networkType\\_t](#) [IDTGeneral\\_GetNetworkType](#) (T\_idtDrvHdlr hdlr)  
*Get the network type, SDH or SONET.*
- void [IDTGeneral\\_SetRevertiveMode](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
*Set the device running in Revertive mode or not.*
- T\_osUInt8 [IDTGeneral\\_GetRevertiveMode](#) (T\_idtDrvHdlr hdlr)  
*Get the device running in Revertive mode or not.*
- void [IDTGeneral\\_SetTimeoutValue](#) (T\_idtDrvHdlr hdlr, [phaseTimeoutMultiplier\\_t](#) nMultiFactor, T\_osUInt8 nTimeout)  
*Set the value of time out of phase alarm  $Timeout(second) = nMultiFactor * nTimeout$ .*
- void [IDTGeneral\\_GetTimeoutValue](#) (T\_idtDrvHdlr hdlr, [phaseTimeoutMultiplier\\_t](#) \*nMultiFactor\_p, T\_osUInt8 \*nTimeout\_p)  
*Get the value of time out of phase alarm  $Timeout(second) = nMultiFactor * nTimeout$ .*
- void [IDTGeneral\\_SetOscActiveEdge](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEdge)  
*Set the activate edge of main oscillator.*
- T\_osUInt8 [IDTGeneral\\_GetOscActiveEdge](#) (T\_idtDrvHdlr hdlr)  
*Get the activate edge of main oscillator.*
- void [IDTGeneral\\_SetProtectMode](#) (T\_idtDrvHdlr hdlr)  
*Enable protected register access mode.*
- void [IDTGeneral\\_SetSingleUnprotectMode](#) (T\_idtDrvHdlr hdlr)  
*Set single access register access mode.*
- void [IDTGeneral\\_SetFullyUnprotectMode](#) (T\_idtDrvHdlr hdlr)  
*Set fully un-protected register access mode.*
- void [IDTGeneral\\_SetFlagOnTDO](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
*Set TDO pin to indicate the fail of selected clock.*
- T\_osUInt8 [IDTGeneral\\_GetFlagOnTDO](#) (T\_idtDrvHdlr hdlr)  
*Get TDO pin to indicate the fail of selected clock.*
- void [IDTGeneral\\_SetUltraFastSwitch](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
*Set Ultra-fast-switch enable or not.*
- T\_osUInt8 [IDTGeneral\\_GetUltraFastSwitch](#) (T\_idtDrvHdlr hdlr)  
*Get Ultra-fast-switch enable or not.*
- void [IDTGeneral\\_SetHardAlarm](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
*Set Hard-alarm enable when the input exceeds the threshold.*
- T\_osUInt8 [IDTGeneral\\_GetHardAlarm](#) (T\_idtDrvHdlr hdlr)  
*Get Hard-alarm enable when the input exceeds the threshold.*
- T\_osUInt8 [IDTGeneral\\_i2cTranslate](#) (T\_osUInt8 i2cAddr, void \*transData)  
*I2C address translation function used internally by the driver.*
- void [IDTGeneral\\_InitDeviceCommon](#) (T\_idtDrvHdlr hdlr)
- T\_idtDrvHdlr [IDTGeneral\\_InitDriverCommon](#) (const char \*name, [T\\_idtDrvAccess](#) drvAccess, [T\\_idtDeviceType](#) deviceType, const [T\\_idtDrvDeviceConfig](#) \*deviceConfig, int useHighLevelApi)
- void [IDTGeneral\\_DeInitDriver](#) (T\_idtDrvHdlr hdlr)  
*De-initialize driver handler.*

## 6.37 common/include/Hal.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
Include dependency graph for Hal.h:
```



### Macros

- `#define IDT_HAL_USE_MACRO 0`
- `#define IDTHAL_ModifyBitfield(regVal, mask, lsb, data)`  
*Define function to set bitfield value of read data from device.*
- `#define IDTHAL_GetBitfield(regVal, mask, lsb) (((regVal) & (mask)) >> (lsb))`  
*Define function to get bitfield value of read data from device.*

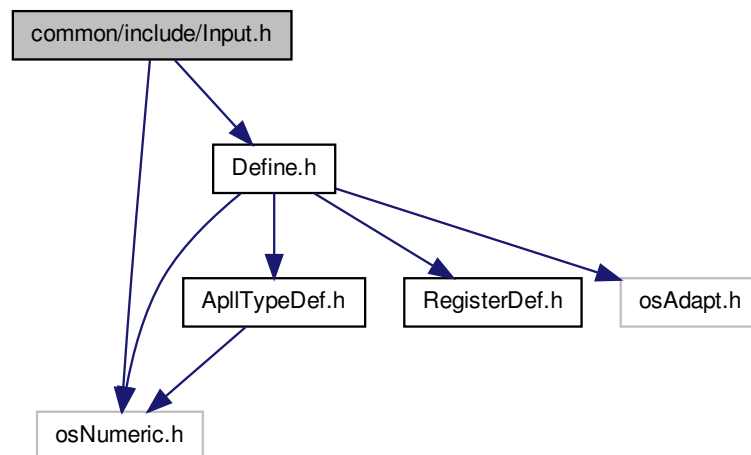
### Functions

- void `IDTHAL_WriteBitfield` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 mask, T\_osUInt8 lsb, T\_osUInt8 data)  
*Define function to write a bitfield to a device.*
- void `IDTHAL_WriteBitfield_Ext` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 mask, T\_osUInt8 lsb, T\_osUInt8 data)  
*Define function to write a bitfield to a device.*
- T\_osUInt8 `IDTHAL_ReadBitfield` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 mask, T\_osUInt8 lsb)  
*Define function to read a bitfield from device.*
- T\_osUInt8 `IDTHAL_ReadBitfield_Ext` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 mask, T\_osUInt8 lsb)  
*Define function to read a bitfield from device.*
- T\_osUInt8 `IDTHal_Read` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset)  
*Function to read from device.*
- void `IDTHal_Write` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 data)  
*Define function to write to device.*
- T\_osUInt8 `IDTHal_Read_Ext` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset)  
*Function to read from device apll address space.*

- void `IDTHal_Write_Ext` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 data)  
*Function to write to device apll address space.*

## 6.38 common/include/Input.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
Include dependency graph for Input.h:
```



## Macros

- `#define Input_Phase_Lock_Alarm` 1
- `#define Input_No_Activity_Alarm` 2
- `#define Input_Freq_Hard_Alarm` 4
- `#define IDT_TRANSLATE_EXT_TO_INT_INPORT`(hdlr, ext, internal)  
*Utility macro to check if input port is supported and to translated from external port to internal port.*
- `#define IDT_EXT_INPORT_POPULATED`(hdlr, ext) (hdlr->deviceConfig\_m->inputExt2IntXlate\_ma[(ext)])  
*Utility macro to check if input port is populated.*

## Enumerations

- enum `T_idtInputDividerMux` { `Divider_Bypass` = 0, `Divider_Lock8k` = 1, `Divider_DivN` = 2, `Divider_MAX` }  
*Enumerated type listing input divider modes.*
- enum `T_idtHighFrequencyDivisor` { `HF_Bypass` = 0, `HF_Divide_4` = 1, `HF_Divide_5` = 2, `HF_MAX` }  
*Enumerated type listing high frequency (> 155.52MHz) dividers.*
- enum `T_idtInputFrequencyBitfieldValue` {  
  `Freq_8K` = 0, `Freq_1544M` = 1, `Freq_2048M` = 1, `Freq_648M` = 2,  
  `Freq_1944M` = 3, `Freq_2592M` = 4, `Freq_3888M` = 5, `Freq_2K` = 9,  
  `Freq_4K` = 10, `Freq_1PPS` = 11, `Freq_625M` = 12, `Freq_10M` = 13,  
  `Freq_MAX` }  
*Enumerated type listing input frequencies.*

- enum [T\\_idtInputFrequencyFamily](#) {  
[InFreqFamily\\_8K](#) = 0, [InFreqFamily\\_1544M](#) = 1, [InFreqFamily\\_2048M](#) = 2, [InFreqFamily\\_648M](#) = 3,  
[InFreqFamily\\_1944M](#) = 4, [InFreqFamily\\_2592M](#) = 5, [InFreqFamily\\_3888M](#) = 6, [InFreqFamily\\_2K](#) = 7,  
[InFreqFamily\\_4K](#) = 8, [InFreqFamily\\_1PPS](#) = 9, [InFreqFamily\\_625M](#) = 10, [InFreqFamily\\_10M](#) = 11,  
[InFreqFamily\\_MAX](#) }  
*Enumerated type listing input frequency families.*
- enum [T\\_idtAmiInputFrequencyBitFieldValue](#) { [AmiFreq\\_64kHzPlus8kHz](#), [AmiFreq\\_64kHzPlus400Hz](#), [AmiFreq\\_MAX](#) }  
*Enumerated type listing AMI Input frequencies.*
- enum [T\\_idtInputSyncFreq](#) {  
[SYNC\\_8kHz](#), [SYNC\\_1PPS](#), [SYNC\\_4kHz](#), [SYNC\\_2kHz](#),  
[SYNC\\_MAX](#) }  
*Enumerated type listing possible input sync frequencies.*
- enum [T\\_externalSyncSampling](#) {  
[externalSyncSampling\\_onTarget](#), [externalSyncSampling\\_0dot5Early](#), [externalSyncSampling\\_1dot0Late](#),  
[externalSyncSampling\\_0dot5Late](#),  
[externalSyncSampling\\_MAX](#) }  
*Enumerated type listing sampling of input external sync.*
- enum [T\\_externalSyncEdge](#) { [externalSyncEdge\\_FallingEdge](#), [externalSyncEdge\\_RisingEdge](#), [externalSyncEdge\\_MAX](#) }  
*Enumerated type listing possible external sync alignment options.*
- enum [T\\_externalSyncAlarmRange](#) {  
[externalSyncAlarmRange\\_1](#), [externalSyncAlarmRange\\_2](#), [externalSyncAlarmRange\\_3](#), [externalSyncAlarmRange\\_4](#),  
[externalSyncAlarmRange\\_5](#), [externalSyncAlarmRange\\_6](#), [externalSyncAlarmRange\\_7](#), [externalSyncAlarmRange\\_8](#),  
[externalSyncAlarmRange\\_MAX](#) }  
*Enumerate type listing ranges for external sync alarm.*
- enum [T\\_externalSyncBypass](#) { [externalSyncBypass\\_ExSync1](#), [externalSyncBypass\\_AutoSel](#), [externalSyncBypass\\_MAX](#) }  
*Enumerated type listing output synchronization with respect to.*
- enum [T\\_externalSyncEnable](#) { [externalSyncEnable\\_Disable](#), [externalSyncEnable\\_Enable](#), [externalSyncEnable\\_Auto](#), [externalSyncEnable\\_MAX](#) }  
*Enumerated type listing enable options for external input sync.*

## Functions

- void [IDTInput\\_SetPriority](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_osUInt8](#) nInputNo, [T\\_idtDpllInstance](#) nDpll, [T\\_osUInt8](#) nPriority)  
*Set specified port to the priority.*
- [T\\_osUInt8](#) [IDTInput\\_GetPriority](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_osUInt8](#) nInputNo, [T\\_idtDpllInstance](#) nDpll)  
*Get the current priority of the specified port.*
- void [IDTInput\\_SetInputFrequency](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_osUInt8](#) nInputNo, [T\\_idtInputFrequencyBitfieldValue](#) nFrequency)  
*Set the frequency of the input clock at the specified port.*
- [T\\_idtInputFrequencyBitfieldValue](#) [IDTInput\\_GetInputFrequency](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_osUInt8](#) nInputNo)  
*Get frequency of the input clock at the specified port.*
- void [IDTInput\\_SetAmiInputFrequency](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_osUInt8](#) nInputNo, [T\\_idtAmiInputFrequencyBitFieldValue](#) amiSignal)  
*Provision AMI signal for input port.*
- [T\\_idtAmiInputFrequencyBitFieldValue](#) [IDTInput\\_GetAmiInputFrequency](#) ([T\\_idtDrvHdlr](#) hdlr, [T\\_osUInt8](#) nInputNo)  
*Function to retrieve AMI input frequency for input port.*

- void [IDTInput\\_SetActivityMonitor](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, T\_osUInt8 nBucket)  
*Select one of the four activity monitor configurations for the specified input port.*
- T\_osUInt8 [IDTInput\\_GetActivityMonitor](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Retrieve selected activity monitoring configuration.*
- void [IDTInput\\_SetBucketConfig](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nBucket, T\_osUInt8 nUpper, T\_osUInt8 nLower, T\_osUInt8 nSize, T\_osUInt8 nDecay)  
*Set the configuration to Leaky Bucket activity monitoring.*
- void [IDTInput\\_GetBucketConfig](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nBucket, T\_osUInt8 \*nUpper\_p, T\_osUInt8 \*nLower\_p, T\_osUInt8 \*nSize\_p, T\_osUInt8 \*nDecay\_p)  
*Get activity monitoring configuration (Leaky Bucket)*
- void [IDTInput\\_SetPreDivider](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, [T\\_idtInputDividerMux](#) nDivider)  
*Assign a pre-divider for the specified input.*
- [T\\_idtInputDividerMux](#) [IDTInput\\_GetPreDivider](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Get configured input port pre-divider.*
- void [IDTInput\\_SetDivisionFactor](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, T\_osUInt16 iFactor)  
*Set the dividen factor to pre-divider assigned to a certain input.*
- T\_osUInt16 [IDTInput\\_GetDivisionFactor](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Get the dividen factor to pre-divider assigned to a certain input.*
- void [IDTInput\\_SetInputHFdivider](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, [T\\_idtHighFrequencyDivisor](#) nUsage)  
*Set the usage of high frequency (> 155.52MHz) divider.*
- [T\\_idtHighFrequencyDivisor](#) [IDTInput\\_GetInputHFdivider](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Get high frequency configuration divider.*
- T\_osUInt8 [IDTInput\\_GetInputFreqOffset](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Get the frequency offset of the specified input clock. The returned value is updated every 16 seconds.*
- T\_osUInt8 [IDTInput\\_GetInputClockStatus](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Get the status of a specified input clock.*
- T\_osUInt8 [IDTInput\\_GetInputClockValidation](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Get the clock validation status of an input port.*
- void [IDTInput\\_EnableAllInputs](#) (T\_idtDrvHdlr hdlr)  
*Enable to select anyone of input clocks.*
- void [IDTInput\\_DisableAllInputs](#) (T\_idtDrvHdlr hdlr)  
*Disable to select anyone of input clocks.*
- void [IDTInput\\_SetInputEnable](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, T\_osUInt8 enable)  
*Enable/Disable selected input.*
- T\_osUInt8 [IDTInput\\_GetInputEnable](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Retrieve enable/disable status of selected input.*
- void [IDTInput\\_SetSyncFrequency](#) (T\_idtDrvHdlr hdlr, [T\\_idtInputSyncFreq](#) syncFreq)  
*Function to select input external sync frequency.*
- [T\\_idtInputSyncFreq](#) [IDTInput\\_GetSyncFrequency](#) (T\_idtDrvHdlr hdlr)  
*Function to retrieve select input external sync frequency.*
- void [IDTInput\\_SetSyncSampling](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 exSyncN, [T\\_externalSyncSampling](#) sampling)  
*Function to provision sampling configuration for external sync.*
- [T\\_externalSyncSampling](#) [IDTInput\\_GetSyncSampling](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 exSyncN)  
*Function to retrieve sampling configuration for external sync.*
- void [IDTInput\\_SetSyncEdge](#) (T\_idtDrvHdlr hdlr, [T\\_externalSyncEdge](#) edge)  
*Function to provision synchronization alignment.*
- [T\\_externalSyncEdge](#) [IDTInput\\_GetSyncEdge](#) (T\_idtDrvHdlr hdlr)  
*Function to retrieve synchronization alignment.*
- void [IDTInput\\_SetSyncAlarmRange](#) (T\_idtDrvHdlr hdlr, [T\\_externalSyncAlarmRange](#) range)

- Function to provision the sync allowable range before alarming.*

• [T\\_externalSyncAlarmRange IDTInput\\_GetSyncAlarmRange](#) (T\_idtDrvHdlr hdlr)

*Function to retrieve the sync allowable range before alarming.*
- void [IDTInput\\_SetSyncBypass](#) (T\_idtDrvHdlr hdlr, [T\\_externalSyncBypass](#) bypass)

*Function to provision sync bypass mode.*
- [T\\_externalSyncBypass IDTInput\\_GetSyncBypass](#) (T\_idtDrvHdlr hdlr)

*Function to retrieve sync bypass mode.*
- void [IDTInput\\_SetSyncEnable](#) (T\_idtDrvHdlr hdlr, [T\\_externalSyncEnable](#) enable)

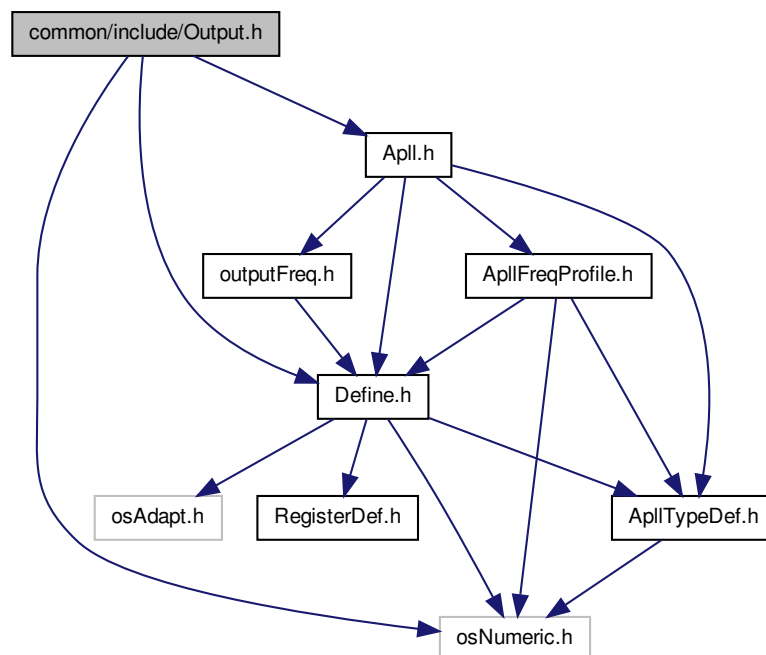
*Function to provision enable mode for input external sync.*
- [T\\_externalSyncEnable IDTInput\\_GetSyncEnable](#) (T\_idtDrvHdlr hdlr)

*Function to retrieve external sync enable mode.*
- [T\\_idtInputFrequencyFamily IDTInput\\_InFreqBitValue2Family](#) (T\_idtDrvHdlr hdlr, [T\\_idtInputFrequencyBitfield-Value](#) inFreqBitValue)

*Translate input frequency bit value to input frequency family.*

## 6.39 common/include/Output.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "Appl.h"
Include dependency graph for Output.h:
```



### Macros

- `#define` [IDT\\_TRANSLATE\\_EXT\\_TO\\_INT\\_OUTPORT](#)(hdlr, ext, internal)

*Utility macro to check if output port is supported and to translated from external port to internal port.*

- `#define IDT_EXT_OUTPUT_PORT_POPULATED(hdlr, ext) (hdlr->deviceConfig_m->outputExt2IntXlate_ma[(ext)])`  
Utility macro to check if output port is populated.

## Enumerations

- enum `T_outputPathType` {  
OutputPath\_SonetGigEth, OutputPath\_7776kHz, OutputPath\_Eth, OutputPath\_16E1\_16T1,  
OutputPath\_12E1\_E3\_T3, OutputPath\_GSM\_16E1\_16T1, OutputPath\_GPS, OutputPath\_OBSAI,  
OutputPath\_24T1, OutputPath\_MAX }  
Enumerated type listing possible output port sources (from DPLL(s)) This enumerated type is used in conjunction with DPLL instance to select output DPLL frequency to output dividers. The following table outlines.
- enum `T_outputInterface` {  
OUTPUTIF\_AMI, OUTPUTIF\_CMOS, OUTPUTIF\_LVDS, OUTPUTIF\_PECL,  
OUTPUTIF\_MAX }  
Enumerated type listing type of supported output port interfaces.
- enum `T_frameSync` { frameSync\_FRSYNC, frameSync\_MFRSYNC, frameSync\_MAX }  
Enumerated type listing supported frame syncs.

## Functions

- void `IDTOutput_SetOutputConfig` (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN, T\_idtDpllInstance nDpll, T\_outputPathType nPath, T\_osUInt8 nFactor, T\_idtApplConfigPerOutput \*applConfig)  
Set the configuration of Output port.
- void `IDTOutput_GetOutputConfig` (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN, T\_idtDpllInstance \*nDpll\_p, T\_outputPathType \*nPath\_p, T\_osUInt8 \*nFactor\_p, T\_idtApplConfigPerOutput \*applConfig)  
Get the configuration of Output port.
- void `IDTOutput_SetOutputInvert` (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN, T\_osUInt8 invert)  
Set the specified output port to generate a invert signal.
- T\_osUInt8 `IDTOutput_GetOutputInvert` (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN)  
Get output signal inversion status.
- void `IDTOutput_SetOutputEnable` (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN, T\_osUInt8 nEnable)  
Enable/Disable output port. Consult data sheet for supported ports.
- T\_osUInt8 `IDTOutput_GetOutputEnable` (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN)  
Get enable/disable status of output port. Consult data sheet for supported ports.
- void `IDTOutput_SetOutputInterface` (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN, T\_outputInterface outIf)  
Set output port interface type. Consult data sheet for supported interfaces per port.
- T\_outputInterface `IDTOutput_GetOutputInterface` (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN)  
Get output port interface type. Consult data sheet for supported interfaces per port.
- void `IDTOutput_SetFrameSync` (T\_idtDrvHdlr hdlr, T\_frameSync frameSync, T\_osUInt8 enable, T\_osUInt8 invert, T\_osUInt8 pulse)  
Function to control output frame syncs.
- void `IDTOutput_GetFrameSync` (T\_idtDrvHdlr hdlr, T\_frameSync frameSync, T\_osUInt8 \*enable\_p, T\_osUInt8 \*invert\_p, T\_osUInt8 \*pulse\_p)  
Function to retrieve output frame sync configuration.
- void `IDTOutput_SetFrameSyncPulseEdge` (T\_idtDrvHdlr hdlr, T\_osUInt8 isRisingEdge)  
Function provision trigger for frame sync pulses.
- T\_osUInt8 `IDTOutput_GetFrameSyncPulseEdge` (T\_idtDrvHdlr hdlr)  
Function retrieve trigger for frame sync pulses.
- void `IDTOutput_DisableApplInput` (T\_idtDrvHdlr hdlr, T\_idtOutputApplConfig outputApplConfig)  
Function disables APLL input.
- void `IDTOutput_DisableAllIntOutput` (T\_idtDrvHdlr hdlr)  
Function disables all internal outputs.

## 6.40 common/include/RegisterDef.h File Reference

This header file contains register and bitfield information for accessing device.

### Enumerations

```

• enum {
    REG_ID_LOW = 0x00, REG_ID_HIGH = 0x01, REG_MPU_PIN_STS = 0x02, REG_NOMINAL_FREQ_LO-
    W_CNFG = 0x04,
    REG_NOMINAL_FREQ_MID_CNFG = 0x05, REG_NOMINAL_FREQ_HIGH_CNFG = 0x06, REG_T4_T0_
    SEL_CNFG = 0x07, REG_PHASE_ALARM_TIME_OUT_CNFG = 0x08,
    REG_INPUT_MODE_CNFG = 0x09, REG_DIFFERENTIAL_IN_OUT_CNFG = 0x0A, REG_MON_SW_HS-
    _CNFG = 0x0B, REG_INTERRUPT_CNFG = 0x0C,
    REG_INTERRUPTS1_STS = 0x0D, REG_INTERRUPTS2_STS = 0x0E, REG_INTERRUPTS3_STS = 0x0F,
    REG_INTERRUPTS1_MASK_CNFG = 0x10,
    REG_INTERRUPTS2_MASK_CNFG = 0x11, REG_INTERRUPTS3_MASK_CNFG = 0x12, REG_MS_SL_
    CTRL_CNFG = 0x13, REG_IN1_CNFG = 0x14,
    REG_IN2_CNFG = 0x15, REG_IN3_CNFG = 0x16, REG_IN4_CNFG = 0x17, REG_IN5_IN6_HF_DIV_CN-
    FG = 0x18,
    REG_IN5_CNFG = 0x19, REG_IN6_CNFG = 0x1A, REG_IN7_CNFG = 0x1B, REG_IN8_CNFG = 0x1C,
    REG_IN9_CNFG = 0x1D, REG_IN10_CNFG = 0x1E, REG_IN11_CNFG = 0x1F, REG_IN12_CNFG = 0x20,
    REG_IN13_CNFG = 0x21, REG_IN14_CNFG = 0x22, REG_PRE_DIV_CH_CNFG = 0x23, REG_PRE_DIV-
    N_LOW_CNFG = 0x24,
    REG_PRE_DIVN_HIGH_CNFG = 0x25, REG_IN1_IN2_SEL_PRIORITY_CNFG = 0x26, REG_IN3_IN4_S-
    EL_PRIORITY_CNFG = 0x27, REG_IN5_IN6_SEL_PRIORITY_CNFG = 0x28,
    REG_IN7_IN8_SEL_PRIORITY_CNFG = 0x29, REG_IN9_IN10_SEL_PRIORITY_CNFG = 0x2A, REG_I-
    N11_IN12_SEL_PRIORITY_CNFG = 0x2B, REG_IN13_IN14_SEL_PRIORITY_CNFG = 0x2C,
    REG_PAGE_POINTER_CNFG = 0x2D, REG_FREQ_MON_FACTOR_CNFG = 0x2E, REG_HARD_FREQ-
    _MON_THRESHOLD_CNFG = 0x2F, REG_SOFT_FREQ_MON_THRESHOLD_CNFG = 0x30,
    REG_UPPER_THRESHOLD_0_CNFG = 0x31, REG_LOWER_THRESHOLD_0_CNFG = 0x32, REG_BU-
    CKET_SIZE_0_CNFG = 0x33, REG_DECAY_RATE_0_CNFG = 0x34,
    REG_UPPER_THRESHOLD_1_CNFG = 0x35, REG_LOWER_THRESHOLD_1_CNFG = 0x36, REG_BU-
    CKET_SIZE_1_CNFG = 0x37, REG_DECAY_RATE_1_CNFG = 0x38,
    REG_UPPER_THRESHOLD_2_CNFG = 0x39, REG_LOWER_THRESHOLD_2_CNFG = 0x3A, REG_BU-
    CKET_SIZE_2_CNFG = 0x3B, REG_DECAY_RATE_2_CNFG = 0x3C,
    REG_UPPER_THRESHOLD_3_CNFG = 0x3D, REG_LOWER_THRESHOLD_3_CNFG = 0x3E, REG_BU-
    CKET_SIZE_3_CNFG = 0x3F, REG_DECAY_RATE_3_CNFG = 0x40,
    REG_IN_FREQ_READ_CH_CNFG = 0x41, REG_IN_FREQ_READ_STS = 0x42, REG_IN1_IN2_STS =
    0x43, REG_IN3_IN4_STS = 0x44,
    REG_IN5_IN6_STS = 0x45, REG_IN7_IN8_STS = 0x46, REG_IN9_IN10_STS = 0x47, REG_IN11_IN12_S-
    TS = 0x48,
    REG_IN13_IN14_STS = 0x49, REG_INPUT_VALID1_STS = 0x4A, REG_INPUT_VALID2_STS = 0x4B,
    REG_REMOTE_INPUT_VALID1_CNFG = 0x4C,
    REG_REMOTE_INPUT_VALID2_CNFG = 0x4D, REG_PRIORITY_TABLE1_STS = 0x4E, REG_PRIORITY-
    _TABLE2_STS = 0x4F, REG_T0_INPUT_SEL_CNFG = 0x50,
    REG_T4_INPUT_SEL_CNFG = 0x51, REG_OPERATING_STS = 0x52, REG_T0_OPERATION_MODE_C-
    NFG = 0x53, REG_T4_OPERATION_MODE_CNFG = 0x54,
    REG_T0_DPLL_APLL_PATH_CNFG = 0x55, REG_T0_DPLL_START_BW_DAMPING_CNFG = 0x56, R-
    EG_T0_DPLL_ACQ_BW_DAMPING_CNFG = 0x57, REG_T0_DPLL_LOCKED_BW_DAMPING_CNFG =
    0x58,
    REG_T0_BW_OVERSHOOT_CNFG = 0x59, REG_PHASE_LOSS_COARSE_LIMIT_CNFG = 0x5A, REG-
    _PHASE_LOSS_FINE_LIMIT_CNFG = 0x5B, REG_T0_HOLDOVER_MODE_CNFG = 0x5C,
    REG_HOLDOVER_FREQ_LOW_CNFG = 0x5D, REG_HOLDOVER_FREQ_MID_CNFG = 0x5E, REG_HO-
    LDOVER_FREQ_HIGH_CNFG = 0x5F, REG_T4_DPLL_APLL_PATH_CNFG = 0x60,
    REG_T4_DPLL_LOCKED_BW_DAMPING_CNFG = 0x61, REG_CURRENT_FREQ_LOW_STS = 0x62,
    REG_CURRENT_FREQ_MID_STS = 0x63, REG_CURRENT_FREQ_HIGH_STS = 0x64,
    REG_DPLL_FREQ_SOFT_LIMIT_CNFG = 0x65, REG_DPLL_FREQ_HARD_LIMIT_LOW_CNFG = 0x66,

```

```

REG_DPLL_FREQ_HARD_LIMIT_MID_CNFG = 0x67, REG_CURRENT_DPLL_PHASE_LOW_STS =
0x68,
REG_CURRENT_DPLL_PHASE_HIGH_STS = 0x69, REG_OUT1_FREQ_CNFG = 0x6B, REG_OUT2_FR-
EQ_CNFG = 0x6C, REG_OUT3_FREQ_CNFG = 0x6D,
REG_OUT4_FREQ_CNFG = 0x6E, REG_OUT5_FREQ_CNFG = 0x6F, REG_OUT6_FREQ_CNFG = 0x70,
REG_OUT7_FREQ_CNFG = 0x71,
REG_OUT8_FREQ_CNFG = 0x72, REG_OUT9_FREQ_CNFG = 0x73, REG_FR_MFR_SYNC_CNFG =
0x74, REG_PHASE_MON_CNFG = 0x78,
REG_PHASE_OFFSET_LOW_CNFG = 0x7A, REG_PHASE_OFFSET_HIGH_CNFG = 0x7B, REG_SYNC-
_MONITOR_CNFG = 0x7C, REG_SYNC_PHASE_CNFG = 0x7D,
REG_PROTECTION_CNFG = 0x7E, REG_MPU_SEL_CNFG = 0x7F, REG_DFS_OFF_CNFG = 0x30, RE-
G_T0_PPS_CNFG = 0x31,
REG_PH_SLOPE_CNFG = 0x32, REG_T0_ICP_CTRL_CNFG = 0x33, REG_T4_ICP_CTRL_CNFG = 0x34,
REG_T4_PPS_CNFG = 0x3E,
BF_ID_A_SIZE = 8, BF_ID_A_MASK = 0xFF, BF_ID_A_LSB = 0, BF_ID_B_SIZE = 8,
BF_ID_B_MASK = 0xFF, BF_ID_B_LSB = 0, BF_MPU_PIN_STS_SIZE = 1, BF_MPU_PIN_STS_MASK =
0x01,
BF_MPU_PIN_STS_LSB = 0, BF_NOMINAL_FREQ_VALUE_A_SIZE = 8, BF_NOMINAL_FREQ_VALUE_-
A_MASK = 0xFF, BF_NOMINAL_FREQ_VALUE_A_LSB = 0,
BF_NOMINAL_FREQ_VALUE_B_SIZE = 8, BF_NOMINAL_FREQ_VALUE_B_MASK = 0xFF, BF_NOMIN-
AL_FREQ_VALUE_B_LSB = 0, BF_NOMINAL_FREQ_VALUE_C_SIZE = 8,
BF_NOMINAL_FREQ_VALUE_C_MASK = 0xFF, BF_NOMINAL_FREQ_VALUE_C_LSB = 0, BF_T4_T0_S-
EL_SIZE = 1, BF_T4_T0_SEL_MASK = 0x10,
BF_T4_T0_SEL_LSB = 4, BF_MULTI_FACTOR_SIZE = 2, BF_MULTI_FACTOR_MASK = 0xC0, BF_MUL-
TI_FACTOR_LSB = 6,
BF_TIMEOUT_VALUE_SIZE = 6, BF_TIMEOUT_VALUE_MASK = 0x3F, BF_TIMEOUT_VALUE_LSB = 0,
BF_AUTO_EXT_SYNC_EN_SIZE = 1,
BF_AUTO_EXT_SYNC_EN_MASK = 0x80, BF_AUTO_EXT_SYNC_EN_LSB = 7, BF_EXT_SYNC_EN_SI-
ZE = 1, BF_EXT_SYNC_EN_MASK = 0x40,
BF_EXT_SYNC_EN_LSB = 6, BF_PH_ALARM_TIMEOUT_SIZE = 1, BF_PH_ALARM_TIMEOUT_MASK =
0x20, BF_PH_ALARM_TIMEOUT_LSB = 5,
BF_SYNC_FREQ_SIZE = 2, BF_SYNC_FREQ_MASK = 0x18, BF_SYNC_FREQ_LSB = 3, BF_IN_SONE-
T_SDH_SIZE = 1,
BF_IN_SONET_SDH_MASK = 0x04, BF_IN_SONET_SDH_LSB = 2, BF_MASTER_SLAVE_SIZE = 1, BF-
_MASTER_SLAVE_MASK = 0x02,
BF_MASTER_SLAVE_LSB = 1, BF_REVERTIVE_MODE_SIZE = 1, BF_REVERTIVE_MODE_MASK =
0x01, BF_REVERTIVE_MODE_LSB = 0,
BF_OSC_EDGE_SIZE = 1, BF_OSC_EDGE_MASK = 0x04, BF_OSC_EDGE_LSB = 2, BF_OUT7_PECL_-
LVDS_SIZE = 1,
BF_OUT7_PECL_LVDS_MASK = 0x02, BF_OUT7_PECL_LVDS_LSB = 1, BF_OUT6_PECL_LVDS_SIZE
= 1, BF_OUT6_PECL_LVDS_MASK = 0x01,
BF_OUT6_PECL_LVDS_LSB = 0, BF_FREQ_MON_CLK_SIZE = 1, BF_FREQ_MON_CLK_MASK = 0x80,
BF_FREQ_MON_CLK_LSB = 7,
BF_LOS_FLAG_TO_TDO_SIZE = 1, BF_LOS_FLAG_TO_TDO_MASK = 0x40, BF_LOS_FLAG_TO_TDO-
_LSB = 6, BF_ULTR_FAST_SW_SIZE = 1,
BF_ULTR_FAST_SW_MASK = 0x20, BF_ULTR_FAST_SW_LSB = 5, BF_EXT_SW_SIZE = 1, BF_EXT_S-
W_MASK = 0x10,
BF_EXT_SW_LSB = 4, BF_HS_FREZ_SIZE = 1, BF_HS_FREZ_MASK = 0x08, BF_HS_FREZ_LSB = 3,
BF_HS_EN_SIZE = 1, BF_HS_EN_MASK = 0x04, BF_HS_EN_LSB = 2, BF_FREQ_MON_HARD_EN_SIZE
= 1,
BF_FREQ_MON_HARD_EN_MASK = 0x01, BF_FREQ_MON_HARD_EN_LSB = 0, BF_HZ_EN_SIZE = 1,

```

```

BF_HZ_EN_MASK = 0x02,
BF_HZ_EN_LSB = 1, BF_INT_POL_SIZE = 1, BF_INT_POL_MASK = 0x01, BF_INT_POL_LSB = 0,
BF_IN8_SIZE = 1, BF_IN8_MASK = 0x80, BF_IN8_LSB = 7, BF_IN7_SIZE = 1,
BF_IN7_MASK = 0x40, BF_IN7_LSB = 6, BF_IN6_SIZE = 1, BF_IN6_MASK = 0x20,
BF_IN6_LSB = 5, BF_IN5_SIZE = 1, BF_IN5_MASK = 0x10, BF_IN5_LSB = 4,
BF_IN4_SIZE = 1, BF_IN4_MASK = 0x08, BF_IN4_LSB = 3, BF_IN3_SIZE = 1,
BF_IN3_MASK = 0x04, BF_IN3_LSB = 2, BF_IN2_SIZE = 1, BF_IN2_MASK = 0x02,
BF_IN2_LSB = 1, BF_IN1_SIZE = 1, BF_IN1_MASK = 0x01, BF_IN1_LSB = 0,
BF_T0_OPERATING_MODE_STS_SIZE = 1, BF_T0_OPERATING_MODE_STS_MASK = 0x80, BF_T0_
OPERATING_MODE_STS_LSB = 7, BF_T0_MAIN_REF_FAIL_SIZE = 1,
BF_T0_MAIN_REF_FAIL_MASK = 0x40, BF_T0_MAIN_REF_FAIL_LSB = 6, BF_IN14_SIZE = 1, BF_IN14_
_MASK = 0x20,
BF_IN14_LSB = 5, BF_IN13_SIZE = 1, BF_IN13_MASK = 0x10, BF_IN13_LSB = 4,
BF_IN12_SIZE = 1, BF_IN12_MASK = 0x08, BF_IN12_LSB = 3, BF_IN11_SIZE = 1,
BF_IN11_MASK = 0x04, BF_IN11_LSB = 2, BF_IN10_SIZE = 1, BF_IN10_MASK = 0x02,
BF_IN10_LSB = 1, BF_IN9_SIZE = 1, BF_IN9_MASK = 0x01, BF_IN9_LSB = 0,
BF_EX_SYNC_ALARM_SIZE = 1, BF_EX_SYNC_ALARM_MASK = 0x80, BF_EX_SYNC_ALARM_LSB =
7, BF_T4_STS_SIZE = 1,
BF_T4_STS_MASK = 0x40, BF_T4_STS_LSB = 6, BF_INPUT_TO_T4_SIZE = 1, BF_INPUT_TO_T4_MA-
SK = 0x10,
BF_INPUT_TO_T4_LSB = 4, BF_AMI2_VIOL_SIZE = 1, BF_AMI2_VIOL_MASK = 0x08, BF_AMI2_VIOL_-
LSB = 3,
BF_AMI2_LOS_SIZE = 1, BF_AMI2_LOS_MASK = 0x04, BF_AMI2_LOS_LSB = 2, BF_AMI1_VIOL_SIZE =
1,
BF_AMI1_VIOL_MASK = 0x02, BF_AMI1_VIOL_LSB = 1, BF_AMI1_LOS_SIZE = 1, BF_AMI1_LOS_MASK
= 0x01,
BF_AMI1_LOS_LSB = 0, BF_PPS_FAST_FREQ_LOCK_EN_SIZE = 1, BF_PPS_FAST_FREQ_LOCK_EN_-
_MASK = 0x10, BF_PPS_FAST_FREQ_LOCK_EN_LSB = 4,
BF_PPS_FAST_PH_LOCK_EN_SIZE = 1, BF_PPS_FAST_PH_LOCK_EN_MASK = 0x08, BF_PPS_FAST_
_PH_LOCK_EN_LSB = 3, BF_MS_SL_CTRL_SIZE = 1,
BF_MS_SL_CTRL_MASK = 0x01, BF_MS_SL_CTRL_LSB = 0, BF_400HZ_SEL_SIZE = 1, BF_400HZ_SE-
L_MASK = 0x40,
BF_400HZ_SEL_LSB = 6, BF_BUCKET_SEL_SIZE = 2, BF_BUCKET_SEL_MASK = 0x30, BF_BUCKET_-
SEL_LSB = 4,
BF_IN_FREQ_SIZE = 4, BF_IN_FREQ_MASK = 0x0F, BF_IN_FREQ_LSB = 0, BF_DIRECT_DIV_SIZE = 1,
BF_DIRECT_DIV_MASK = 0x80, BF_DIRECT_DIV_LSB = 7, BF_LOCK_8K_SIZE = 1, BF_LOCK_8K_MA-
SK = 0x40,
BF_LOCK_8K_LSB = 6, BF_IN6_DIV_SIZE = 2, BF_IN6_DIV_MASK = 0xC0, BF_IN6_DIV_LSB = 6,
BF_IN5_DIV_SIZE = 2, BF_IN5_DIV_MASK = 0x03, BF_IN5_DIV_LSB = 0, BF_PRE_DIV_CH_VALUE_SI-
ZE = 4,
BF_PRE_DIV_CH_VALUE_MASK = 0x0F, BF_PRE_DIV_CH_VALUE_LSB = 0, BF_PRE_DIVN_VALUE_-
A_SIZE = 8, BF_PRE_DIVN_VALUE_A_MASK = 0xFF,
BF_PRE_DIVN_VALUE_A_LSB = 0, BF_PRE_DIVN_VALUE_B_SIZE = 7, BF_PRE_DIVN_VALUE_B_MA-
SK = 0x7F, BF_PRE_DIVN_VALUE_B_LSB = 0,
BF_IN2_SEL_PRIORITY_SIZE = 4, BF_IN2_SEL_PRIORITY_MASK = 0xF0, BF_IN2_SEL_PRIORITY_LS-
B = 4, BF_IN1_SEL_PRIORITY_SIZE = 4,
BF_IN1_SEL_PRIORITY_MASK = 0x0F, BF_IN1_SEL_PRIORITY_LSB = 0, BF_IN4_SEL_PRIORITY_SIZE
= 4, BF_IN4_SEL_PRIORITY_MASK = 0xF0,
BF_IN4_SEL_PRIORITY_LSB = 4, BF_IN3_SEL_PRIORITY_SIZE = 4, BF_IN3_SEL_PRIORITY_MASK =
0x0F, BF_IN3_SEL_PRIORITY_LSB = 0,
BF_IN6_SEL_PRIORITY_SIZE = 4, BF_IN6_SEL_PRIORITY_MASK = 0xF0, BF_IN6_SEL_PRIORITY_LS-
B = 4, BF_IN5_SEL_PRIORITY_SIZE = 4,
BF_IN5_SEL_PRIORITY_MASK = 0x0F, BF_IN5_SEL_PRIORITY_LSB = 0, BF_IN8_SEL_PRIORITY_SIZE
= 4, BF_IN8_SEL_PRIORITY_MASK = 0xF0,
BF_IN8_SEL_PRIORITY_LSB = 4, BF_IN7_SEL_PRIORITY_SIZE = 4, BF_IN7_SEL_PRIORITY_MASK =
0x0F, BF_IN7_SEL_PRIORITY_LSB = 0,
BF_IN10_SEL_PRIORITY_SIZE = 4, BF_IN10_SEL_PRIORITY_MASK = 0xF0, BF_IN10_SEL_PRIORITY-

```

```

_LSB = 4, BF_IN9_SEL_PRIORITY_SIZE = 4,
BF_IN9_SEL_PRIORITY_MASK = 0x0F, BF_IN9_SEL_PRIORITY_LSB = 0, BF_IN12_SEL_PRIORITY_SIZE = 4, BF_IN12_SEL_PRIORITY_MASK = 0xF0,
BF_IN12_SEL_PRIORITY_LSB = 4, BF_IN11_SEL_PRIORITY_SIZE = 4, BF_IN11_SEL_PRIORITY_MASK = 0x0F, BF_IN11_SEL_PRIORITY_LSB = 0,
BF_IN14_SEL_PRIORITY_SIZE = 4, BF_IN14_SEL_PRIORITY_MASK = 0xF0, BF_IN14_SEL_PRIORITY_LSB = 4, BF_IN13_SEL_PRIORITY_SIZE = 4,
BF_IN13_SEL_PRIORITY_MASK = 0x0F, BF_IN13_SEL_PRIORITY_LSB = 0, BF_PAGE_POINTER_SIZE = 1, BF_PAGE_POINTER_MASK = 0x01,
BF_PAGE_POINTER_LSB = 0, BF_FREQ_MON_FACTOR_SIZE = 4, BF_FREQ_MON_FACTOR_MASK = 0x0F, BF_FREQ_MON_FACTOR_LSB = 0,
BF_HARD_FREQ_MON_THRESHOLD_A_SIZE = 4, BF_HARD_FREQ_MON_THRESHOLD_A_MASK = 0xF0, BF_HARD_FREQ_MON_THRESHOLD_A_LSB = 4, BF_HARD_FREQ_MON_THRESHOLD_B_SIZE = 4,
BF_HARD_FREQ_MON_THRESHOLD_B_MASK = 0x0F, BF_HARD_FREQ_MON_THRESHOLD_B_LSB = 0, BF_SOFT_FREQ_MON_THRESHOLD_A_SIZE = 4, BF_SOFT_FREQ_MON_THRESHOLD_A_MASK = 0xF0,
BF_SOFT_FREQ_MON_THRESHOLD_A_LSB = 4, BF_SOFT_FREQ_MON_THRESHOLD_B_SIZE = 4, BF_SOFT_FREQ_MON_THRESHOLD_B_MASK = 0x0F, BF_SOFT_FREQ_MON_THRESHOLD_B_LSB = 0,
BF_UPPER_THRESHOLD_DATA_SIZE = 8, BF_UPPER_THRESHOLD_DATA_MASK = 0xFF, BF_UPPER_THRESHOLD_DATA_LSB = 0, BF_LOWER_THRESHOLD_DATA_SIZE = 8,
BF_LOWER_THRESHOLD_DATA_MASK = 0xFF, BF_LOWER_THRESHOLD_DATA_LSB = 0, BF_BUCKET_SIZE_DATA_SIZE = 8, BF_BUCKET_SIZE_DATA_MASK = 0xFF,
BF_BUCKET_SIZE_DATA_LSB = 0, BF_DECAY_RATE_DATA_SIZE = 2, BF_DECAY_RATE_DATA_MASK = 0x03, BF_DECAY_RATE_DATA_LSB = 0,
BF_IN_FREQ_READ_CH_SIZE = 4, BF_IN_FREQ_READ_CH_MASK = 0x0F, BF_IN_FREQ_READ_CH_LSB = 0, BF_IN_FREQ_VALUE_SIZE = 8,
BF_IN_FREQ_VALUE_MASK = 0xFF, BF_IN_FREQ_VALUE_LSB = 0, BF_IN2_FREQ_SOFT_ALARM_SIZE = 1, BF_IN2_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN2_FREQ_SOFT_ALARM_LSB = 7, BF_IN2_FREQ_HARD_ALARM_SIZE = 1, BF_IN2_FREQ_HARD_ALARM_MASK = 0x40, BF_IN2_FREQ_HARD_ALARM_LSB = 6,
BF_IN2_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN2_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN2_NO_ACTIVITY_ALARM_LSB = 5, BF_IN2_PH_LOCK_ALARM_SIZE = 1,
BF_IN2_PH_LOCK_ALARM_MASK = 0x10, BF_IN2_PH_LOCK_ALARM_LSB = 4, BF_IN1_FREQ_SOFT_ALARM_SIZE = 1, BF_IN1_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN1_FREQ_SOFT_ALARM_LSB = 3, BF_IN1_FREQ_HARD_ALARM_SIZE = 1, BF_IN1_FREQ_HARD_ALARM_MASK = 0x04, BF_IN1_FREQ_HARD_ALARM_LSB = 2,
BF_IN1_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN1_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN1_NO_ACTIVITY_ALARM_LSB = 1, BF_IN1_PH_LOCK_ALARM_SIZE = 1,
BF_IN1_PH_LOCK_ALARM_MASK = 0x01, BF_IN1_PH_LOCK_ALARM_LSB = 0, BF_IN4_FREQ_SOFT_ALARM_SIZE = 1, BF_IN4_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN4_FREQ_SOFT_ALARM_LSB = 7, BF_IN4_FREQ_HARD_ALARM_SIZE = 1, BF_IN4_FREQ_HARD_ALARM_MASK = 0x40, BF_IN4_FREQ_HARD_ALARM_LSB = 6,
BF_IN4_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN4_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN4_NO_ACTIVITY_ALARM_LSB = 5, BF_IN4_PH_LOCK_ALARM_SIZE = 1,
BF_IN4_PH_LOCK_ALARM_MASK = 0x10, BF_IN4_PH_LOCK_ALARM_LSB = 4, BF_IN3_FREQ_SOFT_ALARM_SIZE = 1, BF_IN3_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN3_FREQ_SOFT_ALARM_LSB = 3, BF_IN3_FREQ_HARD_ALARM_SIZE = 1, BF_IN3_FREQ_HARD_ALARM_MASK = 0x04, BF_IN3_FREQ_HARD_ALARM_LSB = 2,
BF_IN3_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN3_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN3_NO_ACTIVITY_ALARM_LSB = 1, BF_IN3_PH_LOCK_ALARM_SIZE = 1,
BF_IN3_PH_LOCK_ALARM_MASK = 0x01, BF_IN3_PH_LOCK_ALARM_LSB = 0, BF_IN6_FREQ_SOFT_ALARM_SIZE = 1, BF_IN6_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN6_FREQ_SOFT_ALARM_LSB = 7, BF_IN6_FREQ_HARD_ALARM_SIZE = 1, BF_IN6_FREQ_HARD_ALARM_MASK = 0x40, BF_IN6_FREQ_HARD_ALARM_LSB = 6,
BF_IN6_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN6_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN6_NO_ACTIVITY_ALARM_LSB = 5,

```

