

82V3911 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	- Add power saving settings. - Add support for APLL pre-divider value 1. - Update Ethernet FEC frequency configuration.

Contents

1	Module Index	1
1.1	Modules	1
2	Data Structure Index	3
2.1	Data Structures	3
3	File Index	5
3.1	File List	5
4	Module Documentation	7
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_DisableApIInput	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	ApI 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	IDTApll_GetInputClkSrc	132
4.8.5.15	IDTApll_GetMasterResetSync	133
4.8.5.16	IDTApll_GetNonActiveApllConfig	133
4.8.5.17	IDTApll_GetOutputEnState	134
4.8.5.18	IDTApll_GetOutputSigType	134
4.8.5.19	IDTApll_GetPreDiv	135
4.8.5.20	IDTApll_GetPreDivRange	135
4.8.5.21	IDTApll_GetQaDiv	135
4.8.5.22	IDTApll_GetQbDiv	136
4.8.5.23	IDTApll_GetShutoff	136
4.8.5.24	IDTApll_GetSupportedXtal	137
4.8.5.25	IDTApll_GetXtalId	137
4.8.5.26	IDTApll_GetXtalIdFromXtalFreq	138
4.8.5.27	IDTApll_NullFreqTableEntry	138
4.8.5.28	IDTApll_SetApllConfig	138
4.8.5.29	IDTApll_SetApllConfigPerOutput	138
4.8.5.30	IDTApll_SetBypass	139
4.8.5.31	IDTApll_SetConfigSel	139
4.8.5.32	IDTApll_SetFeedbackDiv	139
4.8.5.33	IDTApll_SetFreqDoublerSel	140
4.8.5.34	IDTApll_SetInputClkSrc	140
4.8.5.35	IDTApll_SetMasterResetSync	141
4.8.5.36	IDTApll_SetOutputEnState	141
4.8.5.37	IDTApll_SetOutputSigType	141
4.8.5.38	IDTApll_SetPreDiv	142
4.8.5.39	IDTApll_SetQaDiv	142
4.8.5.40	IDTApll_SetQbDiv	143
4.8.5.41	IDTApll_SetShutoff	143
4.8.5.42	IDTApll_SetXtalId	144
4.8.5.43	IDTApll_Switch2NonActiveApllConfig	144
4.8.5.44	IDTApll_ValidXtalInput	144
4.8.5.45	IDTApll_XtalConfigured	145
5	Data Structure Documentation	147
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
6	File Documentation	175
6.1	82V3911/dev/include/idt82V3911.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	82V3911/dev/src/82V3911.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	idt82V3911Config	178
6.3	82V3911/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	82V3911/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	82V3911/sample/include/sample.h File Reference	182
6.5.1	Function Documentation	182
6.5.1.1	IDTSample_ConfigureSampleConfig1	182
6.5.1.2	IDTSample_ConfigureSampleConfig2	183
6.5.1.3	IDTSample_ConfigureSampleConfig3	183

6.5.1.4	IDTSample_ConfigureSampleConfig4	183
6.5.1.5	IDTSample_ConfigureSampleConfig5	184
6.5.1.6	IDTSample_ConfigureSampleConfig6	184
6.5.1.7	IDTSample_ConfigureSampleConfig7	184
6.6	82V3911/sample/src/access.c File Reference	185
6.6.1	Function Documentation	186
6.6.1.1	DeInit_SimWrapper	186
6.6.1.2	IDTSample_GetLogVerbosity	186
6.6.1.3	IDTSample_InitDriverI2c	186
6.6.1.4	IDTSample_InitDriverSimulator	186
6.6.1.5	IDTSample_ResetXtalList	186
6.6.1.6	IDTSample_SetLogVerbosity	187
6.6.1.7	Init_SimWrapper	187
6.6.1.8	ReadByte_SimWrapper	187
6.6.1.9	WriteByte_SimWrapper	187
6.6.2	Variable Documentation	188
6.6.2.1	I2cAccessMethods	188
6.6.2.2	SimAccessMethods	188
6.7	82V3911/sample/src/loadstore.c File Reference	188
6.7.1	Function Documentation	189
6.7.1.1	IDTSample_LoadRgfFile	189
6.7.1.2	IDTSample_SaveRgfFile	189
6.8	82V3911/sample/src/main.c File Reference	189
6.8.1	Macro Definition Documentation	191
6.8.1.1	MODE_DpllProfile	191
6.8.1.2	MODE_Input	191
6.8.1.3	MODE_Load	191
6.8.1.4	MODE_None	191
6.8.1.5	MODE_Output	191
6.8.1.6	MODE_Read	192
6.8.1.7	MODE_Sample	192
6.8.1.8	MODE_Save	192
6.8.1.9	MODE_Write	192
6.8.1.10	no_argument	192
6.8.1.11	optional_argument	192
6.8.1.12	required_argument	192
6.8.1.13	SAMPLE_MAX_FILENAME	192
6.8.2	Typedef Documentation	192
6.8.2.1	T_DpllProfileFunc	192
6.8.2.2	T_SampleConfigDescrFunc	192

6.8.2.3	T_SampleConfigFunc	193
6.8.3	Enumeration Type Documentation	193
6.8.3.1	T_DpllProfile	193
6.8.4	Function Documentation	193
6.8.4.1	IDTSample_DumpRegistersRgf	193
6.8.4.2	main	193
6.8.4.3	printHelp	193
6.8.4.4	processCmdLineArguments	193
6.8.5	Variable Documentation	193
6.8.5.1	globalArgs	193
6.8.5.2	IDTHIApi_Frequency2String_c	193
6.8.5.3	simDebugFlag	194
6.9	82V3911/sample/src/sample.c File Reference	194
6.9.1	Function Documentation	194
6.9.1.1	IDTSample_ConfigureSampleConfig1	195
6.9.1.2	IDTSample_ConfigureSampleConfig2	195
6.9.1.3	IDTSample_ConfigureSampleConfig3	195
6.9.1.4	IDTSample_ConfigureSampleConfig4	195
6.9.1.5	IDTSample_ConfigureSampleConfig5	195
6.9.1.6	IDTSample_ConfigureSampleConfig6	195
6.9.1.7	IDTSample_ConfigureSampleConfig7	195
6.10	apll/access/sim/ApllReadWrite_sim.c File Reference	195
6.10.1	Macro Definition Documentation	196
6.10.1.1	SIM_REG_MAP_SIZE	196
6.10.2	Function Documentation	196
6.10.2.1	ApllDeInit_Sim	196
6.10.2.2	ApllInit_Sim	197
6.10.2.3	ApllRead_Sim	197
6.10.2.4	ApllWrite_Sim	197
6.10.3	Variable Documentation	197
6.10.3.1	simulatedRegMapApll0	197
6.10.3.2	simulatedRegMapApll1	197
6.11	apll/access/sim/ApllReadWrite_sim.h File Reference	197
6.11.1	Function Documentation	198
6.11.1.1	ApllDeInit_Sim	198
6.11.1.2	ApllInit_Sim	198
6.11.1.3	ApllRead_Sim	198
6.11.1.4	ApllWrite_Sim	198
6.12	apll/include/ApllApi.h File Reference	198
6.13	apll/include/ApllFreqProfile.h File Reference	200