```

ACTIVITY_ALARM_LSB = 5, BF_IN6_PH_LOCK_ALARM_SIZE = 1,
BF_IN6_PH_LOCK_ALARM_MASK = 0x10, BF_IN6_PH_LOCK_ALARM_LSB = 4, BF_IN5_FREQ_SOFT-
_ALARM_SIZE = 1, BF_IN5_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN5_FREQ_SOFT_ALARM_LSB = 3, BF_IN5_FREQ_HARD_ALARM_SIZE = 1, BF_IN5_FREQ_HAR-
D_ALARM_MASK = 0x04, BF_IN5_FREQ_HARD_ALARM_LSB = 2,
BF_IN5_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN5_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN5_NO_-
ACTIVITY_ALARM_LSB = 1, BF_IN5_PH_LOCK_ALARM_SIZE = 1,
BF_IN5_PH_LOCK_ALARM_MASK = 0x01, BF_IN5_PH_LOCK_ALARM_LSB = 0, BF_IN8_FREQ_SOFT-
_ALARM_SIZE = 1, BF_IN8_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN8_FREQ_SOFT_ALARM_LSB = 7, BF_IN8_FREQ_HARD_ALARM_SIZE = 1, BF_IN8_FREQ_HAR-
D_ALARM_MASK = 0x40, BF_IN8_FREQ_HARD_ALARM_LSB = 6,
BF_IN8_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN8_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN8_NO_-
ACTIVITY_ALARM_LSB = 5, BF_IN8_PH_LOCK_ALARM_SIZE = 1,
BF_IN8_PH_LOCK_ALARM_MASK = 0x10, BF_IN8_PH_LOCK_ALARM_LSB = 4, BF_IN7_FREQ_SOFT-
_ALARM_SIZE = 1, BF_IN7_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN7_FREQ_SOFT_ALARM_LSB = 3, BF_IN7_FREQ_HARD_ALARM_SIZE = 1, BF_IN7_FREQ_HAR-
D_ALARM_MASK = 0x04, BF_IN7_FREQ_HARD_ALARM_LSB = 2,
BF_IN7_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN7_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN7_NO_-
ACTIVITY_ALARM_LSB = 1, BF_IN7_PH_LOCK_ALARM_SIZE = 1,
BF_IN7_PH_LOCK_ALARM_MASK = 0x01, BF_IN7_PH_LOCK_ALARM_LSB = 0, BF_IN10_FREQ_SOF-
T_ALARM_SIZE = 1, BF_IN10_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN10_FREQ_SOFT_ALARM_LSB = 7, BF_IN10_FREQ_HARD_ALARM_SIZE = 1, BF_IN10_FREQ_H-
ARD_ALARM_MASK = 0x40, BF_IN10_FREQ_HARD_ALARM_LSB = 6,
BF_IN10_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN10_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN10_-
NO_ACTIVITY_ALARM_LSB = 5, BF_IN10_PH_LOCK_ALARM_SIZE = 1,
BF_IN10_PH_LOCK_ALARM_MASK = 0x10, BF_IN10_PH_LOCK_ALARM_LSB = 4, BF_IN9_FREQ_SO-
FT_ALARM_SIZE = 1, BF_IN9_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN9_FREQ_SOFT_ALARM_LSB = 3, BF_IN9_FREQ_HARD_ALARM_SIZE = 1, BF_IN9_FREQ_HAR-
D_ALARM_MASK = 0x04, BF_IN9_FREQ_HARD_ALARM_LSB = 2,
BF_IN9_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN9_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN9_NO_-
ACTIVITY_ALARM_LSB = 1, BF_IN9_PH_LOCK_ALARM_SIZE = 1,
BF_IN9_PH_LOCK_ALARM_MASK = 0x01, BF_IN9_PH_LOCK_ALARM_LSB = 0, BF_IN12_FREQ_SOF-
T_ALARM_SIZE = 1, BF_IN12_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN12_FREQ_SOFT_ALARM_LSB = 7, BF_IN12_FREQ_HARD_ALARM_SIZE = 1, BF_IN12_FREQ_H-
ARD_ALARM_MASK = 0x40, BF_IN12_FREQ_HARD_ALARM_LSB = 6,
BF_IN12_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN12_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN12_-
NO_ACTIVITY_ALARM_LSB = 5, BF_IN12_PH_LOCK_ALARM_SIZE = 1,
BF_IN12_PH_LOCK_ALARM_MASK = 0x10, BF_IN12_PH_LOCK_ALARM_LSB = 4, BF_IN11_FREQ_S-
OFT_ALARM_SIZE = 1, BF_IN11_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN11_FREQ_SOFT_ALARM_LSB = 3, BF_IN11_FREQ_HARD_ALARM_SIZE = 1, BF_IN11_FREQ_H-
ARD_ALARM_MASK = 0x04, BF_IN11_FREQ_HARD_ALARM_LSB = 2,
BF_IN11_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN11_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN11_-
NO_ACTIVITY_ALARM_LSB = 1, BF_IN11_PH_LOCK_ALARM_SIZE = 1,
BF_IN11_PH_LOCK_ALARM_MASK = 0x01, BF_IN11_PH_LOCK_ALARM_LSB = 0, BF_IN14_FREQ_S-
OFT_ALARM_SIZE = 1, BF_IN14_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN14_FREQ_SOFT_ALARM_LSB = 7, BF_IN14_FREQ_HARD_ALARM_SIZE = 1, BF_IN14_FREQ_H-
ARD_ALARM_MASK = 0x40, BF_IN14_FREQ_HARD_ALARM_LSB = 6,
BF_IN14_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN14_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN14_-
NO_ACTIVITY_ALARM_LSB = 5, BF_IN14_PH_LOCK_ALARM_SIZE = 1,
BF_IN14_PH_LOCK_ALARM_MASK = 0x10, BF_IN14_PH_LOCK_ALARM_LSB = 4, BF_IN13_FREQ_S-
OFT_ALARM_SIZE = 1, BF_IN13_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN13_FREQ_SOFT_ALARM_LSB = 3, BF_IN13_FREQ_HARD_ALARM_SIZE = 1, BF_IN13_FREQ_H-
ARD_ALARM_MASK = 0x04, BF_IN13_FREQ_HARD_ALARM_LSB = 2,
BF_IN13_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN13_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN13_-
NO_ACTIVITY_ALARM_LSB = 1, BF_IN13_PH_LOCK_ALARM_SIZE = 1,
BF_IN13_PH_LOCK_ALARM_MASK = 0x01, BF_IN13_PH_LOCK_ALARM_LSB = 0, BF_HIGHEST_PRI-
ORITY_VALIDATED_SIZE = 4, BF_HIGHEST_PRIORITY_VALIDATED_MASK = 0xF0,
BF_HIGHEST_PRIORITY_VALIDATED_LSB = 4, BF_CURRENTLY_SELECTED_INPUTS_SIZE = 4, BF_-

```

CURRENTLY\_SELECTED\_INPUTS\_MASK = 0x0F, BF\_CURRENTLY\_SELECTED\_INPUTS\_LSB = 0,  
 BF\_THIRD\_HIGHEST\_PRIORITY\_VALIDATED\_SIZE = 4, BF\_THIRD\_HIGHEST\_PRIORITY\_VALIDATE-  
 D\_MASK = 0xF0, BF\_THIRD\_HIGHEST\_PRIORITY\_VALIDATED\_LSB = 4, BF\_SECOND\_HIGHEST\_PRI-  
 ORITY\_VALIDATED\_SIZE = 4,  
 BF\_SECOND\_HIGHEST\_PRIORITY\_VALIDATED\_MASK = 0x0F, BF\_SECOND\_HIGHEST\_PRIORITY\_V-  
 ALIDATED\_LSB = 0, BF\_T0\_INPUT\_SEL\_SIZE = 4, BF\_T0\_INPUT\_SEL\_MASK = 0x0F,  
 BF\_T0\_INPUT\_SEL\_LSB = 0, BF\_T4\_LOCK\_T0\_SIZE = 1, BF\_T4\_LOCK\_T0\_MASK = 0x40, BF\_T4\_LOC-  
 K\_T0\_LSB = 6,  
 BF\_T0\_FOR\_T4\_SIZE = 1, BF\_T0\_FOR\_T4\_MASK = 0x20, BF\_T0\_FOR\_T4\_LSB = 5, BF\_T4\_TEST\_T0\_-  
 PH\_SIZE = 1,  
 BF\_T4\_TEST\_T0\_PH\_MASK = 0x10, BF\_T4\_TEST\_T0\_PH\_LSB = 4, BF\_T4\_INPUT\_SEL\_SIZE = 4, BF\_-  
 T4\_INPUT\_SEL\_MASK = 0x0F,  
 BF\_T4\_INPUT\_SEL\_LSB = 0, BF\_EX\_SYNC\_ALARM\_MON\_SIZE = 1, BF\_EX\_SYNC\_ALARM\_MON\_M-  
 ASK = 0x80, BF\_EX\_SYNC\_ALARM\_MON\_LSB = 7,  
 BF\_T4\_DPLL\_LOCK\_SIZE = 1, BF\_T4\_DPLL\_LOCK\_MASK = 0x40, BF\_T4\_DPLL\_LOCK\_LSB = 6, BF\_-  
 T0\_DPLL\_SOFT\_FREQ\_ALARM\_SIZE = 1,  
 BF\_T0\_DPLL\_SOFT\_FREQ\_ALARM\_MASK = 0x20, BF\_T0\_DPLL\_SOFT\_FREQ\_ALARM\_LSB = 5, BF\_-  
 T4\_DPLL\_SOFT\_FREQ\_ALARM\_SIZE = 1, BF\_T4\_DPLL\_SOFT\_FREQ\_ALARM\_MASK = 0x10,  
 BF\_T4\_DPLL\_SOFT\_FREQ\_ALARM\_LSB = 4, BF\_T0\_DPLL\_LOCK\_SIZE = 1, BF\_T0\_DPLL\_LOCK\_MA-  
 SK = 0x08, BF\_T0\_DPLL\_LOCK\_LSB = 3,  
 BF\_T0\_DPLL\_OPERATING\_MODE\_SIZE = 3, BF\_T0\_DPLL\_OPERATING\_MODE\_MASK = 0x07, BF\_T0\_-  
 DPLL\_OPERATING\_MODE\_LSB = 0, BF\_T0\_WDCO\_SIZE = 1,  
 BF\_T0\_WDCO\_MASK = 0x08, BF\_T0\_WDCO\_LSB = 3, BF\_T0\_OPERATING\_MODE\_SIZE = 3, BF\_T0\_-  
 OPERATING\_MODE\_MASK = 0x07,  
 BF\_T0\_OPERATING\_MODE\_LSB = 0, BF\_T4\_WDCO\_SIZE = 1, BF\_T4\_WDCO\_MASK = 0x08, BF\_T4\_-  
 WDCO\_LSB = 3,  
 BF\_T4\_OPERATING\_MODE\_SIZE = 3, BF\_T4\_OPERATING\_MODE\_MASK = 0x07, BF\_T4\_OPERATIN-  
 G\_MODE\_LSB = 0, BF\_T0\_APLL\_PATH\_SIZE = 4,  
 BF\_T0\_APLL\_PATH\_MASK = 0xF0, BF\_T0\_APLL\_PATH\_LSB = 4, BF\_T0\_GSM\_OBSAI\_16E1\_16T1\_SE-  
 L\_SIZE = 2, BF\_T0\_GSM\_OBSAI\_16E1\_16T1\_SEL\_MASK = 0x0C,  
 BF\_T0\_GSM\_OBSAI\_16E1\_16T1\_SEL\_LSB = 2, BF\_T0\_12E1\_GPS\_E3\_T3\_SEL\_SIZE = 2, BF\_T0\_12-  
 E1\_GPS\_E3\_T3\_SEL\_MASK = 0x03, BF\_T0\_12E1\_GPS\_E3\_T3\_SEL\_LSB = 0,  
 BF\_T0\_DPLL\_START\_DAMPING\_SIZE = 3, BF\_T0\_DPLL\_START\_DAMPING\_MASK = 0xE0, BF\_T0\_D-  
 PLL\_START\_DAMPING\_LSB = 5, BF\_T0\_DPLL\_START\_BW\_SIZE = 5,  
 BF\_T0\_DPLL\_START\_BW\_MASK = 0x1F, BF\_T0\_DPLL\_START\_BW\_LSB = 0, BF\_T0\_DPLL\_ACQ\_DAM-  
 PING\_SIZE = 3, BF\_T0\_DPLL\_ACQ\_DAMPING\_MASK = 0xE0,  
 BF\_T0\_DPLL\_ACQ\_DAMPING\_LSB = 5, BF\_T0\_DPLL\_ACQ\_BW\_SIZE = 5, BF\_T0\_DPLL\_ACQ\_BW\_M-  
 ASK = 0x1F, BF\_T0\_DPLL\_ACQ\_BW\_LSB = 0,  
 BF\_T0\_DPLL\_LOCKED\_DAMPING\_SIZE = 3, BF\_T0\_DPLL\_LOCKED\_DAMPING\_MASK = 0xE0, BF\_T0\_-  
 DPLL\_LOCKED\_DAMPING\_LSB = 5, BF\_T0\_DPLL\_LOCKED\_BW\_SIZE = 5,  
 BF\_T0\_DPLL\_LOCKED\_BW\_MASK = 0x1F, BF\_T0\_DPLL\_LOCKED\_BW\_LSB = 0, BF\_AUTO\_BW\_SEL\_-  
 SIZE = 1, BF\_AUTO\_BW\_SEL\_MASK = 0x80,  
 BF\_AUTO\_BW\_SEL\_LSB = 7, BF\_T0\_LIMIT\_SIZE = 1, BF\_T0\_LIMIT\_MASK = 0x08, BF\_T0\_LIMIT\_LSB =  
 3,  
 BF\_COARSE\_PH\_LOS\_LIMT\_EN\_SIZE = 1, BF\_COARSE\_PH\_LOS\_LIMT\_EN\_MASK = 0x80, BF\_COA-  
 RSE\_PH\_LOS\_LIMT\_EN\_LSB = 7, BF\_WIDE\_EN\_SIZE = 1,  
 BF\_WIDE\_EN\_MASK = 0x40, BF\_WIDE\_EN\_LSB = 6, BF\_MULTI\_PH\_APP\_SIZE = 1, BF\_MULTI\_PH\_A-  
 PP\_MASK = 0x20,  
 BF\_MULTI\_PH\_APP\_LSB = 5, BF\_MULTI\_PH\_8K\_4K\_2K\_EN\_SIZE = 1, BF\_MULTI\_PH\_8K\_4K\_2K\_EN-  
 MASK = 0x10, BF\_MULTI\_PH\_8K\_4K\_2K\_EN\_LSB = 4,  
 BF\_PH\_LOS\_COARSE\_LIMT\_SIZE = 4, BF\_PH\_LOS\_COARSE\_LIMT\_MASK = 0x0F, BF\_PH\_LOS\_COA-  
 RSE\_LIMT\_LSB = 0, BF\_FINE\_PH\_LOS\_LIMT\_EN\_SIZE = 1,  
 BF\_FINE\_PH\_LOS\_LIMT\_EN\_MASK = 0x80, BF\_FINE\_PH\_LOS\_LIMT\_EN\_LSB = 7, BF\_FAST\_LOS\_S-  
 W\_SIZE = 1, BF\_FAST\_LOS\_SW\_MASK = 0x40,  
 BF\_FAST\_LOS\_SW\_LSB = 6, BF\_PH\_LOS\_FINE\_LIMT\_SIZE = 3, BF\_PH\_LOS\_FINE\_LIMT\_MASK =  
 0x07, BF\_PH\_LOS\_FINE\_LIMT\_LSB = 0,  
 BF\_MAN\_HOLDOVER\_SIZE = 1, BF\_MAN\_HOLDOVER\_MASK = 0x80, BF\_MAN\_HOLDOVER\_LSB = 7,

```

BF_AUTO_AVG_SIZE = 1,
BF_AUTO_AVG_MASK = 0x40, BF_AUTO_AVG_LSB = 6, BF_FAST_AVG_SIZE = 1, BF_FAST_AVG_M-
ASK = 0x20,
BF_FAST_AVG_LSB = 5, BF_READ_AVG_SIZE = 1, BF_READ_AVG_MASK = 0x10, BF_READ_AVG_L-
SB = 4,
BF_TEMP_HOLD OVER_MODE_SIZE = 2, BF_TEMP_HOLD OVER_MODE_MASK = 0x0C, BF_TEMP_H-
OLD OVER_MODE_LSB = 2, BF_HOLD OVER_FREQ_A_SIZE = 8,
BF_HOLD OVER_FREQ_A_MASK = 0xFF, BF_HOLD OVER_FREQ_A_LSB = 0, BF_HOLD OVER_FREQ_-
B_SIZE = 8, BF_HOLD OVER_FREQ_B_MASK = 0xFF,
BF_HOLD OVER_FREQ_B_LSB = 0, BF_HOLD OVER_FREQ_C_SIZE = 8, BF_HOLD OVER_FREQ_C_M-
ASK = 0xFF, BF_HOLD OVER_FREQ_C_LSB = 0,
BF_T4_APLL_PATH_SIZE = 4, BF_T4_APLL_PATH_MASK = 0xF0, BF_T4_APLL_PATH_LSB = 4, BF_-
T4_GSM_OBSAI_16E1_16T1_SEL_SIZE = 2,
BF_T4_GSM_OBSAI_16E1_16T1_SEL_MASK = 0x0C, BF_T4_GSM_OBSAI_16E1_16T1_SEL_LSB = 2,
BF_T4_12E1_GPS_E3_T3_SEL_SIZE = 2, BF_T4_12E1_GPS_E3_T3_SEL_MASK = 0x03,
BF_T4_12E1_GPS_E3_T3_SEL_LSB = 0, BF_T4_DPLL_LOCKED_DAMPING_SIZE = 3, BF_T4_DPLL_L-
OCKED_DAMPING_MASK = 0xE0, BF_T4_DPLL_LOCKED_DAMPING_LSB = 5,
BF_T4_DPLL_LOCKED_BW_SIZE = 2, BF_T4_DPLL_LOCKED_BW_MASK = 0x03, BF_T4_DPLL_LOCK-
ED_BW_LSB = 0, BF_CURRENT_DPLL_FREQ_A_SIZE = 8,
BF_CURRENT_DPLL_FREQ_A_MASK = 0xFF, BF_CURRENT_DPLL_FREQ_A_LSB = 0, BF_CURRENT_-
DPLL_FREQ_B_SIZE = 8, BF_CURRENT_DPLL_FREQ_B_MASK = 0xFF,
BF_CURRENT_DPLL_FREQ_B_LSB = 0, BF_CURRENT_DPLL_FREQ_C_SIZE = 8, BF_CURRENT_DP-
LL_FREQ_C_MASK = 0xFF, BF_CURRENT_DPLL_FREQ_C_LSB = 0,
BF_FREQ_LIMT_PH_LOS_SIZE = 1, BF_FREQ_LIMT_PH_LOS_MASK = 0x80, BF_FREQ_LIMT_PH_LO-
S_LSB = 7, BF_DPLL_FREQ_SOFT_LIMT_SIZE = 7,
BF_DPLL_FREQ_SOFT_LIMT_MASK = 0x7F, BF_DPLL_FREQ_SOFT_LIMT_LSB = 0, BF_DPLL_FREQ_-
HARD_LIMT_A_SIZE = 8, BF_DPLL_FREQ_HARD_LIMT_A_MASK = 0xFF,
BF_DPLL_FREQ_HARD_LIMT_A_LSB = 0, BF_DPLL_FREQ_HARD_LIMT_B_SIZE = 8, BF_DPLL_FRE-
Q_HARD_LIMT_B_MASK = 0xFF, BF_DPLL_FREQ_HARD_LIMT_B_LSB = 0,
BF_CURRENT_PH_DATA_A_SIZE = 8, BF_CURRENT_PH_DATA_A_MASK = 0xFF, BF_CURRENT_PH_-
DATA_A_LSB = 0, BF_CURRENT_PH_DATA_B_SIZE = 8,
BF_CURRENT_PH_DATA_B_MASK = 0xFF, BF_CURRENT_PH_DATA_B_LSB = 0, BF_OUT1_PATH_S-
EL_SIZE = 4, BF_OUT1_PATH_SEL_MASK = 0xF0,
BF_OUT1_PATH_SEL_LSB = 4, BF_OUT1_DIVIDER_SIZE = 4, BF_OUT1_DIVIDER_MASK = 0x0F, BF_-
OUT1_DIVIDER_LSB = 0,
BF_OUT2_PATH_SEL_SIZE = 4, BF_OUT2_PATH_SEL_MASK = 0xF0, BF_OUT2_PATH_SEL_LSB = 4,
BF_OUT2_DIVIDER_SIZE = 4,
BF_OUT2_DIVIDER_MASK = 0x0F, BF_OUT2_DIVIDER_LSB = 0, BF_OUT3_PATH_SEL_SIZE = 4, BF_-
OUT3_PATH_SEL_MASK = 0xF0,
BF_OUT3_PATH_SEL_LSB = 4, BF_OUT3_DIVIDER_SIZE = 4, BF_OUT3_DIVIDER_MASK = 0x0F, BF_-
OUT3_DIVIDER_LSB = 0,
BF_OUT4_PATH_SEL_SIZE = 4, BF_OUT4_PATH_SEL_MASK = 0xF0, BF_OUT4_PATH_SEL_LSB = 4,
BF_OUT4_DIVIDER_SIZE = 4,
BF_OUT4_DIVIDER_MASK = 0x0F, BF_OUT4_DIVIDER_LSB = 0, BF_OUT5_PATH_SEL_SIZE = 4, BF_-
OUT5_PATH_SEL_MASK = 0xF0,
BF_OUT5_PATH_SEL_LSB = 4, BF_OUT5_DIVIDER_SIZE = 4, BF_OUT5_DIVIDER_MASK = 0x0F, BF_-
OUT5_DIVIDER_LSB = 0,
BF_OUT6_PATH_SEL_SIZE = 4, BF_OUT6_PATH_SEL_MASK = 0xF0, BF_OUT6_PATH_SEL_LSB = 4,
BF_OUT6_DIVIDER_SIZE = 4,
BF_OUT6_DIVIDER_MASK = 0x0F, BF_OUT6_DIVIDER_LSB = 0, BF_OUT7_PATH_SEL_SIZE = 4, BF_-
OUT7_PATH_SEL_MASK = 0xF0,
BF_OUT7_PATH_SEL_LSB = 4, BF_OUT7_DIVIDER_SIZE = 4, BF_OUT7_DIVIDER_MASK = 0x0F, BF_-
OUT7_DIVIDER_LSB = 0,
BF_OUT8_PATH_SEL_SIZE = 1, BF_OUT8_PATH_SEL_MASK = 0x80, BF_OUT8_PATH_SEL_LSB = 7,
BF_OUT8_EN_SIZE = 1,
BF_OUT8_EN_MASK = 0x40, BF_OUT8_EN_LSB = 6, BF_T4_INPUT_FAIL_SIZE = 1, BF_T4_INPUT_FA-
IL_MASK = 0x20,
BF_T4_INPUT_FAIL_LSB = 5, BF_AMI_OUT_DUTY_SIZE = 1, BF_AMI_OUT_DUTY_MASK = 0x10, BF_-

```

```

AMI_OUT_DUTY_LSB = 4,
BF_400HZ_OUT_SEL_SIZE = 1, BF_400HZ_OUT_SEL_MASK = 0x08, BF_400HZ_OUT_SEL_LSB = 3,
BF_OUT9_INV_SIZE = 1,
BF_OUT9_INV_MASK = 0x04, BF_OUT9_INV_LSB = 2, BF_OUT7_INV_SIZE = 1, BF_OUT7_INV_MASK
= 0x02,
BF_OUT7_INV_LSB = 1, BF_OUT6_INV_SIZE = 1, BF_OUT6_INV_MASK = 0x01, BF_OUT6_INV_LSB =
0,
BF_OUT9_PATH_SEL_SIZE = 1, BF_OUT9_PATH_SEL_MASK = 0x80, BF_OUT9_PATH_SEL_LSB = 7,
BF_OUT9_EN_SIZE = 1,
BF_OUT9_EN_MASK = 0x40, BF_OUT9_EN_LSB = 6, BF_OUT5_INV_SIZE = 1, BF_OUT5_INV_MASK =
0x10,
BF_OUT5_INV_LSB = 4, BF_OUT4_INV_SIZE = 1, BF_OUT4_INV_MASK = 0x08, BF_OUT4_INV_LSB =
3,
BF_OUT3_INV_SIZE = 1, BF_OUT3_INV_MASK = 0x04, BF_OUT3_INV_LSB = 2, BF_OUT2_INV_SIZE =
1,
BF_OUT2_INV_MASK = 0x02, BF_OUT2_INV_LSB = 1, BF_OUT1_INV_SIZE = 1, BF_OUT1_INV_MASK
= 0x01,
BF_OUT1_INV_LSB = 0, BF_IN_2K_4K_8K_INV_SIZE = 1, BF_IN_2K_4K_8K_INV_MASK = 0x80, BF_IN-
_2K_4K_8K_INV_LSB = 7,
BF_8K_EN_SIZE = 1, BF_8K_EN_MASK = 0x40, BF_8K_EN_LSB = 6, BF_2K_EN_SIZE = 1,
BF_2K_EN_MASK = 0x20, BF_2K_EN_LSB = 5, BF_2K_8K_PUL_POSITION_SIZE = 1, BF_2K_8K_PUL_-
POSITION_MASK = 0x10,
BF_2K_8K_PUL_POSITION_LSB = 4, BF_8K_INV_SIZE = 1, BF_8K_INV_MASK = 0x08, BF_8K_INV_LSB
= 3,
BF_8K_PUL_SIZE = 1, BF_8K_PUL_MASK = 0x04, BF_8K_PUL_LSB = 2, BF_2K_INV_SIZE = 1,
BF_2K_INV_MASK = 0x02, BF_2K_INV_LSB = 1, BF_2K_PUL_SIZE = 1, BF_2K_PUL_MASK = 0x01,
BF_2K_PUL_LSB = 0, BF_IN_NOISE_WINDOW_SIZE = 1, BF_IN_NOISE_WINDOW_MASK = 0x80, BF_I-
N_NOISE_WINDOW_LSB = 7,
BF_PH_OFFSET_A_SIZE = 8, BF_PH_OFFSET_A_MASK = 0xFF, BF_PH_OFFSET_A_LSB = 0, BF_PH_-
OFFSET_EN_SIZE = 1,
BF_PH_OFFSET_EN_MASK = 0x80, BF_PH_OFFSET_EN_LSB = 7, BF_PH_OFFSET_B_SIZE = 2, BF_-
PH_OFFSET_B_MASK = 0x03,
BF_PH_OFFSET_B_LSB = 0, BF_SYNC_BYPASS_SIZE = 1, BF_SYNC_BYPASS_MASK = 0x80, BF_SY-
NC_BYPASS_LSB = 7,
BF_SYNC_MON_LIMT_SIZE = 3, BF_SYNC_MON_LIMT_MASK = 0x70, BF_SYNC_MON_LIMT_LSB = 4,
BF_SYNC_EDGE_SIZE = 1,
BF_SYNC_EDGE_MASK = 0x40, BF_SYNC_EDGE_LSB = 6, BF_SYNC_PH2_SIZE = 2, BF_SYNC_PH2-
_MASK = 0x0C,
BF_SYNC_PH2_LSB = 2, BF_SYNC_PH1_SIZE = 2, BF_SYNC_PH1_MASK = 0x03, BF_SYNC_PH1_LSB
= 0,
BF_PROTECTION_DATA_SIZE = 8, BF_PROTECTION_DATA_MASK = 0xFF, BF_PROTECTION_DATA-
_LSB = 0, BF_MPU_SEL_CNFG_SIZE = 3,
BF_MPU_SEL_CNFG_MASK = 0x07, BF_MPU_SEL_CNFG_LSB = 0, BF_T0_DFS_OFF_3_SIZE = 1,
BF_T0_DFS_OFF_3_MASK = 0x80,
BF_T0_DFS_OFF_3_LSB = 7, BF_T0_DFS_OFF_2_SIZE = 1, BF_T0_DFS_OFF_2_MASK = 0x40, BF_-
T0_DFS_OFF_2_LSB = 6,
BF_T0_DFS_OFF_1_SIZE = 1, BF_T0_DFS_OFF_1_MASK = 0x20, BF_T0_DFS_OFF_1_LSB = 5, BF_-
T0_DFS_OFF_0_SIZE = 1,
BF_T0_DFS_OFF_0_MASK = 0x10, BF_T0_DFS_OFF_0_LSB = 4, BF_T4_DFS_OFF_3_SIZE = 1, BF_-
T4_DFS_OFF_3_MASK = 0x08,
BF_T4_DFS_OFF_3_LSB = 3, BF_T4_DFS_OFF_2_SIZE = 1, BF_T4_DFS_OFF_2_MASK = 0x04, BF_-
T4_DFS_OFF_2_LSB = 2,
BF_T4_DFS_OFF_1_SIZE = 1, BF_T4_DFS_OFF_1_MASK = 0x02, BF_T4_DFS_OFF_1_LSB = 1, BF_-
T4_DFS_OFF_0_SIZE = 1,
BF_T4_DFS_OFF_0_MASK = 0x01, BF_T4_DFS_OFF_0_LSB = 0, BF_PPS_PHASE_SIZE = 2, BF_PPS-
_PHASE_MASK = 0x30,
BF_PPS_PHASE_LSB = 4, BF_PPS_PULSE_SIZE = 4, BF_PPS_PULSE_MASK = 0x0F, BF_PPS_PULS-

```

```

E_LSB = 0,
BF_T0_PH_SLOPE_SIZE = 2, BF_T0_PH_SLOPE_MASK = 0x0C, BF_T0_PH_SLOPE_LSB = 2, BF_T4_
PH_SLOPE_SIZE = 2,
BF_T4_PH_SLOPE_MASK = 0x03, BF_T4_PH_SLOPE_LSB = 0, BF_ICP_CTRL_CODE_SIZE = 5, BF_IC-
P_CTRL_CODE_MASK = 0x1F,
BF_ICP_CTRL_CODE_LSB = 0, REGMAP_MAX }

```

*Enumerated type listing register offsets and bit field constants.*

### 6.40.1 Detailed Description

This header file contains register and bitfield information for accessing device.

#### See Also

Datasheet for detailed register descriptions.

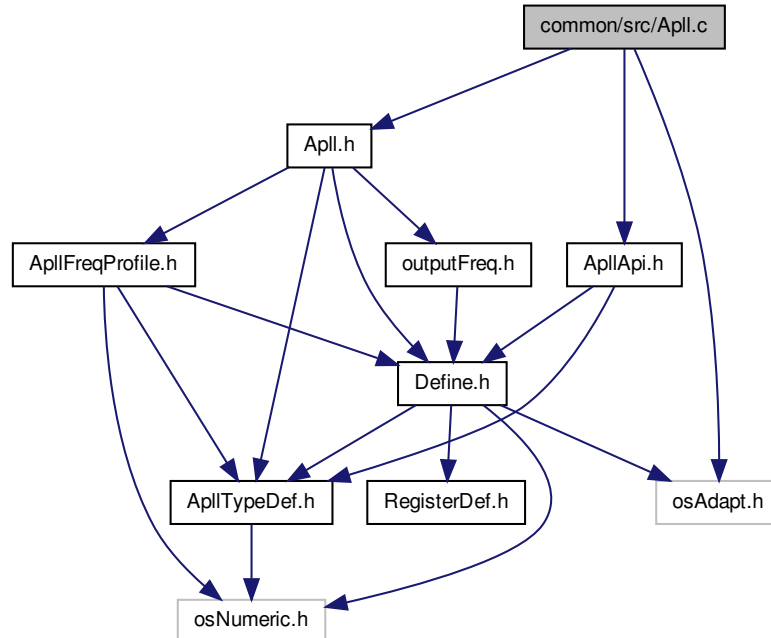
Definition in file [RegisterDef.h](#).

## 6.41 common/src/Apll.c File Reference

```

#include "Apll.h"
#include "ApllApi.h"
#include "osAdapt.h"
Include dependency graph for Apll.c:

```



### Functions

- void [IDTApll\\_SetApllConfigPerOutput](#) (T\_idtDrvHdlr hdlr, T\_idtApllId appllInstId, T\_idtApllOutput output, T\_idtApllConfigPerOutput \*appllConfigPerOutput)

*Set APLL per output configuration.*

- void `IDTApl_GetApllConfigPerOutput` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllInstId, `T_idtApllOutput` output, `T_idtApllConfigPerOutput` \*apllConfigPerOutput)

*Get APLL per output configuration.*

- void `IDTApl_SetApllConfig` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllInstId, `T_idtApllConfig` \*apllConfig)

*Set APLL full configuration.*

- void `IDTApl_GetApllConfig` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllInstId, `T_idtApllConfig` \*apllConfig)

*Get APLL full configuration.*

- `outputFrequency_t` `IDTApl_ApllInput2DpllOutput` (`T_idtApllInputFreqType` apllInputFreq)

*Translate APLL input frequency to DPLL output frequency.*

- `T_idtApllOutputFreqType` `IDTApl_DpllOutput2ApllOutput` (`outputFrequency_t` dpllOutput)

*Translate DPLL output frequency to APLL output frequency.*

- const `T_idtApllFreqMapping` \* `IDTApl_GetApllFreqTableEntry` (`T_idtDrvHdlr` hdlr, `T_idtOutputApllConfig` outputApll, `outputFrequency_t` nOutFreq)

*Get APLL frequency configuration table entry based on the output frequency.*

- `T_idtApllConfigSel` `IDTApl_GetNonActiveApllConfig` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllInst)

*Get non-active APLL configuration.*

- void `IDTApl_Switch2NonActiveApllConfig` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllId)

*Switch to non-active APLL configuration.*

- `T_idtApllXtalId` `IDTApl_GetXtalIdFromXtalFreq` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllInst, `T_idtApllXtalType` apllVcxoFreq)

*Get APLL configured crystal ID from crystal frequency.*

- `T_osBool` `IDTApl_XtalConfigured` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllId)

*Checks if there is any XTAL configured for the APLL.*

- `T_osBool` `IDTApl_ValidXtalInput` (const `T_idtDrvDeviceConfig` \*deviceConfig, `T_idtApllXtal` \*xtalList, `T_osUInt8` numberOfXtal)

*Checks if there is input XTAL information is valid for the device.*

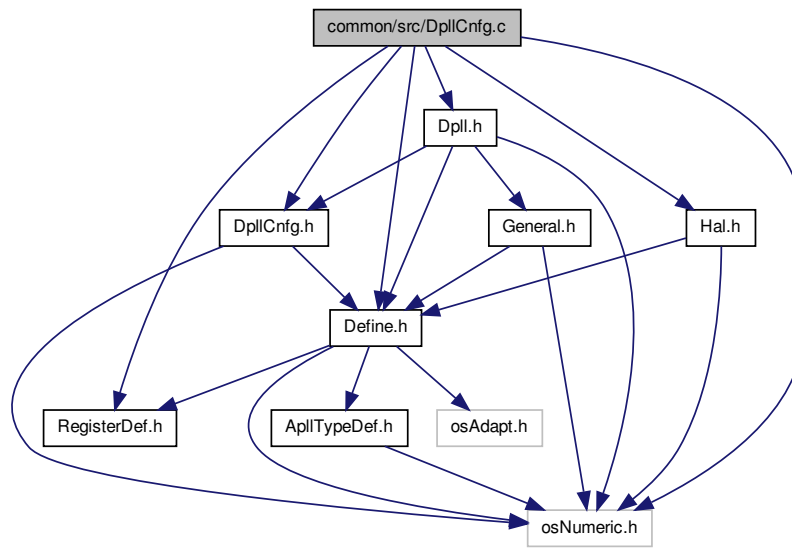
- `T_idtApllXtal` \* `IDTApl_GetSupportedXtal` (const `T_idtDrvDeviceConfig` \*deviceConfig, `T_osUInt8` \*numberOfXtal)

*Return supported xtal configuration by the device.*

## 6.42 common/src/DpllCnfg.c File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "Hal.h"
#include "DpllCnfg.h"
#include "RegisterDef.h"
#include "Dpll.h"
```

Include dependency graph for DpllCnfg.c:



## Data Structures

- struct [T\\_idtSonetGigEthBitfield2PathConfig](#)  
Structure Translate from Sonet/GigE bitfield to path configuration.

## Macros

- #define [INVALID\\_REGISTER\\_OFFSET](#) ((T\_osUInt8)(-1))  
Tag used to indicating invalid register offset.

## Functions

- [T\\_idtDpllType IDTDpll\\_GetTypeOfDpll](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
Function to determine type of DPLL give a DPLL instance.
- void [IDTDpll\\_SetAutoSelInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 mode)  
Set the DPLL as automatic reference clock selection.
- T\_osUInt8 [IDTDpll\\_GetAutoSelInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
Get automatic input selection status.
- void [IDTDpll\\_SetForceSelInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nInputNo)  
Set the DPLL to force to lock to a specified reference clock.
- T\_osUInt8 [IDTDpll\\_GetForceSelInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
Set the DPLL to force to lock to a specified reference clock.
- T\_osUInt8 [IDTDpll\\_Get1stPriorityInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
Get input number of a valid clock with the highest priority.
- T\_osUInt8 [IDTDpll\\_Get2ndPriorityInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
Get input number of a valid clock with the second highest priority.
- T\_osUInt8 [IDTDpll\\_Get3rdPriorityInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
Get input number of a valid clock with the third highest priority.

- T\_osUInt8 [IDTDpll\\_GetCurrentSelectInput](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get the current selected input clock.*
- void [IDTDpll\\_SetDpllOperMod](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtDpllOperMode nMode)  
*Set DPLL working at the specified operating mode.*
- T\_idtDpllOperMode [IDTDpll\\_GetDpllOperMod](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Set DPLL working at the specified operating mode.*
- T\_osUInt8 [IDTDpll\\_IsDpllLocked](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Inquire if the DPLL is in lock state.*
- void [IDTDpll\\_SetBandWidthDamping](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtBandwidthLimitStage stage, T\_idtDpllBandwidth nBandWidth, T\_idtDampingFactor nDamping)  
*Set the bandwidth and damping factor used for bandwidth limiting.*
- void [IDTDpll\\_GetBandWidthDamping](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtBandwidthLimitStage stage, T\_idtDpllBandwidth \*nBandWidth\_p, T\_idtDampingFactor \*nDamping\_p)  
*Get configured bandwidth and damping factor for bandwidth limiting.*
- void [IDTDpll\\_SetAutoSelBandWidth](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nEnable)  
*Set device to select a suitable bandwidth and damping factor automatically.*
- T\_osUInt8 [IDTDpll\\_GetAutoSelBandWidth](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get auto bandwidth/damping factor status automatically.*
- void [IDTDpll\\_SetDpllFrozen](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nEnable)  
*Set DPLL integral path value frozen.*
- T\_osUInt8 [IDTDpll\\_GetDpllFrozen](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get DPLL integral path value frozen.*
- void [IDTDpll\\_SetPhaseCoarseDetector](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nEnable, T\_osUInt8 nWide, T\_osUInt8 nMultiPhase, T\_osUInt8 nMultiPh8kEn)  
*Set the coarse phase lose detector enable.*
- void [IDTDpll\\_GetPhaseCoarseDetector](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 \*nEnable\_p, T\_osUInt8 \*nWide\_p, T\_osUInt8 \*nMultiPhase\_p, T\_osUInt8 \*nMultiPh8kEn\_p)  
*Set the coarse phase lose detector enable.*
- void [IDTDpll\\_SetPhaseCoarseLimit](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtCoarsePhaseLimit nLimit)  
*Set the limitation of the coarse phase lose detector.*
- T\_idtCoarsePhaseLimit [IDTDpll\\_GetPhaseCoarseLimit](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get the limitation of the coarse phase lose detector.*
- void [IDTDpll\\_SetPhaseFineDetectorEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_osUInt8 nEnable)  
*Set the fine phase lose detector enable.*
- T\_osUInt8 [IDTDpll\\_GetPhaseFineDetectorEnable](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Get the fine phase lose detector enable.*
- void [IDTDpll\\_SetPhaseFineLimit](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtFinePhaseLimit nLimit)  
*Set the limitation of the fine phase lose detector.*
- T\_idtFinePhaseLimit [IDTDpll\\_GetPhaseFineLimit](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll)  
*Set the limitation of the fine phase lose detector.*
- void [IDTDpll\\_SetFastLossSwitch](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
*Set reference clock switch if a fast loss occurs.*
- T\_osUInt8 [IDTDpll\\_GetFastLossSwitch](#) (T\_idtDrvHdlr hdlr)  
*Get reference clock switch if a fast loss occurs.*
- void [IDTDpll\\_SetHoldoverMode](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtDpllHoldover nMode, T\_osUInt8 nReadMode)  
*Set calculation mode of holdover frequency and read back mode.*
- void [IDTDpll\\_GetHoldoverMode](#) (T\_idtDrvHdlr hdlr, T\_idtDpllInstance nDpll, T\_idtDpllHoldover \*nMode\_p, T\_osUInt8 \*nReadMode\_p)  
*Get calculation mode of holdover frequency and read back mode.*

- void [IDTDpll\\_SetTempHoldoverMode](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtTemp\\_holdover](#) n-Mode)  
*Set calculation mode of temporary holdover frequency.*
- [T\\_idtTemp\\_holdover](#) [IDTDpll\\_GetTempHoldoverMode](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
*Set calculation mode of temporary holdover frequency.*
- long [IDTDpll\\_GetHoldoverFreq](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
*Get holdover frequency offset (in parts per trillion). In DCO mode, this function returns current DCO offset (in parts per trillion).*
- void [IDTDpll\\_SetHoldoverFreq](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, long pptOffset)  
*Set DCO frequency to device. Note this function should only be called when DCO mode is enabled.*
- long [IDTDpll\\_GetCurrentDpllFreqOffset](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
*Get current frequency offset of DPLL in parts per trillion (PPT)*
- void [IDTDpll\\_SetDpllSoftAlarmLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, T\_osUInt8 nLimit)  
*Set the limitation of Soft alarm of DPLL.*
- T\_osUInt8 [IDTDpll\\_GetDpllSoftAlarmLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
*Get the limitation of Soft alarm of DPLL.*
- void [IDTDpll\\_SetDpllHardAlarmLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, T\_osUInt16 nLimit)  
*Set the limitation of Hard alarm of DPLL.*
- T\_osUInt16 [IDTDpll\\_GetDpllHardAlarmLimit](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
*Get the limitation of Hard alarm of DPLL.*
- void [IDTDpll\\_SetDpllHardAlarmEnable](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
*Set DPLL in unlocked state when a Hard alarm raised.*
- T\_osUInt8 [IDTDpll\\_GetDpllHardAlarmEnable](#) (T\_idtDrvHdlr hdlr)  
*Get DPLL in unlocked state when a Hard alarm raised.*
- T\_osUInt16 [IDTDpll\\_GetCurrentPhaseError](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
*Get the current phase error of DPLL.*
- void [IDTDpll\\_SetSonetGigEthOutputPath](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 instance, [T\\_idtDpllInstance](#) inputDpll, [T\\_idtSonetGigEthOutput](#) outputFreq)  
*Set Sonet/GigE path configuration.*
- void [IDTDpll\\_GetSonetGigEthOutputPath](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 instance, [T\\_idtDpllInstance](#) \*inputDpll\_p, [T\\_idtSonetGigEthOutput](#) \*outputFreq\_p)  
*Retrieve Sonet/GigE path configuration.*
- void [IDTDpll\\_SetDpllOutputPathSelectorA](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtDpllOutput-SelectorA](#) path)  
*Set DPLL output path selector A.*
- [T\\_idtDpllOutputSelectorA](#) [IDTDpll\\_GetDpllOutputPathSelectorA](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
*Get DPLL output path selector A.*
- void [IDTDpll\\_SetDpllOutputPathSelectorB](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtDpllOutput-SelectorB](#) path)  
*Set DPLL output path selector B.*
- [T\\_idtDpllOutputSelectorB](#) [IDTDpll\\_GetDpllOutputPathSelectorB](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
*Get DPLL output path selector B.*
- void [IDTDpll\\_SetDpllEthPathEnable](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, T\_osUInt8 nDisable)  
*Set ETH Enable/Disable.*
- T\_osUInt8 [IDTDpll\\_GetDpllEthPathEnable](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
*Get ETH Enable/Disable.*
- void [IDTDpll\\_SetDpllSelBPathEnable](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, T\_osUInt8 nDisable)  
*Set DPLL output path selector B Enable/Disable.*
- T\_osUInt8 [IDTDpll\\_GetDpllSelBPathEnable](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
*Get DPLL output path selector B Enable/Disable.*

- void [IDTDpll\\_SetDpll16E116T1Enable](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, T\_osUInt8 nDisable)  
*Set 16E1/16T1 Enable/Disable.*
- T\_osUInt8 [IDTDpll\\_GetDpll16E116T1Enable](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
*Get 16E1/16T1 Enable/Disable.*
- void [IDTDpll\\_SetDpllSelAPathEnable](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, T\_osUInt8 nDisable)  
*Set DPLL output path selector A Enable/Disable.*
- T\_osUInt8 [IDTDpll\\_GetDpllSelAPathEnable](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
*Get DPLL output path selector A Enable/Disable.*
- void [IDTDpll\\_SetFrequencyMonitoringFactor](#) (T\_idtDrvHdlr hdlr, [T\\_idtFreqMonFactor](#) freqMonFactor)  
*Function to set the frequency monitoring factor. Hard and soft alarm thresholds use this factor as their base unit.*
- [T\\_idtFreqMonFactor](#) [IDTDpll\\_GetFrequencyMonitoringFactor](#) (T\_idtDrvHdlr hdlr)  
*Function to get the frequency monitoring factor. Hard and soft alarm thresholds use this factor as their base unit.*
- void [IDTDpll\\_SetHardAlarmThreshold](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 accepting, T\_osUInt8 rejecting)  
*Set the frequency hard alarm accepting thresholds of reference clock monitoring.*
- void [IDTDpll\\_GetHardAlarmThreshold](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 \*accepting\_p, T\_osUInt8 \*rejecting\_p)  
*Get the frequency hard alarm accepting thresholds of reference clock monitoring.*
- void [IDTDpll\\_SetSoftAlarmThreshold](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 accepting, T\_osUInt8 rejecting)  
*Set the frequency soft alarm accepting thresholds of reference clock monitoring.*
- void [IDTDpll\\_GetSoftAlarmThreshold](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 \*accepting\_p, T\_osUInt8 \*rejecting\_p)  
*Get the frequency soft alarm accepting thresholds of reference clock monitoring.*
- void [IDTDpll\\_SetIcpCtrl](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, T\_osUInt8 nIcp)  
*Set charge pump control.*
- T\_osUInt8 [IDTDpll\\_GetIcpCtrl](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
*Get charge pump control.*
- void [IDTDpll\\_SetPhaseSlope](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtPhaseSlope](#) nPhaseSlope)  
*Set DPLL phase slope for DPLLs.*
- [T\\_idtPhaseSlope](#) [IDTDpll\\_GetPhaseSlope](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll)  
*Get DPLL phase slope for DPLLs.*
- void [IDTDpll\\_SetPpsFastLock](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnableFreq, T\_osUInt8 nEnablePhase)  
*Enable/Disable 1 PPS fast lock.*
- void [IDTDpll\\_GetPpsFastLock](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 \*nEnableFreq\_p, T\_osUInt8 \*nEnablePhase\_p)  
*Get enable/disable 1 PPS fast lock status.*
- void [IDTDpll\\_SetPhaseTransient](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
*Set the configuration when phase transient occurs.*
- T\_osUInt8 [IDTDpll\\_GetPhaseTransient](#) (T\_idtDrvHdlr hdlr)  
*Get the configuration when phase transient occurs.*
- void [IDTDpll\\_SetHitlessSwitchingFeature](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable, T\_osUInt8 nFrozen)  
*Set the configuration of hitless switching feature.*
- void [IDTDpll\\_GetHitlessSwitchingFeature](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 \*nEnable\_p, T\_osUInt8 \*nFrozen\_p)  
*Get the configuration of hitless switching feature.*
- void [IDTDpll\\_SetPhaseOffset](#) (T\_idtDrvHdlr hdlr, T\_osUInt16 nOffset)  
*Set the phase offset value.*
- T\_osUInt16 [IDTDpll\\_GetPhaseOffset](#) (T\_idtDrvHdlr hdlr)  
*Get the phase offset value.*
- void [IDTDpll\\_SetPpsPhase](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtDpllPpsPhasePostion](#) nPhase, [T\\_idtDpllPpsPulseWidth](#) nPulse)  
*Set PPS phase and pulse for DPLL.*
- void [IDTDpll\\_GetPpsPhase](#) (T\_idtDrvHdlr hdlr, [T\\_idtDpllInstance](#) nDpll, [T\\_idtDpllPpsPhasePostion](#) \*nPhase\_p, [T\\_idtDpllPpsPulseWidth](#) \*nPulse\_p)  
*Get PPS phase and pulse for DPLL.*

## Variables

- const [T\\_IDTPathConfiguration](#) [IDTDpll\\_pathConfigurationNULL\\_c](#)  
*Invalid output path configuration.*
- const [T\\_osUInt8](#) [IDTDpll\\_bandwidthLimitRegisters\\_a](#) [[Dpll\\_MAX](#)][[dpllBandwidthLimit\\_MAX](#)]  
*Structure used to determine which register to access for bandwidth limiting.*
- const [T\\_idtSonetGigEthSelector](#) [IDTDpll\\_sonetGigEthPathConfig2Bitfield](#) [[Dpll\\_MAX](#)][[SonetGigEthOutput\\_MAX](#)]  
*Table to translate from Sonet/GigE path configuration to bitfield value.*

## 6.42.1 Macro Definition Documentation

### 6.42.1.1 #define INVALID\_REGISTER\_OFFSET ((T\_osUInt8)(-1))

Tag used to indicating invalid register offset.

Definition at line 46 of file `DpllCnfg.c`.

## 6.42.2 Variable Documentation

### 6.42.2.1 const T\_osUInt8 IDTDpll\_bandwidthLimitRegisters\_a[Dpll\_MAX][dpllBandwidthLimit\_MAX]

**Initial value:**

```
=
{
    {
        REG_T0_DPLL_START_BW_DAMPING_CNFG,
        REG_T0_DPLL_ACQ_BW_DAMPING_CNFG,
        REG_T0_DPLL_LOCKED_BW_DAMPING_CNFG
    },
    {
        INVALID_REGISTER_OFFSET,
        INVALID_REGISTER_OFFSET,
        REG_T4_DPLL_LOCKED_BW_DAMPING_CNFG
    }
}
```

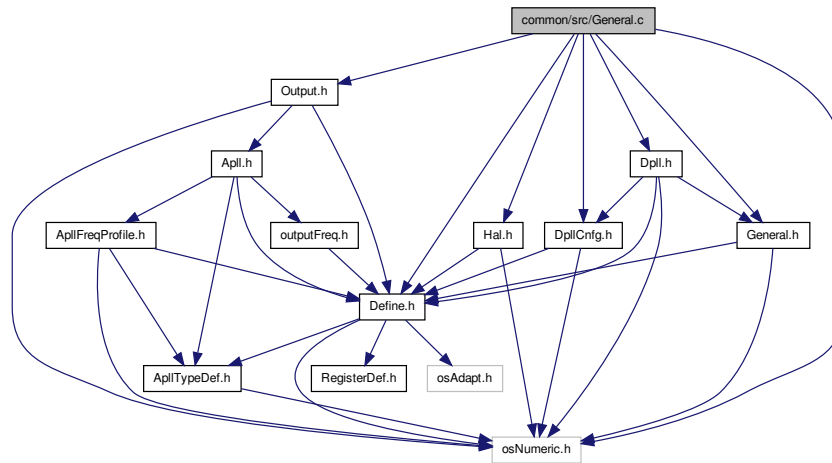
Structure used to determine which register to access for bandwidth limiting.

Definition at line 75 of file `DpllCnfg.c`.

## 6.43 common/src/General.c File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "Hal.h"
#include "DpllCnfg.h"
#include "Dpll.h"
#include "General.h"
#include "Output.h"
```

Include dependency graph for General.c:



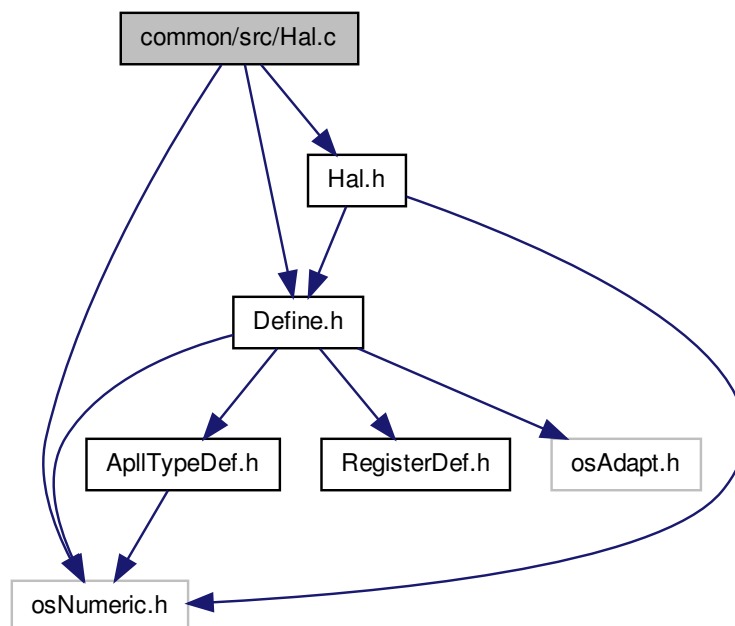
## Functions

- `T_osUInt16 IDTGeneral_GetDeviceID (T_idtDrvHdlr hdlr)`  
*Get ID number of the device.*
- `void IDTGeneral_SetOscFreqOffset (T_idtDrvHdlr hdlr, long pptOffset)`  
*Set compensation to the oscillator offset.*
- `long IDTGeneral_GetOscFreqOffset (T_idtDrvHdlr hdlr)`  
*Get compensated oscillator offset.*
- `void IDTGeneral_SetNetworkType (T_idtDrvHdlr hdlr, networkType_t nType)`  
*Set the network type, SDH or SONET.*
- `networkType_t IDTGeneral_GetNetworkType (T_idtDrvHdlr hdlr)`  
*Get the network type, SDH or SONET.*
- `void IDTGeneral_SetRevertiveMode (T_idtDrvHdlr hdlr, T_osUInt8 nEnable)`  
*Set the device running in Revertive mode or not.*
- `T_osUInt8 IDTGeneral_GetRevertiveMode (T_idtDrvHdlr hdlr)`  
*Get the device running in Revertive mode or not.*
- `void IDTGeneral_SetTimeoutValue (T_idtDrvHdlr hdlr, phaseTimeoutMultiplier_t nMultiFactor, T_osUInt8 n-Timeout)`  
*Set the value of time out of phase alarm Timeout(second)= nMultiFactor \* nTimeout.*
- `void IDTGeneral_GetTimeoutValue (T_idtDrvHdlr hdlr, phaseTimeoutMultiplier_t *nMultiFactor_p, T_osUInt8 *nTimeout_p)`  
*Get the value of time out of phase alarm Timeout(second)= nMultiFactor \* nTimeout.*
- `void IDTGeneral_SetOscActiveEdge (T_idtDrvHdlr hdlr, T_osUInt8 nEdge)`  
*Set the activate edge of main oscillator.*
- `T_osUInt8 IDTGeneral_GetOscActiveEdge (T_idtDrvHdlr hdlr)`  
*Get the activate edge of main oscillator.*
- `void IDTGeneral_SetProtectMode (T_idtDrvHdlr hdlr)`  
*Enable protected register access mode.*
- `void IDTGeneral_SetSingleUnprotectMode (T_idtDrvHdlr hdlr)`  
*Set single access register access mode.*
- `void IDTGeneral_SetFullyUnprotectMode (T_idtDrvHdlr hdlr)`  
*Set fully un-protected register access mode.*

- void [IDTGeneral\\_SetFlagOnTDO](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
*Set TDO pin to indicate the fail of selected clock.*
- T\_osUInt8 [IDTGeneral\\_GetFlagOnTDO](#) (T\_idtDrvHdlr hdlr)  
*Get TDO pin to indicate the fail of selected clock.*
- void [IDTGeneral\\_SetUltraFastSwitch](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
*Set Ultra-fast-switch enable or not.*
- T\_osUInt8 [IDTGeneral\\_GetUltraFastSwitch](#) (T\_idtDrvHdlr hdlr)  
*Get Ultra-fast-switch enable or not.*
- void [IDTGeneral\\_SetHardAlarm](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nEnable)  
*Set Hard-alarm enable when the input exceeds the threshold.*
- T\_osUInt8 [IDTGeneral\\_GetHardAlarm](#) (T\_idtDrvHdlr hdlr)  
*Get Hard-alarm enable when the input exceeds the threshold.*
- T\_osUInt8 [IDTGeneral\\_i2cTranslate](#) (T\_osUInt8 i2cAddr, void \*transData)  
*I2C address translation function used internally by the driver.*
- void [IDTGeneral\\_InitDeviceCommon](#) (T\_idtDrvHdlr hdlr)
- T\_idtDrvHdlr [IDTGeneral\\_InitDriverCommon](#) (const char \*name, [T\\_idtDrvAccess](#) drvAccess, [T\\_idtDeviceType](#) deviceType, const [T\\_idtDrvDeviceConfig](#) \*deviceConfig, int useHighLevelApi)
- void [IDTGeneral\\_DeInitDriver](#) (T\_idtDrvHdlr hdlr)  
*De-initialize driver handler.*

## 6.44 common/src/Hal.c File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "Hal.h"
Include dependency graph for Hal.c:
```



## Macros

- `#define DEBUG_IDT_HAL`

## Functions

- void `IDTHAL_WriteBitfield` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 mask, T\_osUInt8 lsb, T\_osUInt8 data)

*Define function to write a bitfield to a device.*

- void `IDTHAL_WriteBitfield_Ext` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 mask, T\_osUInt8 lsb, T\_osUInt8 data)

*Define function to write a bitfield to a device.*

- T\_osUInt8 `IDTHAL_ReadBitfield` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 mask, T\_osUInt8 lsb)

*Define function to read a bitfield from device.*

- T\_osUInt8 `IDTHAL_ReadBitfield_Ext` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 mask, T\_osUInt8 lsb)

*Define function to read a bitfield from device.*

- T\_osUInt8 `IDTHal_Read` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset)

*Function to read from device address space.*

- void `IDTHal_Write` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 data)

*Function to write to device address space.*

- T\_osUInt8 `IDTHal_Read_Ext` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset)

*Function to read from device ap11 address space.*

- void `IDTHal_Write_Ext` (T\_idtDrvHdlr hdlr, T\_osUInt8 offset, T\_osUInt8 data)

*Function to write to device ap11 address space.*

### 6.44.1 Macro Definition Documentation

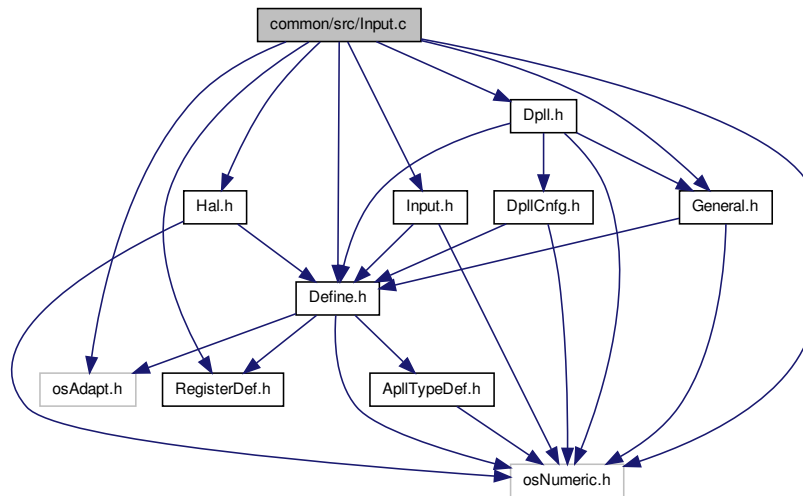
#### 6.44.1.1 `#define DEBUG_IDT_HAL`

Definition at line 41 of file Hal.c.

## 6.45 common/src/Input.c File Reference

```
#include "osNumeric.h"
#include "osAdapt.h"
#include "Define.h"
#include "Hal.h"
#include "General.h"
#include "Input.h"
#include "Dpll.h"
#include "RegisterDef.h"
```

Include dependency graph for Input.c:



## Data Structures

- struct [bucketReg\\_t](#)  
*Structure used to store leaky bucket register offsets.*

## Functions

- [T\\_idtInputFrequencyFamily IDTInput\\_InFreqBitValue2Family](#) (T\_idtDrvHdlr hdlr, [T\\_idtInputFrequencyBitfieldValue](#) inFreqBitValue)  
*Translate input frequency bit value to input frequency family.*
- void [IDTInput\\_SetPriority](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, [T\\_idtDpllInstance](#) nDpll, T\_osUInt8 nPriority)  
*Set specified port to the priority.*
- T\_osUInt8 [IDTInput\\_GetPriority](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, [T\\_idtDpllInstance](#) nDpll)  
*Get the current priority of the specified port.*
- void [IDTInput\\_SetInputFrequency](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, [T\\_idtInputFrequencyBitfieldValue](#) nFrequency)  
*Set the frequency of the input clock at the specified port.*
- [T\\_idtInputFrequencyBitfieldValue IDTInput\\_GetInputFrequency](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Get frequency of the input clock at the specified port.*
- void [IDTInput\\_SetActivityMonitor](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, T\_osUInt8 nBucket)  
*Select one of the four activity monitor configurations for the specified input port.*
- T\_osUInt8 [IDTInput\\_GetActivityMonitor](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Retrieve selected activity monitoring configuration.*
- void [IDTInput\\_SetBucketConfig](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nBucket, T\_osUInt8 nUpper, T\_osUInt8 nLower, T\_osUInt8 nSize, T\_osUInt8 nDecay)  
*Set the configuration to Leaky Bucket activity monitoring.*
- void [IDTInput\\_GetBucketConfig](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nBucket, T\_osUInt8 \*nUpper\_p, T\_osUInt8 \*nLower\_p, T\_osUInt8 \*nSize\_p, T\_osUInt8 \*nDecay\_p)  
*Get activity monitoring configuration (Leaky Bucket)*

- void [IDTInput\\_SetPreDivider](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, [T\\_idtInputDividerMux](#) nDivider)  
*Assign a pre-divider for the specified input.*
- [T\\_idtInputDividerMux IDTInput\\_GetPreDivider](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Get configured input port pre-divider.*
- void [IDTInput\\_SetDivisionFactor](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, T\_osUInt16 nFactor)  
*Set the dividen factor to pre-divider assigned to a certain input.*
- T\_osUInt16 [IDTInput\\_GetDivisionFactor](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Get the dividen factor to pre-divider assigned to a certain input.*
- void [IDTInput\\_SetInputHFdivider](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, [T\\_idtHighFrequencyDivisor](#) n-Usage)  
*Set the usage of high frequency (> 155.52MHz) divider.*
- [T\\_idtHighFrequencyDivisor IDTInput\\_GetInputHFdivider](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Get high frequency configuration divider.*
- T\_osUInt8 [IDTInput\\_GetInputFreqOffset](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Get the frequency offset of the specified input clock. The returned value is updated every 16 seconds.*
- T\_osUInt8 [IDTInput\\_GetInputClockStatus](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Get the status of a specified input clock.*
- T\_osUInt8 [IDTInput\\_GetInputClockValidation](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Get the clock validation status of an input port.*
- void [IDTInput\\_EnableAllInputs](#) (T\_idtDrvHdlr hdlr)  
*Enable to select anyone of input clocks.*
- void [IDTInput\\_DisableAllInputs](#) (T\_idtDrvHdlr hdlr)  
*Disable to select anyone of input clocks.*
- void [IDTInput\\_SetInputEnable](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, T\_osUInt8 enable)  
*Enable/Disable selected input.*
- T\_osUInt8 [IDTInput\\_GetInputEnable](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)  
*Retrieve enable/disable status of selected input.*
- void [IDTInput\\_SetSyncFrequency](#) (T\_idtDrvHdlr hdlr, [T\\_idtInputSyncFreq](#) syncFreq)  
*Function to select input external sync frequency.*
- [T\\_idtInputSyncFreq IDTInput\\_GetSyncFrequency](#) (T\_idtDrvHdlr hdlr)  
*Function to retrieve select input external sync frequency.*
- void [IDTInput\\_SetSyncSampling](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 exSyncN, [T\\_externalSyncSampling](#) sampling)  
*Function to provision sampling configuration for external sync.*
- [T\\_externalSyncSampling IDTInput\\_GetSyncSampling](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 exSyncN)  
*Function to retrieve sampling configuration for external sync.*
- void [IDTInput\\_SetSyncEdge](#) (T\_idtDrvHdlr hdlr, [T\\_externalSyncEdge](#) edge)  
*Function to provision synchronization alignment.*
- [T\\_externalSyncEdge IDTInput\\_GetSyncEdge](#) (T\_idtDrvHdlr hdlr)  
*Function to retrieve synchronization alignment.*
- void [IDTInput\\_SetSyncAlarmRange](#) (T\_idtDrvHdlr hdlr, [T\\_externalSyncAlarmRange](#) range)  
*Function to provision the sync allowable range before alarming.*
- [T\\_externalSyncAlarmRange IDTInput\\_GetSyncAlarmRange](#) (T\_idtDrvHdlr hdlr)  
*Function to retrieve the sync allowable range before alarming.*
- void [IDTInput\\_SetSyncBypass](#) (T\_idtDrvHdlr hdlr, [T\\_externalSyncBypass](#) bypass)  
*Function to provision sync bypass mode.*
- [T\\_externalSyncBypass IDTInput\\_GetSyncBypass](#) (T\_idtDrvHdlr hdlr)  
*Function to retrieve sync bypass mode.*
- void [IDTInput\\_SetSyncEnable](#) (T\_idtDrvHdlr hdlr, [T\\_externalSyncEnable](#) enable)  
*Function to provision enable mode for input external sync.*
- [T\\_externalSyncEnable IDTInput\\_GetSyncEnable](#) (T\_idtDrvHdlr hdlr)

*Function to retrieve external sync enable mode.*

- void [IDTInput\\_SetAmiInputFrequency](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo, [T\\_idtAmiInputFrequencyBitFieldValue](#) amiSignal)

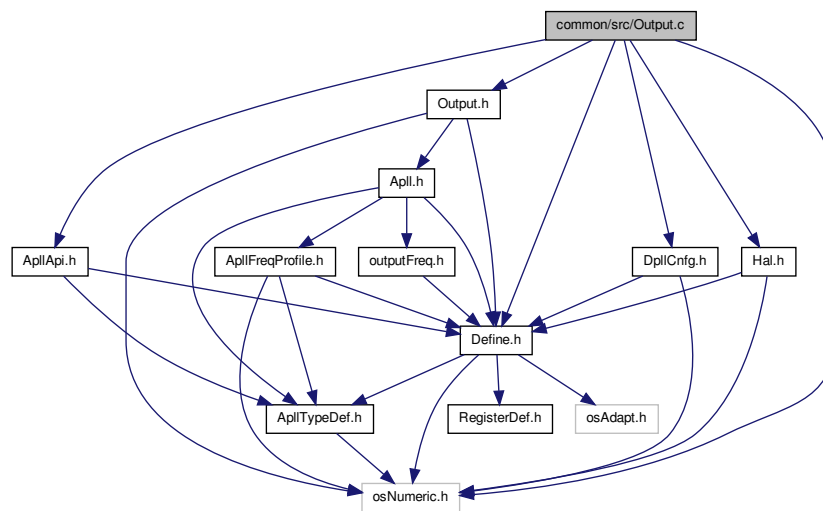
*Provision AMI signal for input port.*

- [T\\_idtAmiInputFrequencyBitFieldValue](#) [IDTInput\\_GetAmiInputFrequency](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nInputNo)