6.14	apll/include/ApllLogging.h File Reference	202
6.15	apll/include/ApllRegDef.h File Reference	202
6.16	apll/include/ApllTypeDef.h File Reference	203
6.17	apll/src/ApllApi.c File Reference	205
6.17.1	Macro Definition Documentation	207
6.17.1.1	APLL1_REG_HIGH_BIT	207
6.17.1.2	APLL1_REG_VALID_OFFSET	207
6.17.1.3	APLL2_REG_HIGH_BIT	208
6.17.1.4	APLL2_REG_VALID_OFFSET	208
6.17.1.5	APLL_DIVIDER_LSB	208
6.17.1.6	APLL_DIVIDER_MSB	208
6.17.1.7	APLL_DIVIDER_VAL	208
6.17.1.8	APLL_FB_DIV_LSB	208
6.17.1.9	APLL_FB_DIV_MAX	208
6.17.1.10	APLL_FB_DIV_MIN	208
6.17.1.11	APLL_FB_DIV_MSB	208
6.17.1.12	APLL_FB_DIV_VAL	208
6.17.1.13	APLL_PRE_DIV_LSB	209
6.17.1.14	APLL_PRE_DIV_MAX	209
6.17.1.15	APLL_PRE_DIV_MIN	209
6.17.1.16	APLL_PRE_DIV_MSB	209
6.17.1.17	APLL_PRE_DIV_VAL	209
6.17.1.18	APLL_REG_HIGH_BIT_OFFSET	209
6.17.1.19	APLL_REG_OFFSET	209
6.17.1.20	INVALID_XTAL_SEL_VALUE	209
6.18	apll/src/ApllFreqProfile.c File Reference	209
6.18.1	Macro Definition Documentation	211
6.18.1.1	APLL_FB_DIV_ETH	211
6.18.1.2	APLL_FB_DIV_ETH_FEC	211
6.18.1.3	APLL_FB_DIV_SONET	211
6.18.1.4	APLL_OUTPUT_DIV_1	211
6.18.1.5	APLL_OUTPUT_DIV_2	211
6.18.1.6	APLL_OUTPUT_DIV_25	211
6.18.1.7	APLL_OUTPUT_DIV_4	211
6.18.1.8	APLL_OUTPUT_DIV_5	211
6.18.1.9	APLL_OUTPUT_DIV_8	211
6.18.1.10	APLL_PRE_DIV_ETH	211
6.18.1.11	APLL_PRE_DIV_ETH_FEC	211
6.18.1.12	APLL_PRE_DIV_SONET	212
6.18.1.13	MAX_CONFIG_COMBO_PER_FREQ	212

6.18.1.14	MAX_CONFIG_COMBO_PER_VC XO_TYPE	212
6.18.1.15	NULL_APLL_FREQ_TABLE_ENTRY	212
6.19	common/access/sim/ReadWrite_sim.c File Reference	212
6.19.1	Macro Definition Documentation	213
6.19.1.1	DRV_MEMMAP_SIZE	213
6.19.1.2	NONE_APLL_REG_SPACE_FLAG	213
6.19.2	Function Documentation	214
6.19.2.1	Delnit_Sim	214
6.19.2.2	Init_Sim	214
6.19.2.3	RDByte_Sim	214
6.19.2.4	WRByte_Sim	214
6.19.3	Variable Documentation	214
6.19.3.1	dpllOverlayRegisters	214
6.19.3.2	page1Registers	214
6.19.3.3	simDebugFlag	215
6.19.3.4	simulatedMemMap	215
6.20	common/access/sim/ReadWrite_sim.h File Reference	215
6.20.1	Function Documentation	215
6.20.1.1	Delnit_Sim	215
6.20.1.2	Init_Sim	215
6.20.1.3	RDByte_Sim	215
6.20.1.4	WRByte_Sim	216
6.21	common/hlapi/include/inputFreq.h File Reference	216
6.21.1	Function Documentation	216
6.21.1.1	IDTHIApi_ConfigureInput1PPS	216
6.21.1.2	IDTHIApi_ConfigureInputAmi	217
6.21.1.3	IDTHIApi_ConfigureInputFrequency	217
6.21.1.4	IDTHIApi_DisableAllInputPriorities	217
6.22	common/hlapi/include/outputFreq.h File Reference	217
6.22.1	Enumeration Type Documentation	219
6.22.1.1	outputFrequency_t	219
6.22.1.2	T_sonetGigeMode	220
6.22.2	Function Documentation	220
6.22.2.1	IDTHIApi_ClearProvisioning	220
6.22.2.2	IDTHIApi_ConfigureOutput	221
6.22.2.3	IDTHIApi_DisableAllOutputs	222
6.22.2.4	IDTHIApi_PrintInput	222
6.22.2.5	IDTHIApi_PrintOutput	222
6.23	common/hlapi/include/outputFreq_priv.h File Reference	222
6.23.1	Macro Definition Documentation	224

6.23.1.1	CHECK_BIT	224
6.23.1.2	SET_BIT	224
6.23.2	Typedef Documentation	224
6.23.2.1	validFreq_t	224
6.23.3	Enumeration Type Documentation	225
6.23.3.1	pathSelectors_t	225
6.23.4	Function Documentation	225
6.23.4.1	IDTHIApi_ApplyFrequencies	225
6.23.4.2	IDTHIApi_ConvertFromFreqListToFreqArray	225
6.23.4.3	IDTHIApi_IsFreqArrayInFreqList	225
6.23.4.4	IDTHIApi_IsFrequencyInFreqList	225
6.23.4.5	IDTHIApi_PrintAvailFrequencies	226
6.23.4.6	IDTHIApi_PrintConfiguration	226
6.23.4.7	IDTHIApi_PrintFrequencyList	226
6.23.4.8	IDTHIApi_PrintOption	226
6.23.4.9	IDTHIApi_PrintOutputHwConfig	227
6.23.4.10	IDTHIApi_PrintSelectedOutputPath	227
6.23.5	Variable Documentation	227
6.23.5.1	IDTHIApi_Frequency2String_c	227
6.24	common/hlapi/include/profiles.h File Reference	227
6.24.1	Function Documentation	228
6.24.1.1	IDTHIApi_g813Option1	228
6.24.1.2	IDTHIApi_g813Option2	228
6.24.1.3	IDTHIApi_g8262EecOption1	228
6.24.1.4	IDTHIApi_g8262EecOption2	229
6.24.1.5	IDTHIApi_gr1244Stratum3	229
6.24.1.6	IDTHIApi_gr253SonetStratum3	229
6.24.1.7	IDTHIApi_pps	229
6.24.1.8	IDTHIApi_smc	229
6.24.1.9	IDTHIApi_wideband	230
6.25	common/hlapi/src/inputFreq.c File Reference	230
6.25.1	Function Documentation	231
6.25.1.1	IDTHIApi_ConfigureInput1PPS	231
6.25.1.2	IDTHIApi_ConfigureInputAmi	231
6.25.1.3	IDTHIApi_ConfigureInputFrequency	231
6.25.1.4	IDTHIApi_DisableAllInputPriorities	231
6.26	common/hlapi/src/outputFreq.c File Reference	232
6.26.1	Macro Definition Documentation	233
6.26.1.1	MAX_SUPPORTED_FREQUENCIES	233
6.26.1.2	OUTMUX_ENTRY	233

6.26.2	Function Documentation	233
6.26.2.1	IDTHIApi_ClearProvisioning	233
6.26.2.2	IDTHIApi_ConfigureOutput	233
6.26.2.3	IDTHIApi_DisableAllOutputs	234
6.26.2.4	IDTHIApi_RetrieveOutputPathMuxFromHardware	234
6.27	common/hlapi/src/outputFreqString.c File Reference	235
6.27.1	Enumeration Type Documentation	236
6.27.1.1	T_outFreqLookupIdx	236
6.27.2	Function Documentation	237
6.27.2.1	IDTHIApi_PrintAvailFrequencies	237
6.27.2.2	IDTHIApi_PrintConfiguration	237
6.27.2.3	IDTHIApi_PrintFrequencyList	237
6.27.2.4	IDTHIApi_PrintInput	237
6.27.2.5	IDTHIApi_PrintInputHwConfig	237
6.27.2.6	IDTHIApi_PrintOption	237
6.27.2.7	IDTHIApi_PrintOutput	238
6.27.2.8	IDTHIApi_PrintOutputHwConfig	238
6.27.2.9	IDTHIApi_PrintSelectedOutputPath	238
6.27.3	Variable Documentation	238
6.27.3.1	IDTHIApi_Frequency2String_c	238
6.28	common/hlapi/src/outputFreqUtil.c File Reference	238
6.28.1	Function Documentation	239
6.28.1.1	IDTHIApi_ApplyFrequencies	239
6.28.1.2	IDTHIApi_ConvertFromFreqListToFreqArray	240
6.28.1.3	IDTHIApi_IsFreqArrayInFreqList	240
6.28.1.4	IDTHIApi_IsFrequencyInFreqList	240
6.29	common/hlapi/src/profiles.c File Reference	240
6.29.1	Detailed Description	241
6.29.2	Function Documentation	242
6.29.2.1	IDTHIApi_g813Option1	242
6.29.2.2	IDTHIApi_g813Option2	242
6.29.2.3	IDTHIApi_g8262EecOption1	242
6.29.2.4	IDTHIApi_g8262EecOption2	242
6.29.2.5	IDTHIApi_gr1244Stratum3	242
6.29.2.6	IDTHIApi_gr253SonetStratum3	243
6.29.2.7	IDTHIApi_pps	243
6.29.2.8	IDTHIApi_smc	243
6.29.2.9	IDTHIApi_wideband	243
6.30	common/include/Apl.h File Reference	243
6.31	common/include/Define.h File Reference	245