*Function to retrieve AMI input frequency for input port.*

## 6.46 common/src/Output.c File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "Hal.h"
#include "Output.h"
#include "DpllCnfg.h"
#include "ApplApi.h"
Include dependency graph for Output.c:
```



## Enumerations

- enum [T\\_idtOutputPathBfVal](#) {  
[OutputPathBfVal\\_T0SonetGigEth](#) = 0, [OutputPathBfVal\\_T0Dpll\\_Eth](#) = 3, [OutputPathBfVal\\_T0Dpll\\_7776kHz](#) = 4, [OutputPathBfVal\\_T0Dpll\\_12E1\\_GPS\\_E3\\_T3](#) = 5,  
[OutputPathBfVal\\_T0Dpll\\_16E1\\_16T1](#) = 6, [OutputPathBfVal\\_T0Dpll\\_GSM\\_OBSAI\\_16E1\\_16T1](#) = 7, [OutputPathBfVal\\_T4SonetGigEth](#) = 8, [OutputPathBfVal\\_T4Dpll\\_Eth](#) = 11,  
[OutputPathBfVal\\_T4Dpll\\_7776kHz](#) = 12, [OutputPathBfVal\\_T4Dpll\\_12E1\\_24T1\\_E3\\_T3](#) = 13, [OutputPathBfVal\\_T4Dpll\\_16E1\\_16T1](#) = 14, [OutputPathBfVal\\_T4Dpll\\_GSM\\_GPS\\_16E1\\_16T1](#) = 15,  
[OutputPathBfVal\\_NULL](#), [OutputPathBfVal\\_MAX](#) = [OutputPathBfVal\\_NULL](#) }

*Enumerated type listing possible output path selections.*

- enum { [OUTPUTIF\\_AMI\\_MASK](#) = 1<<OUTPUTIF\_AMI, [OUTPUTIF\\_CMOS\\_MASK](#) = 1<<OUTPUTIF\_CMOS, [OUTPUTIF\\_LVDS\\_MASK](#) = 1<<OUTPUTIF\_LVDS, [OUTPUTIF\\_PECL\\_MASK](#) = 1<<OUTPUTIF\_PECL }

*Enumerated type supported output port interfaces in bit mask format.*

## Functions

- void [IDTOutput\\_SetOutputConfig](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN, [T\\_idtDpllInstance](#) nDpll, [T\\_outputPathType](#) nPath, T\_osUInt8 nFactor, [T\\_idtApllConfigPerOutput](#) \*apllConfig)  
*Set the configuration of Output port.*
- void [IDTOutput\\_GetOutputConfig](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN, [T\\_idtDpllInstance](#) \*nDpll\_p, [T\\_outputPathType](#) \*nPath\_p, T\_osUInt8 \*nFactor\_p, [T\\_idtApllConfigPerOutput](#) \*apllConfig)  
*Get the configuration of Output port.*
- void [IDTOutput\\_SetOutputInvert](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN, T\_osUInt8 invert)  
*Set the specified output port to generate a invert signal.*
- T\_osUInt8 [IDTOutput\\_GetOutputInvert](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN)  
*Get output signal inversion status.*
- void [IDTOutput\\_SetOutputEnable](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN, T\_osUInt8 nEnable)  
*Enable/Disable output port. Consult data sheet for supported ports.*
- T\_osUInt8 [IDTOutput\\_GetOutputEnable](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN)  
*Get enable/disable status of output port. Consult data sheet for supported ports.*
- void [IDTOutput\\_SetOutputInterface](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN, [T\\_outputInterface](#) outIf)  
*Set output port interface type. Consult data sheet for supported interfaces per port.*
- [T\\_outputInterface](#) [IDTOutput\\_GetOutputInterface](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 nOutN)  
*Get output port interface type. Consult data sheet for supported interfaces per port.*
- void [IDTOutput\\_SetFrameSync](#) (T\_idtDrvHdlr hdlr, [T\\_frameSync](#) frameSync, T\_osUInt8 enable, T\_osUInt8 invert, T\_osUInt8 pulse)  
*Function to control output frame syncs.*
- void [IDTOutput\\_GetFrameSync](#) (T\_idtDrvHdlr hdlr, [T\\_frameSync](#) frameSync, T\_osUInt8 \*enable\_p, T\_osUInt8 \*invert\_p, T\_osUInt8 \*pulse\_p)  
*Function to retrieve output frame sync configuration.*
- void [IDTOutput\\_SetFrameSyncPulseEdge](#) (T\_idtDrvHdlr hdlr, T\_osUInt8 isRisingEdge)  
*Function provision trigger for frame sync pulses.*
- T\_osUInt8 [IDTOutput\\_GetFrameSyncPulseEdge](#) (T\_idtDrvHdlr hdlr)  
*Function retrieve trigger for frame sync pulses.*
- void [IDTOutput\\_DisableApllInput](#) (T\_idtDrvHdlr hdlr, [T\\_idtOutputApllConfig](#) outputApllConfig)  
*Function disables APLL input.*
- void [IDTOutput\\_DisableAllIntOutput](#) (T\_idtDrvHdlr hdlr)  
*Function disables all internal outputs.*

## 6.46.1 Enumeration Type Documentation

### 6.46.1.1 anonymous enum

Enumerated type supported output port interfaces in bit mask format.

#### Enumerator

- OUTPUTIF\_AMI\_MASK** AMI Interface mask
- OUTPUTIF\_CMOS\_MASK** CMOS Interface mask
- OUTPUTIF\_LVDS\_MASK** LVDS Interface mask
- OUTPUTIF\_PECL\_MASK** PECL Interface mask

Definition at line 78 of file Output.c.

## 6.46.1.2 enum T\_idtOutputPathBfVal

Enumerated type listing possible output path selections.

Enumerator

***OutputPathBfVal\_T0SonetGigEth*** From T0 Sonet/GigE path  
***OutputPathBfVal\_T0Dpll\_Eth*** From T0 DPLL ETH path  
***OutputPathBfVal\_T0Dpll\_7776kHz*** From T0 DPLL 77.76 MHz path  
***OutputPathBfVal\_T0Dpll\_12E1\_GPS\_E3\_T3*** From T0 DPLL 12E1/GPS/E3/T3 path  
***OutputPathBfVal\_T0Dpll\_16E1\_16T1*** From T0 DPLL 16E1/16T1 path  
***OutputPathBfVal\_T0Dpll\_GSM\_OBSAI\_16E1\_16T1*** From T0 DPLL GSM/OBSAI/16E1/16T1 path  
***OutputPathBfVal\_T4SonetGigEth*** From T4 Sonet/GigE path  
***OutputPathBfVal\_T4Dpll\_Eth*** From T4 DPLL ETH path  
***OutputPathBfVal\_T4Dpll\_7776kHz*** From T4 DPLL 77.76 MHz path  
***OutputPathBfVal\_T4Dpll\_12E1\_24T1\_E3\_T3*** From T4 DPLL 12E1/E3/T3/24T1 path  
***OutputPathBfVal\_T4Dpll\_16E1\_16T1*** From T4 DPLL 16E1/16T1 path  
***OutputPathBfVal\_T4Dpll\_GSM\_GPS\_16E1\_16T1*** From T4 DPLL GSM/GPS/16E1/16T1 path  
***OutputPathBfVal\_NULL*** Not supported  
***OutputPathBfVal\_MAX***

Definition at line 48 of file Output.c.

# Index

- 82V3910.c
  - IDTGeneral\_InitDevice, [177](#)
  - IDTGeneral\_InitDriver, [177](#)
  - idt82V3910Config, [178](#)
- 82V3910/dev/include/idt82V3910.h, [175](#)
- 82V3910/dev/src/82V3910.c, [176](#)
- 82V3910/sample/include/access.h, [178](#)
- 82V3910/sample/include/loadstore.h, [181](#)
- 82V3910/sample/include/sample.h, [182](#)
- 82V3910/sample/include/sampleAppl.h, [185](#)
- 82V3910/sample/src/access.c, [189](#)
- 82V3910/sample/src/loadstore.c, [192](#)
- 82V3910/sample/src/main.c, [194](#)
- 82V3910/sample/src/sample.c, [198](#)
- 82V3910/sample/src/sampleAppl.c, [201](#)
- APLL\_ACCESS\_I2C
  - Appl Function Module, [124](#)
- APLL\_ACCESS\_MAX
  - Appl Function Module, [124](#)
- APLL\_ACCESS\_SIM
  - Appl Function Module, [124](#)
- APLL\_BYPASS\_BYPASS
  - Appl Function Module, [124](#)
- APLL\_BYPASS\_MAX
  - Appl Function Module, [124](#)
- APLL\_BYPASS\_NORMAL\_OP
  - Appl Function Module, [124](#)
- APLL\_CLK\_SEL\_EXTERNAL
  - Appl Function Module, [125](#)
- APLL\_CLK\_SEL\_INTERNAL
  - Appl Function Module, [125](#)
- APLL\_CLK\_SEL\_MAX
  - Appl Function Module, [125](#)
- APLL\_CONFIG0
  - Appl Function Module, [124](#)
- APLL\_CONFIG1
  - Appl Function Module, [124](#)
- APLL\_CONFIG\_MAX
  - Appl Function Module, [124](#)
- APLL\_FREQ\_SEL\_DOUBLE
  - Appl Function Module, [125](#)
- APLL\_FREQ\_SEL\_MAX
  - Appl Function Module, [125](#)
- APLL\_FREQ\_SEL\_SINGLE
  - Appl Function Module, [125](#)
- APLL\_IN\_MR\_SYNC
  - Appl Function Module, [126](#)
- APLL\_INPUT\_FREQ\_125000KHz
  - Appl Function Module, [125](#)
- APLL\_INPUT\_FREQ\_128906\_DOT\_25KHz
  - Appl Function Module, [125](#)
- APLL\_INPUT\_FREQ\_155520KHz
  - Appl Function Module, [125](#)
- APLL\_INPUT\_FREQ\_156250KHz
  - Appl Function Module, [125](#)
- APLL\_INPUT\_FREQ\_19440KHz
  - Appl Function Module, [125](#)
- APLL\_INPUT\_FREQ\_25000KHz
  - Appl Function Module, [125](#)
- APLL\_INPUT\_FREQ\_25781\_DOT\_25KHz
  - Appl Function Module, [125](#)
- APLL\_INPUT\_FREQ\_312500KHz
  - Appl Function Module, [125](#)
- APLL\_INPUT\_FREQ\_77760KHz
  - Appl Function Module, [125](#)
- APLL\_INPUT\_FREQ\_8KHz
  - Appl Function Module, [125](#)
- APLL\_INPUT\_FREQ\_MAX
  - Appl Function Module, [125](#)
- APLL\_INST\_1
  - Appl Function Module, [125](#)
- APLL\_INST\_2
  - Appl Function Module, [125](#)
- APLL\_INST\_MAX
  - Appl Function Module, [125](#)
- APLL\_MR\_SYNC\_MAX
  - Appl Function Module, [126](#)
- APLL\_OUT\_MAX
  - Appl Function Module, [127](#)
- APLL\_OUT\_MR\_SYNC
  - Appl Function Module, [126](#)
- APLL\_OUT\_QA\_ONLY
  - Appl Function Module, [127](#)
- APLL\_OUT\_QA\_OR\_QB
  - Appl Function Module, [127](#)
- APLL\_OUT\_QB\_ONLY
  - Appl Function Module, [127](#)
- APLL\_OUT\_SIG\_LVDS
  - Appl Function Module, [127](#)
- APLL\_OUT\_SIG\_LVPECL
  - Appl Function Module, [127](#)
- APLL\_OUT\_SIG\_MAX
  - Appl Function Module, [127](#)
- APLL\_OUTPUT\_DISABLE
  - Appl Function Module, [126](#)
- APLL\_OUTPUT\_EN\_MAX
  - Appl Function Module, [126](#)
- APLL\_OUTPUT\_ENABLE

- ApI Function Module, [126](#)
- APLL\_OUTPUT\_FREQ\_124416KHz
  - ApI Function Module, [126](#)
- APLL\_OUTPUT\_FREQ\_125000KHz
  - ApI Function Module, [126](#)
- APLL\_OUTPUT\_FREQ\_128906\_DOT\_25KHz
  - ApI Function Module, [126](#)
- APLL\_OUTPUT\_FREQ\_155520KHz
  - ApI Function Module, [126](#)
- APLL\_OUTPUT\_FREQ\_156250KHz
  - ApI Function Module, [126](#)
- APLL\_OUTPUT\_FREQ\_161132\_DOT\_8125KHz
  - ApI Function Module, [127](#)
- APLL\_OUTPUT\_FREQ\_24883\_DOT\_2KHz
  - ApI Function Module, [126](#)
- APLL\_OUTPUT\_FREQ\_25000KHz
  - ApI Function Module, [126](#)
- APLL\_OUTPUT\_FREQ\_25781\_DOT\_25KHz
  - ApI Function Module, [126](#)
- APLL\_OUTPUT\_FREQ\_311040KHz
  - ApI Function Module, [127](#)
- APLL\_OUTPUT\_FREQ\_312500KHz
  - ApI Function Module, [127](#)
- APLL\_OUTPUT\_FREQ\_322265\_DOT\_625KHz
  - ApI Function Module, [127](#)
- APLL\_OUTPUT\_FREQ\_622080KHz
  - ApI Function Module, [127](#)
- APLL\_OUTPUT\_FREQ\_625000KHz
  - ApI Function Module, [127](#)
- APLL\_OUTPUT\_FREQ\_644531\_DOT\_25KHz
  - ApI Function Module, [127](#)
- APLL\_OUTPUT\_FREQ\_77760KHz
  - ApI Function Module, [126](#)
- APLL\_OUTPUT\_FREQ\_78125KHz
  - ApI Function Module, [126](#)
- APLL\_OUTPUT\_FREQ\_80566\_DOT\_40625KHz
  - ApI Function Module, [126](#)
- APLL\_OUTPUT\_FREQ\_MAX
  - ApI Function Module, [127](#)
- APLL\_OUTPUT\_MAX
  - ApI Function Module, [126](#)
- APLL\_OUTPUT\_QA
  - ApI Function Module, [126](#)
- APLL\_OUTPUT\_QB
  - ApI Function Module, [126](#)
- APLL\_QA\_ODSEL\_DIV\_1
  - ApI Function Module, [127](#)
- APLL\_QA\_ODSEL\_DIV\_2
  - ApI Function Module, [127](#)
- APLL\_QA\_ODSEL\_DIV\_25
  - ApI Function Module, [127](#)
- APLL\_QA\_ODSEL\_DIV\_4
  - ApI Function Module, [127](#)
- APLL\_QA\_ODSEL\_DIV\_5
  - ApI Function Module, [127](#)
- APLL\_QA\_ODSEL\_DIV\_MAX
  - ApI Function Module, [127](#)
- APLL\_QB\_ODSEL\_DIV\_1
  - ApI Function Module, [128](#)
- APLL\_QB\_ODSEL\_DIV\_2
  - ApI Function Module, [128](#)
- APLL\_QB\_ODSEL\_DIV\_4
  - ApI Function Module, [128](#)
- APLL\_QB\_ODSEL\_DIV\_5
  - ApI Function Module, [128](#)
- APLL\_QB\_ODSEL\_DIV\_8
  - ApI Function Module, [128](#)
- APLL\_QB\_ODSEL\_DIV\_MAX
  - ApI Function Module, [128](#)
- APLL\_SHUTOFF\_MAX
  - ApI Function Module, [128](#)
- APLL\_SHUTOFF\_NORMAL\_OP
  - ApI Function Module, [128](#)
- APLL\_SHUTOFF\_SHUTOFF
  - ApI Function Module, [128](#)
- APLL\_XTAL\_1\_APLL1
  - ApI Function Module, [128](#)
- APLL\_XTAL\_2\_APLL2
  - ApI Function Module, [128](#)
- APLL\_XTAL\_3\_APLL1
  - ApI Function Module, [128](#)
- APLL\_XTAL\_4\_APLL2
  - ApI Function Module, [128](#)
- APLL\_XTAL\_FREQ\_24883\_DOT\_2KHz
  - ApI Function Module, [128](#)
- APLL\_XTAL\_FREQ\_25000KHz
  - ApI Function Module, [128](#)
- APLL\_XTAL\_FREQ\_25781\_DOT\_25KHz
  - ApI Function Module, [128](#)
- APLL\_XTAL\_FREQ\_MAX
  - ApI Function Module, [128](#)
- APLL\_XTAL\_MAX
  - ApI Function Module, [128](#)
- APLL1\_REG\_HIGH\_BIT
  - ApIApi.c, [217](#)
- APLL2\_REG\_HIGH\_BIT
  - ApIApi.c, [217](#)
- APLL\_CONFIGURED
  - Define.h, [257](#)
- APLL\_DIVIDER\_LSB
  - ApIApi.c, [218](#)
- APLL\_DIVIDER\_MSB
  - ApIApi.c, [218](#)
- APLL\_DIVIDER\_VAL
  - ApIApi.c, [218](#)
- APLL\_FB\_DIV\_ETH
  - ApIFreqProfile.c, [221](#)
- APLL\_FB\_DIV\_LSB
  - ApIApi.c, [218](#)
- APLL\_FB\_DIV\_MAX
  - ApIApi.c, [218](#)
- APLL\_FB\_DIV\_MIN
  - ApIApi.c, [218](#)
- APLL\_FB\_DIV\_MSB
  - ApIApi.c, [218](#)
- APLL\_FB\_DIV\_SONET

- ApIIFreqProfile.c, 221
- APLL\_FB\_DIV\_VAL
  - ApIIFreqProfile.c, 218
- APLL\_OUTPUT\_DIV\_1
  - ApIIFreqProfile.c, 221
- APLL\_OUTPUT\_DIV\_2
  - ApIIFreqProfile.c, 221
- APLL\_OUTPUT\_DIV\_25
  - ApIIFreqProfile.c, 221
- APLL\_OUTPUT\_DIV\_4
  - ApIIFreqProfile.c, 221
- APLL\_OUTPUT\_DIV\_5
  - ApIIFreqProfile.c, 221
- APLL\_OUTPUT\_DIV\_8
  - ApIIFreqProfile.c, 221
- APLL\_PRE\_DIV\_ETH
  - ApIIFreqProfile.c, 221
- APLL\_PRE\_DIV\_LSB
  - ApIIFreqProfile.c, 218
- APLL\_PRE\_DIV\_MAX
  - ApIIFreqProfile.c, 219
- APLL\_PRE\_DIV\_MIN
  - ApIIFreqProfile.c, 219
- APLL\_PRE\_DIV\_MSB
  - ApIIFreqProfile.c, 219
- APLL\_PRE\_DIV\_VAL
  - ApIIFreqProfile.c, 219
- APLL\_REG\_OFFSET
  - ApIIFreqProfile.c, 219
- access.c
  - DelInit\_SimWrapper, 190
  - I2cAccessMethods, 192
  - IDTSample\_GetLogVerbosity, 190
  - IDTSample\_InitDriverI2c, 190
  - IDTSample\_InitDriverSimulator, 190
  - IDTSample\_ResetXtalList, 191
  - IDTSample\_SetLogVerbosity, 191
  - Init\_SimWrapper, 191
  - ReadByte\_SimWrapper, 191
  - SimAccessMethods, 192
  - WriteByte\_SimWrapper, 192
- access.h
  - IDTSample\_GetLogVerbosity, 179
  - IDTSample\_InitDriverI2c, 179
  - IDTSample\_InitDriverSimulator, 180
  - IDTSample\_ResetXtalList, 180
  - IDTSample\_SetLogVerbosity, 180
- Activate\_Edge\_Falling
  - General Function Module, 48
- Activate\_Edge\_Rising
  - General Function Module, 48
- AmiFreq\_64kHzPlus400Hz
  - Input Function Module, 65
- AmiFreq\_64kHzPlus8kHz
  - Input Function Module, 65
- AmiFreq\_MAX
  - Input Function Module, 65
- ApIIFreq\_Profile.c, 118
- APLL\_ACCESS\_I2C, 124
- APLL\_ACCESS\_MAX, 124
- APLL\_ACCESS\_SIM, 124
- APLL\_BYPASS\_BYPASS, 124
- APLL\_BYPASS\_MAX, 124
- APLL\_BYPASS\_NORMAL\_OP, 124
- APLL\_CLK\_SEL\_EXTERNAL, 125
- APLL\_CLK\_SEL\_INTERNAL, 125
- APLL\_CLK\_SEL\_MAX, 125
- APLL\_CONFIG0, 124
- APLL\_CONFIG1, 124
- APLL\_CONFIG\_MAX, 124
- APLL\_FREQ\_SEL\_DOUBLE, 125
- APLL\_FREQ\_SEL\_MAX, 125
- APLL\_FREQ\_SEL\_SINGLE, 125
- APLL\_IN\_MR\_SYNC, 126
- APLL\_INPUT\_FREQ\_125000KHz, 125
- APLL\_INPUT\_FREQ\_128906\_DOT\_25KHz, 125
- APLL\_INPUT\_FREQ\_155520KHz, 125
- APLL\_INPUT\_FREQ\_156250KHz, 125
- APLL\_INPUT\_FREQ\_19440KHz, 125
- APLL\_INPUT\_FREQ\_25000KHz, 125
- APLL\_INPUT\_FREQ\_25781\_DOT\_25KHz, 125
- APLL\_INPUT\_FREQ\_312500KHz, 125
- APLL\_INPUT\_FREQ\_77760KHz, 125
- APLL\_INPUT\_FREQ\_8KHz, 125
- APLL\_INPUT\_FREQ\_MAX, 125
- APLL\_INST\_1, 125
- APLL\_INST\_2, 125
- APLL\_INST\_MAX, 125
- APLL\_MR\_SYNC\_MAX, 126
- APLL\_OUT\_MAX, 127
- APLL\_OUT\_MR\_SYNC, 126
- APLL\_OUT\_QA\_ONLY, 127
- APLL\_OUT\_QA\_OR\_QB, 127
- APLL\_OUT\_QB\_ONLY, 127
- APLL\_OUT\_SIG\_LVDS, 127
- APLL\_OUT\_SIG\_LVPECL, 127
- APLL\_OUT\_SIG\_MAX, 127
- APLL\_OUTPUT\_DISABLE, 126
- APLL\_OUTPUT\_EN\_MAX, 126
- APLL\_OUTPUT\_ENABLE, 126
- APLL\_OUTPUT\_FREQ\_124416KHz, 126
- APLL\_OUTPUT\_FREQ\_125000KHz, 126
- APLL\_OUTPUT\_FREQ\_128906\_DOT\_25KHz, 126
- APLL\_OUTPUT\_FREQ\_155520KHz, 126
- APLL\_OUTPUT\_FREQ\_156250KHz, 126
- APLL\_OUTPUT\_FREQ\_161132\_DOT\_8125KHz, 127
- APLL\_OUTPUT\_FREQ\_24883\_DOT\_2KHz, 126
- APLL\_OUTPUT\_FREQ\_25000KHz, 126
- APLL\_OUTPUT\_FREQ\_25781\_DOT\_25KHz, 126
- APLL\_OUTPUT\_FREQ\_311040KHz, 127
- APLL\_OUTPUT\_FREQ\_312500KHz, 127
- APLL\_OUTPUT\_FREQ\_322265\_DOT\_625KHz, 127
- APLL\_OUTPUT\_FREQ\_622080KHz, 127

- APLL\_OUTPUT\_FREQ\_625000KHz, [127](#)
- APLL\_OUTPUT\_FREQ\_644531\_DOT\_25KHz, [127](#)
- APLL\_OUTPUT\_FREQ\_77760KHz, [126](#)
- APLL\_OUTPUT\_FREQ\_78125KHz, [126](#)
- APLL\_OUTPUT\_FREQ\_80566\_DOT\_40625KHz, [126](#)
- APLL\_OUTPUT\_FREQ\_MAX, [127](#)
- APLL\_OUTPUT\_MAX, [126](#)
- APLL\_OUTPUT\_QA, [126](#)
- APLL\_OUTPUT\_QB, [126](#)
- APLL\_QA\_ODSEL\_DIV\_1, [127](#)
- APLL\_QA\_ODSEL\_DIV\_2, [127](#)
- APLL\_QA\_ODSEL\_DIV\_25, [127](#)
- APLL\_QA\_ODSEL\_DIV\_4, [127](#)
- APLL\_QA\_ODSEL\_DIV\_5, [127](#)
- APLL\_QA\_ODSEL\_DIV\_MAX, [127](#)
- APLL\_QB\_ODSEL\_DIV\_1, [128](#)
- APLL\_QB\_ODSEL\_DIV\_2, [128](#)
- APLL\_QB\_ODSEL\_DIV\_4, [128](#)
- APLL\_QB\_ODSEL\_DIV\_5, [128](#)
- APLL\_QB\_ODSEL\_DIV\_8, [128](#)
- APLL\_QB\_ODSEL\_DIV\_MAX, [128](#)
- APLL\_SHUTOFF\_MAX, [128](#)
- APLL\_SHUTOFF\_NORMAL\_OP, [128](#)
- APLL\_SHUTOFF\_SHUTOFF, [128](#)
- APLL\_XTAL\_1\_APLL1, [128](#)
- APLL\_XTAL\_2\_APLL2, [128](#)
- APLL\_XTAL\_3\_APLL1, [128](#)
- APLL\_XTAL\_4\_APLL2, [128](#)
- APLL\_XTAL\_FREQ\_24883\_DOT\_2KHz, [128](#)
- APLL\_XTAL\_FREQ\_25000KHz, [128](#)
- APLL\_XTAL\_FREQ\_25781\_DOT\_25KHz, [128](#)
- APLL\_XTAL\_FREQ\_MAX, [128](#)
- APLL\_XTAL\_MAX, [128](#)
- IDT\_APLL\_API\_TRACE, [123](#)
- IDTApll\_ApllInput2DpllOutput, [129](#)
- IDTApll\_DpllOutput2ApllOutput, [129](#)
- IDTApll\_GetApllConfig, [129](#)
- IDTApll\_GetApllConfigPerOutput, [129](#)
- IDTApll\_GetApllFreqTableEntry, [129](#)
- IDTApll\_GetBypass, [130](#)
- IDTApll\_GetConfigListByFreq, [130](#)
- IDTApll\_GetConfigListByXtalType, [130](#)
- IDTApll\_GetConfigSel, [131](#)
- IDTApll\_GetCurrentApllFreqConfig, [131](#)
- IDTApll\_GetFeedbackDiv, [131](#)
- IDTApll\_GetFeedbackDivRange, [132](#)
- IDTApll\_GetFreqDoublerSel, [132](#)
- IDTApll\_GetInputClkSrc, [132](#)
- IDTApll\_GetMasterResetSync, [133](#)
- IDTApll\_GetNonActiveApllConfig, [133](#)
- IDTApll\_GetOutputEnState, [134](#)
- IDTApll\_GetOutputSigType, [134](#)
- IDTApll\_GetPreDiv, [135](#)
- IDTApll\_GetPreDivRange, [135](#)
- IDTApll\_GetQaDiv, [135](#)
- IDTApll\_GetQbDiv, [136](#)
- IDTApll\_GetShutoff, [136](#)
- IDTApll\_GetSupportedXtal, [137](#)
- IDTApll\_GetXtalId, [137](#)
- IDTApll\_GetXtalIdFromXtalFreq, [137](#)
- IDTApll\_NullFreqTableEntry, [138](#)
- IDTApll\_SetApllConfig, [138](#)
- IDTApll\_SetApllConfigPerOutput, [138](#)
- IDTApll\_SetBypass, [139](#)
- IDTApll\_SetConfigSel, [139](#)
- IDTApll\_SetFeedbackDiv, [139](#)
- IDTApll\_SetFreqDoublerSel, [140](#)
- IDTApll\_SetInputClkSrc, [140](#)
- IDTApll\_SetMasterResetSync, [140](#)
- IDTApll\_SetOutputEnState, [141](#)
- IDTApll\_SetOutputSigType, [141](#)
- IDTApll\_SetPreDiv, [142](#)
- IDTApll\_SetQaDiv, [142](#)
- IDTApll\_SetQbDiv, [143](#)
- IDTApll\_SetShutoff, [143](#)
- IDTApll\_SetXtalId, [144](#)
- IDTApll\_Switch2NonActiveApllConfig, [144](#)
- IDTApll\_ValidXtalInput, [144](#)
- IDTApll\_XtalConfigured, [145](#)
- T\_idtApllAccessType, [124](#)
- T\_idtApllBypass, [124](#)
- T\_idtApllConfigSel, [124](#)
- T\_idtApllDividerValue, [123](#)
- T\_idtApllFreqDoublerSel, [124](#)
- T\_idtApllId, [125](#)
- T\_idtApllInputClkSrc, [125](#)
- T\_idtApllInputFreqType, [125](#)
- T\_idtApllMasterResetSync, [125](#)
- T\_idtApllOutput, [126](#)
- T\_idtApllOutputEn, [126](#)
- T\_idtApllOutputFreqType, [126](#)
- T\_idtApllOutputSigType, [127](#)
- T\_idtApllOutputSupportedType, [127](#)
- T\_idtApllQaDiv, [127](#)
- T\_idtApllQbDiv, [127](#)
- T\_idtApllRegAddr, [123](#)
- T\_idtApllRegMask, [123](#)
- T\_idtApllRegVal, [124](#)
- T\_idtApllShutoff, [128](#)
- T\_idtApllXtalId, [128](#)
- T\_idtApllXtalType, [128](#)
- apll/access/sim/ApllReadWrite\_sim.c, [205](#)
- apll/access/sim/ApllReadWrite\_sim.h, [207](#)
- apll/include/ApllApi.h, [208](#)
- apll/include/ApllFreqProfile.h, [210](#)
- apll/include/ApllLogging.h, [212](#)
- apll/include/ApllRegDef.h, [212](#)
- apll/include/ApllTypeDef.h, [213](#)
- apll/src/ApllApi.c, [215](#)
- apll/src/ApllFreqProfile.c, [219](#)
- ApllApi.c
  - APLL1\_REG\_HIGH\_BIT, [217](#)
  - APLL2\_REG\_HIGH\_BIT, [217](#)
  - APLL\_DIVIDER\_LSB, [218](#)

- APLL\_DIVIDER\_MSB, 218
- APLL\_DIVIDER\_VAL, 218
- APLL\_FB\_DIV\_LSB, 218
- APLL\_FB\_DIV\_MAX, 218
- APLL\_FB\_DIV\_MIN, 218
- APLL\_FB\_DIV\_MSB, 218
- APLL\_FB\_DIV\_VAL, 218
- APLL\_PRE\_DIV\_LSB, 218
- APLL\_PRE\_DIV\_MAX, 219
- APLL\_PRE\_DIV\_MIN, 219
- APLL\_PRE\_DIV\_MSB, 219
- APLL\_PRE\_DIV\_VAL, 219
- APLL\_REG\_OFFSET, 219
- apllCfg
  - T\_idtApllConfig, 158
  - T\_idtApllConfigPerOutput, 159
- ApllDelnit\_Sim
  - ApllReadWrite\_sim.c, 206
  - ApllReadWrite\_sim.h, 208
- apllDivVal\_m
  - T\_apllOutputFrequencyEntry, 157
- apllFbDivider
  - T\_idtApllFreqMapping, 161
- ApllFreqProfile.c
  - APLL\_FB\_DIV\_ETH, 221
  - APLL\_OUTPUT\_DIV\_1, 221
  - APLL\_OUTPUT\_DIV\_2, 221
  - APLL\_OUTPUT\_DIV\_25, 221
  - APLL\_OUTPUT\_DIV\_4, 221
  - APLL\_OUTPUT\_DIV\_5, 221
  - APLL\_OUTPUT\_DIV\_8, 221
  - APLL\_PRE\_DIV\_ETH, 221
- apllI2CTransData\_mp
  - drvHdlr\_s, 168
- ApllInit\_Sim
  - ApllReadWrite\_sim.c, 206
  - ApllReadWrite\_sim.h, 208
- apllInput
  - T\_idtOutputApllConfig, 171
- apllInputFreq
  - T\_idtApllFreqMapping, 161
- apllInst
  - T\_idtOutputApllConfig, 171
- apllOutput
  - T\_idtOutputApllConfig, 171
- apllOutputDivider
  - T\_idtApllFreqMapping, 161
- apllOutputFreq
  - T\_idtApllFreqMapping, 161
- apllOutputs
  - T\_idtApllFreqMapping, 161
- apllPreDivider
  - T\_idtApllFreqMapping, 161
- ApllRead\_Sim
  - ApllReadWrite\_sim.c, 206
  - ApllReadWrite\_sim.h, 208
- ApllReadWrite\_sim.c
  - ApllDelnit\_Sim, 206
  - ApllInit\_Sim, 206
  - ApllRead\_Sim, 206
  - ApllWrite\_Sim, 207
  - simulatedRegMapApll0, 207
  - simulatedRegMapApll1, 207
- ApllReadWrite\_sim.h
  - ApllDelnit\_Sim, 208
  - ApllInit\_Sim, 208
  - ApllRead\_Sim, 208
  - ApllWrite\_Sim, 208
- ApllWrite\_Sim
  - ApllReadWrite\_sim.c, 207
  - ApllReadWrite\_sim.h, 208
- apllXtalFreq
  - T\_idtApllFreqMapping, 161
  - T\_idtApllXtal, 162
- apllXtalId
  - T\_idtApllXtal, 162
- Automatic\_Mode
  - DPLL Function Module, 18
- BF\_2K\_8K\_PUL\_POSITION\_LSB
  - Register definitions, 113
- BF\_2K\_8K\_PUL\_POSITION\_MASK
  - Register definitions, 113
- BF\_2K\_8K\_PUL\_POSITION\_SIZE
  - Register definitions, 113
- BF\_2K\_EN\_LSB
  - Register definitions, 113
- BF\_2K\_EN\_MASK
  - Register definitions, 113
- BF\_2K\_EN\_SIZE
  - Register definitions, 113
- BF\_2K\_INV\_LSB
  - Register definitions, 113
- BF\_2K\_INV\_MASK
  - Register definitions, 113
- BF\_2K\_INV\_SIZE
  - Register definitions, 113
- BF\_2K\_PUL\_LSB
  - Register definitions, 113
- BF\_2K\_PUL\_MASK
  - Register definitions, 113
- BF\_2K\_PUL\_SIZE
  - Register definitions, 113
- BF\_400HZ\_OUT\_SEL\_LSB
  - Register definitions, 112
- BF\_400HZ\_OUT\_SEL\_MASK
  - Register definitions, 112
- BF\_400HZ\_OUT\_SEL\_SIZE
  - Register definitions, 112
- BF\_400HZ\_SEL\_LSB
  - Register definitions, 100
- BF\_400HZ\_SEL\_MASK
  - Register definitions, 100
- BF\_400HZ\_SEL\_SIZE
  - Register definitions, 100
- BF\_8K\_EN\_LSB
  - Register definitions, 113

- BF\_8K\_EN\_MASK
  - Register definitions, [113](#)
- BF\_8K\_EN\_SIZE
  - Register definitions, [113](#)
- BF\_8K\_INV\_LSB
  - Register definitions, [113](#)
- BF\_8K\_INV\_MASK
  - Register definitions, [113](#)
- BF\_8K\_INV\_SIZE
  - Register definitions, [113](#)
- BF\_8K\_PUL\_LSB
  - Register definitions, [113](#)
- BF\_8K\_PUL\_MASK
  - Register definitions, [113](#)
- BF\_8K\_PUL\_SIZE
  - Register definitions, [113](#)
- BF\_AMI1\_LOS\_LSB
  - Register definitions, [100](#)
- BF\_AMI1\_LOS\_MASK
  - Register definitions, [100](#)
- BF\_AMI1\_LOS\_SIZE
  - Register definitions, [100](#)
- BF\_AMI1\_VIOL\_LSB
  - Register definitions, [100](#)
- BF\_AMI1\_VIOL\_MASK
  - Register definitions, [100](#)
- BF\_AMI1\_VIOL\_SIZE
  - Register definitions, [100](#)
- BF\_AMI2\_LOS\_LSB
  - Register definitions, [100](#)
- BF\_AMI2\_LOS\_MASK
  - Register definitions, [100](#)
- BF\_AMI2\_LOS\_SIZE
  - Register definitions, [100](#)
- BF\_AMI2\_VIOL\_LSB
  - Register definitions, [100](#)
- BF\_AMI2\_VIOL\_MASK
  - Register definitions, [100](#)
- BF\_AMI2\_VIOL\_SIZE
  - Register definitions, [100](#)
- BF\_AMI\_OUT\_DUTY\_LSB
  - Register definitions, [112](#)
- BF\_AMI\_OUT\_DUTY\_MASK
  - Register definitions, [112](#)
- BF\_AMI\_OUT\_DUTY\_SIZE
  - Register definitions, [112](#)
- BF\_AUTO\_AVG\_LSB
  - Register definitions, [109](#)
- BF\_AUTO\_AVG\_MASK
  - Register definitions, [109](#)
- BF\_AUTO\_AVG\_SIZE
  - Register definitions, [109](#)
- BF\_AUTO\_BW\_SEL\_LSB
  - Register definitions, [108](#)
- BF\_AUTO\_BW\_SEL\_MASK
  - Register definitions, [108](#)
- BF\_AUTO\_BW\_SEL\_SIZE
  - Register definitions, [108](#)
- BF\_AUTO\_EXT\_SYNC\_EN\_LSB
  - Register definitions, [97](#)
- BF\_AUTO\_EXT\_SYNC\_EN\_MASK
  - Register definitions, [97](#)
- BF\_AUTO\_EXT\_SYNC\_EN\_SIZE
  - Register definitions, [97](#)
- BF\_BUCKET\_SEL\_LSB
  - Register definitions, [100](#)
- BF\_BUCKET\_SEL\_MASK
  - Register definitions, [100](#)
- BF\_BUCKET\_SEL\_SIZE
  - Register definitions, [100](#)
- BF\_BUCKET\_SIZE\_DATA\_LSB
  - Register definitions, [103](#)
- BF\_BUCKET\_SIZE\_DATA\_MASK
  - Register definitions, [103](#)
- BF\_BUCKET\_SIZE\_DATA\_SIZE
  - Register definitions, [102](#)
- BF\_BYPASS\_LSB
  - Register definitions, [116](#)
- BF\_BYPASS\_MASK
  - Register definitions, [116](#)
- BF\_BYPASS\_SIZE
  - Register definitions, [116](#)
- BF\_CLK\_SEL\_LSB
  - Register definitions, [115](#)
- BF\_CLK\_SEL\_MASK
  - Register definitions, [115](#)
- BF\_CLK\_SEL\_SIZE
  - Register definitions, [115](#)
- BF\_COARSE\_PH\_LOS\_LIMT\_EN\_LSB
  - Register definitions, [109](#)
- BF\_COARSE\_PH\_LOS\_LIMT\_EN\_MASK
  - Register definitions, [109](#)
- BF\_COARSE\_PH\_LOS\_LIMT\_EN\_SIZE
  - Register definitions, [109](#)
- BF\_CONFIG\_LSB
  - Register definitions, [115](#)
- BF\_CONFIG\_MASK
  - Register definitions, [115](#)
- BF\_CONFIG\_SIZE
  - Register definitions, [115](#)
- BF\_CURRENT\_DPLL\_FREQ\_A\_LSB
  - Register definitions, [110](#)
- BF\_CURRENT\_DPLL\_FREQ\_A\_MASK
  - Register definitions, [110](#)
- BF\_CURRENT\_DPLL\_FREQ\_A\_SIZE
  - Register definitions, [110](#)
- BF\_CURRENT\_DPLL\_FREQ\_B\_LSB
  - Register definitions, [110](#)
- BF\_CURRENT\_DPLL\_FREQ\_B\_MASK
  - Register definitions, [110](#)
- BF\_CURRENT\_DPLL\_FREQ\_B\_SIZE
  - Register definitions, [110](#)
- BF\_CURRENT\_DPLL\_FREQ\_C\_LSB
  - Register definitions, [110](#)
- BF\_CURRENT\_DPLL\_FREQ\_C\_MASK
  - Register definitions, [110](#)

- BF\_CURRENT\_DPLL\_FREQ\_C\_SIZE  
Register definitions, [110](#)
- BF\_CURRENT\_PH\_DATA\_A\_LSB  
Register definitions, [110](#)
- BF\_CURRENT\_PH\_DATA\_A\_MASK  
Register definitions, [110](#)
- BF\_CURRENT\_PH\_DATA\_A\_SIZE  
Register definitions, [110](#)
- BF\_CURRENT\_PH\_DATA\_B\_LSB  
Register definitions, [111](#)
- BF\_CURRENT\_PH\_DATA\_B\_MASK  
Register definitions, [110](#)
- BF\_CURRENT\_PH\_DATA\_B\_SIZE  
Register definitions, [110](#)
- BF\_CURRENTLY\_SELECTED\_INPUTS\_LSB  
Register definitions, [107](#)
- BF\_CURRENTLY\_SELECTED\_INPUTS\_MASK  
Register definitions, [107](#)
- BF\_CURRENTLY\_SELECTED\_INPUTS\_SIZE  
Register definitions, [107](#)
- BF\_DECAY\_RATE\_DATA\_LSB  
Register definitions, [103](#)
- BF\_DECAY\_RATE\_DATA\_MASK  
Register definitions, [103](#)
- BF\_DECAY\_RATE\_DATA\_SIZE  
Register definitions, [103](#)
- BF\_DIRECT\_DIV\_LSB  
Register definitions, [101](#)
- BF\_DIRECT\_DIV\_MASK  
Register definitions, [100](#)
- BF\_DIRECT\_DIV\_SIZE  
Register definitions, [100](#)
- BF\_DPLL\_FREQ\_HARD\_LIMT\_A\_LSB  
Register definitions, [110](#)
- BF\_DPLL\_FREQ\_HARD\_LIMT\_A\_MASK  
Register definitions, [110](#)
- BF\_DPLL\_FREQ\_HARD\_LIMT\_A\_SIZE  
Register definitions, [110](#)
- BF\_DPLL\_FREQ\_HARD\_LIMT\_B\_LSB  
Register definitions, [110](#)
- BF\_DPLL\_FREQ\_HARD\_LIMT\_B\_MASK  
Register definitions, [110](#)
- BF\_DPLL\_FREQ\_HARD\_LIMT\_B\_SIZE  
Register definitions, [110](#)
- BF\_DPLL\_FREQ\_SOFT\_LIMT\_LSB  
Register definitions, [110](#)
- BF\_DPLL\_FREQ\_SOFT\_LIMT\_MASK  
Register definitions, [110](#)
- BF\_DPLL\_FREQ\_SOFT\_LIMT\_SIZE  
Register definitions, [110](#)
- BF\_EX\_SYNC\_ALARM\_LSB  
Register definitions, [100](#)
- BF\_EX\_SYNC\_ALARM\_MASK  
Register definitions, [100](#)
- BF\_EX\_SYNC\_ALARM\_MON\_LSB  
Register definitions, [107](#)
- BF\_EX\_SYNC\_ALARM\_MON\_MASK  
Register definitions, [107](#)
- BF\_EX\_SYNC\_ALARM\_MON\_SIZE  
Register definitions, [107](#)
- BF\_EX\_SYNC\_ALARM\_SIZE  
Register definitions, [100](#)
- BF\_EXT\_SW\_LSB  
Register definitions, [98](#)
- BF\_EXT\_SW\_MASK  
Register definitions, [98](#)
- BF\_EXT\_SW\_SIZE  
Register definitions, [98](#)
- BF\_EXT\_SYNC\_EN\_LSB  
Register definitions, [97](#)
- BF\_EXT\_SYNC\_EN\_MASK  
Register definitions, [97](#)
- BF\_EXT\_SYNC\_EN\_SIZE  
Register definitions, [97](#)
- BF\_FAST\_AVG\_LSB  
Register definitions, [109](#)
- BF\_FAST\_AVG\_MASK  
Register definitions, [109](#)
- BF\_FAST\_AVG\_SIZE  
Register definitions, [109](#)
- BF\_FAST\_LOS\_SW\_LSB  
Register definitions, [109](#)
- BF\_FAST\_LOS\_SW\_MASK  
Register definitions, [109](#)
- BF\_FAST\_LOS\_SW\_SIZE  
Register definitions, [109](#)
- BF\_FINE\_PH\_LOS\_LIMT\_EN\_LSB  
Register definitions, [109](#)
- BF\_FINE\_PH\_LOS\_LIMT\_EN\_MASK  
Register definitions, [109](#)
- BF\_FINE\_PH\_LOS\_LIMT\_EN\_SIZE  
Register definitions, [109](#)
- BF\_FREQ\_LIMT\_PH\_LOS\_LSB  
Register definitions, [110](#)
- BF\_FREQ\_LIMT\_PH\_LOS\_MASK  
Register definitions, [110](#)
- BF\_FREQ\_LIMT\_PH\_LOS\_SIZE  
Register definitions, [110](#)
- BF\_FREQ\_MON\_CLK\_LSB  
Register definitions, [98](#)
- BF\_FREQ\_MON\_CLK\_MASK  
Register definitions, [98](#)
- BF\_FREQ\_MON\_CLK\_SIZE  
Register definitions, [98](#)
- BF\_FREQ\_MON\_FACTOR\_LSB  
Register definitions, [102](#)
- BF\_FREQ\_MON\_FACTOR\_MASK  
Register definitions, [102](#)
- BF\_FREQ\_MON\_FACTOR\_SIZE  
Register definitions, [102](#)
- BF\_FREQ\_MON\_HARD\_EN\_LSB  
Register definitions, [98](#)
- BF\_FREQ\_MON\_HARD\_EN\_MASK  
Register definitions, [98](#)
- BF\_FREQ\_MON\_HARD\_EN\_SIZE  
Register definitions, [98](#)

- BF\_HARD\_FREQ\_MON\_THRESHOLD\_A\_LSB  
Register definitions, [102](#)
- BF\_HARD\_FREQ\_MON\_THRESHOLD\_A\_MASK  
Register definitions, [102](#)
- BF\_HARD\_FREQ\_MON\_THRESHOLD\_A\_SIZE  
Register definitions, [102](#)
- BF\_HARD\_FREQ\_MON\_THRESHOLD\_B\_LSB  
Register definitions, [102](#)
- BF\_HARD\_FREQ\_MON\_THRESHOLD\_B\_MASK  
Register definitions, [102](#)
- BF\_HARD\_FREQ\_MON\_THRESHOLD\_B\_SIZE  
Register definitions, [102](#)
- BF\_HIGHEST\_PRIORITY\_VALIDATED\_LSB  
Register definitions, [107](#)
- BF\_HIGHEST\_PRIORITY\_VALIDATED\_MASK  
Register definitions, [107](#)
- BF\_HIGHEST\_PRIORITY\_VALIDATED\_SIZE  
Register definitions, [107](#)
- BF\_HOLDOVER\_FREQ\_A\_LSB  
Register definitions, [109](#)
- BF\_HOLDOVER\_FREQ\_A\_MASK  
Register definitions, [109](#)
- BF\_HOLDOVER\_FREQ\_A\_SIZE  
Register definitions, [109](#)
- BF\_HOLDOVER\_FREQ\_B\_LSB  
Register definitions, [110](#)
- BF\_HOLDOVER\_FREQ\_B\_MASK  
Register definitions, [109](#)
- BF\_HOLDOVER\_FREQ\_B\_SIZE  
Register definitions, [109](#)
- BF\_HOLDOVER\_FREQ\_C\_LSB  
Register definitions, [110](#)
- BF\_HOLDOVER\_FREQ\_C\_MASK  
Register definitions, [110](#)
- BF\_HOLDOVER\_FREQ\_C\_SIZE  
Register definitions, [110](#)
- BF\_HS\_EN\_LSB  
Register definitions, [98](#)
- BF\_HS\_EN\_MASK  
Register definitions, [98](#)
- BF\_HS\_EN\_SIZE  
Register definitions, [98](#)
- BF\_HS\_FREQ\_LSB  
Register definitions, [98](#)
- BF\_HS\_FREQ\_MASK  
Register definitions, [98](#)
- BF\_HS\_FREQ\_SIZE  
Register definitions, [98](#)
- BF\_HZ\_EN\_LSB  
Register definitions, [98](#)
- BF\_HZ\_EN\_MASK  
Register definitions, [98](#)
- BF\_HZ\_EN\_SIZE  
Register definitions, [98](#)
- BF\_ICP\_CTRL\_CODE\_LSB  
Register definitions, [115](#)
- BF\_ICP\_CTRL\_CODE\_MASK  
Register definitions, [115](#)
- BF\_ICP\_CTRL\_CODE\_SIZE  
Register definitions, [115](#)
- BF\_ID\_A\_LSB  
Register definitions, [96](#)
- BF\_ID\_A\_MASK  
Register definitions, [96](#)
- BF\_ID\_A\_SIZE  
Register definitions, [96](#)
- BF\_ID\_B\_LSB  
Register definitions, [96](#)
- BF\_ID\_B\_MASK  
Register definitions, [96](#)
- BF\_ID\_B\_SIZE  
Register definitions, [96](#)
- BF\_IN10\_FREQ\_HARD\_ALARM\_LSB  
Register definitions, [105](#)
- BF\_IN10\_FREQ\_HARD\_ALARM\_MASK  
Register definitions, [105](#)
- BF\_IN10\_FREQ\_HARD\_ALARM\_SIZE  
Register definitions, [105](#)
- BF\_IN10\_FREQ\_SOFT\_ALARM\_LSB  
Register definitions, [105](#)
- BF\_IN10\_FREQ\_SOFT\_ALARM\_MASK  
Register definitions, [105](#)
- BF\_IN10\_FREQ\_SOFT\_ALARM\_SIZE  
Register definitions, [105](#)
- BF\_IN10\_LSB  
Register definitions, [99](#)
- BF\_IN10\_MASK  
Register definitions, [99](#)
- BF\_IN10\_NO\_ACTIVITY\_ALARM\_LSB  
Register definitions, [105](#)
- BF\_IN10\_NO\_ACTIVITY\_ALARM\_MASK  
Register definitions, [105](#)
- BF\_IN10\_NO\_ACTIVITY\_ALARM\_SIZE  
Register definitions, [105](#)
- BF\_IN10\_PH\_LOCK\_ALARM\_LSB  
Register definitions, [105](#)
- BF\_IN10\_PH\_LOCK\_ALARM\_MASK  
Register definitions, [105](#)
- BF\_IN10\_PH\_LOCK\_ALARM\_SIZE  
Register definitions, [105](#)
- BF\_IN10\_SEL\_PRIORITY\_LSB  
Register definitions, [102](#)
- BF\_IN10\_SEL\_PRIORITY\_MASK  
Register definitions, [102](#)
- BF\_IN10\_SEL\_PRIORITY\_SIZE  
Register definitions, [101](#)
- BF\_IN10\_SIZE  
Register definitions, [99](#)
- BF\_IN11\_FREQ\_HARD\_ALARM\_LSB  
Register definitions, [106](#)
- BF\_IN11\_FREQ\_HARD\_ALARM\_MASK  
Register definitions, [106](#)
- BF\_IN11\_FREQ\_HARD\_ALARM\_SIZE  
Register definitions, [106](#)
- BF\_IN11\_FREQ\_SOFT\_ALARM\_LSB  
Register definitions, [106](#)

- BF\_IN11\_FREQ\_SOFT\_ALARM\_MASK  
Register definitions, [106](#)
- BF\_IN11\_FREQ\_SOFT\_ALARM\_SIZE  
Register definitions, [106](#)
- BF\_IN11\_LSB  
Register definitions, [99](#)
- BF\_IN11\_MASK  
Register definitions, [99](#)
- BF\_IN11\_NO\_ACTIVITY\_ALARM\_LSB  
Register definitions, [106](#)
- BF\_IN11\_NO\_ACTIVITY\_ALARM\_MASK  
Register definitions, [106](#)
- BF\_IN11\_NO\_ACTIVITY\_ALARM\_SIZE  
Register definitions, [106](#)
- BF\_IN11\_PH\_LOCK\_ALARM\_LSB  
Register definitions, [106](#)
- BF\_IN11\_PH\_LOCK\_ALARM\_MASK  
Register definitions, [106](#)
- BF\_IN11\_PH\_LOCK\_ALARM\_SIZE  
Register definitions, [106](#)
- BF\_IN11\_SEL\_PRIORITY\_LSB  
Register definitions, [102](#)
- BF\_IN11\_SEL\_PRIORITY\_MASK  
Register definitions, [102](#)
- BF\_IN11\_SEL\_PRIORITY\_SIZE  
Register definitions, [102](#)
- BF\_IN11\_SIZE  
Register definitions, [99](#)
- BF\_IN12\_FREQ\_HARD\_ALARM\_LSB  
Register definitions, [106](#)
- BF\_IN12\_FREQ\_HARD\_ALARM\_MASK  
Register definitions, [106](#)
- BF\_IN12\_FREQ\_HARD\_ALARM\_SIZE  
Register definitions, [106](#)
- BF\_IN12\_FREQ\_SOFT\_ALARM\_LSB  
Register definitions, [106](#)
- BF\_IN12\_FREQ\_SOFT\_ALARM\_MASK  
Register definitions, [105](#)
- BF\_IN12\_FREQ\_SOFT\_ALARM\_SIZE  
Register definitions, [105](#)
- BF\_IN12\_LSB  
Register definitions, [99](#)
- BF\_IN12\_MASK  
Register definitions, [99](#)
- BF\_IN12\_NO\_ACTIVITY\_ALARM\_LSB  
Register definitions, [106](#)
- BF\_IN12\_NO\_ACTIVITY\_ALARM\_MASK  
Register definitions, [106](#)
- BF\_IN12\_NO\_ACTIVITY\_ALARM\_SIZE  
Register definitions, [106](#)
- BF\_IN12\_PH\_LOCK\_ALARM\_LSB  
Register definitions, [106](#)
- BF\_IN12\_PH\_LOCK\_ALARM\_MASK  
Register definitions, [106](#)
- BF\_IN12\_PH\_LOCK\_ALARM\_SIZE  
Register definitions, [106](#)
- BF\_IN12\_SEL\_PRIORITY\_LSB  
Register definitions, [102](#)
- BF\_IN12\_SEL\_PRIORITY\_MASK  
Register definitions, [102](#)
- BF\_IN12\_SEL\_PRIORITY\_SIZE  
Register definitions, [102](#)
- BF\_IN12\_SIZE  
Register definitions, [99](#)
- BF\_IN13\_FREQ\_HARD\_ALARM\_LSB  
Register definitions, [106](#)
- BF\_IN13\_FREQ\_HARD\_ALARM\_MASK  
Register definitions, [106](#)
- BF\_IN13\_FREQ\_HARD\_ALARM\_SIZE  
Register definitions, [106](#)
- BF\_IN13\_FREQ\_SOFT\_ALARM\_LSB  
Register definitions, [106](#)
- BF\_IN13\_FREQ\_SOFT\_ALARM\_MASK  
Register definitions, [106](#)
- BF\_IN13\_FREQ\_SOFT\_ALARM\_SIZE  
Register definitions, [106](#)
- BF\_IN13\_LSB  
Register definitions, [99](#)
- BF\_IN13\_MASK  
Register definitions, [99](#)
- BF\_IN13\_NO\_ACTIVITY\_ALARM\_LSB  
Register definitions, [106](#)
- BF\_IN13\_NO\_ACTIVITY\_ALARM\_MASK  
Register definitions, [106](#)
- BF\_IN13\_NO\_ACTIVITY\_ALARM\_SIZE  
Register definitions, [106](#)
- BF\_IN13\_PH\_LOCK\_ALARM\_LSB  
Register definitions, [107](#)
- BF\_IN13\_PH\_LOCK\_ALARM\_MASK  
Register definitions, [106](#)
- BF\_IN13\_PH\_LOCK\_ALARM\_SIZE  
Register definitions, [106](#)
- BF\_IN13\_SEL\_PRIORITY\_LSB  
Register definitions, [102](#)
- BF\_IN13\_SEL\_PRIORITY\_MASK  
Register definitions, [102](#)
- BF\_IN13\_SEL\_PRIORITY\_SIZE  
Register definitions, [102](#)
- BF\_IN13\_SIZE  
Register definitions, [99](#)
- BF\_IN14\_FREQ\_HARD\_ALARM\_LSB  
Register definitions, [106](#)
- BF\_IN14\_FREQ\_HARD\_ALARM\_MASK  
Register definitions, [106](#)
- BF\_IN14\_FREQ\_HARD\_ALARM\_SIZE  
Register definitions, [106](#)
- BF\_IN14\_FREQ\_SOFT\_ALARM\_LSB  
Register definitions, [106](#)
- BF\_IN14\_FREQ\_SOFT\_ALARM\_MASK  
Register definitions, [106](#)
- BF\_IN14\_FREQ\_SOFT\_ALARM\_SIZE  
Register definitions, [106](#)
- BF\_IN14\_LSB  
Register definitions, [99](#)
- BF\_IN14\_MASK  
Register definitions, [99](#)

- BF\_IN14\_NO\_ACTIVITY\_ALARM\_LSB
  - Register definitions, [106](#)
- BF\_IN14\_NO\_ACTIVITY\_ALARM\_MASK
  - Register definitions, [106](#)
- BF\_IN14\_NO\_ACTIVITY\_ALARM\_SIZE
  - Register definitions, [106](#)
- BF\_IN14\_PH\_LOCK\_ALARM\_LSB
  - Register definitions, [106](#)
- BF\_IN14\_PH\_LOCK\_ALARM\_MASK
  - Register definitions, [106](#)
- BF\_IN14\_PH\_LOCK\_ALARM\_SIZE
  - Register definitions, [106](#)
- BF\_IN14\_SEL\_PRIORITY\_LSB
  - Register definitions, [102](#)
- BF\_IN14\_SEL\_PRIORITY\_MASK
  - Register definitions, [102](#)
- BF\_IN14\_SEL\_PRIORITY\_SIZE
  - Register definitions, [102](#)
- BF\_IN14\_SIZE
  - Register definitions, [99](#)
- BF\_IN1\_FREQ\_HARD\_ALARM\_LSB
  - Register definitions, [103](#)
- BF\_IN1\_FREQ\_HARD\_ALARM\_MASK
  - Register definitions, [103](#)
- BF\_IN1\_FREQ\_HARD\_ALARM\_SIZE
  - Register definitions, [103](#)
- BF\_IN1\_FREQ\_SOFT\_ALARM\_LSB
  - Register definitions, [103](#)
- BF\_IN1\_FREQ\_SOFT\_ALARM\_MASK
  - Register definitions, [103](#)
- BF\_IN1\_FREQ\_SOFT\_ALARM\_SIZE
  - Register definitions, [103](#)
- BF\_IN1\_LSB
  - Register definitions, [99](#)
- BF\_IN1\_MASK
  - Register definitions, [99](#)
- BF\_IN1\_NO\_ACTIVITY\_ALARM\_LSB
  - Register definitions, [103](#)
- BF\_IN1\_NO\_ACTIVITY\_ALARM\_MASK
  - Register definitions, [103](#)
- BF\_IN1\_NO\_ACTIVITY\_ALARM\_SIZE
  - Register definitions, [103](#)
- BF\_IN1\_PH\_LOCK\_ALARM\_LSB
  - Register definitions, [103](#)
- BF\_IN1\_PH\_LOCK\_ALARM\_MASK
  - Register definitions, [103](#)
- BF\_IN1\_PH\_LOCK\_ALARM\_SIZE
  - Register definitions, [103](#)
- BF\_IN1\_SEL\_PRIORITY\_LSB
  - Register definitions, [101](#)
- BF\_IN1\_SEL\_PRIORITY\_MASK
  - Register definitions, [101](#)
- BF\_IN1\_SEL\_PRIORITY\_SIZE
  - Register definitions, [101](#)
- BF\_IN1\_SIZE
  - Register definitions, [99](#)
- BF\_IN2\_FREQ\_HARD\_ALARM\_LSB
  - Register definitions, [103](#)
- BF\_IN2\_FREQ\_HARD\_ALARM\_MASK
  - Register definitions, [103](#)
- BF\_IN2\_FREQ\_HARD\_ALARM\_SIZE
  - Register definitions, [103](#)
- BF\_IN2\_FREQ\_SOFT\_ALARM\_LSB
  - Register definitions, [103](#)
- BF\_IN2\_FREQ\_SOFT\_ALARM\_MASK
  - Register definitions, [103](#)
- BF\_IN2\_FREQ\_SOFT\_ALARM\_SIZE
  - Register definitions, [103](#)
- BF\_IN2\_LSB
  - Register definitions, [99](#)
- BF\_IN2\_MASK
  - Register definitions, [99](#)
- BF\_IN2\_NO\_ACTIVITY\_ALARM\_LSB
  - Register definitions, [103](#)
- BF\_IN2\_NO\_ACTIVITY\_ALARM\_MASK
  - Register definitions, [103](#)
- BF\_IN2\_NO\_ACTIVITY\_ALARM\_SIZE
  - Register definitions, [103](#)
- BF\_IN2\_PH\_LOCK\_ALARM\_LSB
  - Register definitions, [103](#)
- BF\_IN2\_PH\_LOCK\_ALARM\_MASK
  - Register definitions, [103](#)
- BF\_IN2\_PH\_LOCK\_ALARM\_SIZE
  - Register definitions, [103](#)
- BF\_IN2\_SEL\_PRIORITY\_LSB
  - Register definitions, [101](#)
- BF\_IN2\_SEL\_PRIORITY\_MASK
  - Register definitions, [101](#)
- BF\_IN2\_SEL\_PRIORITY\_SIZE
  - Register definitions, [101](#)
- BF\_IN2\_SIZE
  - Register definitions, [99](#)
- BF\_IN3\_FREQ\_HARD\_ALARM\_LSB
  - Register definitions, [104](#)
- BF\_IN3\_FREQ\_HARD\_ALARM\_MASK
  - Register definitions, [104](#)