6.31.1	Macro Definition Documentation	247
6.31.1.1	APLL_CONFIGURED	247
6.31.1.2	IDT_DRV_MAX_APLL_XTAL	247
6.31.1.3	IDT_DRV_MAX_INPUT	248
6.31.1.4	IDT_DRV_MAX_OUTPUT	248
6.31.1.5	IDT_DRV_MAX_XTAL_PER_APLL	248
6.31.1.6	MPU_I2C_MODE	248
6.31.1.7	MPU_SERIAL_MODE	248
6.31.1.8	NULL_APLL_CONFIG	248
6.31.2	Typedef Documentation	248
6.31.2.1	T_idtAccessDeInitFunc	248
6.31.2.2	T_idtAccessInitFunc	249
6.31.2.3	T_idtI2CAddrTransFunc	249
6.31.2.4	T_idtInputFreqValid	249
6.31.2.5	T_idtOutputFreqValid	249
6.31.2.6	T_idtReadFunc	250
6.31.2.7	T_idtWriteFunc	250
6.31.3	Enumeration Type Documentation	250
6.31.3.1	T_idtDeviceType	250
6.31.3.2	T_idtDpllInstance	250
6.31.3.3	T_idtDpllType	251
6.32	common/include/Dpll.h File Reference	251
6.33	common/include/DpllCfg.h File Reference	252
6.34	common/include/General.h File Reference	258
6.35	common/include/Hal.h File Reference	261
6.36	common/include/Input.h File Reference	262
6.37	common/include/Output.h File Reference	265
6.38	common/include/RegisterDef.h File Reference	267
6.38.1	Detailed Description	275
6.39	common/src/Apll.c File Reference	275
6.40	common/src/DpllCfg.c File Reference	276
6.40.1	Macro Definition Documentation	281
6.40.1.1	INVALID_REGISTER_OFFSET	281
6.40.2	Variable Documentation	281
6.40.2.1	IDTDpll_bandwidthLimitRegisters_a	281
6.41	common/src/General.c File Reference	281
6.42	common/src/Hal.c File Reference	283
6.42.1	Macro Definition Documentation	284
6.42.1.1	DEBUG_IDT_HAL	284
6.43	common/src/Input.c File Reference	284

6.44 common/src/Output.c File Reference	287
6.44.1 Enumeration Type Documentation	288
6.44.1.1 anonymous enum	288
6.44.1.2 T_idtOutputPathBfVal	289
Index	289

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:

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

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

82V3911/dev/include/idt82V3911.h	175
82V3911/dev/src/82V3911.c	176
82V3911/sample/include/access.h	178
82V3911/sample/include/loadstore.h	181
82V3911/sample/include/sample.h	182
82V3911/sample/src/access.c	185
82V3911/sample/src/loadstore.c	188
82V3911/sample/src/main.c	189
82V3911/sample/src/sample.c	194
apll/access/sim/ApllReadWrite_sim.c	195
apll/access/sim/ApllReadWrite_sim.h	197
apll/include/ApllApi.h	198
apll/include/ApllFreqProfile.h	200
apll/include/ApllLogging.h	202
apll/include/ApllRegDef.h	202
apll/include/ApllTypeDef.h	203
apll/src/ApllApi.c	205
apll/src/ApllFreqProfile.c	209
common/access/sim/ReadWrite_sim.c	212
common/access/sim/ReadWrite_sim.h	215
common/hlapi/include/inputFreq.h	216
common/hlapi/include/outputFreq.h	217
common/hlapi/include/outputFreq_priv.h	222
common/hlapi/include/profiles.h	227
common/hlapi/src/inputFreq.c	230
common/hlapi/src/outputFreq.c	232
common/hlapi/src/outputFreqString.c	235
common/hlapi/src/outputFreqUtil.c	238
common/hlapi/src/profiles.c	240
common/include/Apll.h	243
common/include/Define.h	245
common/include/Dpll.h	251
common/include/DpllCnfg.h	252
common/include/General.h	258
common/include/Hal.h	261
common/include/Input.h	262
common/include/Output.h	265

common/include/ RegisterDef.h	
This header file contains register and bitfield information for accessing device	267
common/src/ AplI.c	275
common/src/ DpllCnfg.c	276
common/src/ General.c	281
common/src/ Hal.c	283
common/src/ Input.c	284
common/src/ Output.c	287

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 • 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
<i>nDamping_p</i>	the damping factor of DPLL • DampingFactor_1p2 - 1.2 • DampingFactor_2p5 - 2.5 • DampingFactor_5 - 5 • DampingFactor_10 - 10 • DampingFactor_20 - 20

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>	• 0= Disable the hitless switching feature; • 1= Enable the hitless switching feature;
<i>nFrozen_p</i>	• 0= hitless switching feature is not frozen; • 1= HS feature is frozen, ignore the further hitless switching events

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>	- Holdover_Instantaneous = automatic instantaneous mode - Holdover_Slow_Average = automatic slow average - Holdover_Fast_Average = Automatic fast average - Holdover_Manual = Manual
<i>nReadMode_p</i>	0= Don't read value from device 1= Read value calculated by device back to the holdover frequency register(0x5D,0x5E and 0x5F)

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>	• 0= Disable the coarse phase detector • 1= Enable the coarse phase detector
<i>nWide_p</i>	• 0= Wide range disable • 1= Wide range enable
<i>nMultiPhase_p</i>	• 0= Limit to +/- 1UI • 1= Limit to PH_LOS_COARSE_LIMT register
<i>nMultiPh8kEn_p</i>	• 0= Limit to +/- 1UI when reference clock 8/4/2KHz clock • 1= Limit to PH_LOS_COARSE_LIMT register

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>	• 0= Only locked bandwidth and damping factor selected • 1= enable automatic selection

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 • dppllBandwidth_0p5mHz - 0.5 mHz • dppllBandwidth_1mHz - 1 mHz • dppllBandwidth_2mHz - 2 mHz • dppllBandwidth_4mHz - 4 mHz • dppllBandwidth_8mHz - 8 mHz • dppllBandwidth_15mHz - 15 mHz • dppllBandwidth_30mHz - 30 mHz • dppllBandwidth_60mHz - 60 mHz • dppllBandwidth_0p1Hz - 0.1 Hz • dppllBandwidth_0p3Hz - 0.3 Hz • dppllBandwidth_0p6Hz - 0.6 Hz • dppllBandwidth_1p2Hz - 1.2 Hz • dppllBandwidth_2p5Hz - 2.5 Hz • dppllBandwidth_4Hz - 4 Hz • dppllBandwidth_8Hz - 8 Hz • dppllBandwidth_18Hz - 18 Hz • dppllBandwidth_35Hz - 35 Hz • dppllBandwidth_70Hz - 70 Hz • dppllBandwidth_560Hz - 560 Hz
<i>nDamping</i>	the damping factor of DPLL • DampingFactor_1p2 - 1.2 • DampingFactor_2p5 - 2.5 • DampingFactor_5 - 5 • DampingFactor_10 - 10 • DampingFactor_20 - 20

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>	• 0= Enable • 1= Disable

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>	• 0= Enable • 1= Disable

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>	• 0= Disable the integral path frozen • 1= Enable the integral path frozen

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

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>	• 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 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>	• 0= Enable • 1= Disable

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>	• 0= Enable • 1= Disable

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

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>	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 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>	• 0= Disable the hitless switching feature; • 1= Enable the hitless switching feature;
<i>nFrozen</i>	• 0= hitless switching feature is not frozen; • 1= HS feature is frozen, ignore the further hitless switching events

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>	• Holdover_Instantaneous = automatic instantaneous mode • Holdover_Slow_Average = automatic slow average • Holdover_Fast_Average = Automatic fast average • Holdover_Manual = Manual
<i>nReadMode</i>	• 0= Don't read value from device • 1= Read value calculated by device back to the holdover frequency register(0x5D,0x5E and 0x5F)

Definition at line 1136 of file DpllCnfg.c.