- BF\_IN3\_FREQ\_HARD\_ALARM\_SIZE
  - Register definitions, [104](#)
- BF\_IN3\_FREQ\_SOFT\_ALARM\_LSB
  - Register definitions, [104](#)
- BF\_IN3\_FREQ\_SOFT\_ALARM\_MASK
  - Register definitions, [104](#)
- BF\_IN3\_FREQ\_SOFT\_ALARM\_SIZE
  - Register definitions, [104](#)
- BF\_IN3\_LSB
  - Register definitions, [99](#)
- BF\_IN3\_MASK
  - Register definitions, [99](#)
- BF\_IN3\_NO\_ACTIVITY\_ALARM\_LSB
  - Register definitions, [104](#)
- BF\_IN3\_NO\_ACTIVITY\_ALARM\_MASK
  - Register definitions, [104](#)
- BF\_IN3\_NO\_ACTIVITY\_ALARM\_SIZE
  - Register definitions, [104](#)
- BF\_IN3\_PH\_LOCK\_ALARM\_LSB
  - Register definitions, [104](#)

- BF\_IN3\_PH\_LOCK\_ALARM\_MASK  
Register definitions, [104](#)
- BF\_IN3\_PH\_LOCK\_ALARM\_SIZE  
Register definitions, [104](#)
- BF\_IN3\_SEL\_PRIORITY\_LSB  
Register definitions, [101](#)
- BF\_IN3\_SEL\_PRIORITY\_MASK  
Register definitions, [101](#)
- BF\_IN3\_SEL\_PRIORITY\_SIZE  
Register definitions, [101](#)
- BF\_IN3\_SIZE  
Register definitions, [99](#)
- BF\_IN4\_FREQ\_HARD\_ALARM\_LSB  
Register definitions, [103](#)
- BF\_IN4\_FREQ\_HARD\_ALARM\_MASK  
Register definitions, [103](#)
- BF\_IN4\_FREQ\_HARD\_ALARM\_SIZE  
Register definitions, [103](#)
- BF\_IN4\_FREQ\_SOFT\_ALARM\_LSB  
Register definitions, [103](#)
- BF\_IN4\_FREQ\_SOFT\_ALARM\_MASK  
Register definitions, [103](#)
- BF\_IN4\_FREQ\_SOFT\_ALARM\_SIZE  
Register definitions, [103](#)
- BF\_IN4\_LSB  
Register definitions, [99](#)
- BF\_IN4\_MASK  
Register definitions, [99](#)
- BF\_IN4\_NO\_ACTIVITY\_ALARM\_LSB  
Register definitions, [104](#)
- BF\_IN4\_NO\_ACTIVITY\_ALARM\_MASK  
Register definitions, [103](#)
- BF\_IN4\_NO\_ACTIVITY\_ALARM\_SIZE  
Register definitions, [103](#)
- BF\_IN4\_PH\_LOCK\_ALARM\_LSB  
Register definitions, [104](#)
- BF\_IN4\_PH\_LOCK\_ALARM\_MASK  
Register definitions, [104](#)
- BF\_IN4\_PH\_LOCK\_ALARM\_SIZE  
Register definitions, [104](#)
- BF\_IN4\_SEL\_PRIORITY\_LSB  
Register definitions, [101](#)
- BF\_IN4\_SEL\_PRIORITY\_MASK  
Register definitions, [101](#)
- BF\_IN4\_SEL\_PRIORITY\_SIZE  
Register definitions, [101](#)
- BF\_IN4\_SIZE  
Register definitions, [99](#)
- BF\_IN5\_DIV\_LSB  
Register definitions, [101](#)
- BF\_IN5\_DIV\_MASK  
Register definitions, [101](#)
- BF\_IN5\_DIV\_SIZE  
Register definitions, [101](#)
- BF\_IN5\_FREQ\_HARD\_ALARM\_LSB  
Register definitions, [104](#)
- BF\_IN5\_FREQ\_HARD\_ALARM\_MASK  
Register definitions, [104](#)
- BF\_IN5\_FREQ\_HARD\_ALARM\_SIZE  
Register definitions, [104](#)
- BF\_IN5\_FREQ\_SOFT\_ALARM\_LSB  
Register definitions, [104](#)
- BF\_IN5\_FREQ\_SOFT\_ALARM\_MASK  
Register definitions, [104](#)
- BF\_IN5\_FREQ\_SOFT\_ALARM\_SIZE  
Register definitions, [104](#)
- BF\_IN5\_LSB  
Register definitions, [99](#)
- BF\_IN5\_MASK  
Register definitions, [99](#)
- BF\_IN5\_NO\_ACTIVITY\_ALARM\_LSB  
Register definitions, [104](#)
- BF\_IN5\_NO\_ACTIVITY\_ALARM\_MASK  
Register definitions, [104](#)
- BF\_IN5\_NO\_ACTIVITY\_ALARM\_SIZE  
Register definitions, [104](#)
- BF\_IN5\_PH\_LOCK\_ALARM\_LSB  
Register definitions, [104](#)
- BF\_IN5\_PH\_LOCK\_ALARM\_MASK  
Register definitions, [104](#)
- BF\_IN5\_PH\_LOCK\_ALARM\_SIZE  
Register definitions, [104](#)
- BF\_IN5\_SEL\_PRIORITY\_LSB  
Register definitions, [101](#)
- BF\_IN5\_SEL\_PRIORITY\_MASK  
Register definitions, [101](#)
- BF\_IN5\_SEL\_PRIORITY\_SIZE  
Register definitions, [101](#)
- BF\_IN5\_SIZE  
Register definitions, [98](#)
- BF\_IN6\_DIV\_LSB  
Register definitions, [101](#)
- BF\_IN6\_DIV\_MASK  
Register definitions, [101](#)
- BF\_IN6\_DIV\_SIZE  
Register definitions, [101](#)
- BF\_IN6\_FREQ\_HARD\_ALARM\_LSB  
Register definitions, [104](#)
- BF\_IN6\_FREQ\_HARD\_ALARM\_MASK  
Register definitions, [104](#)
- BF\_IN6\_FREQ\_HARD\_ALARM\_SIZE  
Register definitions, [104](#)
- BF\_IN6\_FREQ\_SOFT\_ALARM\_LSB  
Register definitions, [104](#)
- BF\_IN6\_FREQ\_SOFT\_ALARM\_MASK  
Register definitions, [104](#)
- BF\_IN6\_FREQ\_SOFT\_ALARM\_SIZE  
Register definitions, [104](#)
- BF\_IN6\_LSB  
Register definitions, [98](#)
- BF\_IN6\_MASK  
Register definitions, [98](#)
- BF\_IN6\_NO\_ACTIVITY\_ALARM\_LSB  
Register definitions, [104](#)
- BF\_IN6\_NO\_ACTIVITY\_ALARM\_MASK  
Register definitions, [104](#)

- BF\_IN6\_NO\_ACTIVITY\_ALARM\_SIZE
  - Register definitions, [104](#)
- BF\_IN6\_PH\_LOCK\_ALARM\_LSB
  - Register definitions, [104](#)
- BF\_IN6\_PH\_LOCK\_ALARM\_MASK
  - Register definitions, [104](#)
- BF\_IN6\_PH\_LOCK\_ALARM\_SIZE
  - Register definitions, [104](#)
- BF\_IN6\_SEL\_PRIORITY\_LSB
  - Register definitions, [101](#)
- BF\_IN6\_SEL\_PRIORITY\_MASK
  - Register definitions, [101](#)
- BF\_IN6\_SEL\_PRIORITY\_SIZE
  - Register definitions, [101](#)
- BF\_IN6\_SIZE
  - Register definitions, [98](#)
- BF\_IN7\_FREQ\_HARD\_ALARM\_LSB
  - Register definitions, [105](#)
- BF\_IN7\_FREQ\_HARD\_ALARM\_MASK
  - Register definitions, [105](#)
- BF\_IN7\_FREQ\_HARD\_ALARM\_SIZE
  - Register definitions, [105](#)
- BF\_IN7\_FREQ\_SOFT\_ALARM\_LSB
  - Register definitions, [105](#)
- BF\_IN7\_FREQ\_SOFT\_ALARM\_MASK
  - Register definitions, [105](#)
- BF\_IN7\_FREQ\_SOFT\_ALARM\_SIZE
  - Register definitions, [105](#)
- BF\_IN7\_LSB
  - Register definitions, [98](#)
- BF\_IN7\_MASK
  - Register definitions, [98](#)
- BF\_IN7\_NO\_ACTIVITY\_ALARM\_LSB
  - Register definitions, [105](#)
- BF\_IN7\_NO\_ACTIVITY\_ALARM\_MASK
  - Register definitions, [105](#)
- BF\_IN7\_NO\_ACTIVITY\_ALARM\_SIZE
  - Register definitions, [105](#)
- BF\_IN7\_PH\_LOCK\_ALARM\_LSB
  - Register definitions, [105](#)
- BF\_IN7\_PH\_LOCK\_ALARM\_MASK
  - Register definitions, [105](#)
- BF\_IN7\_PH\_LOCK\_ALARM\_SIZE
  - Register definitions, [105](#)
- BF\_IN7\_SEL\_PRIORITY\_LSB
  - Register definitions, [101](#)
- BF\_IN7\_SEL\_PRIORITY\_MASK
  - Register definitions, [101](#)
- BF\_IN7\_SEL\_PRIORITY\_SIZE
  - Register definitions, [101](#)
- BF\_IN7\_SIZE
  - Register definitions, [98](#)
- BF\_IN8\_FREQ\_HARD\_ALARM\_LSB
  - Register definitions, [105](#)
- BF\_IN8\_FREQ\_HARD\_ALARM\_MASK
  - Register definitions, [104](#)
- BF\_IN8\_FREQ\_HARD\_ALARM\_SIZE
  - Register definitions, [104](#)
- BF\_IN8\_FREQ\_SOFT\_ALARM\_LSB
  - Register definitions, [104](#)
- BF\_IN8\_FREQ\_SOFT\_ALARM\_MASK
  - Register definitions, [104](#)
- BF\_IN8\_FREQ\_SOFT\_ALARM\_SIZE
  - Register definitions, [104](#)
- BF\_IN8\_LSB
  - Register definitions, [98](#)
- BF\_IN8\_MASK
  - Register definitions, [98](#)
- BF\_IN8\_NO\_ACTIVITY\_ALARM\_LSB
  - Register definitions, [105](#)
- BF\_IN8\_NO\_ACTIVITY\_ALARM\_MASK
  - Register definitions, [105](#)
- BF\_IN8\_NO\_ACTIVITY\_ALARM\_SIZE
  - Register definitions, [105](#)
- BF\_IN8\_PH\_LOCK\_ALARM\_LSB
  - Register definitions, [105](#)
- BF\_IN8\_PH\_LOCK\_ALARM\_MASK
  - Register definitions, [105](#)
- BF\_IN8\_PH\_LOCK\_ALARM\_SIZE
  - Register definitions, [105](#)
- BF\_IN8\_SEL\_PRIORITY\_LSB
  - Register definitions, [101](#)
- BF\_IN8\_SEL\_PRIORITY\_MASK
  - Register definitions, [101](#)
- BF\_IN8\_SEL\_PRIORITY\_SIZE
  - Register definitions, [101](#)
- BF\_IN8\_SIZE
  - Register definitions, [98](#)
- BF\_IN9\_FREQ\_HARD\_ALARM\_LSB
  - Register definitions, [105](#)
- BF\_IN9\_FREQ\_HARD\_ALARM\_MASK
  - Register definitions, [105](#)
- BF\_IN9\_FREQ\_HARD\_ALARM\_SIZE
  - Register definitions, [105](#)
- BF\_IN9\_FREQ\_SOFT\_ALARM\_LSB
  - Register definitions, [105](#)
- BF\_IN9\_FREQ\_SOFT\_ALARM\_MASK
  - Register definitions, [105](#)
- BF\_IN9\_FREQ\_SOFT\_ALARM\_SIZE
  - Register definitions, [105](#)
- BF\_IN9\_LSB
  - Register definitions, [100](#)
- BF\_IN9\_MASK
  - Register definitions, [99](#)
- BF\_IN9\_NO\_ACTIVITY\_ALARM\_LSB
  - Register definitions, [105](#)
- BF\_IN9\_NO\_ACTIVITY\_ALARM\_MASK
  - Register definitions, [105](#)
- BF\_IN9\_NO\_ACTIVITY\_ALARM\_SIZE
  - Register definitions, [105](#)
- BF\_IN9\_PH\_LOCK\_ALARM\_LSB
  - Register definitions, [105](#)
- BF\_IN9\_PH\_LOCK\_ALARM\_MASK
  - Register definitions, [105](#)
- BF\_IN9\_PH\_LOCK\_ALARM\_SIZE
  - Register definitions, [105](#)

- BF\_IN9\_SEL\_PRIORITY\_LSB  
Register definitions, [102](#)
- BF\_IN9\_SEL\_PRIORITY\_MASK  
Register definitions, [102](#)
- BF\_IN9\_SEL\_PRIORITY\_SIZE  
Register definitions, [102](#)
- BF\_IN9\_SIZE  
Register definitions, [99](#)
- BF\_IN\_2K\_4K\_8K\_INV\_LSB  
Register definitions, [113](#)
- BF\_IN\_2K\_4K\_8K\_INV\_MASK  
Register definitions, [112](#)
- BF\_IN\_2K\_4K\_8K\_INV\_SIZE  
Register definitions, [112](#)
- BF\_IN\_FREQ\_LSB  
Register definitions, [100](#)
- BF\_IN\_FREQ\_MASK  
Register definitions, [100](#)
- BF\_IN\_FREQ\_READ\_CH\_LSB  
Register definitions, [103](#)
- BF\_IN\_FREQ\_READ\_CH\_MASK  
Register definitions, [103](#)
- BF\_IN\_FREQ\_READ\_CH\_SIZE  
Register definitions, [103](#)
- BF\_IN\_FREQ\_SIZE  
Register definitions, [100](#)
- BF\_IN\_FREQ\_VALUE\_LSB  
Register definitions, [103](#)
- BF\_IN\_FREQ\_VALUE\_MASK  
Register definitions, [103](#)
- BF\_IN\_FREQ\_VALUE\_SIZE  
Register definitions, [103](#)
- BF\_IN\_NOISE\_WINDOW\_LSB  
Register definitions, [113](#)
- BF\_IN\_NOISE\_WINDOW\_MASK  
Register definitions, [113](#)
- BF\_IN\_NOISE\_WINDOW\_SIZE  
Register definitions, [113](#)
- BF\_IN\_SONET\_SDH\_LSB  
Register definitions, [97](#)
- BF\_IN\_SONET\_SDH\_MASK  
Register definitions, [97](#)
- BF\_IN\_SONET\_SDH\_SIZE  
Register definitions, [97](#)
- BF\_INPUT\_TO\_T4\_LSB  
Register definitions, [100](#)
- BF\_INPUT\_TO\_T4\_MASK  
Register definitions, [100](#)
- BF\_INPUT\_TO\_T4\_SIZE  
Register definitions, [100](#)
- BF\_INT\_POL\_LSB  
Register definitions, [98](#)
- BF\_INT\_POL\_MASK  
Register definitions, [98](#)
- BF\_INT\_POL\_SIZE  
Register definitions, [98](#)
- BF\_LOCK\_8K\_LSB  
Register definitions, [101](#)
- BF\_LOCK\_8K\_MASK  
Register definitions, [101](#)
- BF\_LOCK\_8K\_SIZE  
Register definitions, [101](#)
- BF\_LOS\_FLAG\_TO\_TDO\_LSB  
Register definitions, [98](#)
- BF\_LOS\_FLAG\_TO\_TDO\_MASK  
Register definitions, [98](#)
- BF\_LOS\_FLAG\_TO\_TDO\_SIZE  
Register definitions, [98](#)
- BF\_LOWER\_THRESHOLD\_DATA\_LSB  
Register definitions, [102](#)
- BF\_LOWER\_THRESHOLD\_DATA\_MASK  
Register definitions, [102](#)
- BF\_LOWER\_THRESHOLD\_DATA\_SIZE  
Register definitions, [102](#)
- BF\_LVDS\_QA\_LSB  
Register definitions, [116](#)
- BF\_LVDS\_QA\_MASK  
Register definitions, [116](#)
- BF\_LVDS\_QA\_SIZE  
Register definitions, [116](#)
- BF\_LVDS\_QB\_LSB  
Register definitions, [117](#)
- BF\_LVDS\_QB\_MASK  
Register definitions, [117](#)
- BF\_LVDS\_QB\_SIZE  
Register definitions, [117](#)
- BF\_M0\_LSB\_LSB  
Register definitions, [116](#)
- BF\_M0\_LSB\_MASK  
Register definitions, [116](#)
- BF\_M0\_LSB\_SIZE  
Register definitions, [116](#)
- BF\_M0\_MSB\_LSB  
Register definitions, [116](#)
- BF\_M0\_MSB\_MASK  
Register definitions, [116](#)
- BF\_M0\_MSB\_SIZE  
Register definitions, [116](#)
- BF\_M1\_LSB\_LSB  
Register definitions, [116](#)
- BF\_M1\_LSB\_MASK  
Register definitions, [116](#)
- BF\_M1\_LSB\_SIZE  
Register definitions, [116](#)
- BF\_M1\_MSB\_LSB  
Register definitions, [116](#)
- BF\_M1\_MSB\_MASK  
Register definitions, [116](#)
- BF\_M1\_MSB\_SIZE  
Register definitions, [116](#)
- BF\_MAN\_HOLD OVER\_LSB  
Register definitions, [109](#)
- BF\_MAN\_HOLD OVER\_MASK  
Register definitions, [109](#)
- BF\_MAN\_HOLD OVER\_SIZE  
Register definitions, [109](#)

- BF\_MASTER\_SLAVE\_LSB  
Register definitions, [97](#)
- BF\_MASTER\_SLAVE\_MASK  
Register definitions, [97](#)
- BF\_MASTER\_SLAVE\_SIZE  
Register definitions, [97](#)
- BF\_MPU\_PIN\_STS\_LSB  
Register definitions, [97](#)
- BF\_MPU\_PIN\_STS\_MASK  
Register definitions, [97](#)
- BF\_MPU\_PIN\_STS\_SIZE  
Register definitions, [96](#)
- BF\_MPU\_SEL\_CNFG\_LSB  
Register definitions, [114](#)
- BF\_MPU\_SEL\_CNFG\_MASK  
Register definitions, [114](#)
- BF\_MPU\_SEL\_CNFG\_SIZE  
Register definitions, [114](#)
- BF\_MR\_SYNC\_LSB  
Register definitions, [116](#)
- BF\_MR\_SYNC\_MASK  
Register definitions, [116](#)
- BF\_MR\_SYNC\_SIZE  
Register definitions, [116](#)
- BF\_MS\_SL\_CTRL\_LSB  
Register definitions, [100](#)
- BF\_MS\_SL\_CTRL\_MASK  
Register definitions, [100](#)
- BF\_MS\_SL\_CTRL\_SIZE  
Register definitions, [100](#)
- BF\_MULTI\_FACTOR\_LSB  
Register definitions, [97](#)
- BF\_MULTI\_FACTOR\_MASK  
Register definitions, [97](#)
- BF\_MULTI\_FACTOR\_SIZE  
Register definitions, [97](#)
- BF\_MULTI\_PH\_8K\_4K\_2K\_EN\_LSB  
Register definitions, [109](#)
- BF\_MULTI\_PH\_8K\_4K\_2K\_EN\_MASK  
Register definitions, [109](#)
- BF\_MULTI\_PH\_8K\_4K\_2K\_EN\_SIZE  
Register definitions, [109](#)
- BF\_MULTI\_PH\_APP\_LSB  
Register definitions, [109](#)
- BF\_MULTI\_PH\_APP\_MASK  
Register definitions, [109](#)
- BF\_MULTI\_PH\_APP\_SIZE  
Register definitions, [109](#)
- BF\_NOMINAL\_FREQ\_VALUE\_A\_LSB  
Register definitions, [97](#)
- BF\_NOMINAL\_FREQ\_VALUE\_A\_MASK  
Register definitions, [97](#)
- BF\_NOMINAL\_FREQ\_VALUE\_A\_SIZE  
Register definitions, [97](#)
- BF\_NOMINAL\_FREQ\_VALUE\_B\_LSB  
Register definitions, [97](#)
- BF\_NOMINAL\_FREQ\_VALUE\_B\_MASK  
Register definitions, [97](#)
- BF\_NOMINAL\_FREQ\_VALUE\_B\_SIZE  
Register definitions, [97](#)
- BF\_NOMINAL\_FREQ\_VALUE\_C\_LSB  
Register definitions, [97](#)
- BF\_NOMINAL\_FREQ\_VALUE\_C\_MASK  
Register definitions, [97](#)
- BF\_NOMINAL\_FREQ\_VALUE\_C\_SIZE  
Register definitions, [97](#)
- BF\_ODBSELA0\_LSB  
Register definitions, [116](#)
- BF\_ODBSELA0\_MASK  
Register definitions, [116](#)
- BF\_ODBSELA0\_SIZE  
Register definitions, [116](#)
- BF\_ODBSELA1\_LSB  
Register definitions, [116](#)
- BF\_ODBSELA1\_MASK  
Register definitions, [116](#)
- BF\_ODBSELA1\_SIZE  
Register definitions, [116](#)
- BF\_ODBSELB0\_LSB  
Register definitions, [116](#)
- BF\_ODBSELB0\_MASK  
Register definitions, [116](#)
- BF\_ODBSELB0\_SIZE  
Register definitions, [116](#)
- BF\_ODBSELB1\_LSB  
Register definitions, [116](#)
- BF\_ODBSELB1\_MASK  
Register definitions, [116](#)
- BF\_ODBSELB1\_SIZE  
Register definitions, [116](#)
- BF\_OE\_A\_LSB  
Register definitions, [117](#)
- BF\_OE\_A\_MASK  
Register definitions, [117](#)
- BF\_OE\_A\_SIZE  
Register definitions, [116](#)
- BF\_OE\_B\_LSB  
Register definitions, [117](#)
- BF\_OE\_B\_MASK  
Register definitions, [117](#)
- BF\_OE\_B\_SIZE  
Register definitions, [117](#)
- BF\_OSC\_EDGE\_LSB  
Register definitions, [97](#)
- BF\_OSC\_EDGE\_MASK  
Register definitions, [97](#)
- BF\_OSC\_EDGE\_SIZE  
Register definitions, [97](#)
- BF\_OUT1\_DIVIDER\_LSB  
Register definitions, [111](#)
- BF\_OUT1\_DIVIDER\_MASK  
Register definitions, [111](#)
- BF\_OUT1\_DIVIDER\_SIZE  
Register definitions, [111](#)
- BF\_OUT1\_INV\_LSB  
Register definitions, [112](#)

- BF\_OUT1\_INV\_MASK
  - Register definitions, [112](#)
- BF\_OUT1\_INV\_SIZE
  - Register definitions, [112](#)
- BF\_OUT1\_PATH\_SEL\_LSB
  - Register definitions, [111](#)
- BF\_OUT1\_PATH\_SEL\_MASK
  - Register definitions, [111](#)
- BF\_OUT1\_PATH\_SEL\_SIZE
  - Register definitions, [111](#)
- BF\_OUT2\_DIVIDER\_LSB
  - Register definitions, [111](#)
- BF\_OUT2\_DIVIDER\_MASK
  - Register definitions, [111](#)
- BF\_OUT2\_DIVIDER\_SIZE
  - Register definitions, [111](#)
- BF\_OUT2\_INV\_LSB
  - Register definitions, [112](#)
- BF\_OUT2\_INV\_MASK
  - Register definitions, [112](#)
- BF\_OUT2\_INV\_SIZE
  - Register definitions, [112](#)
- BF\_OUT2\_PATH\_SEL\_LSB
  - Register definitions, [111](#)
- BF\_OUT2\_PATH\_SEL\_MASK
  - Register definitions, [111](#)
- BF\_OUT2\_PATH\_SEL\_SIZE
  - Register definitions, [111](#)
- BF\_OUT3\_DIVIDER\_LSB
  - Register definitions, [111](#)
- BF\_OUT3\_DIVIDER\_MASK
  - Register definitions, [111](#)
- BF\_OUT3\_DIVIDER\_SIZE
  - Register definitions, [111](#)
- BF\_OUT3\_INV\_LSB
  - Register definitions, [112](#)
- BF\_OUT3\_INV\_MASK
  - Register definitions, [112](#)
- BF\_OUT3\_INV\_SIZE
  - Register definitions, [112](#)
- BF\_OUT3\_PATH\_SEL\_LSB
  - Register definitions, [111](#)
- BF\_OUT3\_PATH\_SEL\_MASK
  - Register definitions, [111](#)
- BF\_OUT3\_PATH\_SEL\_SIZE
  - Register definitions, [111](#)
- BF\_OUT4\_DIVIDER\_LSB
  - Register definitions, [111](#)
- BF\_OUT4\_DIVIDER\_MASK
  - Register definitions, [111](#)
- BF\_OUT4\_DIVIDER\_SIZE
  - Register definitions, [111](#)
- BF\_OUT4\_INV\_LSB
  - Register definitions, [112](#)
- BF\_OUT4\_INV\_MASK
  - Register definitions, [112](#)
- BF\_OUT4\_INV\_SIZE
  - Register definitions, [112](#)
- BF\_OUT4\_PATH\_SEL\_LSB
  - Register definitions, [111](#)
- BF\_OUT4\_PATH\_SEL\_MASK
  - Register definitions, [111](#)
- BF\_OUT4\_PATH\_SEL\_SIZE
  - Register definitions, [111](#)
- BF\_OUT5\_DIVIDER\_LSB
  - Register definitions, [111](#)
- BF\_OUT5\_DIVIDER\_MASK
  - Register definitions, [111](#)
- BF\_OUT5\_DIVIDER\_SIZE
  - Register definitions, [111](#)
- BF\_OUT5\_INV\_LSB
  - Register definitions, [112](#)
- BF\_OUT5\_INV\_MASK
  - Register definitions, [112](#)
- BF\_OUT5\_INV\_SIZE
  - Register definitions, [112](#)
- BF\_OUT5\_PATH\_SEL\_LSB
  - Register definitions, [111](#)
- BF\_OUT5\_PATH\_SEL\_MASK
  - Register definitions, [111](#)
- BF\_OUT5\_PATH\_SEL\_SIZE
  - Register definitions, [111](#)
- BF\_OUT6\_DIVIDER\_LSB
  - Register definitions, [111](#)
- BF\_OUT6\_DIVIDER\_MASK
  - Register definitions, [111](#)
- BF\_OUT6\_DIVIDER\_SIZE
  - Register definitions, [111](#)
- BF\_OUT6\_INV\_LSB
  - Register definitions, [112](#)
- BF\_OUT6\_INV\_MASK
  - Register definitions, [112](#)
- BF\_OUT6\_INV\_SIZE
  - Register definitions, [112](#)
- BF\_OUT6\_PATH\_SEL\_LSB
  - Register definitions, [111](#)
- BF\_OUT6\_PATH\_SEL\_MASK
  - Register definitions, [111](#)
- BF\_OUT6\_PATH\_SEL\_SIZE
  - Register definitions, [111](#)
- BF\_OUT6\_PECL\_LVDS\_LSB
  - Register definitions, [98](#)
- BF\_OUT6\_PECL\_LVDS\_MASK
  - Register definitions, [98](#)
- BF\_OUT6\_PECL\_LVDS\_SIZE
  - Register definitions, [98](#)
- BF\_OUT7\_DIVIDER\_LSB
  - Register definitions, [111](#)
- BF\_OUT7\_DIVIDER\_MASK
  - Register definitions, [111](#)
- BF\_OUT7\_DIVIDER\_SIZE
  - Register definitions, [111](#)
- BF\_OUT7\_INV\_LSB
  - Register definitions, [112](#)
- BF\_OUT7\_INV\_MASK
  - Register definitions, [112](#)

- BF\_OUT7\_INV\_SIZE
  - Register definitions, [112](#)
- BF\_OUT7\_PATH\_SEL\_LSB
  - Register definitions, [111](#)
- BF\_OUT7\_PATH\_SEL\_MASK
  - Register definitions, [111](#)
- BF\_OUT7\_PATH\_SEL\_SIZE
  - Register definitions, [111](#)
- BF\_OUT7\_PECCLVDS\_LSB
  - Register definitions, [98](#)
- BF\_OUT7\_PECCLVDS\_MASK
  - Register definitions, [98](#)
- BF\_OUT7\_PECCLVDS\_SIZE
  - Register definitions, [97](#)
- BF\_OUT8\_EN\_LSB
  - Register definitions, [112](#)
- BF\_OUT8\_EN\_MASK
  - Register definitions, [112](#)
- BF\_OUT8\_EN\_SIZE
  - Register definitions, [112](#)
- BF\_OUT8\_PATH\_SEL\_LSB
  - Register definitions, [112](#)
- BF\_OUT8\_PATH\_SEL\_MASK
  - Register definitions, [111](#)
- BF\_OUT8\_PATH\_SEL\_SIZE
  - Register definitions, [111](#)
- BF\_OUT9\_EN\_LSB
  - Register definitions, [112](#)
- BF\_OUT9\_EN\_MASK
  - Register definitions, [112](#)
- BF\_OUT9\_EN\_SIZE
  - Register definitions, [112](#)
- BF\_OUT9\_INV\_LSB
  - Register definitions, [112](#)
- BF\_OUT9\_INV\_MASK
  - Register definitions, [112](#)
- BF\_OUT9\_INV\_SIZE
  - Register definitions, [112](#)
- BF\_OUT9\_PATH\_SEL\_LSB
  - Register definitions, [112](#)
- BF\_OUT9\_PATH\_SEL\_MASK
  - Register definitions, [112](#)
- BF\_OUT9\_PATH\_SEL\_SIZE
  - Register definitions, [112](#)
- BF\_PAGE\_POINTER\_LSB
  - Register definitions, [102](#)
- BF\_PAGE\_POINTER\_MASK
  - Register definitions, [102](#)
- BF\_PAGE\_POINTER\_SIZE
  - Register definitions, [102](#)
- BF\_PDSEL0\_LSB\_LSB
  - Register definitions, [115](#)
- BF\_PDSEL0\_LSB\_MASK
  - Register definitions, [115](#)
- BF\_PDSEL0\_LSB\_SIZE
  - Register definitions, [115](#)
- BF\_PDSEL0\_MSB\_LSB
  - Register definitions, [115](#)
- BF\_PDSEL0\_MSB\_MASK
  - Register definitions, [115](#)
- BF\_PDSEL0\_MSB\_SIZE
  - Register definitions, [115](#)
- BF\_PDSEL1\_LSB\_LSB
  - Register definitions, [115](#)
- BF\_PDSEL1\_LSB\_MASK
  - Register definitions, [115](#)
- BF\_PDSEL1\_LSB\_SIZE
  - Register definitions, [115](#)
- BF\_PDSEL1\_MSB\_LSB
  - Register definitions, [116](#)
- BF\_PDSEL1\_MSB\_MASK
  - Register definitions, [116](#)
- BF\_PDSEL1\_MSB\_SIZE
  - Register definitions, [115](#)
- BF\_PH\_ALARM\_TIMEOUT\_LSB
  - Register definitions, [97](#)
- BF\_PH\_ALARM\_TIMEOUT\_MASK
  - Register definitions, [97](#)
- BF\_PH\_ALARM\_TIMEOUT\_SIZE
  - Register definitions, [97](#)
- BF\_PH\_LOS\_COARSE\_LIMIT\_LSB
  - Register definitions, [109](#)
- BF\_PH\_LOS\_COARSE\_LIMIT\_MASK
  - Register definitions, [109](#)
- BF\_PH\_LOS\_COARSE\_LIMIT\_SIZE
  - Register definitions, [109](#)
- BF\_PH\_LOS\_FINE\_LIMIT\_LSB
  - Register definitions, [109](#)
- BF\_PH\_LOS\_FINE\_LIMIT\_MASK
  - Register definitions, [109](#)
- BF\_PH\_LOS\_FINE\_LIMIT\_SIZE
  - Register definitions, [109](#)
- BF\_PH\_OFFSET\_A\_LSB
  - Register definitions, [113](#)
- BF\_PH\_OFFSET\_A\_MASK
  - Register definitions, [113](#)
- BF\_PH\_OFFSET\_A\_SIZE
  - Register definitions, [113](#)
- BF\_PH\_OFFSET\_B\_LSB
  - Register definitions, [113](#)
- BF\_PH\_OFFSET\_B\_MASK
  - Register definitions, [113](#)
- BF\_PH\_OFFSET\_B\_SIZE
  - Register definitions, [113](#)
- BF\_PH\_OFFSET\_EN\_LSB
  - Register definitions, [113](#)
- BF\_PH\_OFFSET\_EN\_MASK
  - Register definitions, [113](#)
- BF\_PH\_OFFSET\_EN\_SIZE
  - Register definitions, [113](#)
- BF\_PPS\_FAST\_FREQ\_LOCK\_EN\_LSB
  - Register definitions, [100](#)
- BF\_PPS\_FAST\_FREQ\_LOCK\_EN\_MASK
  - Register definitions, [100](#)
- BF\_PPS\_FAST\_FREQ\_LOCK\_EN\_SIZE
  - Register definitions, [100](#)

- BF\_PPS\_FAST\_PH\_LOCK\_EN\_LSB
  - Register definitions, [100](#)
- BF\_PPS\_FAST\_PH\_LOCK\_EN\_MASK
  - Register definitions, [100](#)
- BF\_PPS\_FAST\_PH\_LOCK\_EN\_SIZE
  - Register definitions, [100](#)
- BF\_PPS\_PHASE\_LSB
  - Register definitions, [114](#)
- BF\_PPS\_PHASE\_MASK
  - Register definitions, [114](#)
- BF\_PPS\_PHASE\_SIZE
  - Register definitions, [114](#)
- BF\_PPS\_PULSE\_LSB
  - Register definitions, [114](#)
- BF\_PPS\_PULSE\_MASK
  - Register definitions, [114](#)
- BF\_PPS\_PULSE\_SIZE
  - Register definitions, [114](#)
- BF\_PRE\_DIV\_CH\_VALUE\_LSB
  - Register definitions, [101](#)
- BF\_PRE\_DIV\_CH\_VALUE\_MASK
  - Register definitions, [101](#)
- BF\_PRE\_DIV\_CH\_VALUE\_SIZE
  - Register definitions, [101](#)
- BF\_PRE\_DIVN\_VALUE\_A\_LSB
  - Register definitions, [101](#)
- BF\_PRE\_DIVN\_VALUE\_A\_MASK
  - Register definitions, [101](#)
- BF\_PRE\_DIVN\_VALUE\_A\_SIZE
  - Register definitions, [101](#)
- BF\_PRE\_DIVN\_VALUE\_B\_LSB
  - Register definitions, [101](#)
- BF\_PRE\_DIVN\_VALUE\_B\_MASK
  - Register definitions, [101](#)
- BF\_PRE\_DIVN\_VALUE\_B\_SIZE
  - Register definitions, [101](#)
- BF\_PROTECTION\_DATA\_LSB
  - Register definitions, [114](#)
- BF\_PROTECTION\_DATA\_MASK
  - Register definitions, [114](#)
- BF\_PROTECTION\_DATA\_SIZE
  - Register definitions, [114](#)
- BF\_READ\_AVG\_LSB
  - Register definitions, [109](#)
- BF\_READ\_AVG\_MASK
  - Register definitions, [109](#)
- BF\_READ\_AVG\_SIZE
  - Register definitions, [109](#)
- BF\_REVERTIVE\_MODE\_LSB
  - Register definitions, [97](#)
- BF\_REVERTIVE\_MODE\_MASK
  - Register definitions, [97](#)
- BF\_REVERTIVE\_MODE\_SIZE
  - Register definitions, [97](#)
- BF\_SECOND\_HIGHEST\_PRIORITY\_VALIDATED\_LSB
  - Register definitions, [107](#)
- BF\_SECOND\_HIGHEST\_PRIORITY\_VALIDATED\_MASK
  - Register definitions, [107](#)
- BF\_SECOND\_HIGHEST\_PRIORITY\_VALIDATED\_SIZE
  - Register definitions, [107](#)
- BF\_SEL\_DOUBLE\_LSB
  - Register definitions, [116](#)
- BF\_SEL\_DOUBLE\_MASK
  - Register definitions, [116](#)
- BF\_SEL\_DOUBLE\_SIZE
  - Register definitions, [116](#)
- BF\_SHUTOFF\_LSB
  - Register definitions, [116](#)
- BF\_SHUTOFF\_MASK
  - Register definitions, [116](#)
- BF\_SHUTOFF\_SIZE
  - Register definitions, [116](#)
- BF\_SOFT\_FREQ\_MON\_THRESHOLD\_A\_LSB
  - Register definitions, [102](#)
- BF\_SOFT\_FREQ\_MON\_THRESHOLD\_A\_MASK
  - Register definitions, [102](#)
- BF\_SOFT\_FREQ\_MON\_THRESHOLD\_A\_SIZE
  - Register definitions, [102](#)
- BF\_SOFT\_FREQ\_MON\_THRESHOLD\_B\_LSB
  - Register definitions, [102](#)
- BF\_SOFT\_FREQ\_MON\_THRESHOLD\_B\_MASK
  - Register definitions, [102](#)
- BF\_SOFT\_FREQ\_MON\_THRESHOLD\_B\_SIZE
  - Register definitions, [102](#)
- BF\_SYNC\_BYPASS\_LSB
  - Register definitions, [113](#)
- BF\_SYNC\_BYPASS\_MASK
  - Register definitions, [113](#)
- BF\_SYNC\_BYPASS\_SIZE
  - Register definitions, [113](#)
- BF\_SYNC\_EDGE\_LSB
  - Register definitions, [113](#)
- BF\_SYNC\_EDGE\_MASK
  - Register definitions, [113](#)
- BF\_SYNC\_EDGE\_SIZE
  - Register definitions, [113](#)
- BF\_SYNC\_FREQ\_LSB
  - Register definitions, [97](#)
- BF\_SYNC\_FREQ\_MASK
  - Register definitions, [97](#)
- BF\_SYNC\_FREQ\_SIZE
  - Register definitions, [97](#)
- BF\_SYNC\_MON\_LIMT\_LSB
  - Register definitions, [113](#)
- BF\_SYNC\_MON\_LIMT\_MASK
  - Register definitions, [113](#)
- BF\_SYNC\_MON\_LIMT\_SIZE
  - Register definitions, [113](#)
- BF\_SYNC\_PH1\_LSB
  - Register definitions, [114](#)
- BF\_SYNC\_PH1\_MASK
  - Register definitions, [114](#)

- BF\_SYNC\_PH1\_SIZE
  - Register definitions, [114](#)
- BF\_SYNC\_PH2\_LSB
  - Register definitions, [114](#)
- BF\_SYNC\_PH2\_MASK
  - Register definitions, [113](#)
- BF\_SYNC\_PH2\_SIZE
  - Register definitions, [113](#)
- BF\_T0\_12E1\_GPS\_E3\_T3\_SEL\_LSB
  - Register definitions, [108](#)
- BF\_T0\_12E1\_GPS\_E3\_T3\_SEL\_MASK
  - Register definitions, [108](#)
- BF\_T0\_12E1\_GPS\_E3\_T3\_SEL\_SIZE
  - Register definitions, [108](#)
- BF\_T0\_APLL\_PATH\_LSB
  - Register definitions, [108](#)
- BF\_T0\_APLL\_PATH\_MASK
  - Register definitions, [108](#)
- BF\_T0\_APLL\_PATH\_SIZE
  - Register definitions, [108](#)
- BF\_T0\_DFS\_OFF\_0\_LSB
  - Register definitions, [114](#)
- BF\_T0\_DFS\_OFF\_0\_MASK
  - Register definitions, [114](#)
- BF\_T0\_DFS\_OFF\_0\_SIZE
  - Register definitions, [114](#)
- BF\_T0\_DFS\_OFF\_1\_LSB
  - Register definitions, [114](#)
- BF\_T0\_DFS\_OFF\_1\_MASK
  - Register definitions, [114](#)
- BF\_T0\_DFS\_OFF\_1\_SIZE
  - Register definitions, [114](#)
- BF\_T0\_DFS\_OFF\_2\_LSB
  - Register definitions, [114](#)
- BF\_T0\_DFS\_OFF\_2\_MASK
  - Register definitions, [114](#)
- BF\_T0\_DFS\_OFF\_2\_SIZE
  - Register definitions, [114](#)
- BF\_T0\_DFS\_OFF\_3\_LSB
  - Register definitions, [114](#)
- BF\_T0\_DFS\_OFF\_3\_MASK
  - Register definitions, [114](#)
- BF\_T0\_DFS\_OFF\_3\_SIZE
  - Register definitions, [114](#)
- BF\_T0\_DPLL\_ACQ\_BW\_LSB
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_ACQ\_BW\_MASK
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_ACQ\_BW\_SIZE
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_ACQ\_DAMPING\_LSB
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_ACQ\_DAMPING\_MASK
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_ACQ\_DAMPING\_SIZE
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_LOCK\_LSB
  - Register definitions, [107](#)
- BF\_T0\_DPLL\_LOCK\_MASK
  - Register definitions, [107](#)
- BF\_T0\_DPLL\_LOCK\_SIZE
  - Register definitions, [107](#)
- BF\_T0\_DPLL\_LOCKED\_BW\_LSB
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_LOCKED\_BW\_MASK
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_LOCKED\_BW\_SIZE
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_LOCKED\_DAMPING\_LSB
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_LOCKED\_DAMPING\_MASK
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_LOCKED\_DAMPING\_SIZE
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_OPERATING\_MODE\_LSB
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_OPERATING\_MODE\_MASK
  - Register definitions, [107](#)
- BF\_T0\_DPLL\_OPERATING\_MODE\_SIZE
  - Register definitions, [107](#)
- BF\_T0\_DPLL\_SOFT\_FREQ\_ALARM\_LSB
  - Register definitions, [107](#)
- BF\_T0\_DPLL\_SOFT\_FREQ\_ALARM\_MASK
  - Register definitions, [107](#)
- BF\_T0\_DPLL\_SOFT\_FREQ\_ALARM\_SIZE
  - Register definitions, [107](#)
- BF\_T0\_DPLL\_START\_BW\_LSB
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_START\_BW\_MASK
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_START\_BW\_SIZE
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_START\_DAMPING\_LSB
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_START\_DAMPING\_MASK
  - Register definitions, [108](#)
- BF\_T0\_DPLL\_START\_DAMPING\_SIZE
  - Register definitions, [108](#)
- BF\_T0\_FOR\_T4\_LSB
  - Register definitions, [107](#)
- BF\_T0\_FOR\_T4\_MASK
  - Register definitions, [107](#)
- BF\_T0\_FOR\_T4\_SIZE
  - Register definitions, [107](#)
- BF\_T0\_GSM\_OBSAI\_16E1\_16T1\_SEL\_LSB
  - Register definitions, [108](#)
- BF\_T0\_GSM\_OBSAI\_16E1\_16T1\_SEL\_MASK
  - Register definitions, [108](#)
- BF\_T0\_GSM\_OBSAI\_16E1\_16T1\_SEL\_SIZE
  - Register definitions, [108](#)
- BF\_T0\_INPUT\_SEL\_LSB
  - Register definitions, [107](#)
- BF\_T0\_INPUT\_SEL\_MASK
  - Register definitions, [107](#)
- BF\_T0\_INPUT\_SEL\_SIZE
  - Register definitions, [107](#)

- BF\_T0\_LIMIT\_LSB
  - Register definitions, [109](#)
- BF\_T0\_LIMIT\_MASK
  - Register definitions, [108](#)
- BF\_T0\_LIMIT\_SIZE
  - Register definitions, [108](#)
- BF\_T0\_MAIN\_REF\_FAIL\_LSB
  - Register definitions, [99](#)
- BF\_T0\_MAIN\_REF\_FAIL\_MASK
  - Register definitions, [99](#)
- BF\_T0\_MAIN\_REF\_FAIL\_SIZE
  - Register definitions, [99](#)
- BF\_T0\_OPERATING\_MODE\_LSB
  - Register definitions, [108](#)
- BF\_T0\_OPERATING\_MODE\_MASK
  - Register definitions, [108](#)
- BF\_T0\_OPERATING\_MODE\_SIZE
  - Register definitions, [108](#)
- BF\_T0\_OPERATING\_MODE\_STS\_LSB
  - Register definitions, [99](#)
- BF\_T0\_OPERATING\_MODE\_STS\_MASK
  - Register definitions, [99](#)
- BF\_T0\_OPERATING\_MODE\_STS\_SIZE
  - Register definitions, [99](#)
- BF\_T0\_PH\_SLOPE\_LSB
  - Register definitions, [114](#)
- BF\_T0\_PH\_SLOPE\_MASK
  - Register definitions, [114](#)
- BF\_T0\_PH\_SLOPE\_SIZE
  - Register definitions, [114](#)
- BF\_T0\_WDCO\_LSB
  - Register definitions, [108](#)
- BF\_T0\_WDCO\_MASK
  - Register definitions, [108](#)
- BF\_T0\_WDCO\_SIZE
  - Register definitions, [108](#)
- BF\_T4\_12E1\_GPS\_E3\_T3\_SEL\_LSB
  - Register definitions, [110](#)
- BF\_T4\_12E1\_GPS\_E3\_T3\_SEL\_MASK
  - Register definitions, [110](#)
- BF\_T4\_12E1\_GPS\_E3\_T3\_SEL\_SIZE
  - Register definitions, [110](#)
- BF\_T4\_APLL\_PATH\_LSB
  - Register definitions, [110](#)
- BF\_T4\_APLL\_PATH\_MASK
  - Register definitions, [110](#)
- BF\_T4\_APLL\_PATH\_SIZE
  - Register definitions, [110](#)
- BF\_T4\_DFS\_OFF\_0\_LSB
  - Register definitions, [114](#)
- BF\_T4\_DFS\_OFF\_0\_MASK
  - Register definitions, [114](#)
- BF\_T4\_DFS\_OFF\_0\_SIZE
  - Register definitions, [114](#)
- BF\_T4\_DFS\_OFF\_1\_LSB
  - Register definitions, [114](#)
- BF\_T4\_DFS\_OFF\_1\_MASK
  - Register definitions, [114](#)
- BF\_T4\_DFS\_OFF\_1\_SIZE
  - Register definitions, [114](#)
- BF\_T4\_DFS\_OFF\_2\_LSB
  - Register definitions, [114](#)
- BF\_T4\_DFS\_OFF\_2\_MASK
  - Register definitions, [114](#)
- BF\_T4\_DFS\_OFF\_2\_SIZE
  - Register definitions, [114](#)
- BF\_T4\_DFS\_OFF\_3\_LSB
  - Register definitions, [114](#)
- BF\_T4\_DFS\_OFF\_3\_MASK
  - Register definitions, [114](#)
- BF\_T4\_DFS\_OFF\_3\_SIZE
  - Register definitions, [114](#)
- BF\_T4\_DPLL\_LOCK\_LSB
  - Register definitions, [107](#)
- BF\_T4\_DPLL\_LOCK\_MASK
  - Register definitions, [107](#)
- BF\_T4\_DPLL\_LOCK\_SIZE
  - Register definitions, [107](#)
- BF\_T4\_DPLL\_LOCKED\_BW\_LSB
  - Register definitions, [110](#)
- BF\_T4\_DPLL\_LOCKED\_BW\_MASK
  - Register definitions, [110](#)
- BF\_T4\_DPLL\_LOCKED\_BW\_SIZE
  - Register definitions, [110](#)
- BF\_T4\_DPLL\_LOCKED\_DAMPING\_LSB
  - Register definitions, [110](#)
- BF\_T4\_DPLL\_LOCKED\_DAMPING\_MASK
  - Register definitions, [110](#)
- BF\_T4\_DPLL\_LOCKED\_DAMPING\_SIZE
  - Register definitions, [110](#)
- BF\_T4\_DPLL\_SOFT\_FREQ\_ALARM\_LSB
  - Register definitions, [107](#)
- BF\_T4\_DPLL\_SOFT\_FREQ\_ALARM\_MASK
  - Register definitions, [107](#)
- BF\_T4\_DPLL\_SOFT\_FREQ\_ALARM\_SIZE
  - Register definitions, [107](#)
- BF\_T4\_GSM\_OBSAI\_16E1\_16T1\_SEL\_LSB
  - Register definitions, [110](#)
- BF\_T4\_GSM\_OBSAI\_16E1\_16T1\_SEL\_MASK
  - Register definitions, [110](#)
- BF\_T4\_GSM\_OBSAI\_16E1\_16T1\_SEL\_SIZE
  - Register definitions, [110](#)
- BF\_T4\_INPUT\_FAIL\_LSB
  - Register definitions, [112](#)
- BF\_T4\_INPUT\_FAIL\_MASK
  - Register definitions, [112](#)
- BF\_T4\_INPUT\_FAIL\_SIZE
  - Register definitions, [112](#)
- BF\_T4\_INPUT\_SEL\_LSB
  - Register definitions, [107](#)
- BF\_T4\_INPUT\_SEL\_MASK
  - Register definitions, [107](#)
- BF\_T4\_INPUT\_SEL\_SIZE
  - Register definitions, [107](#)
- BF\_T4\_LOCK\_T0\_LSB
  - Register definitions, [107](#)

- BF\_T4\_LOCK\_T0\_MASK
  - Register definitions, [107](#)
- BF\_T4\_LOCK\_T0\_SIZE
  - Register definitions, [107](#)
- BF\_T4\_OPERATING\_MODE\_LSB
  - Register definitions, [108](#)
- BF\_T4\_OPERATING\_MODE\_MASK
  - Register definitions, [108](#)
- BF\_T4\_OPERATING\_MODE\_SIZE
  - Register definitions, [108](#)
- BF\_T4\_PH\_SLOPE\_LSB
  - Register definitions, [115](#)
- BF\_T4\_PH\_SLOPE\_MASK
  - Register definitions, [114](#)
- BF\_T4\_PH\_SLOPE\_SIZE
  - Register definitions, [114](#)
- BF\_T4\_STS\_LSB
  - Register definitions, [100](#)
- BF\_T4\_STS\_MASK
  - Register definitions, [100](#)
- BF\_T4\_STS\_SIZE
  - Register definitions, [100](#)
- BF\_T4\_T0\_SEL\_LSB
  - Register definitions, [97](#)
- BF\_T4\_T0\_SEL\_MASK
  - Register definitions, [97](#)
- BF\_T4\_T0\_SEL\_SIZE
  - Register definitions, [97](#)
- BF\_T4\_TEST\_T0\_PH\_LSB
  - Register definitions, [107](#)
- BF\_T4\_TEST\_T0\_PH\_MASK
  - Register definitions, [107](#)
- BF\_T4\_TEST\_T0\_PH\_SIZE
  - Register definitions, [107](#)
- BF\_T4\_WDCO\_LSB
  - Register definitions, [108](#)
- BF\_T4\_WDCO\_MASK
  - Register definitions, [108](#)
- BF\_T4\_WDCO\_SIZE
  - Register definitions, [108](#)
- BF\_TEMP\_HOLDOVER\_MODE\_LSB
  - Register definitions, [109](#)
- BF\_TEMP\_HOLDOVER\_MODE\_MASK
  - Register definitions, [109](#)
- BF\_TEMP\_HOLDOVER\_MODE\_SIZE
  - Register definitions, [109](#)
- BF\_THIRD\_HIGHEST\_PRIORITY\_VALIDATED\_LSB
  - Register definitions, [107](#)
- BF\_THIRD\_HIGHEST\_PRIORITY\_VALIDATED\_MASK
  - Register definitions, [107](#)
- BF\_THIRD\_HIGHEST\_PRIORITY\_VALIDATED\_SIZE
  - Register definitions, [107](#)
- BF\_TIMEOUT\_VALUE\_LSB
  - Register definitions, [97](#)
- BF\_TIMEOUT\_VALUE\_MASK
  - Register definitions, [97](#)
- BF\_TIMEOUT\_VALUE\_SIZE
  - Register definitions, [97](#)
- BF\_ULTR\_FAST\_SW\_LSB
  - Register definitions, [98](#)
- BF\_ULTR\_FAST\_SW\_MASK
  - Register definitions, [98](#)
- BF\_ULTR\_FAST\_SW\_SIZE
  - Register definitions, [98](#)
- BF\_UPPER\_THRESHOLD\_DATA\_LSB
  - Register definitions, [102](#)
- BF\_UPPER\_THRESHOLD\_DATA\_MASK
  - Register definitions, [102](#)
- BF\_UPPER\_THRESHOLD\_DATA\_SIZE
  - Register definitions, [102](#)
- BF\_VC XO\_SEL\_LSB
  - Register definitions, [116](#)
- BF\_VC XO\_SEL\_MASK
  - Register definitions, [116](#)
- BF\_VC XO\_SEL\_SIZE
  - Register definitions, [116](#)
- BF\_WIDE\_EN\_LSB
  - Register definitions, [109](#)
- BF\_WIDE\_EN\_MASK
  - Register definitions, [109](#)
- BF\_WIDE\_EN\_SIZE
  - Register definitions, [109](#)
- bfVal\_m
  - T\_IDTFreq2bfVal, [169](#)
- bitLocation
  - T\_idtI2CtransData, [171](#)
- bitValue
  - T\_idtI2CtransData, [171](#)
- bucketReg\_t, [147](#)
  - bucketSizeReg\_m, [147](#)
  - decayRateReg\_m, [147](#)
  - lowerThreshReg\_m, [147](#)
  - upperThreshReg\_m, [147](#)
- bucketSizeReg\_m
  - bucketReg\_t, [147](#)
- byPass
  - T\_idtApIConfig, [158](#)
- CHECK\_BIT
  - outputFreq\_priv.h, [234](#)
- coarsePhaseLimit\_1
  - DPLL Function Module, [16](#)
- coarsePhaseLimit\_1023
  - DPLL Function Module, [16](#)
- coarsePhaseLimit\_127
  - DPLL Function Module, [16](#)
- coarsePhaseLimit\_15
  - DPLL Function Module, [16](#)
- coarsePhaseLimit\_255
  - DPLL Function Module, [16](#)
- coarsePhaseLimit\_3
  - DPLL Function Module, [16](#)
- coarsePhaseLimit\_31
  - DPLL Function Module, [16](#)
- coarsePhaseLimit\_511
  - DPLL Function Module, [16](#)
- coarsePhaseLimit\_63

- DPLL Function Module, 16
- coarsePhaseLimit\_7
  - DPLL Function Module, 16
- coarsePhaseLimit\_MAX
  - DPLL Function Module, 16
- common/access/sim/ReadWrite\_sim.c, 222
- common/access/sim/ReadWrite\_sim.h, 225
- common/hlapi/include/inputFreq.h, 226
- common/hlapi/include/outputFreq.h, 227
- common/hlapi/include/outputFreq\_priv.h, 232
- common/hlapi/include/profiles.h, 237
- common/hlapi/src/inputFreq.c, 240
- common/hlapi/src/outputFreq.c, 242
- common/hlapi/src/outputFreqString.c, 245
- common/hlapi/src/outputFreqUtil.c, 248
- common/hlapi/src/profiles.c, 250
- common/include/Apll.h, 253
- common/include/Define.h, 255
- common/include/Dpll.h, 261
- common/include/DpllCnfg.h, 262
- common/include/General.h, 268
- common/include/Hal.h, 271
- common/include/Input.h, 272
- common/include/Output.h, 275
- common/include/RegisterDef.h, 277
- common/src/Apll.c, 285
- common/src/DpllCnfg.c, 286
- common/src/General.c, 291
- common/src/Hal.c, 293
- common/src/Input.c, 294
- common/src/Output.c, 297
- configDescription\_m
  - T\_SampleConfigEntry, 174
- configFunction\_m
  - T\_SampleConfigEntry, 174
- currOutPathCfg\_mp
  - drvHdlr\_s, 168
- DPLL Function Module
  - Automatic\_Mode, 18
  - coarsePhaseLimit\_1, 16
  - coarsePhaseLimit\_1023, 16
  - coarsePhaseLimit\_127, 16
  - coarsePhaseLimit\_15, 16
  - coarsePhaseLimit\_255, 16
  - coarsePhaseLimit\_3, 16
  - coarsePhaseLimit\_31, 16
  - coarsePhaseLimit\_511, 16
  - coarsePhaseLimit\_63, 16
  - coarsePhaseLimit\_7, 16
  - coarsePhaseLimit\_MAX, 16
  - DPLLOutputSelA\_16E1, 18
  - DPLLOutputSelA\_16T1, 18
  - DPLLOutputSelA\_GPS, 18
  - DPLLOutputSelA\_GSM, 18
  - DPLLOutputSelA\_MAX, 18
  - DPLLOutputSelA\_OBSAI, 18
  - DPLLOutputSelB\_12E1, 19
  - DPLLOutputSelB\_24T1, 19
  - DPLLOutputSelB\_E3, 19
  - DPLLOutputSelB\_GPS, 19
  - DPLLOutputSelB\_MAX, 19
  - DPLLOutputSelB\_T3, 19
  - DampingFactor\_10, 17
  - DampingFactor\_1p2, 17
  - DampingFactor\_20, 17
  - DampingFactor\_2p5, 17
  - DampingFactor\_5, 17
  - DampingFactor\_MAX, 17
  - Dco\_Mode, 18
  - dpllBandwidth\_0p1Hz, 17
  - dpllBandwidth\_0p3Hz, 17
  - dpllBandwidth\_0p5mHz, 17
  - dpllBandwidth\_0p6Hz, 17
  - dpllBandwidth\_15mHz, 17
  - dpllBandwidth\_18Hz, 17
  - dpllBandwidth\_1mHz, 17
  - dpllBandwidth\_1p2Hz, 17
  - dpllBandwidth\_2mHz, 17
  - dpllBandwidth\_2p5Hz, 17
  - dpllBandwidth\_30mHz, 17
  - dpllBandwidth\_35Hz, 17
  - dpllBandwidth\_4Hz, 17
  - dpllBandwidth\_4mHz, 17
  - dpllBandwidth\_560Hz, 17
  - dpllBandwidth\_60mHz, 17
  - dpllBandwidth\_70Hz, 17
  - dpllBandwidth\_8Hz, 17
  - dpllBandwidth\_8mHz, 17
  - dpllBandwidth\_MAX, 17
  - dpllBandwidthLimit\_MAX, 16
  - dpllBandwidthLimitAcquire, 16
  - dpllBandwidthLimitLocked, 16
  - dpllBandwidthLimitStart, 16
  - DpllOperMode\_MAX, 18
  - finePhaseLimit\_120\_125nanosec, 20
  - finePhaseLimit\_180\_360degree, 20
  - finePhaseLimit\_20\_25nanosec, 20
  - finePhaseLimit\_45\_90degree, 20
  - finePhaseLimit\_60\_65nanosec, 20
  - finePhaseLimit\_90\_180degree, 20
  - finePhaseLimit\_950\_955nanosec, 20
  - finePhaseLimit\_MAX, 20
  - Force\_FreeRun\_Mode, 18
  - Force\_Holdover\_Mode, 18
  - Force\_Locked\_Mode, 18
  - Force\_Lost\_Phase\_Mode, 18
  - Force\_Pre\_Locked2\_Mode, 18
  - Force\_Pre\_Locked\_Mode, 18
  - FreqMonFactor\_0p0032, 20
  - FreqMonFactor\_0p0064, 20
  - FreqMonFactor\_0p0127, 20
  - FreqMonFactor\_0p0257, 20
  - FreqMonFactor\_0p0514, 20
  - FreqMonFactor\_0p1030, 20