4.2.4.64 void IDTDpll.SetIcpCtrl (T_idtDrvHdlr *hdlr*, T_idtDpllInstance *nDpll*, T_osUInt8 *nlcp*)

Set charge pump control.

Parameters

<i>hdlr</i>	Device driver handler
<i>nDpll</i>	DPLL instance
<i>nlcp</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>	• 0= Disable the coarse phase detector • 1= Enable the coarse phase detector
<i>nWide</i>	• 0= Wide range disable • 1= Wide range enable
<i>nMultiPhase</i>	• 0= Limit to +/- 1UI • 1= Limit to PH_LOS_COARSE_LIMT register

<i>nMultiPh8kEn</i>	• 0= Limit to +/- 1UI when reference clock 8/4/2KHz clock • 1= Limit to PH_LOS_COARSE_LIMT register
---------------------	--

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. • 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 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>	• 0= Disable the fine phase detector • 1= Enable the fine phase detector

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 • 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 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= nOffset * 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>	• 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 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>	• 0= Disable the phase transient monitor • 1= Enable the phase transient monitor

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>	- 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 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](#) }
- enum [phaseTimeoutMultiplier_t](#) {
[phaseTimeoutMultiplier_2](#) = 0, [phaseTimeoutMultiplier_4](#) = 1, [phaseTimeoutMultiplier_8](#) = 2, [phaseTimeoutMultiplier_16](#) = 3,
[phaseTimeoutMultiplier_MAX](#) }
- Enumerated type listing types of selectable network type.
- 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>	• phaseTimeoutMultiplier_2 - Multiple 2 • phaseTimeoutMultiplier_4 - Multiple 4 • phaseTimeoutMultiplier_8 - Multiple 8 • phaseTimeoutMultiplier_16 - Multiple 16
<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>	• 0= Not enable to use TDO pin • 1= enable to use the pin;

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

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>	• Network_SDH = SDH network • Network_SONET = SONET network

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>	• 0= Rising edge is activate edge • 1= Falling edge is activate edge

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

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>	• phaseTimeoutMultiplier_2 - Multiple 2 • phaseTimeoutMultiplier_4 - Multiple 4 • phaseTimeoutMultiplier_8 - Multiple 8 • phaseTimeoutMultiplier_16 - Multiple 16
<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>	• 0= The feature is disabled • 1= The feature is enabled

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_idtAmiInputFrequencyBitFieldValue IDTInput.GetAmiInputFrequency (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 • 0x0 = Bucket decrease by 1 in every 128 ms • 0x1 = Bucket decrease by 1 in every 256 ms • 0x2 = Bucket decrease by 1 in every 512 ms • 0x3 = Bucket decrease by 1 in every 1024 ms

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. • 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 329 of file Input.c.

4.5.4.23 `void` `IDTInput_SetAmiInputFrequency` (`T_idtDrvHdlr` *hdlr*, `T_osUInt8` *nInputNo*, `T_idtAmiInputFrequencyBitField` *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 • <code>AmiFreq_64kHzPlus8kHz</code>= 64 kHz + 8 kHz signal • <code>AmiFreq_64kHzPlus400Hz</code>= 64 kHz + 400 Hz signal

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 • 0x0 = Bucket decrease by 1 in every 128 ms • 0x1 = Bucket decrease by 1 in every 256 ms • 0x2 = Bucket decrease by 1 in every 512 ms • 0x3 = Bucket decrease by 1 in every 1024 ms

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

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(hdlr, 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(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_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_idtApplConfigPerOutput * *applConfig*)

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>applConfig</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>	• 0=Disable port • 1=Enable port

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_PEC-
L_LVDS_SIZE = 1,
BF_OUT7_PEC-
L_LVDS_MASK = 0x02, BF_OUT7_PEC-
L_LVDS_LSB = 1, BF_OUT6_PEC-
L_LVDS_SIZE = 1, BF_OUT6_PEC-
L_LVDS_MASK = 0x01,
BF_OUT6_PEC-
L_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_HOLD OVER_SIZE = 1, BF_MAN_HOLD OVER_MASK = 0x80, BF_MAN_HOLD OVER_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_HOLD OVER_MODE_SIZE = 2, BF_TEMP_HOLD OVER_MODE_MASK = 0x0C, BF_TEMP_HOLD 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_MASK = 0xFF,

```

```

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 offisers 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 offisers 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_HOLDOVER_MODE_CNFG T0 DPLL Holdover Mode Configuration Register
REG_HOLDOVER_FREQ_LOW_CNFG DPLL Holdover Frequency Register 1
REG_HOLDOVER_FREQ_MID_CNFG DPLL Holdover Frequency Register 2
REG_HOLDOVER_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 `IDTAplI_GetPreDivRange` (T_idtDrvHdlr hdlr, T_idtAplIDividerValue *minDiv, T_idtAplIDividerValue *maxDiv)

Function to retrieve pre- (input) divisor range.

- void `IDTAplI_SetPreDiv` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId, T_idtAplIConfigSel apIIConfig, T_idtAplI-DividerValue div)

Function to provision pre- (input) divider value.

- T_idtAplIDividerValue `IDTAplI_GetFeedbackDiv` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId, T_idtAplIConfigSel apIIConfig)

Function to retrieve feedback (oscillator) divider value.

- void `IDTAplI_GetFeedbackDivRange` (T_idtDrvHdlr hdlr, T_idtAplIDividerValue *minDiv, T_idtAplIDivider-Value *maxDiv)

Function to retrieve feedback (oscillator) divisor range.

- void `IDTAplI_SetFeedbackDiv` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId, T_idtAplIConfigSel apIIConfig, T_idt-AplI-DividerValue div)

Function to provision feedback (oscillator) divider value.

- T_idtAplIQaDiv `IDTAplI_GetQaDiv` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId, T_idtAplIConfigSel apIIConfig)

Function to retrieve output port A divisor.

- void `IDTAplI_SetQaDiv` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId, T_idtAplIConfigSel apIIConfig, T_idtAplIQa-Div div)

Function to provision output port A divisor.

- T_idtAplIQaDiv `IDTAplI_GetQbDiv` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId, T_idtAplIConfigSel apIIConfig)

Function to retrieve output port B divisor.

- void `IDTAplI_SetQbDiv` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId, T_idtAplIConfigSel apIIConfig, T_idtAplIQb-Div div)

Function to provision output port B divisor.

- T_idtAplIFreqDoublerSel `IDTAplI_GetFreqDoublerSel` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId)

Function to retrieve whether feedback frequency doubling is enabled.

- void `IDTAplI_SetFreqDoublerSel` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId, T_idtAplIFreqDoublerSel doubler)

Function to enable/disable feedback frequency doubling.

- T_idtAplIXtalId `IDTAplI_GetXtalId` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId)

Function to retrieve oscillator currently used.

- void `IDTAplI_SetXtalId` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId, T_idtAplIXtalId xtal)

Function to select oscillator.

- T_idtAplIOutputSigType `IDTAplI_GetOutputSigType` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId, T_idtAplIOutput output)

Function to retrieve output port signal type.

- void `IDTAplI_SetOutputSigType` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId, T_idtAplIOutput output, T_idtAplI-OutputSigType outputSigType)

Function to provision output port signal type.

- T_idtAplIOutputEn `IDTAplI_GetOutputEnState` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId, T_idtAplIOutput out-put)

Function to retrieve output port status (enable/disable)

- void `IDTAplI_SetOutputEnState` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId, T_idtAplIOutput output, T_idtAplI-OutputEn enable)

Function to control (enable/disable) output port.

- T_idtAplIMasterResetSync `IDTAplI_GetMasterResetSync` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId)

Function to retrieve reset status of device.

- void `IDTAplI_SetMasterResetSync` (T_idtDrvHdlr hdlr, T_idtAplIID apIIInstId, T_idtAplIMasterResetSync re-set)

Function to reset APLL.

- T_idtAplIBypass `IDTAplI_GetBypass` (T_idtDrvHdlr hdlr, T_idtAplIID 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_idtApIIFreqMapping* IDTAplI_GetApIIFreqTableEntry ( T_idtDrvHdlr hdlr, T_idtOutputApIIFreqMapping outputApIIFreq, outputFrequency_t nOutFreq )`

Get APLL frequency configuration table entry based on the output frequency.

Parameters

<i>hdlr</i>	Device driver handler
<i>outputApIIFreq</i>	Output APLL configuration
<i>nOutFreq</i>	Output frequency

Returns

APLL frequency configuration

Definition at line 627 of file ApIIFreq.c.

4.8.5.6 `T_idtApIIBypass IDTAplI_GetBypass ( T_idtDrvHdlr hdlr, T_idtApIIFreqMapping apIIFreq )`

Function to retrieve bypass status of device.

Parameters

<i>hdlr</i>	Device driver handler
<i>apIIFreq</i>	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2

Returns

Bypass status

- APLL_BYPASS_NORMAL_OP - Normal operation
- APLL_BYPASS_BYPASS - Bypass VCXO, connect input to output

Definition at line 1359 of file ApIIFreq.c.

4.8.5.7 `const T_idtApIIFreqMapping* IDTAplI_GetConfigListByFreq ( T_idtApIIFreqMappingType apIIFreq )`

Function to retrieve output configuration given an output frequency.

Parameters

<i>apIIFreq</i>	Output frequency
-----------------	------------------

Returns

Output configuration

Definition at line 545 of file ApIIFreqProfile.c.

4.8.5.8 `const T_idtApIIFreqMapping* IDTAplI_GetConfigListByXtalType ( T_idtApIIFreqMappingType apIIFreq )`

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>apllConfig</i>	APLL configuration • APLL_CONFIG0 - configuration 0 • APLL_CONFIG1 - configuration 1

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>output</i>	Output port • APLL_OUTPUT_QA - Output port A • APLL_OUTPUT_QB - Output port B

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>output</i>	Output port • APLL_OUTPUT_QA - Output port A • APLL_OUTPUT_QB - Output port B

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>apIIConfig</i>	APLL configuration • APLL_CONFIG0 - configuration 0 • APLL_CONFIG1 - configuration 1

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2

<i>config</i>	APLL configuration • APLL_CONFIG0 - configuration 0 • APLL_CONFIG1 - configuration 1
---------------	---

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>config</i>	APLL configuration • APLL_CONFIG0 - configuration 0 • APLL_CONFIG1 - configuration 1

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2

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>apllInstId</i>	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2

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_idtApIIXtalId** IDTAplI_GetXtalIdFromXtalFreq (**T_idtDrvHdlr** *hdlr*, **T_idtApIIId** *apIIInst*, **T_idtApIIXtalType** *apIIVcxoFreq*)

Get APLL configured crystal ID from crystal frequency.

Parameters

<i>hdlr</i>	Device driver handler
<i>apIIInst</i>	APLL instance
<i>apIIVcxoFreq</i>	Crytal frequency

Returns

Crytal ID

Definition at line 753 of file ApII.c.

4.8.5.27 **T_osBool** IDTAplI_NullFreqTableEntry (**const T_idtApIIFreqMapping** * *apIIFreqConfig*)

Function to check whether the input APLL frequency mapping is NULL (not configured)

Parameters

<i>apIIFreqConfig</i>	APLL frequency mapping configuration
-----------------------	--------------------------------------

Returns

E_osTrue - not configured, E_osFalse - configured

Definition at line 583 of file ApIIFreqProfile.c.

4.8.5.28 **void** IDTAplI_SetApIIConfig (**T_idtDrvHdlr** *hdlr*, **T_idtApIIId** *apIIInstId*, **T_idtApIIConfig** * *apIIConfig*)

Set APLL full configuration.

Parameters

<i>hdlr</i>	Device driver handler
<i>apIIInstId</i>	APLL instance
<i>apIIConfig</i>	APLL configuration

Definition at line 357 of file ApII.c.

4.8.5.29 **void** IDTAplI_SetApIIConfigPerOutput (**T_idtDrvHdlr** *hdlr*, **T_idtApIIId** *apIIInstId*, **T_idtApIIOutput** *output*, **T_idtApIIConfigPerOutput** * *apIIConfigPerOutput*)

Set 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 272 of file ApI.c.

4.8.5.30 void IDTAplI.SetBypass (T_idtDrvHdlr *hdlr*, T_idtApIId *apIInstId*, T_idtApIBypass *bypass*)

Function to set APLL bypass setting.

Parameters

<i>hdlr</i>	Device driver handler
<i>apIInstId</i>	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>bypass</i>	Value to set bypass • APLL_BYPASS_NORMAL_OP - Normal operation • APLL_BYPASS_BYPASS - Bypass VCXO, connect input to output

Definition at line 1322 of file ApIApi.c.

4.8.5.31 void IDTAplI.SetConfigSel (T_idtDrvHdlr *hdlr*, T_idtApIId *apIInstId*, T_idtApIConfigSel *apIConfig*)

Function to provision APLL based on input configuration.

Parameters

<i>hdlr</i>	Device driver handler
<i>apIInstId</i>	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>apIConfig</i>	APLL configuration • APLL_CONFIG0 - configuration 0 • APLL_CONFIG1 - configuration 1

Definition at line 250 of file ApIApi.c.

4.8.5.32 void IDTAplI.SetFeedbackDiv (T_idtDrvHdlr *hdlr*, T_idtApIId *apIInstId*, T_idtApIConfigSel *apIConfig*, T_idtApIDividerValue *div*)

Function to provision feedback (oscillator) divider value.

Parameters

<i>hdlr</i>	Device driver handler
<i>apllInstId</i>	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>apllConfig</i>	APLL configuration • APLL_CONFIG0 - configuration 0 • APLL_CONFIG1 - configuration 1
<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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>doubler</i>	Feedback frequency doubling value • APLL_FREQ_SEL_SINGLE - Feedback frequency not doubled • APLL_FREQ_SEL_DOUBLE - Feedback frequency doubled

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>inputClk</i>	Input clock source for APLL • APLL_CLK_SEL_EXTERNAL - external clock source • APLL_CLK_SEL_INTERNAL - internal clock source

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>reset</i>	Value to set master reset sync • APLL_OUT_MR_SYNC - Normal operation • APLL_IN_MR_SYNC - In master sync reset

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>output</i>	Output port • APLL_OUTPUT_QA - Output port A • APLL_OUTPUT_QB - Output port B
<i>enable</i>	Enable /Disable output • APLL_OUTPUT_DISABLE - Disable output • APLL_OUTPUT_ENABLE - Enable output

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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2

<i>output</i>	Output port • APLL_OUTPUT_QA - Output port A • APLL_OUTPUT_QB - Output port B
<i>outputSigType</i>	output signal type • APLL_OUT_SIG_LVPECL - Output LVPECL signal • APLL_OUT_SIG_LVDS - Output LVDS signal

Definition at line 1088 of file AplApi.c.

4.8.5.38 void IDTAplI.SetPreDiv (T_idtDrvHdlr *hdlr*, T_idtAplIId *apllInstId*, T_idtAplIConfigSel *apllConfig*, T_idtAplIDividerValue *div*)

Function to provision pre- (input) divider value.