  - FreqMonFactor\_0p2060, 20
  - FreqMonFactor\_0p4120, 20

- FreqMonFactor\_0p8230, [20](#)
- FreqMonFactor\_1p6460, [20](#)
- FreqMonFactor\_3p2920, [20](#)
- FreqMonFactor\_3p8100, [20](#)
- FreqMonFactor\_4p6000, [20](#)
- FreqMonFactor\_MAX, [20](#)
- Holdover\_Fast\_Average, [18](#)
- Holdover\_Instantaneous, [18](#)
- Holdover\_MAX, [18](#)
- Holdover\_Manual, [18](#)
- Holdover\_Slow\_Average, [18](#)
- OperDpllMode\_FreeRun, [20](#)
- OperDpllMode\_HoldOver, [20](#)
- OperDpllMode\_Locked, [20](#)
- OperDpllMode\_LostPhase, [20](#)
- OperDpllMode\_PreLocked, [20](#)
- OperDpllMode\_PreLocked2, [20](#)
- phaseSlope\_MAX, [21](#)
- phaseSlope\_gr1244st3, [21](#)
- phaseSlope\_gr1244st3e, [21](#)
- phaseSlope\_gr813, [21](#)
- phaseSlope\_noLimit, [21](#)
- PpsPhasePostion\_0degree, [19](#)
- PpsPhasePostion\_100ns, [19](#)
- PpsPhasePostion\_50ns, [19](#)
- PpsPhasePostion\_MAX, [19](#)
- PpsPulseWidth\_0pt5sec, [19](#)
- PpsPulseWidth\_100us, [19](#)
- PpsPulseWidth\_10ns, [19](#)
- PpsPulseWidth\_1pt2ms, [19](#)
- PpsPulseWidth\_1us, [19](#)
- PpsPulseWidth\_200ns, [19](#)
- PpsPulseWidth\_200us, [19](#)
- PpsPulseWidth\_20us, [19](#)
- PpsPulseWidth\_400ns, [19](#)
- PpsPulseWidth\_50us, [19](#)
- PpsPulseWidth\_800ns, [19](#)
- PpsPulseWidth\_800us, [19](#)
- PpsPulseWidth\_MAX, [19](#)
- SonetGigEthOutput\_10GbE, [21](#)
- SonetGigEthOutput\_10GbE\_6664, [21](#)
- SonetGigEthOutput\_MAX, [21](#)
- SonetGigEthOutput\_Sonet, [21](#)
- Temp\_Holdover\_Fast\_Average, [21](#)
- Temp\_Holdover\_Instantaneous, [21](#)
- Temp\_Holdover\_MAX, [21](#)
- Temp\_Holdover\_Slow\_Average, [21](#)
- Temp\_Same\_As\_Full\_Holdover, [21](#)
- DPLL Private header file
  - SonetGigEthSel\_MAX, [8](#)
  - SonetGigEthSel\_T0Dpll10GbE, [8](#)
  - SonetGigEthSel\_T0Dpll10GbE\_6664, [8](#)
  - SonetGigEthSel\_T0DpllSonet, [8](#)
  - SonetGigEthSel\_T4Dpll10GbE, [8](#)
  - SonetGigEthSel\_T4Dpll10GbE\_6664, [8](#)
  - SonetGigEthSel\_T4DpllSonet, [8](#)
  - T0DpllSelA\_16E1, [8](#)
  - T0DpllSelA\_16T1, [8](#)
  - T0DpllSelA\_GSM, [8](#)
  - T0DpllSelA\_MAX, [8](#)
  - T0DpllSelA\_OBSAI, [8](#)
  - T0DpllSelB\_12E1, [9](#)
  - T0DpllSelB\_E3, [9](#)
  - T0DpllSelB\_GPS, [9](#)
  - T0DpllSelB\_MAX, [9](#)
  - T0DpllSelB\_T3, [9](#)
  - T4DpllSelA\_16E1, [9](#)
  - T4DpllSelA\_16T1, [9](#)
  - T4DpllSelA\_GPS, [9](#)
  - T4DpllSelA\_GSM, [9](#)
  - T4DpllSelA\_MAX, [9](#)
  - T4DpllSelB\_12E1, [9](#)
  - T4DpllSelB\_24T1, [9](#)
  - T4DpllSelB\_E3, [9](#)
  - T4DpllSelB\_MAX, [9](#)
  - T4DpllSelB\_T3, [9](#)
- DPLL\_INSTANCE\_1
  - Define.h, [261](#)
- DPLL\_INSTANCE\_2
  - Define.h, [261](#)
- DPLL\_INSTANCE\_MAX
  - Define.h, [261](#)
- DPLLOutputSelA\_16E1
  - DPLL Function Module, [18](#)
- DPLLOutputSelA\_16T1
  - DPLL Function Module, [18](#)
- DPLLOutputSelA\_GPS
  - DPLL Function Module, [18](#)
- DPLLOutputSelA\_GSM
  - DPLL Function Module, [18](#)
- DPLLOutputSelA\_MAX
  - DPLL Function Module, [18](#)
- DPLLOutputSelA\_OBSAI
  - DPLL Function Module, [18](#)
- DPLLOutputSelB\_12E1
  - DPLL Function Module, [19](#)
- DPLLOutputSelB\_24T1
  - DPLL Function Module, [19](#)
- DPLLOutputSelB\_E3
  - DPLL Function Module, [19](#)
- DPLLOutputSelB\_GPS
  - DPLL Function Module, [19](#)
- DPLLOutputSelB\_MAX
  - DPLL Function Module, [19](#)
- DPLLOutputSelB\_T3
  - DPLL Function Module, [19](#)
- DCO\_MAX\_PPT
  - DPLL Function Module, [15](#)
- DCO\_MIN\_PPT
  - DPLL Function Module, [15](#)
- DCO\_PPT2DCOUNTS
  - DPLL Function Module, [15](#)
- DEBUG\_IDT\_HAL
  - Hal.c, [294](#)
- DPLL Function Module, [10](#)
  - DCO\_MAX\_PPT, [15](#)

- DCO\_MIN\_PPT, 15
- DCO\_PPT2DCOUNTS, 15
- IDTDpll\_Get1stPriorityInput, 21
- IDTDpll\_Get2ndPriorityInput, 22
- IDTDpll\_Get3rdPriorityInput, 22
- IDTDpll\_GetAutoSelBandWidth, 22
- IDTDpll\_GetAutoSelInput, 22
- IDTDpll\_GetBandWidthDamping, 23
- IDTDpll\_GetCurrentDpllFreqOffset, 24
- IDTDpll\_GetCurrentPhaseError, 24
- IDTDpll\_GetCurrentSelectInput, 24
- IDTDpll\_GetDpll16E116T1Enable, 24
- IDTDpll\_GetDpllEthPathEnable, 25
- IDTDpll\_GetDpllFrozen, 25
- IDTDpll\_GetDpllHardAlarmEnable, 25
- IDTDpll\_GetDpllHardAlarmLimit, 25
- IDTDpll\_GetDpllOperMod, 26
- IDTDpll\_GetDpllOutputPathSelectorA, 26
- IDTDpll\_GetDpllOutputPathSelectorB, 26
- IDTDpll\_GetDpllSelAPathEnable, 27
- IDTDpll\_GetDpllSelBPathEnable, 27
- IDTDpll\_GetDpllSoftAlarmLimit, 27
- IDTDpll\_GetFastLossSwitch, 28
- IDTDpll\_GetForceSelInput, 28
- IDTDpll\_GetFrequencyMonitoringFactor, 28
- IDTDpll\_GetHardAlarmThreshold, 29
- IDTDpll\_GetHitlessSwitchingFeature, 29
- IDTDpll\_GetHoldoverFreq, 29
- IDTDpll\_GetHoldoverMode, 30
- IDTDpll\_GetIcpCtrl, 30
- IDTDpll\_GetPhaseCoarseDetector, 30
- IDTDpll\_GetPhaseCoarseLimit, 31
- IDTDpll\_GetPhaseFineDetectorEnable, 31
- IDTDpll\_GetPhaseFineLimit, 32
- IDTDpll\_GetPhaseOffset, 32
- IDTDpll\_GetPhaseSlope, 32
- IDTDpll\_GetPhaseTransient, 33
- IDTDpll\_GetPpsFastLock, 33
- IDTDpll\_GetPpsPhase, 33
- IDTDpll\_GetSoftAlarmThreshold, 34
- IDTDpll\_GetSonetGigEthOutputPath, 34
- IDTDpll\_GetTempHoldoverMode, 34
- IDTDpll\_GetTypeOfDpll, 35
- IDTDpll\_IsDpllLocked, 35
- IDTDpll\_SetAutoSelBandWidth, 35
- IDTDpll\_SetAutoSelInput, 35
- IDTDpll\_SetBandWidthDamping, 36
- IDTDpll\_SetDpll16E116T1Enable, 37
- IDTDpll\_SetDpllEthPathEnable, 37
- IDTDpll\_SetDpllFrozen, 37
- IDTDpll\_SetDpllHardAlarmEnable, 37
- IDTDpll\_SetDpllHardAlarmLimit, 38
- IDTDpll\_SetDpllOperMod, 38
- IDTDpll\_SetDpllOutputPathSelectorA, 38
- IDTDpll\_SetDpllOutputPathSelectorB, 39
- IDTDpll\_SetDpllSelAPathEnable, 39
- IDTDpll\_SetDpllSelBPathEnable, 39
- IDTDpll\_SetDpllSoftAlarmLimit, 39
- IDTDpll\_SetFastLossSwitch, 40
- IDTDpll\_SetForceSelInput, 40
- IDTDpll\_SetFrequencyMonitoringFactor, 40
- IDTDpll\_SetHardAlarmThreshold, 40
- IDTDpll\_SetHitlessSwitchingFeature, 41
- IDTDpll\_SetHoldoverFreq, 41
- IDTDpll\_SetHoldoverMode, 41
- IDTDpll\_SetIcpCtrl, 42
- IDTDpll\_SetPhaseCoarseDetector, 42
- IDTDpll\_SetPhaseCoarseLimit, 43
- IDTDpll\_SetPhaseFineDetectorEnable, 43
- IDTDpll\_SetPhaseFineLimit, 43
- IDTDpll\_SetPhaseOffset, 44
- IDTDpll\_SetPhaseSlope, 44
- IDTDpll\_SetPhaseTransient, 44
- IDTDpll\_SetPpsFastLock, 45
- IDTDpll\_SetPpsPhase, 45
- IDTDpll\_SetSoftAlarmThreshold, 45
- IDTDpll\_SetSonetGigEthOutputPath, 46
- IDTDpll\_SetTempHoldoverMode, 46
- T\_idtBandwidthLimitStage, 16
- T\_idtCoarsePhaseLimit, 16
- T\_idtDampingFactor, 16
- T\_idtDpllBandwidth, 17
- T\_idtDpllHoldover, 17
- T\_idtDpllOperMode, 18
- T\_idtDpllOutputSelectorA, 18
- T\_idtDpllOutputSelectorB, 18
- T\_idtDpllPpsPhasePostion, 19
- T\_idtDpllPpsPulseWidth, 19
- T\_idtFinePhaseLimit, 19
- T\_idtFreqMonFactor, 20
- T\_idtOperDpllMode, 20
- T\_idtPhaseSlope, 20
- T\_idtSonetGigEthOutput, 21
- T\_idtTemp\_holdover, 21
- DPLL Private header file, 7
  - IDTDpll\_pathConfigurationNULL\_c, 9
  - IDTDpll\_sonetGigEthPathConfig2Bitfield, 9
  - SonetGigEthSel\_NUMMAX, 8
  - T\_idtSonetGigEthSelector, 8
  - T\_idtT0DpllSelectorA, 8
  - T\_idtT0DpllSelectorB, 8
  - T\_idtT4DpllSelectorA, 9
  - T\_idtT4DpllSelectorB, 9
- DRV\_MEMMAP\_SIZE
  - ReadWrite\_sim.c, 223
- DampingFactor\_10
  - DPLL Function Module, 17
- DampingFactor\_1p2
  - DPLL Function Module, 17
- DampingFactor\_20
  - DPLL Function Module, 17
- DampingFactor\_2p5
  - DPLL Function Module, 17
- DampingFactor\_5
  - DPLL Function Module, 17
- DampingFactor\_MAX

- DPLL Function Module, 17
- dbleSel
  - T\_idtApIConfig, 158
  - T\_idtApIConfigPerOutput, 159
- Dco\_Mode
  - DPLL Function Module, 18
- dcoMode\_m
  - globalArgs\_t, 149
- dcoModeSupport\_ma
  - T\_idtDrvDeviceConfig, 165
- DeInit\_Sim
  - ReadWrite\_sim.c, 224
  - ReadWrite\_sim.h, 225
- DeInit\_SimWrapper
  - access.c, 190
- decayRateReg\_m
  - bucketReg\_t, 147
- Define.h
  - DPLL\_INSTANCE\_1, 261
  - DPLL\_INSTANCE\_2, 261
  - DPLL\_INSTANCE\_MAX, 261
  - Dpll\_MAX, 261
  - Dpll\_T0, 261
  - Dpll\_T4, 261
  - IDT\_82V33930, 260
  - IDT\_82V3910, 260
  - IDT\_82V3911, 260
  - IDT\_8V89316, 260
  - IDT\_8V89317, 260
  - IDT\_DEVICE\_MAX, 260
- Define.h
  - APLL\_CONFIGURED, 257
  - IDT\_DRV\_MAX\_INPUT, 257
  - IDT\_DRV\_MAX\_OUTPUT, 258
  - MPU\_I2C\_MODE, 258
  - MPU\_SERIAL\_MODE, 258
  - NULL\_APLL\_CONFIG, 258
  - T\_idtAccessDeInitFunc, 258
  - T\_idtAccessInitFunc, 258
  - T\_idtDeviceType, 260
  - T\_idtDpllInstance, 260
  - T\_idtDpllType, 261
  - T\_idtI2CAddrTransFunc, 259
  - T\_idtInputFreqValid, 259
  - T\_idtOutputFreqValid, 259
  - T\_idtReadFunc, 260
  - T\_idtWriteFunc, 260
- deinitFunc\_m
  - T\_idtDrvAccess, 164
- deviceConfig\_m
  - drvHdlr\_s, 168
- deviceType\_m
  - drvHdlr\_s, 168
- divValue\_m
  - T\_idtHfDivEntry, 170
- Divider\_Bypass
  - Input Function Module, 65
- Divider\_DivN
  - Input Function Module, 65
- Divider\_Lock8k
  - Input Function Module, 65
- Divider\_MAX
  - Input Function Module, 65
- Dpll\_MAX
  - Define.h, 261
- Dpll\_T0
  - Define.h, 261
- Dpll\_T4
  - Define.h, 261
- dpIIBandwidth\_0p1Hz
  - DPLL Function Module, 17
- dpIIBandwidth\_0p3Hz
  - DPLL Function Module, 17
- dpIIBandwidth\_0p5mHz
  - DPLL Function Module, 17
- dpIIBandwidth\_0p6Hz
  - DPLL Function Module, 17
- dpIIBandwidth\_15mHz
  - DPLL Function Module, 17
- dpIIBandwidth\_18Hz
  - DPLL Function Module, 17
- dpIIBandwidth\_1mHz
  - DPLL Function Module, 17
- dpIIBandwidth\_1p2Hz
  - DPLL Function Module, 17
- dpIIBandwidth\_2mHz
  - DPLL Function Module, 17
- dpIIBandwidth\_2p5Hz
  - DPLL Function Module, 17
- dpIIBandwidth\_30mHz
  - DPLL Function Module, 17
- dpIIBandwidth\_35Hz
  - DPLL Function Module, 17
- dpIIBandwidth\_4Hz
  - DPLL Function Module, 17
- dpIIBandwidth\_4mHz
  - DPLL Function Module, 17
- dpIIBandwidth\_560Hz
  - DPLL Function Module, 17
- dpIIBandwidth\_60mHz
  - DPLL Function Module, 17
- dpIIBandwidth\_70Hz
  - DPLL Function Module, 17
- dpIIBandwidth\_8Hz
  - DPLL Function Module, 17
- dpIIBandwidth\_8mHz
  - DPLL Function Module, 17
- dpIIBandwidth\_MAX
  - DPLL Function Module, 17
- dpIIBandwidthLimit\_MAX
  - DPLL Function Module, 16
- dpIIBandwidthLimitAcquire
  - DPLL Function Module, 16
- dpIIBandwidthLimitLocked
  - DPLL Function Module, 16
- dpIIBandwidthLimitStart

- DPLL Function Module, 16
- DpllOperMode\_MAX
  - DPLL Function Module, 18
- DpllProfile\_1PPS
  - main.c, 197
- DpllProfile\_G813Option1
  - main.c, 197
- DpllProfile\_G813Option2
  - main.c, 197
- DpllProfile\_G8262Option1
  - main.c, 197
- DpllProfile\_G8262Option2
  - main.c, 197
- DpllProfile\_GR1244
  - main.c, 197
- DpllProfile\_GR253
  - main.c, 197
- DpllProfile\_MAX
  - main.c, 197
- DpllProfile\_SMC
  - main.c, 197
- dpll\_m
  - globalArgs\_t, 149
- DpllCfg.c
  - IDTDpll\_bandwidthLimitRegisters\_a, 291
- dpllI2CTransData\_mp
  - drvHdlr\_s, 168
- dpllOverlayRegisters
  - ReadWrite\_sim.c, 224
- dpllProfile\_ma
  - globalArgs\_t, 149
- drvAccess\_m
  - drvHdlr\_s, 168
- drvHdlr\_s
  - apllI2CTransData\_mp, 168
  - currOutPathCfg\_mp, 168
  - deviceConfig\_m, 168
  - deviceType\_m, 168
  - dpllI2CTransData\_mp, 168
  - drvAccess\_m, 168
  - i2cAddr, 168
  - i2cAddrTransFunc\_m, 168
  - name\_m, 169
  - numberOfApplXtal\_m, 169
  - xtalList, 169
- enOutput
  - T\_idtApplConfig::outputConfig\_s::qaConfig\_s, 154
  - T\_idtApplConfig::outputConfig\_s::qbConfig\_s, 156
  - T\_idtApplConfigPerOutput, 159
- extOutput
  - T\_idtOutputApplConfig, 172
- externalSyncAlarmRange\_1
  - Input Function Module, 64
- externalSyncAlarmRange\_2
  - Input Function Module, 64
- externalSyncAlarmRange\_3
  - Input Function Module, 64
- externalSyncAlarmRange\_4
  - Input Function Module, 64
- externalSyncAlarmRange\_5
  - Input Function Module, 64
- externalSyncAlarmRange\_6
  - Input Function Module, 64
- externalSyncAlarmRange\_7
  - Input Function Module, 64
- externalSyncAlarmRange\_8
  - Input Function Module, 64
- externalSyncAlarmRange\_MAX
  - Input Function Module, 64
- externalSyncBypass\_AutoSel
  - Input Function Module, 64
- externalSyncBypass\_ExSync1
  - Input Function Module, 64
- externalSyncBypass\_MAX
  - Input Function Module, 64
- externalSyncEdge\_FallingEdge
  - Input Function Module, 64
- externalSyncEdge\_MAX
  - Input Function Module, 64
- externalSyncEdge\_RisingEdge
  - Input Function Module, 64
- externalSyncEnable\_Auto
  - Input Function Module, 64
- externalSyncEnable\_Disable
  - Input Function Module, 64
- externalSyncEnable\_Enable
  - Input Function Module, 64
- externalSyncEnable\_MAX
  - Input Function Module, 64
- externalSyncSampling\_0dot5Early
  - Input Function Module, 65
- externalSyncSampling\_0dot5Late
  - Input Function Module, 65
- externalSyncSampling\_1dot0Late
  - Input Function Module, 65
- externalSyncSampling\_MAX
  - Input Function Module, 65
- externalSyncSampling\_onTarget
  - Input Function Module, 65
- fbDiv
  - T\_idtApplConfig, 158
  - T\_idtApplConfigPerOutput, 159
- fbDivConfig0
  - T\_idtApplConfig::fbDiv\_s, 148
- fbDivConfig1
  - T\_idtApplConfig::fbDiv\_s, 148
- filename\_ma
  - globalArgs\_t, 149
- finePhaseLimit\_120\_125nanosec
  - DPLL Function Module, 20
- finePhaseLimit\_180\_360degree
  - DPLL Function Module, 20
- finePhaseLimit\_20\_25nanosec
  - DPLL Function Module, 20
- finePhaseLimit\_45\_90degree
  - DPLL Function Module, 20

- finePhaseLimit\_60\_65nanosec
  - DPLL Function Module, [20](#)
- finePhaseLimit\_90\_180degree
  - DPLL Function Module, [20](#)
- finePhaseLimit\_950\_955nanosec
  - DPLL Function Module, [20](#)
- finePhaseLimit\_MAX
  - DPLL Function Module, [20](#)
- Force\_FreeRun\_Mode
  - DPLL Function Module, [18](#)
- Force\_Holdover\_Mode
  - DPLL Function Module, [18](#)
- Force\_Locked\_Mode
  - DPLL Function Module, [18](#)
- Force\_Lost\_Phase\_Mode
  - DPLL Function Module, [18](#)
- Force\_Pre\_Locked2\_Mode
  - DPLL Function Module, [18](#)
- Force\_Pre\_Locked\_Mode
  - DPLL Function Module, [18](#)
- frameSync\_FRSYNC
  - Output Function Module, [80](#)
- frameSync\_MAX
  - Output Function Module, [80](#)
- frameSync\_MFRSYNC
  - Output Function Module, [80](#)
- Freq\_10M
  - Input Function Module, [66](#)
- Freq\_1544M
  - Input Function Module, [66](#)
- Freq\_1944M
  - Input Function Module, [66](#)
- Freq\_1PPS
  - Input Function Module, [66](#)
- Freq\_2048M
  - Input Function Module, [66](#)
- Freq\_2592M
  - Input Function Module, [66](#)
- Freq\_2K
  - Input Function Module, [66](#)
- Freq\_3888M
  - Input Function Module, [66](#)
- Freq\_4K
  - Input Function Module, [66](#)
- Freq\_625M
  - Input Function Module, [66](#)
- Freq\_648M
  - Input Function Module, [66](#)
- Freq\_8K
  - Input Function Module, [66](#)
- Freq\_MAX
  - Input Function Module, [66](#)
- FreqMonFactor\_0p0032
  - DPLL Function Module, [20](#)
- FreqMonFactor\_0p0064
  - DPLL Function Module, [20](#)
- FreqMonFactor\_0p0127
  - DPLL Function Module, [20](#)
- FreqMonFactor\_0p0257
  - DPLL Function Module, [20](#)
- FreqMonFactor\_0p0514
  - DPLL Function Module, [20](#)
- FreqMonFactor\_0p1030
  - DPLL Function Module, [20](#)
- FreqMonFactor\_0p2060
  - DPLL Function Module, [20](#)
- FreqMonFactor\_0p4120
  - DPLL Function Module, [20](#)
- FreqMonFactor\_0p8230
  - DPLL Function Module, [20](#)
- FreqMonFactor\_1p6460
  - DPLL Function Module, [20](#)
- FreqMonFactor\_3p2920
  - DPLL Function Module, [20](#)
- FreqMonFactor\_3p8100
  - DPLL Function Module, [20](#)
- FreqMonFactor\_4p6000
  - DPLL Function Module, [20](#)
- FreqMonFactor\_MAX
  - DPLL Function Module, [20](#)
- freq\_m
  - T\_IDTFreq2bfVal, [169](#)
- frequencies\_m
  - T\_apllOutputFrequencyEntry, [157](#)
- General Function Module, [47](#)
  - Activate\_Edge\_Falling, [48](#)
  - Activate\_Edge\_Rising, [48](#)
  - IDTGeneral\_DelInitDriver, [50](#)
  - IDTGeneral\_GetDeviceID, [50](#)
  - IDTGeneral\_GetFlagOnTDO, [50](#)
  - IDTGeneral\_GetHardAlarm, [50](#)
  - IDTGeneral\_GetNetworkType, [51](#)
  - IDTGeneral\_GetOscActiveEdge, [51](#)
  - IDTGeneral\_GetOscFreqOffset, [51](#)
  - IDTGeneral\_GetRevertiveMode, [51](#)
  - IDTGeneral\_GetTimeoutValue, [52](#)
  - IDTGeneral\_GetUltraFastSwitch, [52](#)
  - IDTGeneral\_InitDeviceCommon, [53](#)
  - IDTGeneral\_InitDriverCommon, [53](#)
  - IDTGeneral\_SetFlagOnTDO, [53](#)
  - IDTGeneral\_SetFullyUnprotectMode, [53](#)
  - IDTGeneral\_SetHardAlarm, [53](#)
  - IDTGeneral\_SetNetworkType, [54](#)
  - IDTGeneral\_SetOscActiveEdge, [54](#)
  - IDTGeneral\_SetOscFreqOffset, [54](#)
  - IDTGeneral\_SetProtectMode, [54](#)
  - IDTGeneral\_SetRevertiveMode, [54](#)
  - IDTGeneral\_SetSingleUnprotectMode, [55](#)
  - IDTGeneral\_SetTimeoutValue, [55](#)
  - IDTGeneral\_SetUltraFastSwitch, [55](#)
  - IDTGeneral\_i2cTranslate, [52](#)
  - INT\_Active\_High, [48](#)
  - INT\_Active\_Low, [49](#)
  - NOMINAL\_MAX\_PPT, [49](#)
  - NOMINAL\_MIN\_PPT, [49](#)
  - Network\_MAX, [49](#)

- Network\_SDH, 49
- Network\_SONET, 49
- networkType\_t, 49
- phaseTimeoutMultiplier\_16, 50
- phaseTimeoutMultiplier\_2, 49
- phaseTimeoutMultiplier\_4, 49
- phaseTimeoutMultiplier\_8, 49
- phaseTimeoutMultiplier\_MAX, 50
- phaseTimeoutMultiplier\_t, 49
- globalArgs
  - main.c, 198
- globalArgs\_t, 148
  - dcoMode\_m, 149
  - dpll\_m, 149
  - dpllProfile\_ma, 149
  - filename\_ma, 149
  - inputFreq\_m, 149
  - isSim\_m, 149
  - modes\_m, 150
  - numberOfXtal, 150
  - offset\_m, 150
  - outFreq\_m, 150
  - regOffset\_m, 150
  - regValue\_m, 150
  - sampleConfig\_m, 150
  - xtalList, 150
- HF\_Bypass
  - Input Function Module, 65
- HF\_Divide\_4
  - Input Function Module, 65
- HF\_Divide\_5
  - Input Function Module, 65
- HF\_MAX
  - Input Function Module, 65
- Hal.c
  - DEBUG\_IDT\_HAL, 294
- Hardware Abstraction Layer Module, 56
  - IDT\_HAL\_USE\_MACRO, 56
  - IDTHAL\_GetBitfield, 56
  - IDTHAL\_ModifyBitfield, 57
  - IDTHAL\_ReadBitfield, 57
  - IDTHAL\_ReadBitfield\_Ext, 58
  - IDTHAL\_WriteBitfield, 59
  - IDTHAL\_WriteBitfield\_Ext, 59
  - IDTHal\_Read, 57
  - IDTHal\_Read\_Ext, 57
  - IDTHal\_Write, 58
  - IDTHal\_Write\_Ext, 58
- hfDivValue\_m
  - T\_idtHfDivEntry, 170
- Holdover\_Fast\_Average
  - DPLL Function Module, 18
- Holdover\_Instantaneous
  - DPLL Function Module, 18
- Holdover\_MAX
  - DPLL Function Module, 18
- Holdover\_Manual
  - DPLL Function Module, 18
- Holdover\_Slow\_Average
  - DPLL Function Module, 18
- I2cAccessMethods
  - access.c, 192
- i2cAddr
  - drvHdlr\_s, 168
- i2cAddrTransFunc\_m
  - drvHdlr\_s, 168
- IDT\_82V33930
  - Define.h, 260
- IDT\_82V3910
  - Define.h, 260
- IDT\_82V3911
  - Define.h, 260
- IDT\_8V89316
  - Define.h, 260
- IDT\_8V89317
  - Define.h, 260
- IDT\_DEVICE\_MAX
  - Define.h, 260
- IDT\_APLL\_API\_TRACE
  - Appl Function Module, 123
- IDT\_DRV\_MAX\_INPUT
  - Define.h, 257
- IDT\_DRV\_MAX\_OUTPUT
  - Define.h, 258
- IDT\_HAL\_USE\_MACRO
  - Hardware Abstraction Layer Module, 56
- IDTApll\_ApllInput2DpllOutput
  - Appl Function Module, 129
- IDTApll\_DpllOutput2ApplOutput
  - Appl Function Module, 129
- IDTApll\_GetApplConfig
  - Appl Function Module, 129
- IDTApll\_GetApplConfigPerOutput
  - Appl Function Module, 129
- IDTApll\_GetApplFreqTableEntry
  - Appl Function Module, 129
- IDTApll\_GetBypass
  - Appl Function Module, 130
- IDTApll\_GetConfigListByFreq
  - Appl Function Module, 130
- IDTApll\_GetConfigListByXtalType
  - Appl Function Module, 130
- IDTApll\_GetConfigSel
  - Appl Function Module, 131
- IDTApll\_GetCurrentApplFreqConfig
  - Appl Function Module, 131
- IDTApll\_GetFeedbackDiv
  - Appl Function Module, 131
- IDTApll\_GetFeedbackDivRange
  - Appl Function Module, 132
- IDTApll\_GetFreqDoublersSel
  - Appl Function Module, 132
- IDTApll\_GetInputClkSrc
  - Appl Function Module, 132
- IDTApll\_GetMasterResetSync
  - Appl Function Module, 133

- IDTApll\_GetNonActiveApplConfig
  - Appl Function Module, [133](#)
- IDTApll\_GetOutputEnState
  - Appl Function Module, [134](#)
- IDTApll\_GetOutputSigType
  - Appl Function Module, [134](#)
- IDTApll\_GetPreDiv
  - Appl Function Module, [135](#)
- IDTApll\_GetPreDivRange
  - Appl Function Module, [135](#)
- IDTApll\_GetQaDiv
  - Appl Function Module, [135](#)
- IDTApll\_GetQbDiv
  - Appl Function Module, [136](#)
- IDTApll\_GetShutoff
  - Appl Function Module, [136](#)
- IDTApll\_GetSupportedXtal
  - Appl Function Module, [137](#)
- IDTApll\_GetXtalId
  - Appl Function Module, [137](#)
- IDTApll\_GetXtalIdFromXtalFreq
  - Appl Function Module, [137](#)
- IDTApll\_NullFreqTableEntry
  - Appl Function Module, [138](#)
- IDTApll\_SetApplConfig
  - Appl Function Module, [138](#)
- IDTApll\_SetApplConfigPerOutput
  - Appl Function Module, [138](#)
- IDTApll\_SetBypass
  - Appl Function Module, [139](#)
- IDTApll\_SetConfigSel
  - Appl Function Module, [139](#)
- IDTApll\_SetFeedbackDiv
  - Appl Function Module, [139](#)
- IDTApll\_SetFreqDoublerSel
  - Appl Function Module, [140](#)
- IDTApll\_SetInputClkSrc
  - Appl Function Module, [140](#)
- IDTApll\_SetMasterResetSync
  - Appl Function Module, [140](#)
- IDTApll\_SetOutputEnState
  - Appl Function Module, [141](#)
- IDTApll\_SetOutputSigType
  - Appl Function Module, [141](#)
- IDTApll\_SetPreDiv
  - Appl Function Module, [142](#)
- IDTApll\_SetQaDiv
  - Appl Function Module, [142](#)
- IDTApll\_SetQbDiv
  - Appl Function Module, [143](#)
- IDTApll\_SetShutoff
  - Appl Function Module, [143](#)
- IDTApll\_SetXtalId
  - Appl Function Module, [144](#)
- IDTApll\_Switch2NonActiveApplConfig
  - Appl Function Module, [144](#)
- IDTApll\_ValidXtalInput
  - Appl Function Module, [144](#)
- IDTApll\_XtalConfigured
  - Appl Function Module, [145](#)
- IDTDpll\_Get1stPriorityInput
  - DPLL Function Module, [21](#)
- IDTDpll\_Get2ndPriorityInput
  - DPLL Function Module, [22](#)
- IDTDpll\_Get3rdPriorityInput
  - DPLL Function Module, [22](#)
- IDTDpll\_GetAutoSelBandWidth
  - DPLL Function Module, [22](#)
- IDTDpll\_GetAutoSelInput
  - DPLL Function Module, [22](#)
- IDTDpll\_GetBandWidthDamping
  - DPLL Function Module, [23](#)
- IDTDpll\_GetCurrentDpllFreqOffset
  - DPLL Function Module, [24](#)
- IDTDpll\_GetCurrentPhaseError
  - DPLL Function Module, [24](#)
- IDTDpll\_GetCurrentSelectInput
  - DPLL Function Module, [24](#)
- IDTDpll\_GetDpll16E116T1Enable
  - DPLL Function Module, [24](#)
- IDTDpll\_GetDpllEthPathEnable
  - DPLL Function Module, [25](#)
- IDTDpll\_GetDpllFrozen
  - DPLL Function Module, [25](#)
- IDTDpll\_GetDpllHardAlarmEnable
  - DPLL Function Module, [25](#)
- IDTDpll\_GetDpllHardAlarmLimit
  - DPLL Function Module, [25](#)
- IDTDpll\_GetDpllOperMod
  - DPLL Function Module, [26](#)
- IDTDpll\_GetDpllOutputPathSelectorA
  - DPLL Function Module, [26](#)
- IDTDpll\_GetDpllOutputPathSelectorB
  - DPLL Function Module, [26](#)
- IDTDpll\_GetDpllSelAPathEnable
  - DPLL Function Module, [27](#)
- IDTDpll\_GetDpllSelBPathEnable
  - DPLL Function Module, [27](#)
- IDTDpll\_GetDpllSoftAlarmLimit
  - DPLL Function Module, [27](#)
- IDTDpll\_GetFastLossSwitch
  - DPLL Function Module, [28](#)
- IDTDpll\_GetForceSelInput
  - DPLL Function Module, [28](#)
- IDTDpll\_GetFrequencyMonitoringFactor
  - DPLL Function Module, [28](#)
- IDTDpll\_GetHardAlarmThreshold
  - DPLL Function Module, [29](#)
- IDTDpll\_GetHitlessSwitchingFeature
  - DPLL Function Module, [29](#)
- IDTDpll\_GetHoldoverFreq
  - DPLL Function Module, [29](#)
- IDTDpll\_GetHoldoverMode
  - DPLL Function Module, [30](#)
- IDTDpll\_GetIcpCtrl
  - DPLL Function Module, [30](#)

- IDTDpll\_GetPhaseCoarseDetector
  - DPLL Function Module, [30](#)
- IDTDpll\_GetPhaseCoarseLimit
  - DPLL Function Module, [31](#)
- IDTDpll\_GetPhaseFineDetectorEnable
  - DPLL Function Module, [31](#)
- IDTDpll\_GetPhaseFineLimit
  - DPLL Function Module, [32](#)
- IDTDpll\_GetPhaseOffset
  - DPLL Function Module, [32](#)
- IDTDpll\_GetPhaseSlope
  - DPLL Function Module, [32](#)
- IDTDpll\_GetPhaseTransient
  - DPLL Function Module, [33](#)
- IDTDpll\_GetPpsFastLock
  - DPLL Function Module, [33](#)
- IDTDpll\_GetPpsPhase
  - DPLL Function Module, [33](#)
- IDTDpll\_GetSoftAlarmThreshold
  - DPLL Function Module, [34](#)
- IDTDpll\_GetSonetGigEthOutputPath
  - DPLL Function Module, [34](#)
- IDTDpll\_GetTempHoldoverMode
  - DPLL Function Module, [34](#)
- IDTDpll\_GetTypeOfDpll
  - DPLL Function Module, [35](#)
- IDTDpll\_IsDpllLocked
  - DPLL Function Module, [35](#)
- IDTDpll\_SetAutoSelBandWidth
  - DPLL Function Module, [35](#)
- IDTDpll\_SetAutoSelInput
  - DPLL Function Module, [35](#)
- IDTDpll\_SetBandWidthDamping
  - DPLL Function Module, [36](#)
- IDTDpll\_SetDpll16E116T1Enable
  - DPLL Function Module, [37](#)
- IDTDpll\_SetDpllEthPathEnable
  - DPLL Function Module, [37](#)
- IDTDpll\_SetDpllFrozen
  - DPLL Function Module, [37](#)
- IDTDpll\_SetDpllHardAlarmEnable
  - DPLL Function Module, [37](#)
- IDTDpll\_SetDpllHardAlarmLimit
  - DPLL Function Module, [38](#)
- IDTDpll\_SetDpllOperMod
  - DPLL Function Module, [38](#)
- IDTDpll\_SetDpllOutputPathSelectorA
  - DPLL Function Module, [38](#)
- IDTDpll\_SetDpllOutputPathSelectorB
  - DPLL Function Module, [39](#)
- IDTDpll\_SetDpllSelAPathEnable
  - DPLL Function Module, [39](#)
- IDTDpll\_SetDpllSelBPathEnable
  - DPLL Function Module, [39](#)
- IDTDpll\_SetDpllSoftAlarmLimit
  - DPLL Function Module, [39](#)
- IDTDpll\_SetFastLossSwitch
  - DPLL Function Module, [40](#)
- IDTDpll\_SetForceSelInput
  - DPLL Function Module, [40](#)
- IDTDpll\_SetFrequencyMonitoringFactor
  - DPLL Function Module, [40](#)
- IDTDpll\_SetHardAlarmThreshold
  - DPLL Function Module, [40](#)
- IDTDpll\_SetHitlessSwitchingFeature
  - DPLL Function Module, [41](#)
- IDTDpll\_SetHoldoverFreq
  - DPLL Function Module, [41](#)
- IDTDpll\_SetHoldoverMode
  - DPLL Function Module, [41](#)
- IDTDpll\_SetIcpCtrl
  - DPLL Function Module, [42](#)
- IDTDpll\_SetPhaseCoarseDetector
  - DPLL Function Module, [42](#)
- IDTDpll\_SetPhaseCoarseLimit
  - DPLL Function Module, [43](#)
- IDTDpll\_SetPhaseFineDetectorEnable
  - DPLL Function Module, [43](#)
- IDTDpll\_SetPhaseFineLimit
  - DPLL Function Module, [43](#)
- IDTDpll\_SetPhaseOffset
  - DPLL Function Module, [44](#)
- IDTDpll\_SetPhaseSlope
  - DPLL Function Module, [44](#)
- IDTDpll\_SetPhaseTransient
  - DPLL Function Module, [44](#)
- IDTDpll\_SetPpsFastLock
  - DPLL Function Module, [45](#)
- IDTDpll\_SetPpsPhase
  - DPLL Function Module, [45](#)
- IDTDpll\_SetSoftAlarmThreshold
  - DPLL Function Module, [45](#)
- IDTDpll\_SetSonetGigEthOutputPath
  - DPLL Function Module, [46](#)
- IDTDpll\_SetTempHoldoverMode
  - DPLL Function Module, [46](#)
- IDTDpll\_bandwidthLimitRegisters\_a
  - DpllCfg.c, [291](#)
- IDTDpll\_pathConfigurationNULL\_c
  - DPLL Private header file, [9](#)
- IDTDpll\_sonetGigEthPathConfig2Bitfield
  - DPLL Private header file, [9](#)
- IDTGeneral\_DeInitDriver
  - General Function Module, [50](#)
- IDTGeneral\_GetDeviceID
  - General Function Module, [50](#)
- IDTGeneral\_GetFlagOnTDO
  - General Function Module, [50](#)
- IDTGeneral\_GetHardAlarm
  - General Function Module, [50](#)
- IDTGeneral\_GetNetworkType
  - General Function Module, [51](#)
- IDTGeneral\_GetOscActiveEdge
  - General Function Module, [51](#)
- IDTGeneral\_GetOscFreqOffset
  - General Function Module, [51](#)

- IDTGeneral\_GetRevertiveMode
  - General Function Module, [51](#)
- IDTGeneral\_GetTimeoutValue
  - General Function Module, [52](#)
- IDTGeneral\_GetUltraFastSwitch
  - General Function Module, [52](#)
- IDTGeneral\_InitDevice
  - 82V3910.c, [177](#)
  - idt82V3910.h, [176](#)
- IDTGeneral\_InitDeviceCommon
  - General Function Module, [53](#)
- IDTGeneral\_InitDriver
  - 82V3910.c, [177](#)
  - idt82V3910.h, [176](#)
- IDTGeneral\_InitDriverCommon
  - General Function Module, [53](#)
- IDTGeneral\_SetFlagOnTDO
  - General Function Module, [53](#)
- IDTGeneral\_SetFullyUnprotectMode
  - General Function Module, [53](#)
- IDTGeneral\_SetHardAlarm
  - General Function Module, [53](#)
- IDTGeneral\_SetNetworkType
  - General Function Module, [54](#)
- IDTGeneral\_SetOscActiveEdge
  - General Function Module, [54](#)
- IDTGeneral\_SetOscFreqOffset
  - General Function Module, [54](#)
- IDTGeneral\_SetProtectMode
  - General Function Module, [54](#)
- IDTGeneral\_SetRevertiveMode
  - General Function Module, [54](#)
- IDTGeneral\_SetSingleUnprotectMode
  - General Function Module, [55](#)
- IDTGeneral\_SetTimeoutValue
  - General Function Module, [55](#)
- IDTGeneral\_SetUltraFastSwitch
  - General Function Module, [55](#)
- IDTGeneral\_i2cTranslate
  - General Function Module, [52](#)
- IDTHAL\_GetBitfield
  - Hardware Abstraction Layer Module, [56](#)
- IDTHAL\_ModifyBitfield
  - Hardware Abstraction Layer Module, [57](#)
- IDTHAL\_ReadBitfield
  - Hardware Abstraction Layer Module, [57](#)
- IDTHAL\_ReadBitfield\_Ext
  - Hardware Abstraction Layer Module, [58](#)
- IDTHAL\_WriteBitfield
  - Hardware Abstraction Layer Module, [59](#)
- IDTHAL\_WriteBitfield\_Ext
  - Hardware Abstraction Layer Module, [59](#)
- IDTHal\_Read
  - Hardware Abstraction Layer Module, [57](#)
- IDTHal\_Read\_Ext
  - Hardware Abstraction Layer Module, [57](#)
- IDTHal\_Write
  - Hardware Abstraction Layer Module, [58](#)
- IDTHal\_Write\_Ext
  - Hardware Abstraction Layer Module, [58](#)
- IDTHIApi\_ApplyFrequencies
  - outputFreq\_priv.h, [235](#)
  - outputFreqUtil.c, [249](#)
- IDTHIApi\_ClearProvisioning
  - outputFreq.c, [243](#)
  - outputFreq.h, [230](#)
- IDTHIApi\_ConfigureInput1PPS
  - inputFreq.c, [241](#)
  - inputFreq.h, [226](#)
- IDTHIApi\_ConfigureInputAmi
  - inputFreq.c, [241](#)
  - inputFreq.h, [227](#)
- IDTHIApi\_ConfigureInputFrequency
  - inputFreq.c, [241](#)
  - inputFreq.h, [227](#)
- IDTHIApi\_ConfigureOutput
  - outputFreq.c, [243](#)
  - outputFreq.h, [231](#)
- IDTHIApi\_ConvertFromFreqListToFreqArray
  - outputFreq\_priv.h, [235](#)
  - outputFreqUtil.c, [249](#)
- IDTHIApi\_DisableAllInputPriorities
  - inputFreq.c, [241](#)
  - inputFreq.h, [227](#)
- IDTHIApi\_DisableAllOutputs
  - outputFreq.c, [244](#)
  - outputFreq.h, [232](#)
- IDTHIApi\_Frequency2String\_c
  - main.c, [198](#)
  - outputFreq\_priv.h, [237](#)
  - outputFreqString.c, [248](#)
- IDTHIApi\_IsFreqArrayInFreqList
  - outputFreq\_priv.h, [235](#)
  - outputFreqUtil.c, [250](#)
- IDTHIApi\_IsFrequencyInFreqList
  - outputFreq\_priv.h, [235](#)
  - outputFreqUtil.c, [250](#)
- IDTHIApi\_PrintAvailFrequencies
  - outputFreq\_priv.h, [236](#)
  - outputFreqString.c, [247](#)
- IDTHIApi\_PrintConfiguration
  - outputFreq\_priv.h, [236](#)
  - outputFreqString.c, [247](#)
- IDTHIApi\_PrintFrequencyList
  - outputFreq\_priv.h, [236](#)
  - outputFreqString.c, [247](#)
- IDTHIApi\_PrintInput
  - outputFreq.h, [232](#)
  - outputFreqString.c, [247](#)
- IDTHIApi\_PrintInputHwConfig
  - outputFreqString.c, [247](#)
- IDTHIApi\_PrintOption
  - outputFreq\_priv.h, [236](#)
  - outputFreqString.c, [247](#)
- IDTHIApi\_PrintOutput
  - outputFreq.h, [232](#)

- outputFreqString.c, [247](#)
- IDTHIApi\_PrintOutputHwConfig
  - outputFreq\_priv.h, [236](#)
  - outputFreqString.c, [248](#)
- IDTHIApi\_PrintSelectedOutputPath
  - outputFreq\_priv.h, [237](#)
  - outputFreqString.c, [248](#)
- IDTHIApi\_RetrieveOutputPathMuxFromHardware
  - outputFreq.c, [244](#)
- IDTHIApi\_g813Option1
  - profiles.c, [252](#)
  - profiles.h, [238](#)
- IDTHIApi\_g813Option2
  - profiles.c, [252](#)
  - profiles.h, [238](#)
- IDTHIApi\_g8262EecOption1
  - profiles.c, [252](#)
  - profiles.h, [238](#)
- IDTHIApi\_g8262EecOption2
  - profiles.c, [252](#)
  - profiles.h, [239](#)
- IDTHIApi\_gr1244Stratum3
  - profiles.c, [252](#)
  - profiles.h, [239](#)
- IDTHIApi\_gr253SonetStratum3
  - profiles.c, [252](#)
  - profiles.h, [239](#)
- IDTHIApi\_pps
  - profiles.c, [253](#)
  - profiles.h, [239](#)
- IDTHIApi\_smc
  - profiles.c, [253](#)
  - profiles.h, [239](#)
- IDTHIApi\_wideband
  - profiles.c, [253](#)
  - profiles.h, [240](#)
- IDTInput\_DisableAllInputs
  - Input Function Module, [67](#)
- IDTInput\_EnableAllInputs
  - Input Function Module, [67](#)
- IDTInput\_GetActivityMonitor
  - Input Function Module, [67](#)
- IDTInput\_GetAmiInputFrequency
  - Input Function Module, [67](#)
- IDTInput\_GetBucketConfig
  - Input Function Module, [68](#)
- IDTInput\_GetDivisionFactor
  - Input Function Module, [68](#)
- IDTInput\_GetInputClockStatus
  - Input Function Module, [68](#)
- IDTInput\_GetInputClockValidation
  - Input Function Module, [69](#)
- IDTInput\_GetInputEnable
  - Input Function Module, [69](#)
- IDTInput\_GetInputFreqOffset
  - Input Function Module, [69](#)
- IDTInput\_GetInputFrequency
  - Input Function Module, [69](#)
- IDTInput\_GetInputHFdivider
  - Input Function Module, [70](#)
- IDTInput\_GetPreDivider
  - Input Function Module, [70](#)
- IDTInput\_GetPriority
  - Input Function Module, [71](#)
- IDTInput\_GetSyncAlarmRange
  - Input Function Module, [71](#)
- IDTInput\_GetSyncBypass
  - Input Function Module, [71](#)
- IDTInput\_GetSyncEdge
  - Input Function Module, [71](#)
- IDTInput\_GetSyncEnable
  - Input Function Module, [72](#)
- IDTInput\_GetSyncFrequency
  - Input Function Module, [72](#)
- IDTInput\_GetSyncSampling
  - Input Function Module, [72](#)
- IDTInput\_InFreqBitValue2Family
  - Input Function Module, [72](#)
- IDTInput\_SetActivityMonitor
  - Input Function Module, [73](#)
- IDTInput\_SetAmiInputFrequency
  - Input Function Module, [73](#)
- IDTInput\_SetBucketConfig
  - Input Function Module, [73](#)
- IDTInput\_SetDivisionFactor
  - Input Function Module, [74](#)
- IDTInput\_SetInputEnable
  - Input Function Module, [74](#)
- IDTInput\_SetInputFrequency
  - Input Function Module, [74](#)
- IDTInput\_SetInputHFdivider
  - Input Function Module, [75](#)
- IDTInput\_SetPreDivider
  - Input Function Module, [75](#)
- IDTInput\_SetPriority
  - Input Function Module, [75](#)
- IDTInput\_SetSyncAlarmRange
  - Input Function Module, [76](#)
- IDTInput\_SetSyncBypass
  - Input Function Module, [76](#)
- IDTInput\_SetSyncEdge
  - Input Function Module, [76](#)
- IDTInput\_SetSyncEnable
  - Input Function Module, [76](#)
- IDTInput\_SetSyncFrequency
  - Input Function Module, [76](#)
- IDTInput\_SetSyncSampling
  - Input Function Module, [77](#)
- IDTOutput\_DisableAllIntOutput
  - Output Function Module, [80](#)
- IDTOutput\_DisableApIInput
  - Output Function Module, [81](#)
- IDTOutput\_GetFrameSync
  - Output Function Module, [81](#)
- IDTOutput\_GetFrameSyncPulseEdge
  - Output Function Module, [81](#)

- IDTOutput\_GetOutputpulInterface
  - Output Function Module, [81](#)
- IDTOutput\_GetOutputConfig
  - Output Function Module, [82](#)
- IDTOutput\_GetOutputEnable
  - Output Function Module, [82](#)
- IDTOutput\_GetOutputInvert
  - Output Function Module, [82](#)
- IDTOutput\_SetFrameSync
  - Output Function Module, [82](#)
- IDTOutput\_SetFrameSyncPulseEdge
  - Output Function Module, [83](#)
- IDTOutput\_SetOutputpulInterface
  - Output Function Module, [83](#)
- IDTOutput\_SetOutputConfig
  - Output Function Module, [83](#)
- IDTOutput\_SetOutputEnable
  - Output Function Module, [83](#)
- IDTOutput\_SetOutputInvert
  - Output Function Module, [84](#)
- IDTSample\_Configure1Pps
  - sample.c, [199](#)
  - sample.h, [183](#)
- IDTSample\_Configure1PpsHoldover
  - sample.c, [199](#)
  - sample.h, [183](#)
- IDTSample\_ConfigureDcoMode
  - sample.c, [199](#)
  - sample.h, [183](#)
- IDTSample\_ConfigureSampleConfig1
  - sample.c, [199](#)
  - sample.h, [183](#)
- IDTSample\_ConfigureSampleConfig2
  - sample.c, [200](#)
  - sample.h, [183](#)
- IDTSample\_ConfigureSampleConfig3
  - sample.c, [200](#)
  - sample.h, [184](#)
- IDTSample\_ConfigureSampleConfig4
  - sample.c, [200](#)
  - sample.h, [184](#)
- IDTSample\_ConfigureSampleConfig5
  - sample.c, [201](#)
  - sample.h, [184](#)
- IDTSample\_ConfigureSampleConfigOutput6\_156\_dot-
  - \_25MHz
  - sampleAppl.c, [202](#)
  - sampleAppl.h, [186](#)
- IDTSample\_ConfigureSampleConfigureOutput10\_25\_-
  - MHz
  - sampleAppl.c, [203](#)
  - sampleAppl.h, [186](#)
- IDTSample\_ConfigureSampleConfigureOutput7\_77\_-
  - dot\_76MHz
  - sampleAppl.c, [203](#)
  - sampleAppl.h, [186](#)
- IDTSample\_DualMode
  - sampleAppl.c, [203](#)
- sampleAppl.h, [187](#)
- IDTSample\_DumpRegistersRgf
  - main.c, [197](#)
- IDTSample\_GPS
  - sampleAppl.c, [203](#)
  - sampleAppl.h, [187](#)
- IDTSample\_GPS\_1PPS
  - sampleAppl.c, [204](#)
  - sampleAppl.h, [187](#)
- IDTSample\_GetCombinedDcoOffset
  - sample.c, [201](#)
  - sample.h, [184](#)
- IDTSample\_GetLogVerbosity
  - access.c, [190](#)
  - access.h, [179](#)
- IDTSample\_InitDriverl2c
  - access.c, [190](#)
  - access.h, [179](#)
- IDTSample\_InitDriverSimulator
  - access.c, [190](#)
  - access.h, [180](#)
- IDTSample\_LoadRgfFile
  - loadstore.c, [193](#)
  - loadstore.h, [181](#)
- IDTSample\_ResetXtalList
  - access.c, [191](#)
  - access.h, [180](#)
- IDTSample\_SaveRgfFile
  - loadstore.c, [193](#)
  - loadstore.h, [181](#)
- IDTSample\_SetCombinedDcoOffset
  - sample.c, [201](#)
  - sample.h, [185](#)
- IDTSample\_SetLogVerbosity
  - access.c, [191](#)
  - access.h, [180](#)
- IDTSample\_Sonet
  - sampleAppl.c, [204](#)
  - sampleAppl.h, [187](#)
- IDTSample\_SyncE\_LAN
  - sampleAppl.c, [204](#)
  - sampleAppl.h, [188](#)
- IDTSample\_SyncE\_Sonet
  - sampleAppl.c, [205](#)
  - sampleAppl.h, [188](#)
- IDTSample\_SyncE\_WAN
  - sampleAppl.c, [205](#)
  - sampleAppl.h, [188](#)
- INT\_Active\_High
  - General Function Module, [48](#)
- INT\_Active\_Low
  - General Function Module, [49](#)
- idt82V3910.h
  - IDTGeneral\_InitDevice, [176](#)
  - IDTGeneral\_InitDriver, [176](#)
- idt82V3910Config
  - 82V3910.c, [178](#)
- InFreqFamily\_10M

- Input Function Module, [66](#)
- InFreqFamily\_1544M
  - Input Function Module, [66](#)
- InFreqFamily\_1944M
  - Input Function Module, [66](#)
- InFreqFamily\_1PPS
  - Input Function Module, [66](#)
- InFreqFamily\_2048M
  - Input Function Module, [66](#)
- InFreqFamily\_2592M
  - Input Function Module, [66](#)
- InFreqFamily\_2K
  - Input Function Module, [66](#)
- InFreqFamily\_3888M
  - Input Function Module, [66](#)
- InFreqFamily\_4K
  - Input Function Module, [66](#)
- InFreqFamily\_625M
  - Input Function Module, [66](#)
- InFreqFamily\_648M
  - Input Function Module, [66](#)
- InFreqFamily\_8K
  - Input Function Module, [66](#)
- InFreqFamily\_MAX
  - Input Function Module, [66](#)
- inSyncXlate\_ma
  - T\_idtDrvDeviceConfig, [165](#)
- Init\_Sim
  - ReadWrite\_sim.c, [224](#)
  - ReadWrite\_sim.h, [225](#)
- Init\_SimWrapper
  - access.c, [191](#)
- initFunc\_m
  - T\_idtDrvAccess, [164](#)
- Input Function Module, [60](#)
  - AmiFreq\_64kHzPlus400Hz, [65](#)
  - AmiFreq\_64kHzPlus8kHz, [65](#)
  - AmiFreq\_MAX, [65](#)
  - Divider\_Bypass, [65](#)
  - Divider\_DivN, [65](#)
  - Divider\_Lock8k, [65](#)
  - Divider\_MAX, [65](#)
  - externalSyncAlarmRange\_1, [64](#)
  - externalSyncAlarmRange\_2, [64](#)
  - externalSyncAlarmRange\_3, [64](#)
  - externalSyncAlarmRange\_4, [64](#)
  - externalSyncAlarmRange\_5, [64](#)
  - externalSyncAlarmRange\_6, [64](#)
  - externalSyncAlarmRange\_7, [64](#)
  - externalSyncAlarmRange\_8, [64](#)
  - externalSyncAlarmRange\_MAX, [64](#)
  - externalSyncBypass\_AutoSel, [64](#)
  - externalSyncBypass\_ExSync1, [64](#)
  - externalSyncBypass\_MAX, [64](#)
  - externalSyncEdge\_FallingEdge, [64](#)
  - externalSyncEdge\_MAX, [64](#)
  - externalSyncEdge\_RisingEdge, [64](#)
  - externalSyncEnable\_Auto, [64](#)
  - externalSyncEnable\_Disable, [64](#)
  - externalSyncEnable\_Enable, [64](#)
  - externalSyncEnable\_MAX, [64](#)
  - externalSyncSampling\_0dot5Early, [65](#)
  - externalSyncSampling\_0dot5Late, [65](#)
  - externalSyncSampling\_1dot0Late, [65](#)
  - externalSyncSampling\_MAX, [65](#)
  - externalSyncSampling\_onTarget, [65](#)
  - Freq\_10M, [66](#)
  - Freq\_1544M, [66](#)
  - Freq\_1944M, [66](#)
  - Freq\_1PPS, [66](#)
  - Freq\_2048M, [66](#)
  - Freq\_2592M, [66](#)
  - Freq\_2K, [66](#)
  - Freq\_3888M, [66](#)
  - Freq\_4K, [66](#)
  - Freq\_625M, [66](#)
  - Freq\_648M, [66](#)
  - Freq\_8K, [66](#)
  - Freq\_MAX, [66](#)
  - HF\_Bypass, [65](#)
  - HF\_Divide\_4, [65](#)
  - HF\_Divide\_5, [65](#)
  - HF\_MAX, [65](#)
  - IDTInput\_DisableAllInputs, [67](#)