Parameters

<i>hdlr</i>	Device driver handler
<i>apllInstId</i>	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>apllConfig</i>	APLL configuration • APLL_CONFIG0 - configuration 0 • APLL_CONFIG1 - configuration 1
<i>div</i>	Divisor value

Definition at line 438 of file AplApi.c.

4.8.5.39 void IDTAplI.SetQaDiv (T_idtDrvHdlr *hdlr*, T_idtAplIId *apllInstId*, T_idtAplIConfigSel *config*, T_idtAplIQaDiv *div*)

Function to provision output port A divisor.

Parameters

<i>hdlr</i>	Device driver handler
<i>apIiInstId</i>	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>config</i>	APLL configuration • APLL_CONFIG0 - configuration 0 • APLL_CONFIG1 - configuration 1
<i>div</i>	Divisor value • 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 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 • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>config</i>	APLL configuration • APLL_CONFIG0 - configuration 0 • APLL_CONFIG1 - configuration 1
<i>div</i>	Divisor value • 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 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>hdlr</i>	Device driver handler
<i>apllInstId</i>	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>shutoff</i>	Value to set shutoff • APLL_SHUTOFF_NORMAL_OP - Normal operation • APLL_SHUTOFF_SHUTOFF - VCXO shutoff

Definition at line 1390 of file AplApi.c.

4.8.5.42 void IDTApl_SetXtalId (T_idtDrvHdlr *hdlr*, T_idtAplId *apllInstId*, T_idtAplXtalId *xtal*)

Function to select oscillator.

Parameters

<i>hdlr</i>	Device driver handler
<i>apllInstId</i>	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
<i>xtal</i>	Oscillator to select • 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 991 of file AplApi.c.

4.8.5.43 void IDTApl_Switch2NonActiveAplConfig (T_idtDrvHdlr *hdlr*, T_idtAplId *apllId*)

Switch to non-active APLL configuration.

Parameters

<i>hdlr</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 IDTAplI_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_idtApIldivConfig::fbDiv_s Struct Reference

```
#include <ApIldiv.h>
```

Data Fields

- [T_idtApIldivDividerValue fbDivConfig0](#)
- [T_idtApIldivDividerValue fbDivConfig1](#)

5.2.1 Detailed Description

Definition at line 95 of file ApIldiv.h.

5.2.2 Field Documentation

5.2.2.1 T_idtApIldivDividerValue fbDivConfig0

APLL feedback divider value for configuration 0

Definition at line 97 of file ApIldiv.h.

5.2.2.2 T_idtApIldivDividerValue fbDivConfig1

APLL feedback divider value for configuration 1

Definition at line 98 of file ApIldiv.h.

The documentation for this struct was generated from the following file:

- [common/include/ApIldiv.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 126 of file main.c.

5.3.2 Field Documentation

5.3.2.1 T_osUInt8 dcoMode_m

Enable/Disable

DCO mode

Definition at line 142 of file main.c.

5.3.2.2 T_idtDpllType dpll_m[IDT_DRV_MAX_OUTPUT]

Select DPLL T0 / T4

Definition at line 137 of file main.c.

5.3.2.3 T_DpllProfile dpllProfile_ma[DPLL_INSTANCE_MAX]

Dpll Profile

Definition at line 147 of file main.c.

5.3.2.4 char filename_ma[SAMPLE_MAX_FILENAME]

File name for

load/store

Definition at line 129 of file main.c.

5.3.2.5 T_osUInt32 inputFreq_m[IDT_DRV_MAX_INPUT]

Input frequency (kHz)

Definition at line 131 of file main.c.

5.3.2.6 T_osUInt8 isSim_m

Switch to

simulated mode

Definition at line 140 of file main.c.

5.3.2.7 T_osUInt32 modes_m

Operation modes

for prgram

Definition at line 127 of file main.c.

5.3.2.8 T_osUInt8 numberOfXtal

Number APLL external Xtals

Definition at line 145 of file main.c.

5.3.2.9 long offset_m

DCO Offset

Definition at line 144 of file main.c.

5.3.2.10 outputFrequency_t outFreq_m[IDT_DRV_MAX_OUTPUT]

Output frequency (enum)

Definition at line 134 of file main.c.

5.3.2.11 T_osUInt8 regOffset_m

register offset

Definition at line 150 of file main.c.

5.3.2.12 T_osUInt8 regValue_m

register value

Definition at line 151 of file main.c.

5.3.2.13 T_osUInt8 sampleConfig_m

Sample configuration

Definition at line 149 of file main.c.

5.3.2.14 T_idtApIIXtal* xtalList

APLL external Xtal list

Definition at line 146 of file main.c.

The documentation for this struct was generated from the following file:

- 82V3911/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 [xtal1ApI1](#)
- [T_idtApIIXtalType xtal1ApI1Freq](#)
- int [xtal3ApI1](#)
- [T_idtApIIXtalType xtal3ApI1Freq](#)
- int [xtal2ApI2](#)
- [T_idtApIIXtalType xtal2ApI2Freq](#)
- int [xtal4ApI2](#)
- [T_idtApIIXtalType xtal4ApI2Freq](#)

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:

- 82V3911/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_idtl2CaddrTransFunc 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 74 of file main.c.

5.28.2 Field Documentation

5.28.2.1 T_SampleConfigDescrFunc configDescription_m

Function to output description

Definition at line 76 of file main.c.

5.28.2.2 T_SampleConfigFunc configFunction_m

Function to configure

Definition at line 75 of file main.c.

The documentation for this struct was generated from the following file:

- 82V3911/sample/src/[main.c](#)

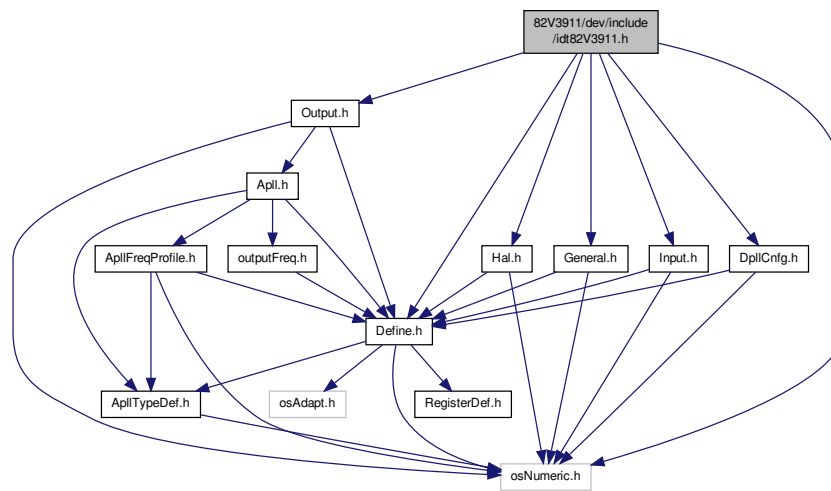
Chapter 6

File Documentation

6.1 82V3911/dev/include/idt82V3911.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 idt82V3911.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 82V3911.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

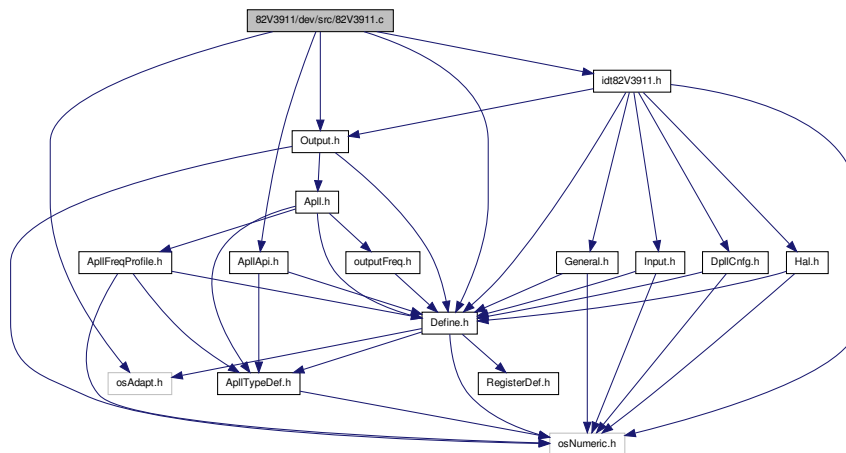
Device driver handler (used in all APIs)

Definition at line 174 of file 82V3911.c.