  - IDTInput\_EnableAllInputs, [67](#)
  - IDTInput\_GetActivityMonitor, [67](#)
  - IDTInput\_GetAmiInputFrequency, [67](#)
  - IDTInput\_GetBucketConfig, [68](#)
  - IDTInput\_GetDivisionFactor, [68](#)
  - IDTInput\_GetInputClockStatus, [68](#)
  - IDTInput\_GetInputClockValidation, [69](#)
  - IDTInput\_GetInputEnable, [69](#)
  - IDTInput\_GetInputFreqOffset, [69](#)
  - IDTInput\_GetInputFrequency, [69](#)
  - IDTInput\_GetInputHFdivider, [70](#)
  - IDTInput\_GetPreDivider, [70](#)
  - IDTInput\_GetPriority, [71](#)
  - IDTInput\_GetSyncAlarmRange, [71](#)
  - IDTInput\_GetSyncBypass, [71](#)
  - IDTInput\_GetSyncEdge, [71](#)
  - IDTInput\_GetSyncEnable, [72](#)
  - IDTInput\_GetSyncFrequency, [72](#)
  - IDTInput\_GetSyncSampling, [72](#)
  - IDTInput\_InFreqBitValue2Family, [72](#)
  - IDTInput\_SetActivityMonitor, [73](#)
  - IDTInput\_SetAmiInputFrequency, [73](#)
  - IDTInput\_SetBucketConfig, [73](#)
  - IDTInput\_SetDivisionFactor, [74](#)
  - IDTInput\_SetInputEnable, [74](#)
  - IDTInput\_SetInputFrequency, [74](#)
  - IDTInput\_SetInputHFdivider, [75](#)
  - IDTInput\_SetPreDivider, [75](#)
  - IDTInput\_SetPriority, [75](#)
  - IDTInput\_SetSyncAlarmRange, [76](#)
  - IDTInput\_SetSyncBypass, [76](#)
  - IDTInput\_SetSyncEdge, [76](#)

- IDTInput\_SetSyncEnable, [76](#)
  - IDTInput\_SetSyncFrequency, [76](#)
  - IDTInput\_SetSyncSampling, [77](#)
  - InFreqFamily\_10M, [66](#)
  - InFreqFamily\_1544M, [66](#)
  - InFreqFamily\_1944M, [66](#)
  - InFreqFamily\_1PPS, [66](#)
  - InFreqFamily\_2048M, [66](#)
  - InFreqFamily\_2592M, [66](#)
  - InFreqFamily\_2K, [66](#)
  - InFreqFamily\_3888M, [66](#)
  - InFreqFamily\_4K, [66](#)
  - InFreqFamily\_625M, [66](#)
  - InFreqFamily\_648M, [66](#)
  - InFreqFamily\_8K, [66](#)
  - InFreqFamily\_MAX, [66](#)
  - Input\_Freq\_Hard\_Alarm, [63](#)
  - Input\_No\_Activity\_Alarm, [63](#)
  - Input\_Phase\_Lock\_Alarm, [63](#)
  - SYNC\_1PPS, [67](#)
  - SYNC\_2kHz, [67](#)
  - SYNC\_4kHz, [67](#)
  - SYNC\_8kHz, [67](#)
  - SYNC\_MAX, [67](#)
  - T\_externalSyncAlarmRange, [63](#)
  - T\_externalSyncBypass, [64](#)
  - T\_externalSyncEdge, [64](#)
  - T\_externalSyncEnable, [64](#)
  - T\_externalSyncSampling, [64](#)
  - T\_idtAmiInputFrequencyBitFieldValue, [65](#)
  - T\_idtHighFrequencyDivisor, [65](#)
  - T\_idtInputDividerMux, [65](#)
  - T\_idtInputFrequencyBitfieldValue, [65](#)
  - T\_idtInputFrequencyFamily, [66](#)
  - T\_idtInputSyncFreq, [66](#)
- Input\_Freq\_Hard\_Alarm
  - Input Function Module, [63](#)
- Input\_No\_Activity\_Alarm
  - Input Function Module, [63](#)
- Input\_Phase\_Lock\_Alarm
  - Input Function Module, [63](#)
- inputClk
  - T\_idtApIConfig, [158](#)
  - T\_idtApIConfigPerOutput, [159](#)
- inputDpll\_m
  - T\_idtSonetGigEthBitfield2PathConfig, [173](#)
- inputExt2IntXlate\_ma
  - T\_idtDrvDeviceConfig, [165](#)
- inputFreq.c
  - IDTHIApi\_ConfigureInput1PPS, [241](#)
  - IDTHIApi\_ConfigureInputAmi, [241](#)
  - IDTHIApi\_ConfigureInputFrequency, [241](#)
  - IDTHIApi\_DisableAllInputPriorities, [241](#)
- inputFreq.h
  - IDTHIApi\_ConfigureInput1PPS, [226](#)
  - IDTHIApi\_ConfigureInputAmi, [227](#)
  - IDTHIApi\_ConfigureInputFrequency, [227](#)
  - IDTHIApi\_DisableAllInputPriorities, [227](#)
- inputFreq\_m
  - globalArgs\_t, [149](#)
- inputFreqValidFunc\_m
  - T\_idtDrvDeviceConfig, [165](#)
- instance\_m
  - pathSelectorConfig\_t, [152](#)
- isSim\_m
  - globalArgs\_t, [149](#)
- loadstore.c
  - IDTSample\_LoadRgfFile, [193](#)
  - IDTSample\_SaveRgfFile, [193](#)
- loadstore.h
  - IDTSample\_LoadRgfFile, [181](#)
  - IDTSample\_SaveRgfFile, [181](#)
- lowerThreshReg\_m
  - bucketReg\_t, [147](#)
- MODE\_DpllProfile
  - main.c, [195](#)
- MODE\_Input
  - main.c, [195](#)
- MODE\_Load
  - main.c, [195](#)
- MODE\_None
  - main.c, [195](#)
- MODE\_Output
  - main.c, [196](#)
- MODE\_Read
  - main.c, [196](#)
- MODE\_Sample
  - main.c, [196](#)
- MODE\_Save
  - main.c, [196](#)
- MODE\_Write
  - main.c, [196](#)
- MPU\_I2C\_MODE
  - Define.h, [258](#)
- MPU\_SERIAL\_MODE
  - Define.h, [258](#)
- main
  - main.c, [197](#)
- main.c
  - DpllProfile\_1PPS, [197](#)
  - DpllProfile\_G813Option1, [197](#)
  - DpllProfile\_G813Option2, [197](#)
  - DpllProfile\_G8262Option1, [197](#)
  - DpllProfile\_G8262Option2, [197](#)
  - DpllProfile\_GR1244, [197](#)
  - DpllProfile\_GR253, [197](#)
  - DpllProfile\_MAX, [197](#)
  - DpllProfile\_SMC, [197](#)
- main.c
  - globalArgs, [198](#)
  - IDTHIApi\_Frequency2String\_c, [198](#)
  - IDTSample\_DumpRegistersRgf, [197](#)
  - MODE\_DpllProfile, [195](#)
  - MODE\_Input, [195](#)
  - MODE\_Load, [195](#)

- MODE\_None, 195
- MODE\_Output, 196
- MODE\_Read, 196
- MODE\_Sample, 196
- MODE\_Save, 196
- MODE\_Write, 196
- main, 197
- no\_argument, 196
- optional\_argument, 196
- printHelp, 197
- processCmdLineArguments, 197
- required\_argument, 196
- SAMPLE\_MAX\_FILENAME, 196
- simDebugFlag, 198
- T\_DpllProfile, 197
- T\_DpllProfileFunc, 197
- T\_SampleConfigDescrFunc, 197
- T\_SampleConfigFunc, 197
- maxNumXtalPerAppl\_m
  - T\_idtDrvDeviceConfig, 166
- modes\_m
  - globalArgs\_t, 150
- mrSync
  - T\_idtApplConfig, 158
- NOMINAL\_MAX\_PPT
  - General Function Module, 49
- NOMINAL\_MIN\_PPT
  - General Function Module, 49
- NULL\_APLL\_CONFIG
  - Define.h, 258
- name\_m
  - drvHdlr\_s, 169
- Network\_MAX
  - General Function Module, 49
- Network\_SDH
  - General Function Module, 49
- Network\_SONET
  - General Function Module, 49
- networkType\_m
  - T\_IDTPathConfiguration, 172
- networkType\_t
  - General Function Module, 49
- no\_argument
  - main.c, 196
- numberOfAppl\_m
  - T\_idtDrvDeviceConfig, 166
- numberOfApplXtal\_m
  - drvHdlr\_s, 169
- numberOfSonetGigEthPath\_m
  - T\_idtDrvDeviceConfig, 166
- numberOfXtal
  - globalArgs\_t, 150
- OUTPUTIF\_AMI
  - Output Function Module, 80
- OUTPUTIF\_AMI\_MASK
  - Output.c, 298
- OUTPUTIF\_CMOS
  - Output Function Module, 80
- OUTPUTIF\_CMOS\_MASK
  - Output.c, 298
- OUTPUTIF\_LVDS
  - Output Function Module, 80
- OUTPUTIF\_LVDS\_MASK
  - Output.c, 298
- OUTPUTIF\_MAX
  - Output Function Module, 80
- OUTPUTIF\_PECL
  - Output Function Module, 80
- OUTPUTIF\_PECL\_MASK
  - Output.c, 298
- OUTMUX\_ENTRY
  - outputFreq.c, 243
- offset\_m
  - globalArgs\_t, 150
- OperDpllMode\_FreeRun
  - DPLL Function Module, 20
- OperDpllMode\_HoldOver
  - DPLL Function Module, 20
- OperDpllMode\_Locked
  - DPLL Function Module, 20
- OperDpllMode\_LostPhase
  - DPLL Function Module, 20
- OperDpllMode\_PreLocked
  - DPLL Function Module, 20
- OperDpllMode\_PreLocked2
  - DPLL Function Module, 20
- optional\_argument
  - main.c, 196
- OutFreq\_1
  - outputFreq.h, 229
- OutFreq\_10000k
  - outputFreq.h, 230
- OutFreq\_12288k
  - outputFreq.h, 229
- OutFreq\_12352k
  - outputFreq.h, 229
- OutFreq\_124416k
  - outputFreq.h, 230
- OutFreq\_125000k
  - outputFreq.h, 230
- OutFreq\_125000k\_66\_64
  - outputFreq.h, 230
- OutFreq\_13000k
  - outputFreq.h, 229
- OutFreq\_15360k
  - outputFreq.h, 229
- OutFreq\_1544k
  - outputFreq.h, 229
- OutFreq\_155520k
  - outputFreq.h, 230
- OutFreq\_156250k
  - outputFreq.h, 230
- OutFreq\_156250k\_66\_64
  - outputFreq.h, 230
- OutFreq\_16384k

- outputFreq.h, [229](#)
- OutFreq\_18528k
  - outputFreq.h, [229](#)
- OutFreq\_19440k
  - outputFreq.h, [230](#)
- OutFreq\_20000k
  - outputFreq.h, [229](#)
- OutFreq\_2048k
  - outputFreq.h, [229](#)
- OutFreq\_24576k
  - outputFreq.h, [229](#)
- OutFreq\_24704k
  - outputFreq.h, [229](#)
- OutFreq\_24883\_DOT\_2k
  - outputFreq.h, [230](#)
- OutFreq\_25000k
  - outputFreq.h, [230](#)
- OutFreq\_25000k\_66\_64
  - outputFreq.h, [230](#)
- OutFreq\_25920k
  - outputFreq.h, [230](#)
- OutFreq\_26000k
  - outputFreq.h, [229](#)
- OutFreq\_2k
  - outputFreq.h, [229](#)
- OutFreq\_30720k
  - outputFreq.h, [229](#)
- OutFreq\_3088k
  - outputFreq.h, [229](#)
- OutFreq\_311040k
  - outputFreq.h, [230](#)
- OutFreq\_312500k
  - outputFreq.h, [230](#)
- OutFreq\_312500k\_66\_64
  - outputFreq.h, [230](#)
- OutFreq\_32768k
  - outputFreq.h, [229](#)
- OutFreq\_34368k
  - outputFreq.h, [229](#)
- OutFreq\_37056k
  - outputFreq.h, [229](#)
- OutFreq\_3840k
  - outputFreq.h, [229](#)
- OutFreq\_38880k
  - outputFreq.h, [230](#)
- OutFreq\_400
  - outputFreq.h, [229](#)
- OutFreq\_40000k
  - outputFreq.h, [229](#)
- OutFreq\_4096k
  - outputFreq.h, [229](#)
- OutFreq\_44736k
  - outputFreq.h, [229](#)
- OutFreq\_4632k
  - outputFreq.h, [229](#)
- OutFreq\_5000k
  - outputFreq.h, [230](#)
- OutFreq\_51840k
  - outputFreq.h, [230](#)
- OutFreq\_6144k
  - outputFreq.h, [229](#)
- OutFreq\_6176k
  - outputFreq.h, [229](#)
- OutFreq\_622080k
  - outputFreq.h, [230](#)
- OutFreq\_625000k
  - outputFreq.h, [230](#)
- OutFreq\_625000k\_66\_64
  - outputFreq.h, [230](#)
- OutFreq\_6480k
  - outputFreq.h, [230](#)
- OutFreq\_64k
  - outputFreq.h, [229](#)
- OutFreq\_7680k
  - outputFreq.h, [229](#)
- OutFreq\_77760k
  - outputFreq.h, [230](#)
- OutFreq\_78125k
  - outputFreq.h, [230](#)
- OutFreq\_78125k\_66\_64
  - outputFreq.h, [230](#)
- OutFreq\_8192k
  - outputFreq.h, [229](#)
- OutFreq\_8k
  - outputFreq.h, [229](#)
- OutFreq\_9264k
  - outputFreq.h, [229](#)
- OutFreq\_Invalid
  - outputFreq.h, [229](#)
- OutFreq\_MAX
  - outputFreq.h, [230](#)
- outFreqLookupIdx\_Dpll12E1
  - outputFreqString.c, [246](#)
- outFreqLookupIdx\_Dpll16E1
  - outputFreqString.c, [246](#)
- outFreqLookupIdx\_Dpll16T1
  - outputFreqString.c, [246](#)
- outFreqLookupIdx\_Dpll24T1
  - outputFreqString.c, [246](#)
- outFreqLookupIdx\_Dpll77p76
  - outputFreqString.c, [246](#)
- outFreqLookupIdx\_DpllE3
  - outputFreqString.c, [246](#)
- outFreqLookupIdx\_DpllEth
  - outputFreqString.c, [246](#)
- outFreqLookupIdx\_DpllGps
  - outputFreqString.c, [246](#)
- outFreqLookupIdx\_DpllGsm
  - outputFreqString.c, [246](#)
- outFreqLookupIdx\_DpllObsai
  - outputFreqString.c, [246](#)
- outFreqLookupIdx\_DpllT3
  - outputFreqString.c, [246](#)
- outFreqLookupIdx\_MAX
  - outputFreqString.c, [246](#)
- outFreqLookupIdx\_SonetGigEthEth

- outputFreqString.c, 246
- outFreqLookupIdx\_SonetGigEthFec
  - outputFreqString.c, 246
- outFreqLookupIdx\_SonetGigEthSonet
  - outputFreqString.c, 246
- outFreq\_m
  - globalArgs\_t, 150
- outPutApplConfig
  - T\_idtDrvDeviceConfig, 166
- outSyntXlate\_ma
  - T\_idtDrvDeviceConfig, 166
- Output Function Module, 78
  - frameSync\_FRSYNC, 80
  - frameSync\_MAX, 80
  - frameSync\_MFRSYNC, 80
  - IDTOutput\_DisableAllIntOutput, 80
  - IDTOutput\_DisableApplInput, 81
  - IDTOutput\_GetFrameSync, 81
  - IDTOutput\_GetFrameSyncPulseEdge, 81
  - IDTOutput\_GetOutputInterface, 81
  - IDTOutput\_GetOutputConfig, 82
  - IDTOutput\_GetOutputEnable, 82
  - IDTOutput\_GetOutputInvert, 82
  - IDTOutput\_SetFrameSync, 82
  - IDTOutput\_SetFrameSyncPulseEdge, 83
  - IDTOutput\_SetOutputInterface, 83
  - IDTOutput\_SetOutputConfig, 83
  - IDTOutput\_SetOutputEnable, 83
  - IDTOutput\_SetOutputInvert, 84
  - OUTPUTIF\_AMI, 80
  - OUTPUTIF\_CMOS, 80
  - OUTPUTIF\_LVDS, 80
  - OUTPUTIF\_MAX, 80
  - OUTPUTIF\_PECL, 80
  - OutputPath\_12E1\_E3\_T3, 80
  - OutputPath\_16E1\_16T1, 80
  - OutputPath\_24T1, 80
  - OutputPath\_7776kHz, 80
  - OutputPath\_Eth, 80
  - OutputPath\_GPS, 80
  - OutputPath\_GSM\_16E1\_16T1, 80
  - OutputPath\_MAX, 80
  - OutputPath\_OBSAI, 80
  - OutputPath\_SonetGigEth, 80
  - T\_frameSync, 79
  - T\_outputInterface, 80
  - T\_outputPathType, 80
- Output.c
  - OUTPUTIF\_AMI\_MASK, 298
  - OUTPUTIF\_CMOS\_MASK, 298
  - OUTPUTIF\_LVDS\_MASK, 298
  - OUTPUTIF\_PECL\_MASK, 298
  - OutputPathBfVal\_MAX, 299
  - OutputPathBfVal\_NULL, 299
  - OutputPathBfVal\_T0Dpll\_12E1\_GPS\_E3\_T3, 299
  - OutputPathBfVal\_T0Dpll\_16E1\_16T1, 299
  - OutputPathBfVal\_T0Dpll\_7776kHz, 299
  - OutputPathBfVal\_T0Dpll\_Eth, 299
  - OutputPathBfVal\_T0Dpll\_GSM\_OBSAI\_16E1\_16T1, 299
  - OutputPathBfVal\_T0SonetGigEth, 299
  - OutputPathBfVal\_T4Dpll\_12E1\_24T1\_E3\_T3, 299
  - OutputPathBfVal\_T4Dpll\_16E1\_16T1, 299
  - OutputPathBfVal\_T4Dpll\_7776kHz, 299
  - OutputPathBfVal\_T4Dpll\_Eth, 299
  - OutputPathBfVal\_T4Dpll\_GSM\_GPS\_16E1\_16T1, 299
  - OutputPathBfVal\_T4SonetGigEth, 299
- Output.c
  - T\_idtOutputPathBfVal, 298
- outputFreq.h
  - OutFreq\_1, 229
  - OutFreq\_10000k, 230
  - OutFreq\_12288k, 229
  - OutFreq\_12352k, 229
  - OutFreq\_124416k, 230
  - OutFreq\_125000k, 230
  - OutFreq\_125000k\_66\_64, 230
  - OutFreq\_13000k, 229
  - OutFreq\_15360k, 229
  - OutFreq\_1544k, 229
  - OutFreq\_155520k, 230
  - OutFreq\_156250k, 230
  - OutFreq\_156250k\_66\_64, 230
  - OutFreq\_16384k, 229
  - OutFreq\_18528k, 229
  - OutFreq\_19440k, 230
  - OutFreq\_20000k, 229
  - OutFreq\_2048k, 229
  - OutFreq\_24576k, 229
  - OutFreq\_24704k, 229
  - OutFreq\_24883\_DOT\_2k, 230
  - OutFreq\_25000k, 230
  - OutFreq\_25000k\_66\_64, 230
  - OutFreq\_25920k, 230
  - OutFreq\_26000k, 229
  - OutFreq\_2k, 229
  - OutFreq\_30720k, 229
  - OutFreq\_3088k, 229
  - OutFreq\_311040k, 230
  - OutFreq\_312500k, 230
  - OutFreq\_312500k\_66\_64, 230
  - OutFreq\_32768k, 229
  - OutFreq\_34368k, 229
  - OutFreq\_37056k, 229
  - OutFreq\_3840k, 229
  - OutFreq\_38880k, 230
  - OutFreq\_400, 229
  - OutFreq\_40000k, 229
  - OutFreq\_4096k, 229
  - OutFreq\_44736k, 229
  - OutFreq\_4632k, 229
  - OutFreq\_5000k, 230
  - OutFreq\_51840k, 230
  - OutFreq\_6144k, 229
  - OutFreq\_6176k, 229

- OutFreq\_622080k, [230](#)
- OutFreq\_625000k, [230](#)
- OutFreq\_625000k\_66\_64, [230](#)
- OutFreq\_6480k, [230](#)
- OutFreq\_64k, [229](#)
- OutFreq\_7680k, [229](#)
- OutFreq\_77760k, [230](#)
- OutFreq\_78125k, [230](#)
- OutFreq\_78125k\_66\_64, [230](#)
- OutFreq\_8192k, [229](#)
- OutFreq\_8k, [229](#)
- OutFreq\_9264k, [229](#)
- OutFreq\_Invalid, [229](#)
- OutFreq\_MAX, [230](#)
- sonetGigeMode\_AllFromDpll1, [230](#)
- sonetGigeMode\_AllFromDpll2, [230](#)
- sonetGigeMode\_MAX, [230](#)
- sonetGigeMode\_Matching, [230](#)
- outputFreq\_priv.h
  - Selector\_Any, [235](#)
  - Selector\_MAX, [235](#)
  - Selector\_Network, [235](#)
  - Selector\_SonetGigEth, [235](#)
  - Selector\_T0DpllSelA, [235](#)
  - Selector\_T0DpllSelB, [235](#)
  - Selector\_T4DpllSelA, [235](#)
  - Selector\_T4DpllSelB, [235](#)
- outputFreqString.c
  - outFreqLookupIdx\_Dpll12E1, [246](#)
  - outFreqLookupIdx\_Dpll16E1, [246](#)
  - outFreqLookupIdx\_Dpll16T1, [246](#)
  - outFreqLookupIdx\_Dpll24T1, [246](#)
  - outFreqLookupIdx\_Dpll77p76, [246](#)
  - outFreqLookupIdx\_DpllE3, [246](#)
  - outFreqLookupIdx\_DpllEth, [246](#)
  - outFreqLookupIdx\_DpllGps, [246](#)
  - outFreqLookupIdx\_DpllGsm, [246](#)
  - outFreqLookupIdx\_DpllObsai, [246](#)
  - outFreqLookupIdx\_DpllT3, [246](#)
  - outFreqLookupIdx\_MAX, [246](#)
  - outFreqLookupIdx\_SonetGigEthEth, [246](#)
  - outFreqLookupIdx\_SonetGigEthFec, [246](#)
  - outFreqLookupIdx\_SonetGigEthSonet, [246](#)
- OutputPath\_12E1\_E3\_T3
  - Output Function Module, [80](#)
- OutputPath\_16E1\_16T1
  - Output Function Module, [80](#)
- OutputPath\_24T1
  - Output Function Module, [80](#)
- OutputPath\_7776kHz
  - Output Function Module, [80](#)
- OutputPath\_Eth
  - Output Function Module, [80](#)
- OutputPath\_GPS
  - Output Function Module, [80](#)
- OutputPath\_GSM\_16E1\_16T1
  - Output Function Module, [80](#)
- OutputPath\_MAX
  - Output Function Module, [80](#)
- OutputPath\_OBSAI
  - Output Function Module, [80](#)
- OutputPath\_SonetGigEth
  - Output Function Module, [80](#)
- OutputPathBfVal\_MAX
  - Output.c, [299](#)
- OutputPathBfVal\_NULL
  - Output.c, [299](#)
- OutputPathBfVal\_T0Dpll\_12E1\_GPS\_E3\_T3
  - Output.c, [299](#)
- OutputPathBfVal\_T0Dpll\_16E1\_16T1
  - Output.c, [299](#)
- OutputPathBfVal\_T0Dpll\_7776kHz
  - Output.c, [299](#)
- OutputPathBfVal\_T0Dpll\_Eth
  - Output.c, [299](#)
- OutputPathBfVal\_T0Dpll\_GSM\_OBSAI\_16E1\_16T1
  - Output.c, [299](#)
- OutputPathBfVal\_T0SonetGigEth
  - Output.c, [299](#)
- OutputPathBfVal\_T4Dpll\_12E1\_24T1\_E3\_T3
  - Output.c, [299](#)
- OutputPathBfVal\_T4Dpll\_16E1\_16T1
  - Output.c, [299](#)
- OutputPathBfVal\_T4Dpll\_7776kHz
  - Output.c, [299](#)
- OutputPathBfVal\_T4Dpll\_Eth
  - Output.c, [299](#)
- OutputPathBfVal\_T4Dpll\_GSM\_GPS\_16E1\_16T1
  - Output.c, [299](#)
- OutputPathBfVal\_T4SonetGigEth
  - Output.c, [299](#)
- outputConfig
  - T\_idtApplConfig, [158](#)
- outputDiv
  - T\_idtApplConfig, [158](#)
  - T\_idtApplConfigPerOutput, [160](#)
- outputDivConfig0
  - T\_idtApplConfig::outputDiv\_s::qaDiv\_s, [155](#)
  - T\_idtApplConfig::outputDiv\_s::qbDiv\_s, [156](#)
- outputDivConfig1
  - T\_idtApplConfig::outputDiv\_s::qaDiv\_s, [155](#)
  - T\_idtApplConfig::outputDiv\_s::qbDiv\_s, [156](#)
- outputDivider\_m
  - pathSelectorConfig\_t, [152](#)
- outputExt2IntXlate\_ma
  - T\_idtDrvDeviceConfig, [166](#)
- outputFreq.c
  - IDTHIApi\_ClearProvisioning, [243](#)
  - IDTHIApi\_ConfigureOutput, [243](#)
  - IDTHIApi\_DisableAllOutputs, [244](#)
  - IDTHIApi\_RetrieveOutputPathMuxFromHardware, [244](#)
  - OUTMUX\_ENTRY, [243](#)
- outputFreq.h
  - IDTHIApi\_ClearProvisioning, [230](#)
  - IDTHIApi\_ConfigureOutput, [231](#)

- IDTHIApi\_DisableAllOutputs, [232](#)
  - IDTHIApi\_PrintInput, [232](#)
  - IDTHIApi\_PrintOutput, [232](#)
  - outputFrequency\_t, [229](#)
  - T\_sonetGigeMode, [230](#)
- outputFreq\_m
  - T\_idtSonetGigEthBitfield2PathConfig, [173](#)
- outputFreq\_priv.h
  - CHECK\_BIT, [234](#)
  - IDTHIApi\_ApplyFrequencies, [235](#)
  - IDTHIApi\_ConvertFromFreqListToFreqArray, [235](#)
  - IDTHIApi\_Frequency2String\_c, [237](#)
  - IDTHIApi\_IsFreqArrayInFreqList, [235](#)
  - IDTHIApi\_IsFrequencyInFreqList, [235](#)
  - IDTHIApi\_PrintAvailFrequencies, [236](#)
  - IDTHIApi\_PrintConfiguration, [236](#)
  - IDTHIApi\_PrintFrequencyList, [236](#)
  - IDTHIApi\_PrintOption, [236](#)
  - IDTHIApi\_PrintOutputHwConfig, [236](#)
  - IDTHIApi\_PrintSelectedOutputPath, [237](#)
  - pathSelectors\_t, [235](#)
  - SET\_BIT, [234](#)
  - validFreq\_t, [234](#)
- outputFreqString.c
  - IDTHIApi\_Frequency2String\_c, [248](#)
  - IDTHIApi\_PrintAvailFrequencies, [247](#)
  - IDTHIApi\_PrintConfiguration, [247](#)
  - IDTHIApi\_PrintFrequencyList, [247](#)
  - IDTHIApi\_PrintInput, [247](#)
  - IDTHIApi\_PrintInputHwConfig, [247](#)
  - IDTHIApi\_PrintOption, [247](#)
  - IDTHIApi\_PrintOutput, [247](#)
  - IDTHIApi\_PrintOutputHwConfig, [248](#)
  - IDTHIApi\_PrintSelectedOutputPath, [248](#)
  - T\_outFreqLookupIdx, [246](#)
- outputFreqUtil.c
  - IDTHIApi\_ApplyFrequencies, [249](#)
  - IDTHIApi\_ConvertFromFreqListToFreqArray, [249](#)
  - IDTHIApi\_IsFreqArrayInFreqList, [250](#)
  - IDTHIApi\_IsFrequencyInFreqList, [250](#)
- outputFreqValidFunc\_m
  - T\_idtDrvDeviceConfig, [166](#)
- outputFrequency\_t
  - outputFreq.h, [229](#)
- outputPathSelector\_m
  - pathSelectorConfig\_t, [152](#)
- outputSigType
  - T\_idtApIConfig::outputConfig\_s::qaConfig\_s, [154](#)
  - T\_idtApIConfig::outputConfig\_s::qbConfig\_s, [156](#)
- page1Registers
  - ReadWrite\_sim.c, [224](#)
- pathSelectorConfig\_t, [152](#)
  - instance\_m, [152](#)
  - outputDivider\_m, [152](#)
  - outputPathSelector\_m, [152](#)
  - selector\_m, [152](#)
  - value\_m, [153](#)
- pathSelectorIndex\_t, [153](#)
  - size\_m, [153](#)
  - start\_m, [153](#)
- pathSelectors\_t
  - outputFreq\_priv.h, [235](#)
- phaseSlope\_MAX
  - DPLL Function Module, [21](#)
- phaseSlope\_gr1244st3
  - DPLL Function Module, [21](#)
- phaseSlope\_gr1244st3e
  - DPLL Function Module, [21](#)
- phaseSlope\_gr813
  - DPLL Function Module, [21](#)
- phaseSlope\_noLimit
  - DPLL Function Module, [21](#)
- phaseTimeoutMultiplier\_16
  - General Function Module, [50](#)
- phaseTimeoutMultiplier\_2
  - General Function Module, [49](#)
- phaseTimeoutMultiplier\_4
  - General Function Module, [49](#)
- phaseTimeoutMultiplier\_8
  - General Function Module, [49](#)
- phaseTimeoutMultiplier\_MAX
  - General Function Module, [50](#)
- phaseSlopeLimiting\_ma
  - T\_idtDrvDeviceConfig, [166](#)
- phaseTimeoutMultiplier\_t
  - General Function Module, [49](#)
- pllExt2IntXlate\_ma
  - T\_idtDrvDeviceConfig, [166](#)
- pllInt2ExtXlate\_ma
  - T\_idtDrvDeviceConfig, [167](#)
- populatedApI\_ma
  - T\_idtDrvDeviceConfig, [167](#)
- PpsPhasePostion\_0degree
  - DPLL Function Module, [19](#)
- PpsPhasePostion\_100ns
  - DPLL Function Module, [19](#)
- PpsPhasePostion\_50ns
  - DPLL Function Module, [19](#)
- PpsPhasePostion\_MAX
  - DPLL Function Module, [19](#)
- PpsPulseWidth\_0pt5sec
  - DPLL Function Module, [19](#)
- PpsPulseWidth\_100us
  - DPLL Function Module, [19](#)
- PpsPulseWidth\_10ns
  - DPLL Function Module, [19](#)
- PpsPulseWidth\_1pt2ms
  - DPLL Function Module, [19](#)
- PpsPulseWidth\_1us
  - DPLL Function Module, [19](#)
- PpsPulseWidth\_200ns
  - DPLL Function Module, [19](#)
- PpsPulseWidth\_200us
  - DPLL Function Module, [19](#)
- PpsPulseWidth\_20us
  - DPLL Function Module, [19](#)

- PpsPulseWidth\_400ns
  - DPLL Function Module, [19](#)
- PpsPulseWidth\_50us
  - DPLL Function Module, [19](#)
- PpsPulseWidth\_800ns
  - DPLL Function Module, [19](#)
- PpsPulseWidth\_800us
  - DPLL Function Module, [19](#)
- PpsPulseWidth\_MAX
  - DPLL Function Module, [19](#)
- preDiv
  - T\_idtApIConfig, [158](#)
  - T\_idtApIConfigPerOutput, [160](#)
- preDivConfig0
  - T\_idtApIConfig::preDiv\_s, [154](#)
- preDivConfig1
  - T\_idtApIConfig::preDiv\_s, [154](#)
- printHelp
  - main.c, [197](#)
- processCmdLineArguments
  - main.c, [197](#)
- profiles.c
  - IDTHIApi\_g813Option1, [252](#)
  - IDTHIApi\_g813Option2, [252](#)
  - IDTHIApi\_g8262EecOption1, [252](#)
  - IDTHIApi\_g8262EecOption2, [252](#)
  - IDTHIApi\_gr1244Stratum3, [252](#)
  - IDTHIApi\_gr253SonetStratum3, [252](#)
  - IDTHIApi\_pps, [253](#)
  - IDTHIApi\_smc, [253](#)
  - IDTHIApi\_wideband, [253](#)
- profiles.h
  - IDTHIApi\_g813Option1, [238](#)
  - IDTHIApi\_g813Option2, [238](#)
  - IDTHIApi\_g8262EecOption1, [238](#)
  - IDTHIApi\_g8262EecOption2, [239](#)
  - IDTHIApi\_gr1244Stratum3, [239](#)
  - IDTHIApi\_gr253SonetStratum3, [239](#)
  - IDTHIApi\_pps, [239](#)
  - IDTHIApi\_smc, [239](#)
  - IDTHIApi\_wideband, [240](#)
- qaConfig
  - T\_idtApIConfig::outputConfig\_s, [151](#)
- qaDiv
  - T\_idtApIConfig::outputDiv\_s, [152](#)
- qbConfig
  - T\_idtApIConfig::outputConfig\_s, [151](#)
- qbDiv
  - T\_idtApIConfig::outputDiv\_s, [152](#)
- REG\_BUCKET\_SIZE\_0\_CNFG
  - Register definitions, [94](#)
- REG\_BUCKET\_SIZE\_1\_CNFG
  - Register definitions, [95](#)
- REG\_BUCKET\_SIZE\_2\_CNFG
  - Register definitions, [95](#)
- REG\_BUCKET\_SIZE\_3\_CNFG
  - Register definitions, [95](#)
- REG\_CLK\_SEL
  - Register definitions, [115](#)
- REG\_CONFIG\_QA
  - Register definitions, [115](#)
- REG\_CONFIG\_QB
  - Register definitions, [115](#)
- REG\_CONTROL
  - Register definitions, [115](#)
- REG\_CURRENT\_DPLL\_PHASE\_HIGH\_STS
  - Register definitions, [96](#)
- REG\_CURRENT\_DPLL\_PHASE\_LOW\_STS
  - Register definitions, [96](#)
- REG\_CURRENT\_FREQ\_HIGH\_STS
  - Register definitions, [96](#)
- REG\_CURRENT\_FREQ\_LOW\_STS
  - Register definitions, [96](#)
- REG\_CURRENT\_FREQ\_MID\_STS
  - Register definitions, [96](#)
- REG\_DECAY\_RATE\_0\_CNFG
  - Register definitions, [95](#)
- REG\_DECAY\_RATE\_1\_CNFG
  - Register definitions, [95](#)
- REG\_DECAY\_RATE\_2\_CNFG
  - Register definitions, [95](#)
- REG\_DECAY\_RATE\_3\_CNFG
  - Register definitions, [95](#)
- REG\_DFS\_OFF\_CNFG
  - Register definitions, [96](#)
- REG\_DIFFERENTIAL\_IN\_OUT\_CNFG
  - Register definitions, [94](#)
- REG\_DPLL\_FREQ\_HARD\_LIMIT\_LOW\_CNFG
  - Register definitions, [96](#)
- REG\_DPLL\_FREQ\_HARD\_LIMIT\_MID\_CNFG
  - Register definitions, [96](#)
- REG\_DPLL\_FREQ\_SOFT\_LIMIT\_CNFG
  - Register definitions, [96](#)
- REG\_FR\_MFR\_SYNC\_CNFG
  - Register definitions, [96](#)
- REG\_FREQ\_MON\_FACTOR\_CNFG
  - Register definitions, [94](#)
- REG\_HARD\_FREQ\_MON\_THRESHOLD\_CNFG
  - Register definitions, [94](#)
- REG\_HOLDOVER\_FREQ\_HIGH\_CNFG
  - Register definitions, [96](#)
- REG\_HOLDOVER\_FREQ\_LOW\_CNFG
  - Register definitions, [96](#)
- REG\_HOLDOVER\_FREQ\_MID\_CNFG
  - Register definitions, [96](#)
- REG\_ID\_HIGH
  - Register definitions, [93](#)
- REG\_ID\_LOW
  - Register definitions, [93](#)
- REG\_IN10\_CNFG
  - Register definitions, [94](#)
- REG\_IN11\_CNFG
  - Register definitions, [94](#)
- REG\_IN11\_IN12\_SEL\_PRIORITY\_CNFG
  - Register definitions, [94](#)

- REG\_IN11\_IN12\_STS
  - Register definitions, [95](#)
- REG\_IN12\_CNFG
  - Register definitions, [94](#)
- REG\_IN13\_CNFG
  - Register definitions, [94](#)
- REG\_IN13\_IN14\_SEL\_PRIORITY\_CNFG
  - Register definitions, [94](#)
- REG\_IN13\_IN14\_STS
  - Register definitions, [95](#)
- REG\_IN14\_CNFG
  - Register definitions, [94](#)
- REG\_IN1\_CNFG
  - Register definitions, [94](#)
- REG\_IN1\_IN2\_SEL\_PRIORITY\_CNFG
  - Register definitions, [94](#)
- REG\_IN1\_IN2\_STS
  - Register definitions, [95](#)
- REG\_IN2\_CNFG
  - Register definitions, [94](#)
- REG\_IN3\_CNFG
  - Register definitions, [94](#)
- REG\_IN3\_IN4\_SEL\_PRIORITY\_CNFG
  - Register definitions, [94](#)
- REG\_IN3\_IN4\_STS
  - Register definitions, [95](#)
- REG\_IN4\_CNFG
  - Register definitions, [94](#)
- REG\_IN5\_CNFG
  - Register definitions, [94](#)
- REG\_IN5\_IN6\_HF\_DIV\_CNFG
  - Register definitions, [94](#)
- REG\_IN5\_IN6\_SEL\_PRIORITY\_CNFG
  - Register definitions, [94](#)
- REG\_IN5\_IN6\_STS
  - Register definitions, [95](#)
- REG\_IN6\_CNFG
  - Register definitions, [94](#)
- REG\_IN7\_CNFG
  - Register definitions, [94](#)
- REG\_IN7\_IN8\_SEL\_PRIORITY\_CNFG
  - Register definitions, [94](#)
- REG\_IN7\_IN8\_STS
  - Register definitions, [95](#)
- REG\_IN8\_CNFG
  - Register definitions, [94](#)
- REG\_IN9\_CNFG
  - Register definitions, [94](#)
- REG\_IN9\_IN10\_SEL\_PRIORITY\_CNFG
  - Register definitions, [94](#)
- REG\_IN9\_IN10\_STS
  - Register definitions, [95](#)
- REG\_IN\_FREQ\_READ\_CH\_CNFG
  - Register definitions, [95](#)
- REG\_IN\_FREQ\_READ\_STS
  - Register definitions, [95](#)
- REG\_INPUT\_MODE\_CNFG
  - Register definitions, [93](#)
- REG\_INPUT\_VALID1\_STS
  - Register definitions, [95](#)
- REG\_INPUT\_VALID2\_STS
  - Register definitions, [95](#)
- REG\_INTERRUPT\_CNFG
  - Register definitions, [94](#)
- REG\_INTERRUPTS1\_MASK\_CNFG
  - Register definitions, [94](#)
- REG\_INTERRUPTS1\_STS
  - Register definitions, [94](#)
- REG\_INTERRUPTS2\_MASK\_CNFG
  - Register definitions, [94](#)
- REG\_INTERRUPTS2\_STS
  - Register definitions, [94](#)
- REG\_INTERRUPTS3\_MASK\_CNFG
  - Register definitions, [94](#)
- REG\_INTERRUPTS3\_STS
  - Register definitions, [94](#)
- REG\_LOWER\_THRESHOLD\_0\_CNFG
  - Register definitions, [94](#)
- REG\_LOWER\_THRESHOLD\_1\_CNFG
  - Register definitions, [95](#)
- REG\_LOWER\_THRESHOLD\_2\_CNFG
  - Register definitions, [95](#)
- REG\_LOWER\_THRESHOLD\_3\_CNFG
  - Register definitions, [95](#)
- REG\_M0\_LSB
  - Register definitions, [115](#)
- REG\_M0\_MSB
  - Register definitions, [115](#)
- REG\_M1\_LSB
  - Register definitions, [115](#)
- REG\_M1\_MSB
  - Register definitions, [115](#)
- REG\_MON\_SW\_HS\_CNFG
  - Register definitions, [94](#)
- REG\_MPU\_PIN\_STS
  - Register definitions, [93](#)
- REG\_MPU\_SEL\_CNFG
  - Register definitions, [96](#)
- REG\_MS\_SL\_CTRL\_CNFG
  - Register definitions, [94](#)
- REG\_NOMINAL\_FREQ\_HIGH\_CNFG
  - Register definitions, [93](#)
- REG\_NOMINAL\_FREQ\_LOW\_CNFG
  - Register definitions, [93](#)
- REG\_NOMINAL\_FREQ\_MID\_CNFG
  - Register definitions, [93](#)
- REG\_ODBSEL0
  - Register definitions, [115](#)
- REG\_ODBSEL1
  - Register definitions, [115](#)
- REG\_ODBSELB0
  - Register definitions, [115](#)
- REG\_ODBSELB1
  - Register definitions, [115](#)
- REG\_OPERATING\_STS
  - Register definitions, [95](#)

- REG\_OUT1\_FREQ\_CNFG
  - Register definitions, [96](#)
- REG\_OUT2\_FREQ\_CNFG
  - Register definitions, [96](#)
- REG\_OUT3\_FREQ\_CNFG
  - Register definitions, [96](#)
- REG\_OUT4\_FREQ\_CNFG
  - Register definitions, [96](#)
- REG\_OUT5\_FREQ\_CNFG
  - Register definitions, [96](#)
- REG\_OUT6\_FREQ\_CNFG
  - Register definitions, [96](#)
- REG\_OUT7\_FREQ\_CNFG
  - Register definitions, [96](#)
- REG\_OUT8\_FREQ\_CNFG
  - Register definitions, [96](#)
- REG\_OUT9\_FREQ\_CNFG
  - Register definitions, [96](#)
- REG\_PAGE\_POINTER\_CNFG
  - Register definitions, [94](#)
- REG\_PDSEL0\_LSB
  - Register definitions, [115](#)
- REG\_PDSEL0\_MSB
  - Register definitions, [115](#)
- REG\_PDSEL1\_LSB
  - Register definitions, [115](#)
- REG\_PDSEL1\_MSB
  - Register definitions, [115](#)
- REG\_PH\_SLOPE\_CNFG
  - Register definitions, [96](#)
- REG\_PHASE\_ALARM\_TIME\_OUT\_CNFG
  - Register definitions, [93](#)
- REG\_PHASE\_LOSS\_COARSE\_LIMIT\_CNFG
  - Register definitions, [95](#)
- REG\_PHASE\_LOSS\_FINE\_LIMIT\_CNFG
  - Register definitions, [95](#)
- REG\_PHASE\_MON\_CNFG
  - Register definitions, [96](#)
- REG\_PHASE\_OFFSET\_HIGH\_CNFG
  - Register definitions, [96](#)
- REG\_PHASE\_OFFSET\_LOW\_CNFG
  - Register definitions, [96](#)
- REG\_PRE\_DIV\_CH\_CNFG
  - Register definitions, [94](#)
- REG\_PRE\_DIVN\_HIGH\_CNFG
  - Register definitions, [94](#)
- REG\_PRE\_DIVN\_LOW\_CNFG
  - Register definitions, [94](#)
- REG\_PRIORITY\_TABLE1\_STS
  - Register definitions, [95](#)
- REG\_PRIORITY\_TABLE2\_STS
  - Register definitions, [95](#)
- REG\_PROTECTION\_CNFG
  - Register definitions, [96](#)
- REG\_REMOTE\_INPUT\_VALID1\_CNFG
  - Register definitions, [95](#)
- REG\_REMOTE\_INPUT\_VALID2\_CNFG
  - Register definitions, [95](#)
- REG\_SOFT\_FREQ\_MON\_THRESHOLD\_CNFG
  - Register definitions, [94](#)
- REG\_SYNC\_MONITOR\_CNFG
  - Register definitions, [96](#)
- REG\_SYNC\_PHASE\_CNFG
  - Register definitions, [96](#)
- REG\_T0\_BW\_OVERSHOOT\_CNFG
  - Register definitions, [95](#)
- REG\_T0\_DPLL\_ACQ\_BW\_DAMPING\_CNFG
  - Register definitions, [95](#)
- REG\_T0\_DPLL\_APLL\_PATH\_CNFG
  - Register definitions, [95](#)
- REG\_T0\_DPLL\_LOCKED\_BW\_DAMPING\_CNFG
  - Register definitions, [95](#)
- REG\_T0\_DPLL\_START\_BW\_DAMPING\_CNFG
  - Register definitions, [95](#)
- REG\_T0\_HOLDOVER\_MODE\_CNFG
  - Register definitions, [95](#)
- REG\_T0\_ICP\_CTRL\_CNFG
  - Register definitions, [96](#)
- REG\_T0\_INPUT\_SEL\_CNFG
  - Register definitions, [95](#)
- REG\_T0\_OPERATION\_MODE\_CNFG
  - Register definitions, [95](#)
- REG\_T0\_PPS\_CNFG
  - Register definitions, [96](#)
- REG\_T4\_DPLL\_APLL\_PATH\_CNFG
  - Register definitions, [96](#)
- REG\_T4\_DPLL\_LOCKED\_BW\_DAMPING\_CNFG
  - Register definitions, [96](#)
- REG\_T4\_ICP\_CTRL\_CNFG
  - Register definitions, [96](#)
- REG\_T4\_INPUT\_SEL\_CNFG
  - Register definitions, [95](#)
- REG\_T4\_OPERATION\_MODE\_CNFG
  - Register definitions, [95](#)
- REG\_T4\_PPS\_CNFG
  - Register definitions, [96](#)
- REG\_T4\_T0\_SEL\_CNFG
  - Register definitions, [93](#)
- REG\_UPPER\_THRESHOLD\_0\_CNFG
  - Register definitions, [94](#)
- REG\_UPPER\_THRESHOLD\_1\_CNFG
  - Register definitions, [95](#)
- REG\_UPPER\_THRESHOLD\_2\_CNFG
  - Register definitions, [95](#)
- REG\_UPPER\_THRESHOLD\_3\_CNFG
  - Register definitions, [95](#)
- REG\_VCXO\_SEL
  - Register definitions, [115](#)
- REGMAP\_APLL\_MAX
  - Register definitions, [117](#)
- REGMAP\_MAX
  - Register definitions, [115](#)
- RByte\_Sim
  - ReadWrite\_sim.c, [224](#)
  - ReadWrite\_sim.h, [225](#)
- ReadByte\_SimWrapper

- access.c, 191
- readFunc\_m
  - T\_idtDrvAccess, 164
- ReadWrite\_sim.c
  - DRV\_MEMMAP\_SIZE, 223
  - DelInit\_Sim, 224
  - dpllOverlayRegisters, 224
  - Init\_Sim, 224
  - page1Registers, 224
  - RDByte\_Sim, 224
  - simDebugFlag, 224
  - simulatedMemMap, 225
  - WRByte\_Sim, 224
- ReadWrite\_sim.h
  - DelInit\_Sim, 225
  - Init\_Sim, 225
  - RDByte\_Sim, 225
  - WRByte\_Sim, 225
- regOffset\_m
  - globalArgs\_t, 150
- regValue\_m
  - globalArgs\_t, 150
- Register definitions, 85
  - BF\_2K\_8K\_PUL\_POSITION\_LSB, 113
  - BF\_2K\_8K\_PUL\_POSITION\_MASK, 113
  - BF\_2K\_8K\_PUL\_POSITION\_SIZE, 113
  - BF\_2K\_EN\_LSB, 113
  - BF\_2K\_EN\_MASK, 113
  - BF\_2K\_EN\_SIZE, 113
  - BF\_2K\_INV\_LSB, 113
  - BF\_2K\_INV\_MASK, 113
  - BF\_2K\_INV\_SIZE, 113
  - BF\_2K\_PUL\_LSB, 113
  - BF\_2K\_PUL\_MASK, 113
  - BF\_2K\_PUL\_SIZE, 113
  - BF\_400HZ\_OUT\_SEL\_LSB, 112
  - BF\_400HZ\_OUT\_SEL\_MASK, 112
  - BF\_400HZ\_OUT\_SEL\_SIZE, 112
  - BF\_400HZ\_SEL\_LSB, 100
  - BF\_400HZ\_SEL\_MASK, 100
  - BF\_400HZ\_SEL\_SIZE, 100
  - BF\_8K\_EN\_LSB, 113
  - BF\_8K\_EN\_MASK, 113
  - BF\_8K\_EN\_SIZE, 113
  - BF\_8K\_INV\_LSB, 113
  - BF\_8K\_INV\_MASK, 113
  - BF\_8K\_INV\_SIZE, 113
  - BF\_8K\_PUL\_LSB, 113
  - BF\_8K\_PUL\_MASK, 113
  - BF\_8K\_PUL\_SIZE, 113
  - BF\_AMI1\_LOS\_LSB, 100
  - BF\_AMI1\_LOS\_MASK, 100
  - BF\_AMI1\_LOS\_SIZE, 100
  - BF\_AMI1\_VIOL\_LSB, 100
  - BF\_AMI1\_VIOL\_MASK, 100
  - BF\_AMI1\_VIOL\_SIZE, 100
  - BF\_AMI2\_LOS\_LSB, 100
  - BF\_AMI2\_LOS\_MASK, 100
  - BF\_AMI2\_LOS\_SIZE, 100
  - BF\_AMI2\_VIOL\_LSB, 100
  - BF\_AMI2\_VIOL\_MASK, 100
  - BF\_AMI2\_VIOL\_SIZE, 100
  - BF\_AMI\_OUT\_DUTY\_LSB, 112
  - BF\_AMI\_OUT\_DUTY\_MASK, 112
  - BF\_AMI\_OUT\_DUTY\_SIZE, 112
  - BF\_AUTO\_AVG\_LSB, 109
  - BF\_AUTO\_AVG\_MASK, 109
  - BF\_AUTO\_AVG\_SIZE, 109
  - BF\_AUTO\_BW\_SEL\_LSB, 108
  - BF\_AUTO\_BW\_SEL\_MASK, 108
  - BF\_AUTO\_BW\_SEL\_SIZE, 108
  - BF\_AUTO\_EXT\_SYNC\_EN\_LSB, 97
  - BF\_AUTO\_EXT\_SYNC\_EN\_MASK, 97
  - BF\_AUTO\_EXT\_SYNC\_EN\_SIZE, 97
  - BF\_BUCKET\_SEL\_LSB, 100
  - BF\_BUCKET\_SEL\_MASK, 100
  - BF\_BUCKET\_SEL\_SIZE, 100
  - BF\_BUCKET\_SIZE\_DATA\_LSB, 103
  - BF\_BUCKET\_SIZE\_DATA\_MASK, 103
  - BF\_BUCKET\_SIZE\_DATA\_SIZE, 102
  - BF\_BYPASS\_LSB, 116
  - BF\_BYPASS\_MASK, 116
  - BF\_BYPASS\_SIZE, 116
  - BF\_CLK\_SEL\_LSB, 115
  - BF\_CLK\_SEL\_MASK, 115
  - BF\_CLK\_SEL\_SIZE, 115
  - BF\_COARSE\_PH\_LOS\_LIMT\_EN\_LSB, 109
  - BF\_COARSE\_PH\_LOS\_LIMT\_EN\_MASK, 109
  - BF\_COARSE\_PH\_LOS\_LIMT\_EN\_SIZE, 109
  - BF\_CONFIG\_LSB, 115
  - BF\_CONFIG\_MASK, 115
  - BF\_CONFIG\_SIZE, 115
  - BF\_CURRENT\_DPLL\_FREQ\_A\_LSB, 110
  - BF\_CURRENT\_DPLL\_FREQ\_A\_MASK, 110
  - BF\_CURRENT\_DPLL\_FREQ\_A\_SIZE, 110
  - BF\_CURRENT\_DPLL\_FREQ\_B\_LSB, 110
  - BF\_CURRENT\_DPLL\_FREQ\_B\_MASK, 110
  - BF\_CURRENT\_DPLL\_FREQ\_B\_SIZE, 110
  - BF\_CURRENT\_DPLL\_FREQ\_C\_LSB, 110
  - BF\_CURRENT\_DPLL\_FREQ\_C\_MASK, 110
  - BF\_CURRENT\_DPLL\_FREQ\_C\_SIZE, 110
  - BF\_CURRENT\_PH\_DATA\_A\_LSB, 110
  - BF\_CURRENT\_PH\_DATA\_A\_MASK, 110
  - BF\_CURRENT\_PH\_DATA\_A\_SIZE, 110
  - BF\_CURRENT\_PH\_DATA\_B\_LSB, 111
  - BF\_CURRENT\_PH\_DATA\_B\_MASK, 110
  - BF\_CURRENT\_PH\_DATA\_B\_SIZE, 110
  - BF\_CURRENTLY\_SELECTED\_INPUTS\_LSB, 107
  - BF\_CURRENTLY\_SELECTED\_INPUTS\_MASK, 107
  - BF\_CURRENTLY\_SELECTED\_INPUTS\_SIZE, 107
  - BF\_DECAY\_RATE\_DATA\_LSB, 103
  - BF\_DECAY\_RATE\_DATA\_MASK, 103
  - BF\_DECAY\_RATE\_DATA\_SIZE, 103

- BF\_DIRECT\_DIV\_LSB, [101](#)
- BF\_DIRECT\_DIV\_MASK, [100](#)
- BF\_DIRECT\_DIV\_SIZE, [100](#)
- BF\_DPLL\_FREQ\_HARD\_LIMIT\_A\_LSB, [110](#)
- BF\_DPLL\_FREQ\_HARD\_LIMIT\_A\_MASK, [110](#)
- BF\_DPLL\_FREQ\_HARD\_LIMIT\_A\_SIZE, [110](#)
- BF\_DPLL\_FREQ\_HARD\_LIMIT\_B\_LSB, [110](#)
- BF\_DPLL\_FREQ\_HARD\_LIMIT\_B\_MASK, [110](#)
- BF\_DPLL\_FREQ\_HARD\_LIMIT\_B\_SIZE, [110](#)
- BF\_DPLL\_FREQ\_SOFT\_LIMIT\_LSB, [110](#)
- BF\_DPLL\_FREQ\_SOFT\_LIMIT\_MASK, [110](#)
- BF\_DPLL\_FREQ\_SOFT\_LIMIT\_SIZE, [110](#)
- BF\_EX\_SYNC\_ALARM\_LSB, [100](#)
- BF\_EX\_SYNC\_ALARM\_MASK, [100](#)
- BF\_EX\_SYNC\_ALARM\_MON\_LSB, [107](#)
- BF\_EX\_SYNC\_ALARM\_MON\_MASK, [107](#)
- BF\_EX\_SYNC\_ALARM\_MON\_SIZE, [107](#)
- BF\_EX\_SYNC\_ALARM\_SIZE, [100](#)
- BF\_EXT\_SW\_LSB, [98](#)
- BF\_EXT\_SW\_MASK, [98](#)
- BF\_EXT\_SW\_SIZE, [98](#)
- BF\_EXT\_SYNC\_EN\_LSB, [97](#)
- BF\_EXT\_SYNC\_EN\_MASK, [97](#)
- BF\_EXT\_SYNC\_EN\_SIZE, [97](#)
- BF\_FAST\_AVG\_LSB, [109](#)
- BF\_FAST\_AVG\_MASK, [109](#)
- BF\_FAST\_AVG\_SIZE, [109](#)
- BF\_FAST\_LOS\_SW\_LSB, [109](#)
- BF\_FAST\_LOS\_SW\_MASK, [109](#)
- BF\_FAST\_LOS\_SW\_SIZE, [109](#)
- BF\_FINE\_PH\_LOS\_LIMIT\_EN\_LSB, [109](#)
- BF\_FINE\_PH\_LOS\_LIMIT\_EN\_MASK, [109](#)
- BF\_FINE\_PH\_LOS\_LIMIT\_EN\_SIZE, [109](#)
- BF\_FREQ\_LIMIT\_PH\_LOS\_LSB, [110](#)
- BF\_FREQ\_LIMIT\_PH\_LOS\_MASK, [110](#)
- BF\_FREQ\_LIMIT\_PH\_LOS\_SIZE, [110](#)
- BF\_FREQ\_MON\_CLK\_LSB, [98](#)
- BF\_FREQ\_MON\_CLK\_MASK, [98](#)
- BF\_FREQ\_MON\_CLK\_SIZE, [98](#)
- BF\_FREQ\_MON\_FACTOR\_LSB, [102](#)
- BF\_FREQ\_MON\_FACTOR\_MASK, [102](#)
- BF\_FREQ\_MON\_FACTOR\_SIZE, [102](#)
- BF\_FREQ\_MON\_HARD\_EN\_LSB, [98](#)
- BF\_FREQ\_MON\_HARD\_EN\_MASK, [98](#)
- BF\_FREQ\_MON\_HARD\_EN\_SIZE, [98](#)
- BF\_HARD\_FREQ\_MON\_THRESHOLD\_A\_LSB, [102](#)
- BF\_HARD\_FREQ\_MON\_THRESHOLD\_A\_MASK, [102](#)
- BF\_HARD\_FREQ\_MON\_THRESHOLD\_A\_SIZE, [102](#)
- BF\_HARD\_FREQ\_MON\_THRESHOLD\_B\_LSB, [102](#)
- BF\_HARD\_FREQ\_MON\_THRESHOLD\_B\_MASK, [102](#)
- BF\_HARD\_FREQ\_MON\_THRESHOLD\_B\_SIZE, [102](#)
- BF\_HIGHEST\_PRIORITY\_VALIDATED\_LSB, [107](#)
- BF\_HIGHEST\_PRIORITY\_VALIDATED\_MASK, [107](#)
- BF\_HIGHEST\_PRIORITY\_VALIDATED\_SIZE, [107](#)
- BF\_HOLDOVER\_FREQ\_A\_LSB, [109](#)
- BF\_HOLDOVER\_FREQ\_A\_MASK, [109](#)
- BF\_HOLDOVER\_FREQ\_A\_SIZE, [109](#)
- BF\_HOLDOVER\_FREQ\_B\_LSB, [110](#)
- BF\_HOLDOVER\_FREQ\_B\_MASK, [109](#)
- BF\_HOLDOVER\_FREQ\_B\_SIZE, [109](#)
- BF\_HOLDOVER\_FREQ\_C\_LSB, [110](#)
- BF\_HOLDOVER\_FREQ\_C\_MASK, [110](#)
- BF\_HOLDOVER\_FREQ\_C\_SIZE, [110](#)
- BF\_HS\_EN\_LSB, [98](#)
- BF\_HS\_EN\_MASK, [98](#)
- BF\_HS\_EN\_SIZE, [98](#)
- BF\_HS\_FREZ\_LSB, [98](#)
- BF\_HS\_FREZ\_MASK, [98](#)
- BF\_HS\_FREZ\_SIZE, [98](#)
- BF\_HZ\_EN\_LSB, [98](#)
- BF\_HZ\_EN\_MASK, [98](#)
- BF\_HZ\_EN\_SIZE, [98](#)
- BF\_ICP\_CTRL\_CODE\_LSB, [115](#)
- BF\_ICP\_CTRL\_CODE\_MASK, [115](#)
- BF\_ICP\_CTRL\_CODE\_SIZE, [115](#)
- BF\_ID\_A\_LSB, [96](#)
- BF\_ID\_A\_MASK, [96](#)
- BF\_ID\_A\_SIZE, [96](#)
- BF\_ID\_B\_LSB, [96](#)
- BF\_ID\_B\_MASK, [96](#)
- BF\_ID\_B\_SIZE, [96](#)
- BF\_IN10\_FREQ\_HARD\_ALARM\_LSB, [105](#)
- BF\_IN10\_FREQ\_HARD\_ALARM\_MASK, [105](#)
- BF\_IN10\_FREQ\_HARD\_ALARM\_SIZE, [105](#)
- BF\_IN10\_FREQ\_SOFT\_ALARM\_LSB, [105](#)
- BF\_IN10\_FREQ\_SOFT\_ALARM\_MASK, [105](#)
- BF\_IN10\_FREQ\_SOFT\_ALARM\_SIZE, [105](#)
- BF\_IN10\_LSB, [99](#)
- BF\_IN10\_MASK, [99](#)
- BF\_IN10\_NO\_ACTIVITY\_ALARM\_LSB, [105](#)
- BF\_IN10\_NO\_ACTIVITY\_ALARM\_MASK, [105](#)
- BF\_IN10\_NO\_ACTIVITY\_ALARM\_SIZE, [105](#)
- BF\_IN10\_PH\_LOCK\_ALARM\_LSB, [105](#)
- BF\_IN10\_PH\_LOCK\_ALARM\_MASK, [105](#)
- BF\_IN10\_PH\_LOCK\_ALARM\_SIZE, [105](#)
- BF\_IN10\_SEL\_PRIORITY\_LSB, [102](#)
- BF\_IN10\_SEL\_PRIORITY\_MASK, [102](#)
- BF\_IN10\_SEL\_PRIORITY\_SIZE, [101](#)
- BF\_IN10\_SIZE, [99](#)
- BF\_IN11\_FREQ\_HARD\_ALARM\_LSB, [106](#)
- BF\_IN11\_FREQ\_HARD\_ALARM\_MASK, [106](#)
- BF\_IN11\_FREQ\_HARD\_ALARM\_SIZE, [106](#)
- BF\_IN11\_FREQ\_SOFT\_ALARM\_LSB, [106](#)
- BF\_IN11\_FREQ\_SOFT\_ALARM\_MASK, [106](#)
- BF\_IN11\_FREQ\_SOFT\_ALARM\_SIZE, [106](#)
- BF\_IN11\_LSB, [99](#)
- BF\_IN11\_MASK, [99](#)
- BF\_IN11\_NO\_ACTIVITY\_ALARM\_LSB, [106](#)

BF\_IN11\_NO\_ACTIVITY\_ALARM\_MASK, 106  
 BF\_IN11\_NO\_ACTIVITY\_ALARM\_SIZE, 106  
 BF\_IN11\_PH\_LOCK\_ALARM\_LSB, 106  
 BF\_IN11\_PH\_LOCK\_ALARM\_MASK, 106  
 BF\_IN11\_PH\_LOCK\_ALARM\_SIZE, 106  
 BF\_IN11\_SEL\_PRIORITY\_LSB, 102  
 BF\_IN11\_SEL\_PRIORITY\_MASK, 102  
 BF\_IN11\_SEL\_PRIORITY\_SIZE, 102  
 BF\_IN11\_SIZE, 99  
 BF\_IN12\_FREQ\_HARD\_ALARM\_LSB, 106  
 BF\_IN12\_FREQ\_HARD\_ALARM\_MASK, 106  
 BF\_IN12\_FREQ\_HARD\_ALARM\_SIZE, 106  
 BF\_IN12\_FREQ\_SOFT\_ALARM\_LSB, 106  
 BF\_IN12\_FREQ\_SOFT\_ALARM\_MASK, 105  
 BF\_IN12\_FREQ\_SOFT\_ALARM\_SIZE, 105  
 BF\_IN12\_LSB, 99  
 BF\_IN12\_MASK, 99  
 BF\_IN12\_NO\_ACTIVITY\_ALARM\_LSB, 106  
 BF\_IN12\_NO\_ACTIVITY\_ALARM\_MASK, 106  
 BF\_IN12\_NO\_ACTIVITY\_ALARM\_SIZE, 106  
 BF\_IN12\_PH\_LOCK\_ALARM\_LSB, 106  
 BF\_IN12\_PH\_LOCK\_ALARM\_MASK, 106  
 BF\_IN12\_PH\_LOCK\_ALARM\_SIZE, 106  
 BF\_IN12\_SEL\_PRIORITY\_LSB, 102  
 BF\_IN12\_SEL\_PRIORITY\_MASK, 102  
 BF\_IN12\_SEL\_PRIORITY\_SIZE, 102  
 BF\_IN12\_SIZE, 99  
 BF\_IN13\_FREQ\_HARD\_ALARM\_LSB, 106  
 BF\_IN13\_FREQ\_HARD\_ALARM\_MASK, 106  
 BF\_IN13\_FREQ\_HARD\_ALARM\_SIZE, 106  
 BF\_IN13\_FREQ\_SOFT\_ALARM\_LSB, 106  
 BF\_IN13\_FREQ\_SOFT\_ALARM\_MASK, 106  
 BF\_IN13\_FREQ\_SOFT\_ALARM\_SIZE, 106  
 BF\_IN13\_LSB, 99  
 BF\_IN13\_MASK, 99  
 BF\_IN13\_NO\_ACTIVITY\_ALARM\_LSB, 106  
 BF\_IN13\_NO\_ACTIVITY\_ALARM\_MASK, 106  
 BF\_IN13\_NO\_ACTIVITY\_ALARM\_SIZE, 106  
 BF\_IN13\_PH\_LOCK\_ALARM\_LSB, 107  
 BF\_IN13\_PH\_LOCK\_ALARM\_MASK, 106  
 BF\_IN13\_PH\_LOCK\_ALARM\_SIZE, 106  
 BF\_IN13\_SEL\_PRIORITY\_LSB, 102  
 BF\_IN13\_SEL\_PRIORITY\_MASK, 102  
 BF\_IN13\_SEL\_PRIORITY\_SIZE, 102  
 BF\_IN13\_SIZE, 99  
 BF\_IN14\_FREQ\_HARD\_ALARM\_LSB, 106  
 BF\_IN14\_FREQ\_HARD\_ALARM\_MASK, 106  
 BF\_IN14\_FREQ\_HARD\_ALARM\_SIZE, 106  
 BF\_IN14\_FREQ\_SOFT\_ALARM\_LSB, 106  
 BF\_IN14\_FREQ\_SOFT\_ALARM\_MASK, 106  
 BF\_IN14\_FREQ\_SOFT\_ALARM\_SIZE, 106  
 BF\_IN14\_LSB, 99  
 BF\_IN14\_MASK, 99  
 BF\_IN14\_NO\_ACTIVITY\_ALARM\_LSB, 106  
 BF\_IN14\_NO\_ACTIVITY\_ALARM\_MASK, 106  
 BF\_IN14\_NO\_ACTIVITY\_ALARM\_SIZE, 106  
 BF\_IN14\_PH\_LOCK\_ALARM\_LSB, 106  
 BF\_IN14\_PH\_LOCK\_ALARM\_MASK, 106  
 BF\_IN14\_PH\_LOCK\_ALARM\_SIZE, 106  
 BF\_IN14\_SEL\_PRIORITY\_LSB, 102  
 BF\_IN14\_SEL\_PRIORITY\_MASK, 102  
 BF\_IN14\_SEL\_PRIORITY\_SIZE, 102  
 BF\_IN14\_SIZE, 99  
 BF\_IN1\_FREQ\_HARD\_ALARM\_LSB, 103  
 BF\_IN1\_FREQ\_HARD\_ALARM\_MASK, 103  
 BF\_IN1\_FREQ\_HARD\_ALARM\_SIZE, 103  
 BF\_IN1\_FREQ\_SOFT\_ALARM\_LSB, 103  
 BF\_IN1\_FREQ\_SOFT\_ALARM\_MASK, 103  
 BF\_IN1\_FREQ\_SOFT\_ALARM\_SIZE, 103  
 BF\_IN1\_LSB, 99  
 BF\_IN1\_MASK, 99  
 BF\_IN1\_NO\_ACTIVITY\_ALARM\_LSB, 103  
 BF\_IN1\_NO\_ACTIVITY\_ALARM\_MASK, 103  
 BF\_IN1\_NO\_ACTIVITY\_ALARM\_SIZE, 103  
 BF\_IN1\_PH\_LOCK\_ALARM\_LSB, 103  
 BF\_IN1\_PH\_LOCK\_ALARM\_MASK, 103  
 BF\_IN1\_PH\_LOCK\_ALARM\_SIZE, 103  
 BF\_IN1\_SEL\_PRIORITY\_LSB, 101  
 BF\_IN1\_SEL\_PRIORITY\_MASK, 101  
 BF\_IN1\_SEL\_PRIORITY\_SIZE, 101  
 BF\_IN1\_SIZE, 99  
 BF\_IN2\_FREQ\_HARD\_ALARM\_LSB, 103  
 BF\_IN2\_FREQ\_HARD\_ALARM\_MASK, 103  
 BF\_IN2\_FREQ\_HARD\_ALARM\_SIZE, 103  
 BF\_IN2\_FREQ\_SOFT\_ALARM\_LSB, 103  
 BF\_IN2\_FREQ\_SOFT\_ALARM\_MASK, 103  
 BF\_IN2\_FREQ\_SOFT\_ALARM\_SIZE, 103  
 BF\_IN2\_LSB, 99  
 BF\_IN2\_MASK, 99  
 BF\_IN2\_NO\_ACTIVITY\_ALARM\_LSB, 103  
 BF\_IN2\_NO\_ACTIVITY\_ALARM\_MASK, 103  
 BF\_IN2\_NO\_ACTIVITY\_ALARM\_SIZE, 103  
 BF\_IN2\_PH\_LOCK\_ALARM\_LSB, 103  
 BF\_IN2\_PH\_LOCK\_ALARM\_MASK, 103  
 BF\_IN2\_PH\_LOCK\_ALARM\_SIZE, 103  
 BF\_IN2\_SEL\_PRIORITY\_LSB, 101  
 BF\_IN2\_SEL\_PRIORITY\_MASK, 101  
 BF\_IN2\_SEL\_PRIORITY\_SIZE, 101  
 BF\_IN2\_SIZE, 99  
 BF\_IN3\_FREQ\_HARD\_ALARM\_LSB, 104  
 BF\_IN3\_FREQ\_HARD\_ALARM\_MASK, 104  
 BF\_IN3\_FREQ\_HARD\_ALARM\_SIZE, 104  
 BF\_IN3\_FREQ\_SOFT\_ALARM\_LSB, 104  
 BF\_IN3\_FREQ\_SOFT\_ALARM\_MASK, 104  
 BF\_IN3\_FREQ\_SOFT\_ALARM\_SIZE, 104  
 BF\_IN3\_LSB, 99  
 BF\_IN3\_MASK, 99  
 BF\_IN3\_NO\_ACTIVITY\_ALARM\_LSB, 104  
 BF\_IN3\_NO\_ACTIVITY\_ALARM\_MASK, 104  
 BF\_IN3\_NO\_ACTIVITY\_ALARM\_SIZE, 104  
 BF\_IN3\_PH\_LOCK\_ALARM\_LSB, 104  
 BF\_IN3\_PH\_LOCK\_ALARM\_MASK, 104  
 BF\_IN3\_PH\_LOCK\_ALARM\_SIZE, 104  
 BF\_IN3\_SEL\_PRIORITY\_LSB, 101  
 BF\_IN3\_SEL\_PRIORITY\_MASK, 101  
 BF\_IN3\_SEL\_PRIORITY\_SIZE, 101

BF\_IN3\_SIZE, 99  
BF\_IN4\_FREQ\_HARD\_ALARM\_LSB, 103  
BF\_IN4\_FREQ\_HARD\_ALARM\_MASK, 103  
BF\_IN4\_FREQ\_HARD\_ALARM\_SIZE, 103  
BF\_IN4\_FREQ\_SOFT\_ALARM\_LSB, 103  
BF\_IN4\_FREQ\_SOFT\_ALARM\_MASK, 103  
BF\_IN4\_FREQ\_SOFT\_ALARM\_SIZE, 103  
BF\_IN4\_LSB, 99  
BF\_IN4\_MASK, 99  
BF\_IN4\_NO\_ACTIVITY\_ALARM\_LSB, 104  
BF\_IN4\_NO\_ACTIVITY\_ALARM\_MASK, 103  
BF\_IN4\_NO\_ACTIVITY\_ALARM\_SIZE, 103  
BF\_IN4\_PH\_LOCK\_ALARM\_LSB, 104  
BF\_IN4\_PH\_LOCK\_ALARM\_MASK, 104  
BF\_IN4\_PH\_LOCK\_ALARM\_SIZE, 104  
BF\_IN4\_SEL\_PRIORITY\_LSB, 101  
BF\_IN4\_SEL\_PRIORITY\_MASK, 101  
BF\_IN4\_SEL\_PRIORITY\_SIZE, 101  
BF\_IN4\_SIZE, 99  
BF\_IN5\_DIV\_LSB, 101  
BF\_IN5\_DIV\_MASK, 101  
BF\_IN5\_DIV\_SIZE, 101  
BF\_IN5\_FREQ\_HARD\_ALARM\_LSB, 104  
BF\_IN5\_FREQ\_HARD\_ALARM\_MASK, 104  
BF\_IN5\_FREQ\_HARD\_ALARM\_SIZE, 104  
BF\_IN5\_FREQ\_SOFT\_ALARM\_LSB, 104  
BF\_IN5\_FREQ\_SOFT\_ALARM\_MASK, 104  
BF\_IN5\_FREQ\_SOFT\_ALARM\_SIZE, 104  
BF\_IN5\_LSB, 99  
BF\_IN5\_MASK, 99  
BF\_IN5\_NO\_ACTIVITY\_ALARM\_LSB, 104  
BF\_IN5\_NO\_ACTIVITY\_ALARM\_MASK, 104  
BF\_IN5\_NO\_ACTIVITY\_ALARM\_SIZE, 104  
BF\_IN5\_PH\_LOCK\_ALARM\_LSB, 104  
BF\_IN5\_PH\_LOCK\_ALARM\_MASK, 104  
BF\_IN5\_PH\_LOCK\_ALARM\_SIZE, 104  
BF\_IN5\_SEL\_PRIORITY\_LSB, 101  
BF\_IN5\_SEL\_PRIORITY\_MASK, 101  
BF\_IN5\_SEL\_PRIORITY\_SIZE, 101  
BF\_IN5\_SIZE, 98  
BF\_IN6\_DIV\_LSB, 101  
BF\_IN6\_DIV\_MASK, 101  
BF\_IN6\_DIV\_SIZE, 101  
BF\_IN6\_FREQ\_HARD\_ALARM\_LSB, 104  
BF\_IN6\_FREQ\_HARD\_ALARM\_MASK, 104  
BF\_IN6\_FREQ\_HARD\_ALARM\_SIZE, 104  
BF\_IN6\_FREQ\_SOFT\_ALARM\_LSB, 104  
BF\_IN6\_FREQ\_SOFT\_ALARM\_MASK, 104  
BF\_IN6\_FREQ\_SOFT\_ALARM\_SIZE, 104  
BF\_IN6\_LSB, 98  
BF\_IN6\_MASK, 98  
BF\_IN6\_NO\_ACTIVITY\_ALARM\_LSB, 104  
BF\_IN6\_NO\_ACTIVITY\_ALARM\_MASK, 104  
BF\_IN6\_NO\_ACTIVITY\_ALARM\_SIZE, 104  
BF\_IN6\_PH\_LOCK\_ALARM\_LSB, 104  
BF\_IN6\_PH\_LOCK\_ALARM\_MASK, 104  
BF\_IN6\_PH\_LOCK\_ALARM\_SIZE, 104  
BF\_IN6\_SEL\_PRIORITY\_LSB, 101  
BF\_IN6\_SEL\_PRIORITY\_MASK, 101  
BF\_IN6\_SEL\_PRIORITY\_SIZE, 101  
BF\_IN6\_SIZE, 98  
BF\_IN6\_SEL\_PRIORITY\_MASK, 101  
BF\_IN6\_SEL\_PRIORITY\_SIZE, 101  
BF\_IN6\_SIZE, 98  
BF\_IN7\_FREQ\_HARD\_ALARM\_LSB, 105  
BF\_IN7\_FREQ\_HARD\_ALARM\_MASK, 105  
BF\_IN7\_FREQ\_HARD\_ALARM\_SIZE, 105  
BF\_IN7\_FREQ\_SOFT\_ALARM\_LSB, 105  
BF\_IN7\_FREQ\_SOFT\_ALARM\_MASK, 105  
BF\_IN7\_FREQ\_SOFT\_ALARM\_SIZE, 105  
BF\_IN7\_LSB, 98  
BF\_IN7\_MASK, 98  
BF\_IN7\_NO\_ACTIVITY\_ALARM\_LSB, 105  
BF\_IN7\_NO\_ACTIVITY\_ALARM\_MASK, 105  
BF\_IN7\_NO\_ACTIVITY\_ALARM\_SIZE, 105  
BF\_IN7\_PH\_LOCK\_ALARM\_LSB, 105  
BF\_IN7\_PH\_LOCK\_ALARM\_MASK, 105  
BF\_IN7\_PH\_LOCK\_ALARM\_SIZE, 105  
BF\_IN7\_SEL\_PRIORITY\_LSB, 101  
BF\_IN7\_SEL\_PRIORITY\_MASK, 101  
BF\_IN7\_SEL\_PRIORITY\_SIZE, 101  
BF\_IN7\_SIZE, 98  
BF\_IN8\_FREQ\_HARD\_ALARM\_LSB, 105  
BF\_IN8\_FREQ\_HARD\_ALARM\_MASK, 104  
BF\_IN8\_FREQ\_HARD\_ALARM\_SIZE, 104  
BF\_IN8\_FREQ\_SOFT\_ALARM\_LSB, 104  
BF\_IN8\_FREQ\_SOFT\_ALARM\_MASK, 104  
BF\_IN8\_FREQ\_SOFT\_ALARM\_SIZE, 104  
BF\_IN8\_LSB, 98  
BF\_IN8\_MASK, 98  
BF\_IN8\_NO\_ACTIVITY\_ALARM\_LSB, 105  
BF\_IN8\_NO\_ACTIVITY\_ALARM\_MASK, 105  
BF\_IN8\_NO\_ACTIVITY\_ALARM\_SIZE, 105  
BF\_IN8\_PH\_LOCK\_ALARM\_LSB, 105  
BF\_IN8\_PH\_LOCK\_ALARM\_MASK, 105  
BF\_IN8\_PH\_LOCK\_ALARM\_SIZE, 105  
BF\_IN8\_SEL\_PRIORITY\_LSB, 101  
BF\_IN8\_SEL\_PRIORITY\_MASK, 101  
BF\_IN8\_SEL\_PRIORITY\_SIZE, 101  
BF\_IN8\_SIZE, 98  
BF\_IN9\_FREQ\_HARD\_ALARM\_LSB, 105  
BF\_IN9\_FREQ\_HARD\_ALARM\_MASK, 105  
BF\_IN9\_FREQ\_HARD\_ALARM\_SIZE, 105  
BF\_IN9\_FREQ\_SOFT\_ALARM\_LSB, 105  
BF\_IN9\_FREQ\_SOFT\_ALARM\_MASK, 105  
BF\_IN9\_FREQ\_SOFT\_ALARM\_SIZE, 105  
BF\_IN9\_LSB, 100  
BF\_IN9\_MASK, 99  
BF\_IN9\_NO\_ACTIVITY\_ALARM\_LSB, 105  
BF\_IN9\_NO\_ACTIVITY\_ALARM\_MASK, 105  
BF\_IN9\_NO\_ACTIVITY\_ALARM\_SIZE, 105  
BF\_IN9\_PH\_LOCK\_ALARM\_LSB, 105  
BF\_IN9\_PH\_LOCK\_ALARM\_MASK, 105  
BF\_IN9\_PH\_LOCK\_ALARM\_SIZE, 105  
BF\_IN9\_SEL\_PRIORITY\_LSB, 102  
BF\_IN9\_SEL\_PRIORITY\_MASK, 102  
BF\_IN9\_SEL\_PRIORITY\_SIZE, 102  
BF\_IN9\_SIZE, 99  
BF\_IN\_2K\_4K\_8K\_INV\_LSB, 113

BF\_IN\_2K\_4K\_8K\_INV\_MASK, 112  
 BF\_IN\_2K\_4K\_8K\_INV\_SIZE, 112  
 BF\_IN\_FREQ\_LSB, 100  
 BF\_IN\_FREQ\_MASK, 100  
 BF\_IN\_FREQ\_READ\_CH\_LSB, 103  
 BF\_IN\_FREQ\_READ\_CH\_MASK, 103  
 BF\_IN\_FREQ\_READ\_CH\_SIZE, 103  
 BF\_IN\_FREQ\_SIZE, 100  
 BF\_IN\_FREQ\_VALUE\_LSB, 103  
 BF\_IN\_FREQ\_VALUE\_MASK, 103  
 BF\_IN\_FREQ\_VALUE\_SIZE, 103  
 BF\_IN\_NOISE\_WINDOW\_LSB, 113  
 BF\_IN\_NOISE\_WINDOW\_MASK, 113  
 BF\_IN\_NOISE\_WINDOW\_SIZE, 113  
 BF\_IN\_SONET\_SDH\_LSB, 97  
 BF\_IN\_SONET\_SDH\_MASK, 97  
 BF\_IN\_SONET\_SDH\_SIZE, 97  
 BF\_INPUT\_TO\_T4\_LSB, 100  
 BF\_INPUT\_TO\_T4\_MASK, 100  
 BF\_INPUT\_TO\_T4\_SIZE, 100  
 BF\_INT\_POL\_LSB, 98  
 BF\_INT\_POL\_MASK, 98  
 BF\_INT\_POL\_SIZE, 98  
 BF\_LOCK\_8K\_LSB, 101  
 BF\_LOCK\_8K\_MASK, 101  
 BF\_LOCK\_8K\_SIZE, 101  
 BF\_LOS\_FLAG\_TO\_TDO\_LSB, 98  
 BF\_LOS\_FLAG\_TO\_TDO\_MASK, 98  
 BF\_LOS\_FLAG\_TO\_TDO\_SIZE, 98  
 BF\_LOWER\_THRESHOLD\_DATA\_LSB, 102  
 BF\_LOWER\_THRESHOLD\_DATA\_MASK, 102  
 BF\_LOWER\_THRESHOLD\_DATA\_SIZE, 102  
 BF\_LVDS\_QA\_LSB, 116  
 BF\_LVDS\_QA\_MASK, 116  
 BF\_LVDS\_QA\_SIZE, 116  
 BF\_LVDS\_QB\_LSB, 117  
 BF\_LVDS\_QB\_MASK, 117  
 BF\_LVDS\_QB\_SIZE, 117  
 BF\_M0\_LSB\_LSB, 116  
 BF\_M0\_LSB\_MASK, 116  
 BF\_M0\_LSB\_SIZE, 116  
 BF\_M0\_MSB\_LSB, 116  
 BF\_M0\_MSB\_MASK, 116  
 BF\_M0\_MSB\_SIZE, 116  
 BF\_M1\_LSB\_LSB, 116  
 BF\_M1\_LSB\_MASK, 116  
 BF\_M1\_LSB\_SIZE, 116  
 BF\_M1\_MSB\_LSB, 116  
 BF\_M1\_MSB\_MASK, 116  
 BF\_M1\_MSB\_SIZE, 116  
 BF\_MAN\_HOLD OVER\_LSB, 109  
 BF\_MAN\_HOLD OVER\_MASK, 109  
 BF\_MAN\_HOLD OVER\_SIZE, 109  
 BF\_MASTER\_SLAVE\_LSB, 97  
 BF\_MASTER\_SLAVE\_MASK, 97  
 BF\_MASTER\_SLAVE\_SIZE, 97  
 BF\_MPU\_PIN\_STS\_LSB, 97  
 BF\_MPU\_PIN\_STS\_MASK, 97  
 BF\_MPU\_PIN\_STS\_SIZE, 96  
 BF\_MPU\_SEL\_CNFG\_LSB, 114  
 BF\_MPU\_SEL\_CNFG\_MASK, 114  
 BF\_MPU\_SEL\_CNFG\_SIZE, 114  
 BF\_MR\_SYNC\_LSB, 116  
 BF\_MR\_SYNC\_MASK, 116  
 BF\_MR\_SYNC\_SIZE, 116  
 BF\_MS\_SL\_CTRL\_LSB, 100  
 BF\_MS\_SL\_CTRL\_MASK, 100  
 BF\_MS\_SL\_CTRL\_SIZE, 100  
 BF\_MULTI\_FACTOR\_LSB, 97  
 BF\_MULTI\_FACTOR\_MASK, 97  
 BF\_MULTI\_FACTOR\_SIZE, 97  
 BF\_MULTI\_PH\_8K\_4K\_2K\_EN\_LSB, 109  
 BF\_MULTI\_PH\_8K\_4K\_2K\_EN\_MASK, 109  
 BF\_MULTI\_PH\_8K\_4K\_2K\_EN\_SIZE, 109  
 BF\_MULTI\_PH\_APP\_LSB, 109  
 BF\_MULTI\_PH\_APP\_MASK, 109  
 BF\_MULTI\_PH\_APP\_SIZE, 109  
 BF\_NOMINAL\_FREQ\_VALUE\_A\_LSB, 97  
 BF\_NOMINAL\_FREQ\_VALUE\_A\_MASK, 97  
 BF\_NOMINAL\_FREQ\_VALUE\_A\_SIZE, 97  
 BF\_NOMINAL\_FREQ\_VALUE\_B\_LSB, 97  
 BF\_NOMINAL\_FREQ\_VALUE\_B\_MASK, 97  
 BF\_NOMINAL\_FREQ\_VALUE\_B\_SIZE, 97  
 BF\_NOMINAL\_FREQ\_VALUE\_C\_LSB, 97  
 BF\_NOMINAL\_FREQ\_VALUE\_C\_MASK, 97  
 BF\_NOMINAL\_FREQ\_VALUE\_C\_SIZE, 97  
 BF\_ODBSELA0\_LSB, 116  
 BF\_ODBSELA0\_MASK, 116  
 BF\_ODBSELA0\_SIZE, 116  
 BF\_ODBSELA1\_LSB, 116  
 BF\_ODBSELA1\_MASK, 116  
 BF\_ODBSELA1\_SIZE, 116  
 BF\_ODBSELB0\_LSB, 116  
 BF\_ODBSELB0\_MASK, 116  
 BF\_ODBSELB0\_SIZE, 116  
 BF\_ODBSELB1\_LSB, 116  
 BF\_ODBSELB1\_MASK, 116  
 BF\_ODBSELB1\_SIZE, 116  
 BF\_OE\_A\_LSB, 117  
 BF\_OE\_A\_MASK, 117  
 BF\_OE\_A\_SIZE, 116  
 BF\_OE\_B\_LSB, 117  
 BF\_OE\_B\_MASK, 117  
 BF\_OE\_B\_SIZE, 117  
 BF\_OSC\_EDGE\_LSB, 97  
 BF\_OSC\_EDGE\_MASK, 97  
 BF\_OSC\_EDGE\_SIZE, 97  
 BF\_OUT1\_DIVIDER\_LSB, 111  
 BF\_OUT1\_DIVIDER\_MASK, 111  
 BF\_OUT1\_DIVIDER\_SIZE, 111  
 BF\_OUT1\_INV\_LSB, 112  
 BF\_OUT1\_INV\_MASK, 112  
 BF\_OUT1\_INV\_SIZE, 112  
 BF\_OUT1\_PATH\_SEL\_LSB, 111  
 BF\_OUT1\_PATH\_SEL\_MASK, 111  
 BF\_OUT1\_PATH\_SEL\_SIZE, 111

BF\_OUT2\_DIVIDER\_LSB, 111  
BF\_OUT2\_DIVIDER\_MASK, 111  
BF\_OUT2\_DIVIDER\_SIZE, 111  
BF\_OUT2\_INV\_LSB, 112  
BF\_OUT2\_INV\_MASK, 112  
BF\_OUT2\_INV\_SIZE, 112  
BF\_OUT2\_PATH\_SEL\_LSB, 111  
BF\_OUT2\_PATH\_SEL\_MASK, 111  
BF\_OUT2\_PATH\_SEL\_SIZE, 111  
BF\_OUT3\_DIVIDER\_LSB, 111  
BF\_OUT3\_DIVIDER\_MASK, 111  
BF\_OUT3\_DIVIDER\_SIZE, 111  
BF\_OUT3\_INV\_LSB, 112  
BF\_OUT3\_INV\_MASK, 112  
BF\_OUT3\_INV\_SIZE, 112  
BF\_OUT3\_PATH\_SEL\_LSB, 111  
BF\_OUT3\_PATH\_SEL\_MASK, 111  
BF\_OUT3\_PATH\_SEL\_SIZE, 111  
BF\_OUT4\_DIVIDER\_LSB, 111  
BF\_OUT4\_DIVIDER\_MASK, 111  
BF\_OUT4\_DIVIDER\_SIZE, 111  
BF\_OUT4\_INV\_LSB, 112  
BF\_OUT4\_INV\_MASK, 112  
BF\_OUT4\_INV\_SIZE, 112  
BF\_OUT4\_PATH\_SEL\_LSB, 111  
BF\_OUT4\_PATH\_SEL\_MASK, 111  
BF\_OUT4\_PATH\_SEL\_SIZE, 111  
BF\_OUT5\_DIVIDER\_LSB, 111  
BF\_OUT5\_DIVIDER\_MASK, 111  
BF\_OUT5\_DIVIDER\_SIZE, 111  
BF\_OUT5\_INV\_LSB, 112  
BF\_OUT5\_INV\_MASK, 112  
BF\_OUT5\_INV\_SIZE, 112  
BF\_OUT5\_PATH\_SEL\_LSB, 111  
BF\_OUT5\_PATH\_SEL\_MASK, 111  
BF\_OUT5\_PATH\_SEL\_SIZE, 111  
BF\_OUT6\_DIVIDER\_LSB, 111  
BF\_OUT6\_DIVIDER\_MASK, 111  
BF\_OUT6\_DIVIDER\_SIZE, 111  
BF\_OUT6\_INV\_LSB, 112  
BF\_OUT6\_INV\_MASK, 112  
BF\_OUT6\_INV\_SIZE, 112  
BF\_OUT6\_PATH\_SEL\_LSB, 111  
BF\_OUT6\_PATH\_SEL\_MASK, 111  
BF\_OUT6\_PATH\_SEL\_SIZE, 111  
BF\_OUT6\_PECL\_LVDS\_LSB, 98  
BF\_OUT6\_PECL\_LVDS\_MASK, 98  
BF\_OUT6\_PECL\_LVDS\_SIZE, 98  
BF\_OUT7\_DIVIDER\_LSB, 111  
BF\_OUT7\_DIVIDER\_MASK, 111  
BF\_OUT7\_DIVIDER\_SIZE, 111  
BF\_OUT7\_INV\_LSB, 112  
BF\_OUT7\_INV\_MASK, 112  
BF\_OUT7\_INV\_SIZE, 112  
BF\_OUT7\_PATH\_SEL\_LSB, 111  
BF\_OUT7\_PATH\_SEL\_MASK, 111  
BF\_OUT7\_PATH\_SEL\_SIZE, 111  
BF\_OUT7\_PECL\_LVDS\_LSB, 98  
BF\_OUT7\_PECL\_LVDS\_MASK, 98  
BF\_OUT7\_PECL\_LVDS\_SIZE, 97  
BF\_OUT8\_EN\_LSB, 112  
BF\_OUT8\_EN\_MASK, 112  
BF\_OUT8\_EN\_SIZE, 112  
BF\_OUT8\_PATH\_SEL\_LSB, 112  
BF\_OUT8\_PATH\_SEL\_MASK, 111  
BF\_OUT8\_PATH\_SEL\_SIZE, 111  
BF\_OUT9\_EN\_LSB, 112  
BF\_OUT9\_EN\_MASK, 112  
BF\_OUT9\_EN\_SIZE, 112  
BF\_OUT9\_INV\_LSB, 112  
BF\_OUT9\_INV\_MASK, 112  
BF\_OUT9\_INV\_SIZE, 112  
BF\_OUT9\_PATH\_SEL\_LSB, 112  
BF\_OUT9\_PATH\_SEL\_MASK, 112  
BF\_OUT9\_PATH\_SEL\_SIZE, 112  
BF\_PAGE\_POINTER\_LSB, 102  
BF\_PAGE\_POINTER\_MASK, 102  
BF\_PAGE\_POINTER\_SIZE, 102  
BF\_PDSEL0\_LSB\_LSB, 115  
BF\_PDSEL0\_LSB\_MASK, 115  
BF\_PDSEL0\_LSB\_SIZE, 115  
BF\_PDSEL0\_MSB\_LSB, 115  
BF\_PDSEL0\_MSB\_MASK, 115  
BF\_PDSEL0\_MSB\_SIZE, 115  
BF\_PDSEL1\_LSB\_LSB, 115  
BF\_PDSEL1\_LSB\_MASK, 115  
BF\_PDSEL1\_LSB\_SIZE, 115  
BF\_PDSEL1\_MSB\_LSB, 116  
BF\_PDSEL1\_MSB\_MASK, 116  
BF\_PDSEL1\_MSB\_SIZE, 115  
BF\_PH\_ALARM\_TIMEOUT\_LSB, 97  
BF\_PH\_ALARM\_TIMEOUT\_MASK, 97  
BF\_PH\_ALARM\_TIMEOUT\_SIZE, 97  
BF\_PH\_LOS\_COARSE\_LIMIT\_LSB, 109  
BF\_PH\_LOS\_COARSE\_LIMIT\_MASK, 109  
BF\_PH\_LOS\_COARSE\_LIMIT\_SIZE, 109  
BF\_PH\_LOS\_FINE\_LIMIT\_LSB, 109  
BF\_PH\_LOS\_FINE\_LIMIT\_MASK, 109  
BF\_PH\_LOS\_FINE\_LIMIT\_SIZE, 109  
BF\_PH\_OFFSET\_A\_LSB, 113  
BF\_PH\_OFFSET\_A\_MASK, 113  
BF\_PH\_OFFSET\_A\_SIZE, 113  
BF\_PH\_OFFSET\_B\_LSB, 113  
BF\_PH\_OFFSET\_B\_MASK, 113  
BF\_PH\_OFFSET\_B\_SIZE, 113  
BF\_PH\_OFFSET\_EN\_LSB, 113  
BF\_PH\_OFFSET\_EN\_MASK, 113  
BF\_PH\_OFFSET\_EN\_SIZE, 113  
BF\_PPS\_FAST\_FREQ\_LOCK\_EN\_LSB, 100  
BF\_PPS\_FAST\_FREQ\_LOCK\_EN\_MASK, 100  
BF\_PPS\_FAST\_FREQ\_LOCK\_EN\_SIZE, 100  
BF\_PPS\_FAST\_PH\_LOCK\_EN\_LSB, 100  
BF\_PPS\_FAST\_PH\_LOCK\_EN\_MASK, 100  
BF\_PPS\_FAST\_PH\_LOCK\_EN\_SIZE, 100  
BF\_PPS\_PHASE\_LSB, 114  
BF\_PPS\_PHASE\_MASK, 114

- BF\_PPS\_PHASE\_SIZE, 114
- BF\_PPS\_PULSE\_LSB, 114
- BF\_PPS\_PULSE\_MASK, 114
- BF\_PPS\_PULSE\_SIZE, 114
- BF\_PRE\_DIV\_CH\_VALUE\_LSB, 101
- BF\_PRE\_DIV\_CH\_VALUE\_MASK, 101
- BF\_PRE\_DIV\_CH\_VALUE\_SIZE, 101
- BF\_PRE\_DIVN\_VALUE\_A\_LSB, 101
- BF\_PRE\_DIVN\_VALUE\_A\_MASK, 101
- BF\_PRE\_DIVN\_VALUE\_A\_SIZE, 101
- BF\_PRE\_DIVN\_VALUE\_B\_LSB, 101
- BF\_PRE\_DIVN\_VALUE\_B\_MASK, 101
- BF\_PRE\_DIVN\_VALUE\_B\_SIZE, 101
- BF\_PROTECTION\_DATA\_LSB, 114
- BF\_PROTECTION\_DATA\_MASK, 114
- BF\_PROTECTION\_DATA\_SIZE, 114
- BF\_READ\_AVG\_LSB, 109
- BF\_READ\_AVG\_MASK, 109
- BF\_READ\_AVG\_SIZE, 109
- BF\_REVERTIVE\_MODE\_LSB, 97
- BF\_REVERTIVE\_MODE\_MASK, 97
- BF\_REVERTIVE\_MODE\_SIZE, 97
- BF\_SECOND\_HIGHEST\_PRIORITY\_VALIDATE-  
D\_LSB, 107
- BF\_SECOND\_HIGHEST\_PRIORITY\_VALIDATE-  
D\_MASK, 107
- BF\_SECOND\_HIGHEST\_PRIORITY\_VALIDATE-  
D\_SIZE, 107
- BF\_SEL\_DOUBLE\_LSB, 116
- BF\_SEL\_DOUBLE\_MASK, 116
- BF\_SEL\_DOUBLE\_SIZE, 116
- BF\_SHUTOFF\_LSB, 116
- BF\_SHUTOFF\_MASK, 116
- BF\_SHUTOFF\_SIZE, 116
- BF\_SOFT\_FREQ\_MON\_THRESHOLD\_A\_LSB,  
102
- BF\_SOFT\_FREQ\_MON\_THRESHOLD\_A\_MASK,  
102
- BF\_SOFT\_FREQ\_MON\_THRESHOLD\_A\_SIZE,  
102
- BF\_SOFT\_FREQ\_MON\_THRESHOLD\_B\_LSB,  
102
- BF\_SOFT\_FREQ\_MON\_THRESHOLD\_B\_MASK,  
102
- BF\_SOFT\_FREQ\_MON\_THRESHOLD\_B\_SIZE,  
102
- BF\_SYNC\_BYPASS\_LSB, 113
- BF\_SYNC\_BYPASS\_MASK, 113
- BF\_SYNC\_BYPASS\_SIZE, 113
- BF\_SYNC\_EDGE\_LSB, 113
- BF\_SYNC\_EDGE\_MASK, 113
- BF\_SYNC\_EDGE\_SIZE, 113
- BF\_SYNC\_FREQ\_LSB, 97
- BF\_SYNC\_FREQ\_MASK, 97
- BF\_SYNC\_FREQ\_SIZE, 97
- BF\_SYNC\_MON\_LIMT\_LSB, 113
- BF\_SYNC\_MON\_LIMT\_MASK, 113
- BF\_SYNC\_MON\_LIMT\_SIZE, 113
- BF\_SYNC\_PH1\_LSB, 114
- BF\_SYNC\_PH1\_MASK, 114
- BF\_SYNC\_PH1\_SIZE, 114
- BF\_SYNC\_PH2\_LSB, 114
- BF\_SYNC\_PH2\_MASK, 113
- BF\_SYNC\_PH2\_SIZE, 113
- BF\_T0\_12E1\_GPS\_E3\_T3\_SEL\_LSB, 108
- BF\_T0\_12E1\_GPS\_E3\_T3\_SEL\_MASK, 108
- BF\_T0\_12E1\_GPS\_E3\_T3\_SEL\_SIZE, 108
- BF\_T0\_APLL\_PATH\_LSB, 108
- BF\_T0\_APLL\_PATH\_MASK, 108
- BF\_T0\_APLL\_PATH\_SIZE, 108
- BF\_T0\_DFS\_OFF\_0\_LSB, 114
- BF\_T0\_DFS\_OFF\_0\_MASK, 114
- BF\_T0\_DFS\_OFF\_0\_SIZE, 114
- BF\_T0\_DFS\_OFF\_1\_LSB, 114
- BF\_T0\_DFS\_OFF\_1\_MASK, 114
- BF\_T0\_DFS\_OFF\_1\_SIZE, 114
- BF\_T0\_DFS\_OFF\_2\_LSB, 114
- BF\_T0\_DFS\_OFF\_2\_MASK, 114
- BF\_T0\_DFS\_OFF\_2\_SIZE, 114
- BF\_T0\_DFS\_OFF\_3\_LSB, 114
- BF\_T0\_DFS\_OFF\_3\_MASK, 114
- BF\_T0\_DFS\_OFF\_3\_SIZE, 114
- BF\_T0\_DPLL\_ACQ\_BW\_LSB, 108
- BF\_T0\_DPLL\_ACQ\_BW\_MASK, 108
- BF\_T0\_DPLL\_ACQ\_BW\_SIZE, 108
- BF\_T0\_DPLL\_ACQ\_DAMPING\_LSB, 108
- BF\_T0\_DPLL\_ACQ\_DAMPING\_MASK, 108
- BF\_T0\_DPLL\_ACQ\_DAMPING\_SIZE, 108
- BF\_T0\_DPLL\_LOCK\_LSB, 107
- BF\_T0\_DPLL\_LOCK\_MASK, 107
- BF\_T0\_DPLL\_LOCK\_SIZE, 107
- BF\_T0\_DPLL\_LOCKED\_BW\_LSB, 108
- BF\_T0\_DPLL\_LOCKED\_BW\_MASK, 108
- BF\_T0\_DPLL\_LOCKED\_BW\_SIZE, 108
- BF\_T0\_DPLL\_LOCKED\_DAMPING\_LSB, 108
- BF\_T0\_DPLL\_LOCKED\_DAMPING\_MASK, 108
- BF\_T0\_DPLL\_LOCKED\_DAMPING\_SIZE, 108
- BF\_T0\_DPLL\_OPERATING\_MODE\_LSB, 108
- BF\_T0\_DPLL\_OPERATING\_MODE\_MASK, 107
- BF\_T0\_DPLL\_OPERATING\_MODE\_SIZE, 107
- BF\_T0\_DPLL\_SOFT\_FREQ\_ALARM\_LSB, 107
- BF\_T0\_DPLL\_SOFT\_FREQ\_ALARM\_MASK, 107
- BF\_T0\_DPLL\_SOFT\_FREQ\_ALARM\_SIZE, 107
- BF\_T0\_DPLL\_START\_BW\_LSB, 108
- BF\_T0\_DPLL\_START\_BW\_MASK, 108
- BF\_T0\_DPLL\_START\_BW\_SIZE, 108
- BF\_T0\_DPLL\_START\_DAMPING\_LSB, 108
- BF\_T0\_DPLL\_START\_DAMPING\_MASK, 108
- BF\_T0\_DPLL\_START\_DAMPING\_SIZE, 108
- BF\_T0\_FOR\_T4\_LSB, 107
- BF\_T0\_FOR\_T4\_MASK, 107
- BF\_T0\_FOR\_T4\_SIZE, 107
- BF\_T0\_GSM\_OBSAI\_16E1\_16T1\_SEL\_LSB, 108
- BF\_T0\_GSM\_OBSAI\_16E1\_16T1\_SEL\_MASK,  
108

- BF\_T0\_GSM\_OBSAI\_16E1\_16T1\_SEL\_SIZE, 108
- BF\_T0\_INPUT\_SEL\_LSB, 107
- BF\_T0\_INPUT\_SEL\_MASK, 107
- BF\_T0\_INPUT\_SEL\_SIZE, 107
- BF\_T0\_LIMIT\_LSB, 109
- BF\_T0\_LIMIT\_MASK, 108
- BF\_T0\_LIMIT\_SIZE, 108
- BF\_T0\_MAIN\_REF\_FAIL\_LSB, 99
- BF\_T0\_MAIN\_REF\_FAIL\_MASK, 99
- BF\_T0\_MAIN\_REF\_FAIL\_SIZE, 99
- BF\_T0\_OPERATING\_MODE\_LSB, 108
- BF\_T0\_OPERATING\_MODE\_MASK, 108
- BF\_T0\_OPERATING\_MODE\_SIZE, 108
- BF\_T0\_OPERATING\_MODE\_STS\_LSB, 99
- BF\_T0\_OPERATING\_MODE\_STS\_MASK, 99
- BF\_T0\_OPERATING\_MODE\_STS\_SIZE, 99
- BF\_T0\_PH\_SLOPE\_LSB, 114
- BF\_T0\_PH\_SLOPE\_MASK, 114
- BF\_T0\_PH\_SLOPE\_SIZE, 114
- BF\_T0\_WDCO\_LSB, 108
- BF\_T0\_WDCO\_MASK, 108
- BF\_T0\_WDCO\_SIZE, 108
- BF\_T4\_12E1\_GPS\_E3\_T3\_SEL\_LSB, 110
- BF\_T4\_12E1\_GPS\_E3\_T3\_SEL\_MASK, 110
- BF\_T4\_12E1\_GPS\_E3\_T3\_SEL\_SIZE, 110
- BF\_T4\_APLL\_PATH\_LSB, 110
- BF\_T4\_APLL\_PATH\_MASK, 110
- BF\_T4\_APLL\_PATH\_SIZE, 110
- BF\_T4\_DFS\_OFF\_0\_LSB, 114
- BF\_T4\_DFS\_OFF\_0\_MASK, 114
- BF\_T4\_DFS\_OFF\_0\_SIZE, 114
- BF\_T4\_DFS\_OFF\_1\_LSB, 114
- BF\_T4\_DFS\_OFF\_1\_MASK, 114
- BF\_T4\_DFS\_OFF\_1\_SIZE, 114
- BF\_T4\_DFS\_OFF\_2\_LSB, 114
- BF\_T4\_DFS\_OFF\_2\_MASK, 114
- BF\_T4\_DFS\_OFF\_2\_SIZE, 114
- BF\_T4\_DFS\_OFF\_3\_LSB, 114
- BF\_T4\_DFS\_OFF\_3\_MASK, 114
- BF\_T4\_DFS\_OFF\_3\_SIZE, 114
- BF\_T4\_DPLL\_LOCK\_LSB, 107
- BF\_T4\_DPLL\_LOCK\_MASK, 107
- BF\_T4\_DPLL\_LOCK\_SIZE, 107
- BF\_T4\_DPLL\_LOCKED\_BW\_LSB, 110
- BF\_T4\_DPLL\_LOCKED\_BW\_MASK, 110
- BF\_T4\_DPLL\_LOCKED\_BW\_SIZE, 110
- BF\_T4\_DPLL\_LOCKED\_DAMPING\_LSB, 110
- BF\_T4\_DPLL\_LOCKED\_DAMPING\_MASK, 110
- BF\_T4\_DPLL\_LOCKED\_DAMPING\_SIZE, 110
- BF\_T4\_DPLL\_SOFT\_FREQ\_ALARM\_LSB, 107
- BF\_T4\_DPLL\_SOFT\_FREQ\_ALARM\_MASK, 107
- BF\_T4\_DPLL\_SOFT\_FREQ\_ALARM\_SIZE, 107
- BF\_T4\_GSM\_OBSAI\_16E1\_16T1\_SEL\_LSB, 110
- BF\_T4\_GSM\_OBSAI\_16E1\_16T1\_SEL\_MASK, 110
- BF\_T4\_GSM\_OBSAI\_16E1\_16T1\_SEL\_SIZE, 110
- BF\_T4\_INPUT\_FAIL\_LSB, 112
- BF\_T4\_INPUT\_FAIL\_MASK, 112
- BF\_T4\_INPUT\_FAIL\_SIZE, 112
- BF\_T4\_INPUT\_SEL\_LSB, 107
- BF\_T4\_INPUT\_SEL\_MASK, 107
- BF\_T4\_INPUT\_SEL\_SIZE, 107
- BF\_T4\_LOCK\_T0\_LSB, 107
- BF\_T4\_LOCK\_T0\_MASK, 107
- BF\_T4\_LOCK\_T0\_SIZE, 107
- BF\_T4\_OPERATING\_MODE\_LSB, 108
- BF\_T4\_OPERATING\_MODE\_MASK, 108
- BF\_T4\_OPERATING\_MODE\_SIZE, 108
- BF\_T4\_PH\_SLOPE\_LSB, 115
- BF\_T4\_PH\_SLOPE\_MASK, 114
- BF\_T4\_PH\_SLOPE\_SIZE, 114
- BF\_T4\_STS\_LSB, 100
- BF\_T4\_STS\_MASK, 100
- BF\_T4\_STS\_SIZE, 100
- BF\_T4\_T0\_SEL\_LSB, 97
- BF\_T4\_T0\_SEL\_MASK, 97
- BF\_T4\_T0\_SEL\_SIZE, 97
- BF\_T4\_TEST\_T0\_PH\_LSB, 107
- BF\_T4\_TEST\_T0\_PH\_MASK, 107
- BF\_T4\_TEST\_T0\_PH\_SIZE, 107
- BF\_T4\_WDCO\_LSB, 108
- BF\_T4\_WDCO\_MASK, 108
- BF\_T4\_WDCO\_SIZE, 108
- BF\_TEMP\_HOLD OVER\_MODE\_LSB, 109
- BF\_TEMP\_HOLD OVER\_MODE\_MASK, 109
- BF\_TEMP\_HOLD OVER\_MODE\_SIZE, 109
- BF\_THIRD\_HIGHEST\_PRIORITY\_VALIDATED\_LSB, 107
- BF\_THIRD\_HIGHEST\_PRIORITY\_VALIDATED\_MASK, 107
- BF\_THIRD\_HIGHEST\_PRIORITY\_VALIDATED\_SIZE, 107
- BF\_TIMEOUT\_VALUE\_LSB, 97
- BF\_TIMEOUT\_VALUE\_MASK, 97
- BF\_TIMEOUT\_VALUE\_SIZE, 97
- BF\_ULTR\_FAST\_SW\_LSB, 98
- BF\_ULTR\_FAST\_SW\_MASK, 98
- BF\_ULTR\_FAST\_SW\_SIZE, 98
- BF\_UPPER\_THRESHOLD\_DATA\_LSB, 102
- BF\_UPPER\_THRESHOLD\_DATA\_MASK, 102
- BF\_UPPER\_THRESHOLD\_DATA\_SIZE, 102
- BF\_VC XO\_SEL\_LSB, 116
- BF\_VC XO\_SEL\_MASK, 116
- BF\_VC XO\_SEL\_SIZE, 116
- BF\_WIDE\_EN\_LSB, 109
- BF\_WIDE\_EN\_MASK, 109
- BF\_WIDE\_EN\_SIZE, 109
- REG\_BUCKET\_SIZE\_0\_CNFG, 94
- REG\_BUCKET\_SIZE\_1\_CNFG, 95
- REG\_BUCKET\_SIZE\_2\_CNFG, 95
- REG\_BUCKET\_SIZE\_3\_CNFG, 95
- REG\_CLK\_SEL, 115
- REG\_CONFIG\_QA, 115
- REG\_CONFIG\_QB, 115

- REG\_CONTROL, 115
- REG\_CURRENT\_DPLL\_PHASE\_HIGH\_STS, 96
- REG\_CURRENT\_DPLL\_PHASE\_LOW\_STS, 96
- REG\_CURRENT\_FREQ\_HIGH\_STS, 96
- REG\_CURRENT\_FREQ\_LOW\_STS, 96
- REG\_CURRENT\_FREQ\_MID\_STS, 96
- REG\_DECAY\_RATE\_0\_CNFG, 95
- REG\_DECAY\_RATE\_1\_CNFG, 95
- REG\_DECAY\_RATE\_2\_CNFG, 95
- REG\_DECAY\_RATE\_3\_CNFG, 95
- REG\_DFS\_OFF\_CNFG, 96
- REG\_DIFFERENTIAL\_IN\_OUT\_CNFG, 94
- REG\_DPLL\_FREQ\_HARD\_LIMIT\_LOW\_CNFG, 96
- REG\_DPLL\_FREQ\_HARD\_LIMIT\_MID\_CNFG, 96
- REG\_DPLL\_FREQ\_SOFT\_LIMIT\_CNFG, 96
- REG\_FR\_MFR\_SYNC\_CNFG, 96
- REG\_FREQ\_MON\_FACTOR\_CNFG, 94
- REG\_HARD\_FREQ\_MON\_THRESHOLD\_CNFG, 94
- REG\_HOLDOVER\_FREQ\_HIGH\_CNFG, 96
- REG\_HOLDOVER\_FREQ\_LOW\_CNFG, 96
- REG\_HOLDOVER\_FREQ\_MID\_CNFG, 96
- REG\_ID\_HIGH, 93
- REG\_ID\_LOW, 93
- REG\_IN10\_CNFG, 94
- REG\_IN11\_CNFG, 94
- REG\_IN11\_IN12\_SEL\_PRIORITY\_CNFG, 94
- REG\_IN11\_IN12\_STS, 95
- REG\_IN12\_CNFG, 94
- REG\_IN13\_CNFG, 94
- REG\_IN13\_IN14\_SEL\_PRIORITY\_CNFG, 94
- REG\_IN13\_IN14\_STS, 95
- REG\_IN14\_CNFG, 94
- REG\_IN1\_CNFG, 94
- REG\_IN1\_IN2\_SEL\_PRIORITY\_CNFG, 94
- REG\_IN1\_IN2\_STS, 95
- REG\_IN2\_CNFG, 94
- REG\_IN3\_CNFG, 94
- REG\_IN3\_IN4\_SEL\_PRIORITY\_CNFG, 94
- REG\_IN3\_IN4\_STS, 95
- REG\_IN4\_CNFG, 94
- REG\_IN5\_CNFG, 94
- REG\_IN5\_IN6\_HF\_DIV\_CNFG, 94
- REG\_IN5\_IN6\_SEL\_PRIORITY\_CNFG, 94
- REG\_IN5\_IN6\_STS, 95
- REG\_IN6\_CNFG, 94
- REG\_IN7\_CNFG, 94
- REG\_IN7\_IN8\_SEL\_PRIORITY\_CNFG, 94
- REG\_IN7\_IN8\_STS, 95
- REG\_IN8\_CNFG, 94
- REG\_IN9\_CNFG, 94
- REG\_IN9\_IN10\_SEL\_PRIORITY\_CNFG, 94
- REG\_IN9\_IN10\_STS, 95
- REG\_IN\_FREQ\_READ\_CH\_CNFG, 95
- REG\_IN\_FREQ\_READ\_STS, 95
- REG\_INPUT\_MODE\_CNFG, 93
- REG\_INPUT\_VALID1\_STS, 95
- REG\_INPUT\_VALID2\_STS, 95
- REG\_INTERRUPT\_CNFG, 94
- REG\_INTERRUPTS1\_MASK\_CNFG, 94
- REG\_INTERRUPTS1\_STS, 94
- REG\_INTERRUPTS2\_MASK\_CNFG, 94
- REG\_INTERRUPTS2\_STS, 94
- REG\_INTERRUPTS3\_MASK\_CNFG, 94
- REG\_INTERRUPTS3\_STS, 94
- REG\_LOWER\_THRESHOLD\_0\_CNFG, 94
- REG\_LOWER\_THRESHOLD\_1\_CNFG, 95
- REG\_LOWER\_THRESHOLD\_2\_CNFG, 95
- REG\_LOWER\_THRESHOLD\_3\_CNFG, 95
- REG\_M0\_LSB, 115
- REG\_M0\_MSB, 115
- REG\_M1\_LSB, 115
- REG\_M1\_MSB, 115
- REG\_MON\_SW\_HS\_CNFG, 94
- REG\_MPU\_PIN\_STS, 93
- REG\_MPU\_SEL\_CNFG, 96
- REG\_MS\_SL\_CTRL\_CNFG, 94
- REG\_NOMINAL\_FREQ\_HIGH\_CNFG, 93
- REG\_NOMINAL\_FREQ\_LOW\_CNFG, 93
- REG\_NOMINAL\_FREQ\_MID\_CNFG, 93
- REG\_ODBSELA0, 115
- REG\_ODBSELA1, 115
- REG\_ODBSELB0, 115
- REG\_ODBSELB1, 115
- REG\_OPERATING\_STS, 95
- REG\_OUT1\_FREQ\_CNFG, 96
- REG\_OUT2\_FREQ\_CNFG, 96
- REG\_OUT3\_FREQ\_CNFG, 96
- REG\_OUT4\_FREQ\_CNFG, 96
- REG\_OUT5\_FREQ\_CNFG, 96
- REG\_OUT6\_FREQ\_CNFG, 96
- REG\_OUT7\_FREQ\_CNFG, 96
- REG\_OUT8\_FREQ\_CNFG, 96
- REG\_OUT9\_FREQ\_CNFG, 96
- REG\_PAGE\_POINTER\_CNFG, 94
- REG\_PDSEL0\_LSB, 115
- REG\_PDSEL0\_MSB, 115
- REG\_PDSEL1\_LSB, 115
- REG\_PDSEL1\_MSB, 115
- REG\_PH\_SLOPE\_CNFG, 96
- REG\_PHASE\_ALARM\_TIME\_OUT\_CNFG, 93
- REG\_PHASE\_LOSS\_COARSE\_LIMIT\_CNFG, 95
- REG\_PHASE\_LOSS\_FINE\_LIMIT\_CNFG, 95
- REG\_PHASE\_MON\_CNFG, 96
- REG\_PHASE\_OFFSET\_HIGH\_CNFG, 96
- REG\_PHASE\_OFFSET\_LOW\_CNFG, 96
- REG\_PRE\_DIV\_CH\_CNFG, 94
- REG\_PRE\_DIVN\_HIGH\_CNFG, 94
- REG\_PRE\_DIVN\_LOW\_CNFG, 94
- REG\_PRIORITY\_TABLE1\_STS, 95
- REG\_PRIORITY\_TABLE2\_STS, 95
- REG\_PROTECTION\_CNFG, 96
- REG\_REMOTE\_INPUT\_VALID1\_CNFG, 95
- REG\_REMOTE\_INPUT\_VALID2\_CNFG, 95

- REG\_SOFT\_FREQ\_MON\_THRESHOLD\_CNFG, 94
- REG\_SYNC\_MONITOR\_CNFG, 96
- REG\_SYNC\_PHASE\_CNFG, 96
- REG\_T0\_BW\_OVERSHOOT\_CNFG, 95
- REG\_T0\_DPLL\_ACQ\_BW\_DAMPING\_CNFG, 95
- REG\_T0\_DPLL\_APLL\_PATH\_CNFG, 95
- REG\_T0\_DPLL\_LOCKED\_BW\_DAMPING\_CNFG, 95
- REG\_T0\_DPLL\_START\_BW\_DAMPING\_CNFG, 95
- REG\_T0\_HOLDOVER\_MODE\_CNFG, 95
- REG\_T0\_ICP\_CTRL\_CNFG, 96
- REG\_T0\_INPUT\_SEL\_CNFG, 95
- REG\_T0\_OPERATION\_MODE\_CNFG, 95
- REG\_T0\_PPS\_CNFG, 96
- REG\_T4\_DPLL\_APLL\_PATH\_CNFG, 96
- REG\_T4\_DPLL\_LOCKED\_BW\_DAMPING\_CNFG, 96
- REG\_T4\_ICP\_CTRL\_CNFG, 96
- REG\_T4\_INPUT\_SEL\_CNFG, 95
- REG\_T4\_OPERATION\_MODE\_CNFG, 95
- REG\_T4\_PPS\_CNFG, 96
- REG\_T4\_T0\_SEL\_CNFG, 93
- REG\_UPPER\_THRESHOLD\_0\_CNFG, 94
- REG\_UPPER\_THRESHOLD\_1\_CNFG, 95
- REG\_UPPER\_THRESHOLD\_2\_CNFG, 95
- REG\_UPPER\_THRESHOLD\_3\_CNFG, 95
- REG\_VCXO\_SEL, 115
- REGMAP\_APLL\_MAX, 117
- REGMAP\_MAX, 115
- required\_argument
  - main.c, 196
- SYNC\_1PPS
  - Input Function Module, 67
- SYNC\_2kHz
  - Input Function Module, 67
- SYNC\_4kHz
  - Input Function Module, 67
- SYNC\_8kHz
  - Input Function Module, 67
- SYNC\_MAX
  - Input Function Module, 67
- SAMPLE\_MAX\_FILENAME
  - main.c, 196
- SET\_BIT
  - outputFreq\_priv.h, 234
- SIM\_REG\_MAP\_SIZE
  - ApplReadWrite\_sim.c, 206
- sample.c
  - IDTSample\_Configure1Pps, 199
  - IDTSample\_Configure1PpsHoldover, 199
  - IDTSample\_ConfigureDcoMode, 199
  - IDTSample\_ConfigureSampleConfig1, 199
  - IDTSample\_ConfigureSampleConfig2, 200
  - IDTSample\_ConfigureSampleConfig3, 200
  - IDTSample\_ConfigureSampleConfig4, 200
  - IDTSample\_ConfigureSampleConfig5, 201
  - IDTSample\_GetCombinedDcoOffset, 201
  - IDTSample\_SetCombinedDcoOffset, 201
- sample.h
  - IDTSample\_Configure1Pps, 183
  - IDTSample\_Configure1PpsHoldover, 183
  - IDTSample\_ConfigureDcoMode, 183
  - IDTSample\_ConfigureSampleConfig1, 183
  - IDTSample\_ConfigureSampleConfig2, 183
  - IDTSample\_ConfigureSampleConfig3, 184
  - IDTSample\_ConfigureSampleConfig4, 184
  - IDTSample\_ConfigureSampleConfig5, 184
  - IDTSample\_GetCombinedDcoOffset, 184
  - IDTSample\_SetCombinedDcoOffset, 185
- sampleAppl.c
  - IDTSample\_ConfigureSampleConfigOutput6\_156-dot\_25MHz, 202
  - IDTSample\_ConfigureSampleConfigureOutput10-25\_MHz, 203
  - IDTSample\_ConfigureSampleConfigureOutput7-77-dot\_76MHz, 203
  - IDTSample\_DualMode, 203
  - IDTSample\_GPS, 203
  - IDTSample\_GPS\_1PPS, 204
  - IDTSample\_Sonet, 204
  - IDTSample\_SyncE\_LAN, 204
  - IDTSample\_SyncE\_Sonet, 205
  - IDTSample\_SyncE\_WAN, 205
- sampleAppl.h
  - IDTSample\_ConfigureSampleConfigOutput6\_156-dot\_25MHz, 186
  - IDTSample\_ConfigureSampleConfigureOutput10-25\_MHz, 186
  - IDTSample\_ConfigureSampleConfigureOutput7-77-dot\_76MHz, 186
  - IDTSample\_DualMode, 187
  - IDTSample\_GPS, 187
  - IDTSample\_GPS\_1PPS, 187
  - IDTSample\_Sonet, 187
  - IDTSample\_SyncE\_LAN, 188
  - IDTSample\_SyncE\_Sonet, 188
  - IDTSample\_SyncE\_WAN, 188
- sampleConfig\_m
  - globalArgs\_t, 150
- Selector\_Any
  - outputFreq\_priv.h, 235
- Selector\_MAX
  - outputFreq\_priv.h, 235
- Selector\_Network
  - outputFreq\_priv.h, 235
- Selector\_SonetGigEth
  - outputFreq\_priv.h, 235
- Selector\_T0DpllSelA
  - outputFreq\_priv.h, 235
- Selector\_T0DpllSelB
  - outputFreq\_priv.h, 235
- Selector\_T4DpllSelA
  - outputFreq\_priv.h, 235
- Selector\_T4DpllSelB
  - outputFreq\_priv.h, 235

- outputFreq\_priv.h, [235](#)
- selector\_m
  - pathSelectorConfig\_t, [152](#)
- shutOff
  - T\_idtApIConfig, [158](#)
- sigType
  - T\_idtApIConfigPerOutput, [160](#)
- SimAccessMethods
  - access.c, [192](#)
- simDebugFlag
  - main.c, [198](#)
  - ReadWrite\_sim.c, [224](#)
- simulatedMemMap
  - ReadWrite\_sim.c, [225](#)
- simulatedRegMapApI0
  - ApIReadWrite\_sim.c, [207](#)
- simulatedRegMapApI1
  - ApIReadWrite\_sim.c, [207](#)
- size\_m
  - pathSelectorIndex\_t, [153](#)
- SonetGigEthOutput\_10GbE
  - DPLL Function Module, [21](#)
- SonetGigEthOutput\_10GbE\_6664
  - DPLL Function Module, [21](#)
- SonetGigEthOutput\_MAX
  - DPLL Function Module, [21](#)
- SonetGigEthOutput\_Sonet
  - DPLL Function Module, [21](#)
- SonetGigEthSel\_MAX
  - DPLL Private header file, [8](#)
- SonetGigEthSel\_T0DpII10GbE
  - DPLL Private header file, [8](#)
- SonetGigEthSel\_T0DpII10GbE\_6664
  - DPLL Private header file, [8](#)
- SonetGigEthSel\_T0DpIISonet
  - DPLL Private header file, [8](#)
- SonetGigEthSel\_T4DpII10GbE
  - DPLL Private header file, [8](#)
- SonetGigEthSel\_T4DpII10GbE\_6664
  - DPLL Private header file, [8](#)
- SonetGigEthSel\_T4DpIISonet
  - DPLL Private header file, [8](#)
- sonetGigeMode\_AllFromDpII1
  - outputFreq.h, [230](#)
- sonetGigeMode\_AllFromDpII2
  - outputFreq.h, [230](#)
- sonetGigeMode\_MAX
  - outputFreq.h, [230](#)
- sonetGigeMode\_Matching
  - outputFreq.h, [230](#)
- SonetGigEthSel\_NUMMAX
  - DPLL Private header file, [8](#)
- sonetGigEthSel\_ma
  - T\_IDTPathConfiguration, [172](#)
- start\_m
  - pathSelectorIndex\_t, [153](#)
- supportedXtalFreq\_ma
  - T\_idtDrvDeviceConfig, [167](#)
- supportedXtalId\_ma
  - T\_idtDrvDeviceConfig, [167](#)
- T0DpIISelA\_16E1
  - DPLL Private header file, [8](#)
- T0DpIISelA\_16T1
  - DPLL Private header file, [8](#)
- T0DpIISelA\_GSM
  - DPLL Private header file, [8](#)
- T0DpIISelA\_MAX
  - DPLL Private header file, [8](#)
- T0DpIISelA\_OBSAI
  - DPLL Private header file, [8](#)
- T0DpIISelB\_12E1
  - DPLL Private header file, [9](#)
- T0DpIISelB\_E3
  - DPLL Private header file, [9](#)
- T0DpIISelB\_GPS
  - DPLL Private header file, [9](#)
- T0DpIISelB\_MAX
  - DPLL Private header file, [9](#)
- T0DpIISelB\_T3
  - DPLL Private header file, [9](#)
- t0PIIMode\_m
  - T\_idtDrvDeviceConfig, [167](#)
- T0SelA\_m
  - T\_IDTPathConfiguration, [172](#)
- T0SelB\_m
  - T\_IDTPathConfiguration, [172](#)
- T4DpIISelA\_16E1
  - DPLL Private header file, [9](#)
- T4DpIISelA\_16T1
  - DPLL Private header file, [9](#)
- T4DpIISelA\_GPS
  - DPLL Private header file, [9](#)
- T4DpIISelA\_GSM
  - DPLL Private header file, [9](#)
- T4DpIISelA\_MAX
  - DPLL Private header file, [9](#)
- T4DpIISelB\_12E1
  - DPLL Private header file, [9](#)
- T4DpIISelB\_24T1
  - DPLL Private header file, [9](#)
- T4DpIISelB\_E3
  - DPLL Private header file, [9](#)
- T4DpIISelB\_MAX
  - DPLL Private header file, [9](#)
- T4DpIISelB\_T3
  - DPLL Private header file, [9](#)
- T4SelA\_m
  - T\_IDTPathConfiguration, [173](#)
- T4SelB\_m
  - T\_IDTPathConfiguration, [173](#)
- T\_DpIIProfile
  - main.c, [197](#)
- T\_DpIIProfileFunc
  - main.c, [197](#)
- T\_IDTFreq2bfVal, [169](#)
  - bfVal\_m, [169](#)

- freq\_m, 169
- T\_IDTPathConfiguration, 172
  - networkType\_m, 172
  - sonetGigEthSel\_ma, 172
  - T0SelA\_m, 172
  - T0SelB\_m, 172
  - T4SelA\_m, 173
  - T4SelB\_m, 173
- T\_SampleConfigDescrFunc
  - main.c, 197
- T\_SampleConfigEntry, 174
  - configDescription\_m, 174
  - configFunction\_m, 174
- T\_SampleConfigFunc
  - main.c, 197
- T\_apllOutputFrequencyEntry, 156
  - apllDivVal\_m, 157
  - frequencies\_m, 157
- T\_externalSyncAlarmRange
  - Input Function Module, 63
- T\_externalSyncBypass
  - Input Function Module, 64
- T\_externalSyncEdge
  - Input Function Module, 64
- T\_externalSyncEnable
  - Input Function Module, 64
- T\_externalSyncSampling
  - Input Function Module, 64
- T\_frameSync
  - Output Function Module, 79
- T\_idtAccessDeInitFunc
  - Define.h, 258
- T\_idtAccessInitFunc
  - Define.h, 258
- T\_idtAmiInputFrequencyBitFieldValue
  - Input Function Module, 65
- T\_idtApplAccessType
  - Appl Function Module, 124
- T\_idtApplBypass
  - Appl Function Module, 124
- T\_idtApplConfig, 157
  - apllCfg, 158
  - byPass, 158
  - dbleSel, 158
  - fbDiv, 158
  - inputClk, 158
  - mrSync, 158
  - outputConfig, 158
  - outputDiv, 158
  - preDiv, 158
  - shutOff, 158
  - xtal, 158
- T\_idtApplConfig::fbDiv\_s, 148
  - fbDivConfig0, 148
  - fbDivConfig1, 148
- T\_idtApplConfig::outputConfig\_s, 151
  - qaConfig, 151
  - qbConfig, 151
- T\_idtApplConfig::outputConfig\_s::qaConfig\_s, 154
  - enOutput, 154
  - outputSigType, 154
- T\_idtApplConfig::outputConfig\_s::qbConfig\_s, 155
  - enOutput, 156
  - outputSigType, 156
- T\_idtApplConfig::outputDiv\_s, 151
  - qaDiv, 152
  - qbDiv, 152
- T\_idtApplConfig::outputDiv\_s::qaDiv\_s, 155
  - outputDivConfig0, 155
  - outputDivConfig1, 155
- T\_idtApplConfig::outputDiv\_s::qbDiv\_s, 156
  - outputDivConfig0, 156
  - outputDivConfig1, 156
- T\_idtApplConfig::preDiv\_s, 153
  - preDivConfig0, 154
  - preDivConfig1, 154
- T\_idtApplConfigPerOutput, 159
  - apllCfg, 159
  - dbleSel, 159
  - enOutput, 159
  - fbDiv, 159
  - inputClk, 159
  - outputDiv, 160
  - preDiv, 160
  - sigType, 160
  - xtal, 160
- T\_idtApplConfigSel
  - Appl Function Module, 124
- T\_idtApplDividerValue
  - Appl Function Module, 123
- T\_idtApplFreqDoublerSel
  - Appl Function Module, 124
- T\_idtApplFreqMapping, 160
  - apllFbDivider, 161
  - apllInputFreq, 161
  - apllOutputDivider, 161
  - apllOutputFreq, 161
  - apllOutputs, 161
  - apllPreDivider, 161
  - apllXtalFreq, 161
- T\_idtApplId
  - Appl Function Module, 125
- T\_idtApplInputClkSrc
  - Appl Function Module, 125
- T\_idtApplInputFreqType
  - Appl Function Module, 125
- T\_idtApplMasterResetSync
  - Appl Function Module, 125
- T\_idtApplOutput
  - Appl Function Module, 126
- T\_idtApplOutputEn
  - Appl Function Module, 126
- T\_idtApplOutputFreqType
  - Appl Function Module, 126
- T\_idtApplOutputSigType
  - Appl Function Module, 127

- T\_idtApIIOutputSupportedType
  - ApIIO Function Module, 127
- T\_idtApIIQaDiv
  - ApIIO Function Module, 127
- T\_idtApIIQbDiv
  - ApIIO Function Module, 127
- T\_idtApIIRegAddr
  - ApIIO Function Module, 123
- T\_idtApIIRegMask
  - ApIIO Function Module, 123
- T\_idtApIIRegVal
  - ApIIO Function Module, 124
- T\_idtApIIShutoff
  - ApIIO Function Module, 128
- T\_idtApIIXtal, 161
  - apIIXtalFreq, 162
  - apIIXtalId, 162
- T\_idtApIIXtalId
  - ApIIO Function Module, 128
- T\_idtApIIXtalList, 162
  - xtal1ApI1, 163
  - xtal1ApI1Freq, 163
  - xtal2ApI2, 163
  - xtal2ApI2Freq, 163
  - xtal3ApI1, 163
  - xtal3ApI1Freq, 163
  - xtal4ApI2, 163
  - xtal4ApI2Freq, 163
- T\_idtApIIXtalType
  - ApIIO Function Module, 128
- T\_idtBandwidthLimitStage
  - DPLL Function Module, 16
- T\_idtCoarsePhaseLimit
  - DPLL Function Module, 16
- T\_idtDampingFactor
  - DPLL Function Module, 16
- T\_idtDeviceType
  - Define.h, 260
- T\_idtDpllBandwidth
  - DPLL Function Module, 17
- T\_idtDpllHoldover
  - DPLL Function Module, 17
- T\_idtDpllInstance
  - Define.h, 260
- T\_idtDpllOperMode
  - DPLL Function Module, 18
- T\_idtDpllOutputSelectorA
  - DPLL Function Module, 18
- T\_idtDpllOutputSelectorB
  - DPLL Function Module, 18
- T\_idtDpllPpsPhasePostion
  - DPLL Function Module, 19
- T\_idtDpllPpsPulseWidth
  - DPLL Function Module, 19
- T\_idtDpllType
  - Define.h, 261
- T\_idtDrvAccess, 164
  - deinitFunc\_m, 164
  - initFunc\_m, 164
  - readFunc\_m, 164
  - userData\_m, 164
  - writeFunc\_m, 164
- T\_idtDrvDeviceConfig, 165
  - dcoModeSupport\_ma, 165
  - inSyncXlate\_ma, 165
  - inputExt2IntXlate\_ma, 165
  - inputFreqValidFunc\_m, 165
  - maxNumXtalPerApI\_m, 166
  - numberOfApI\_m, 166
  - numberOfSonetGigEthPath\_m, 166
  - outPutApIConfig, 166
  - outSyntXlate\_ma, 166
  - outputExt2IntXlate\_ma, 166
  - outputFreqValidFunc\_m, 166
  - phaseSlopeLimiting\_ma, 166
  - pllExt2IntXlate\_ma, 166
  - pllInt2ExtXlate\_ma, 167
  - populatedApI\_ma, 167
  - supportedXtalFreq\_ma, 167
  - supportedXtalId\_ma, 167
  - tOPllMode\_m, 167
- T\_idtDrvHdlr, 167
- T\_idtFinePhaseLimit
  - DPLL Function Module, 19
- T\_idtFreqMonFactor
  - DPLL Function Module, 20
- T\_idtHfDivEntry, 170
  - divValue\_m, 170
  - hfDivValue\_m, 170
- T\_idtHighFrequencyDivisor
  - Input Function Module, 65
- T\_idtI2CaddrTransFunc
  - Define.h, 259
- T\_idtI2CtransData, 170
  - bitLocation, 171
  - bitValue, 171
- T\_idtInputDividerMux
  - Input Function Module, 65
- T\_idtInputFreqValid
  - Define.h, 259
- T\_idtInputFrequencyBitfieldValue
  - Input Function Module, 65
- T\_idtInputFrequencyFamily
  - Input Function Module, 66
- T\_idtInputSyncFreq
  - Input Function Module, 66
- T\_idtOperDpllMode
  - DPLL Function Module, 20
- T\_idtOuputFreqValid
  - Define.h, 259
- T\_idtOutputApIConfig, 171
  - apIInput, 171
  - apIInst, 171
  - apIOutput, 171
  - extOutput, 172
- T\_idtOutputPathBfVal

- Output.c, [298](#)
- T\_idtPhaseSlope
  - DPLL Function Module, [20](#)
- T\_idtReadFunc
  - Define.h, [260](#)
- T\_idtSonetGigEthBitfield2PathConfig, [173](#)
  - inputDpll\_m, [173](#)
  - outputFreq\_m, [173](#)
- T\_idtSonetGigEthOutput
  - DPLL Function Module, [21](#)
- T\_idtSonetGigEthSelector
  - DPLL Private header file, [8](#)
- T\_idtT0DpllSelectorA
  - DPLL Private header file, [8](#)
- T\_idtT0DpllSelectorB
  - DPLL Private header file, [8](#)
- T\_idtT4DpllSelectorA
  - DPLL Private header file, [9](#)
- T\_idtT4DpllSelectorB
  - DPLL Private header file, [9](#)
- T\_idtTemp\_holdover
  - DPLL Function Module, [21](#)
- T\_idtWriteFunc
  - Define.h, [260](#)
- T\_outFreqLookupIdx
  - outputFreqString.c, [246](#)
- T\_outputInterface
  - Output Function Module, [80](#)
- T\_outputPathType
  - Output Function Module, [80](#)
- T\_sonetGigeMode
  - outputFreq.h, [230](#)
- Temp\_Holdover\_Fast\_Average
  - DPLL Function Module, [21](#)
- Temp\_Holdover\_Instantaneous
  - DPLL Function Module, [21](#)
- Temp\_Holdover\_MAX
  - DPLL Function Module, [21](#)
- Temp\_Holdover\_Slow\_Average
  - DPLL Function Module, [21](#)
- Temp\_Same\_As\_Full\_Holdover
  - DPLL Function Module, [21](#)
- upperThreshReg\_m
  - bucketReg\_t, [147](#)
- userData\_m
  - T\_idtDrvAccess, [164](#)
- validFreq\_t
  - outputFreq\_priv.h, [234](#)
- value\_m
  - pathSelectorConfig\_t, [153](#)
- WRByte\_Sim
  - ReadWrite\_sim.c, [224](#)
  - ReadWrite\_sim.h, [225](#)
- WriteByte\_SimWrapper
  - access.c, [192](#)
- writeFunc\_m
  - T\_idtDrvAccess, [164](#)
- xtal
  - T\_idtApIConfig, [158](#)
  - T\_idtApIConfigPerOutput, [160](#)
- xtal1ApI1
  - T\_idtApIXtalList, [163](#)
- xtal1ApI1Freq
  - T\_idtApIXtalList, [163](#)
- xtal2ApI2
  - T\_idtApIXtalList, [163](#)
- xtal2ApI2Freq
  - T\_idtApIXtalList, [163](#)
- xtal3ApI1
  - T\_idtApIXtalList, [163](#)
- xtal3ApI1Freq
  - T\_idtApIXtalList, [163](#)
- xtal4ApI2
  - T\_idtApIXtalList, [163](#)
- xtal4ApI2Freq
  - T\_idtApIXtalList, [163](#)
- xtalList
  - drvHdlr\_s, [169](#)
  - globalArgs\_t, [150](#)