6.2 82V3911/dev/src/82V3911.c File Reference

```
#include "osAdapt.h"
#include "idt82V3911.h"
#include "Define.h"
#include "ApplApi.h"
#include "Output.h"
```

Include dependency graph for 82V3911.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](#) [idt82V3911Config](#)
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 82V3911.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 174 of file 82V3911.c.

6.2.2 Variable Documentation

6.2.2.1 `const T_idtDrvDeviceConfig idt82V3911Config`

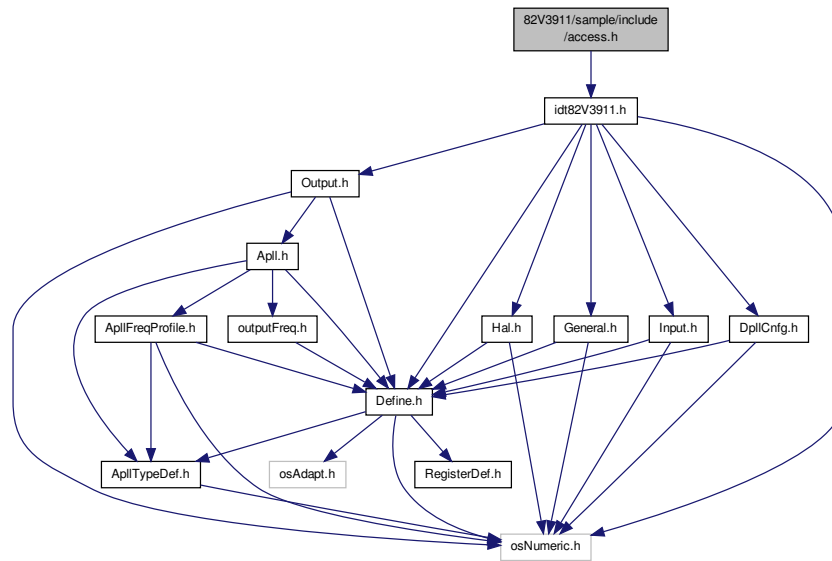
Specific device profile.

Definition at line 50 of file 82V3911.c.

6.3 82V3911/sample/include/access.h File Reference

```
#include "idt82V3911.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 82V3911/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

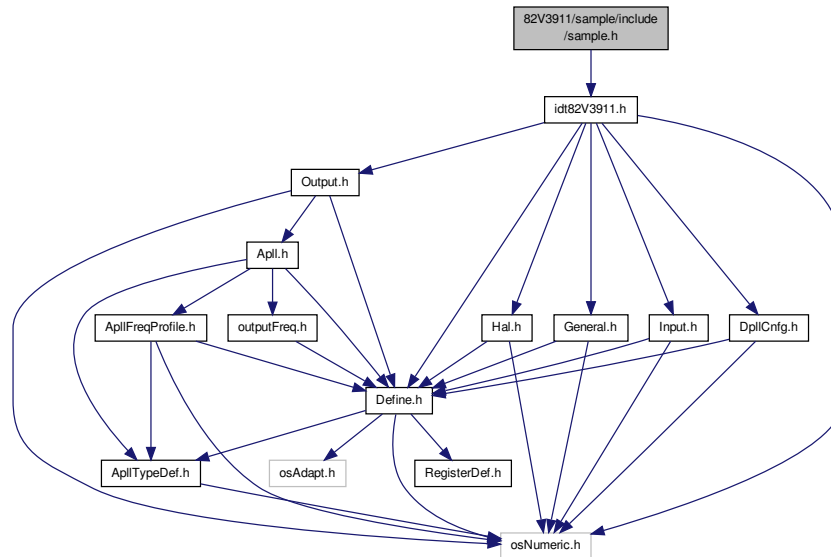
<i>hdlr</i>	Device driver handler
<i>filename</i>	Output file name, NULL for standard IO

Definition at line 94 of file loadstore.c.

6.5 82V3911/sample/include/sample.h File Reference

```
#include "idt82V3911.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_ConfigureSampleConfig6](#) (T_idtDrvHdlr hdlr)
Sample configuration 6.
- void [IDTSample_ConfigureSampleConfig7](#) (T_idtDrvHdlr hdlr)
Sample configuration 7.

6.5.1 Function Documentation

6.5.1.1 void IDTSample_ConfigureSampleConfig1 (T_idtDrvHdlr hdlr)

Sample configuration 1.

```

IN3 19.44MHz -> DPLL #1 -> OUT6 25 MHz
IN3 19.44MHz -> DPLL #1 -> OUT7 125 MHz
IN4 25MHz     -> DPLL #2 -> OUT2 19.44 MHz
APLL1 XTAL1 at 25MHz

```


Parameters

<i>hdlr</i>	Device driver handle
-------------	----------------------

Sample Configuration 1 (see header file for description)

Definition at line 54 of file sample.c.

6.5.1.2 void IDTSample_ConfigureSampleConfig2 (T_idtDrvHdlr *hdlr*)

Sample configuration 2.

```
IN3 25 MHz    -> DPLL #1 -> OUT1 19.44 MHz
IN3 25 MHz    -> DPLL #1 -> OUT3 77.76 MHz
                DPLL #1 -> OUT6 155.52 MHz
IN4 19.44 MHz -> DPLL #2 -> OUT8 25 MHz
APLL1 XTAL1 at 24.8832MHz
APLL2 XTAL2 at 25MHz
```

Parameters

<i>hdlr</i>	Device driver handle
-------------	----------------------

Sample Configuration 2 (see header file for description)

Definition at line 130 of file sample.c.

6.5.1.3 void IDTSample_ConfigureSampleConfig3 (T_idtDrvHdlr *hdlr*)

Sample configuration 3.

```
IN3 25 MHz    -> DPLL #1 -> OUT1 19.44 MHz
IN4 25 MHz    -> DPLL #1 -> OUT8 622.08 MHz
IN5 19.44 MHz -> DPLL #2 -> OUT6 25 MHz
APLL1 XTAL1 at 25MHz
APLL2 XTAL2 at 24.8832MHz
Hitless switch is enabled across input ports { IN3, IN4 }
```

Parameters

<i>hdlr</i>	Device driver handle
-------------	----------------------

Sample Configuration 3 (see header file for description)

Definition at line 221 of file sample.c.

6.5.1.4 void IDTSample_ConfigureSampleConfig4 (T_idtDrvHdlr *hdlr*)

Sample configuration 4.

```
IN1 10 MHz -> DPLL #1 -> OUT 6 25 MHz
IN2 10 MHz -> DPLL #1 -> OUT 7 156.25 MHz
IN3 25 MHz -> DPLL #2 -> OUT 1 2.048 MHz
IN4 25 MHz -> DPLL #2 -> OUT 2 25 MHz
APLL1 XTAL1 at 25MHz
Hitless switch is enabled across input ports { IN1, IN2 }
```

Parameters

<i>hdlr</i>	Device driver handle
-------------	----------------------

Sample Configuration 4 (see header file for description)

Definition at line 309 of file sample.c.

6.5.1.5 void IDTSample_ConfigureSampleConfig5 (T_idtDrvHdlr *hdlr*)

Sample configuration 5.

```
IN4 10 MHz      -> DPLL #1 -> OUT6 25 MHz
                  -> DPLL #1 -> OUT7 625 MHz
EX_SYNC2 1PPS ->          -> FRSYNC 1PPS
APLL1 XTAL1 at 25MHz
```

Parameters

<i>hdlr</i>	Device driver handle
-------------	----------------------

Sample Configuration 5 (see header file for description)

Definition at line 416 of file sample.c.

6.5.1.6 void IDTSample_ConfigureSampleConfig6 (T_idtDrvHdlr *hdlr*)

Sample configuration 6.

```
IN4 6.48 MHz    -> DPLL #1 -> OUT1 19.44 MHz
                  -> DPLL #1 -> OUT6 155.52 MHz
EX_SYNC2 8kHz   ->          -> FRSYNC 8 kHz
APLL1 XTAL1 at 24.8832 MHz
```

Parameters

<i>hdlr</i>	Device driver handle
-------------	----------------------

Sample Configuration 6 (see header file for description)

Definition at line 488 of file sample.c.

6.5.1.7 void IDTSample_ConfigureSampleConfig7 (T_idtDrvHdlr *hdlr*)

Sample configuration 7.

```
IN1 25 MHz -> DPLL #1 -> OUT1 10 MHz
IN3 10 MHz -> DPLL #1 -> OUT2 1PPS
                  DPLL #1 -> OUT3 125 MHz
                  DPLL #1 -> OUT4 8 KHz
                  DPLL #1 -> OUT5 8 KHz
                  DPLL #1 -> OUT6 156.25 MHz
APLL1 XTAL1 at 25 MHz
```

Parameters

<i>hdlr</i>	Device driver handle
-------------	----------------------

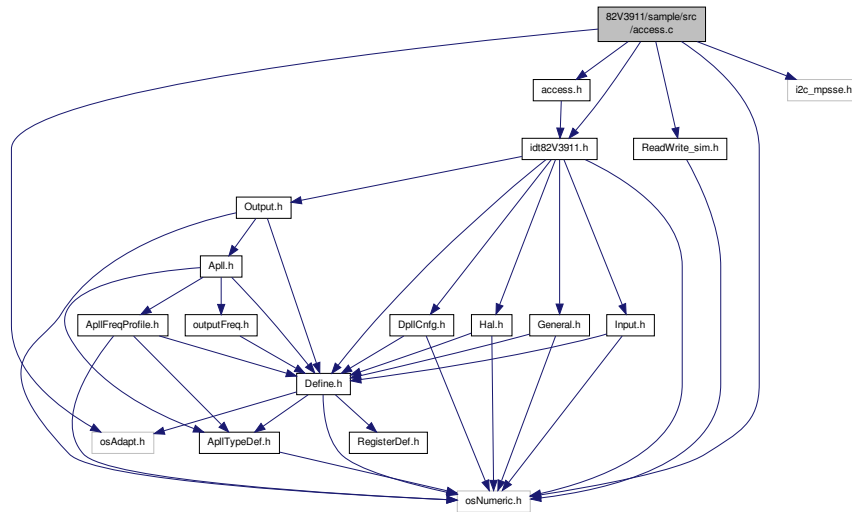
Sample Configuration 7 (see header file for description)

Definition at line 562 of file sample.c.

6.6 82V3911/sample/src/access.c File Reference

```
#include "osNumeric.h"
#include "osAdapt.h"
#include "idt82V3911.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_idtApplXtalList *xtalList, T_osUInt8 coreI2CAddr)
Utility function to initialize device driver for simulator access.
- void [IDTSample_ResetXtalList](#) (T_idtApplXtalList *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](#)

6.6.1 Function Documentation

6.6.1.1 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.

6.6.1.2 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.

6.6.1.3 T_idtDrvHdlr IDTSample_InitDriverI2c (T_idtApIIXtalList * *xtalList*, T_osUInt8 *coreI2CAddr*)

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>coreI2CAddr</i>	Device I2C address

Returns

Device driver handle

Definition at line 161 of file access.c.

6.6.1.4 T_idtDrvHdlr IDTSample_InitDriverSimulator (T_idtApIIXtalList * *xtalList*, T_osUInt8 *coreI2CAddr*)

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.

6.6.1.5 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.

6.6.1.6 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.

6.6.1.7 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.

6.6.1.8 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.

6.6.1.9 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.

6.6.2 Variable Documentation

6.6.2.1 `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.

6.6.2.2 `const T_idtDrvAccess SimAccessMethods`

Initial value:

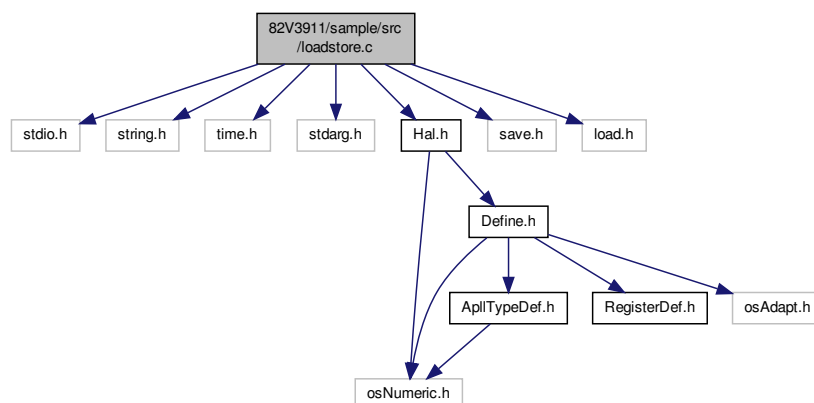
```
= {
    Init_SimWrapper,
    DeInit_SimWrapper,
    ReadByte_SimWrapper,
    WriteByte_SimWrapper,
    NULL
}
```

Definition at line 89 of file access.c.

6.7 82V3911/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.

6.7.1 Function Documentation

6.7.1.1 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.

6.7.1.2 void IDTSample_SaveRgfFile (T_idtDrvHdlr hdlr, char * filename)

Function to output register contents of a device to standard IO or to a file.

Parameters

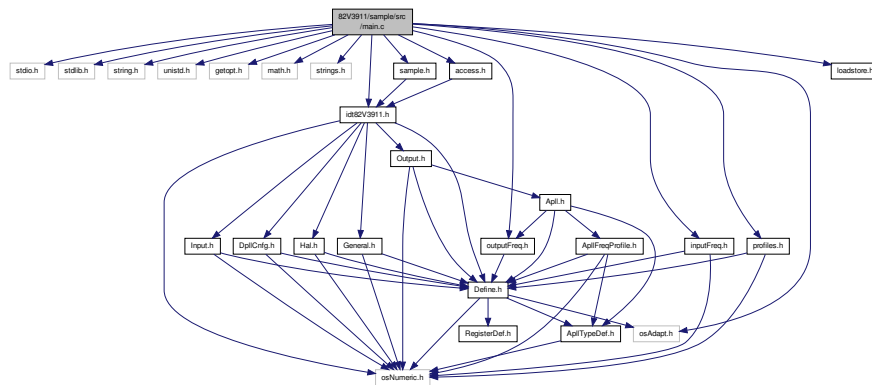
<i>hdlr</i>	Device driver handler
<i>filename</i>	Output file name, NULL for standard IO

Definition at line 94 of file loadstore.c.

6.8 82V3911/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 "idt82V3911.h"
#include "osAdapt.h"
#include "sample.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_DpllProfile](#) (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`]

6.8.1 Macro Definition Documentation

6.8.1.1 `#define MODE_DpllProfile (1<<4)`

Provision DPLL profile mode

Definition at line 110 of file main.c.

6.8.1.2 `#define MODE_Input (1<<2)`

Provision input port mode

Definition at line 104 of file main.c.

6.8.1.3 `#define MODE_Load (1<<0)`

Load from file mode

Definition at line 98 of file main.c.

6.8.1.4 `#define MODE_None (0)`

No-op mode

Definition at line 95 of file main.c.

6.8.1.5 `#define MODE_Output (1<<3)`

Provision output port mode

Definition at line 107 of file main.c.

6.8.1.6 `#define MODE_Read (1<<6)`

Read register mode

Definition at line 116 of file main.c.

6.8.1.7 `#define MODE_Sample (1<<5)`

Sample configuration mode

Definition at line 113 of file main.c.

6.8.1.8 `#define MODE_Save (1<<1)`

Save to file mode

Definition at line 101 of file main.c.

6.8.1.9 `#define MODE_Write (1<<7)`

Read register mode

Definition at line 119 of file main.c.

6.8.1.10 `#define no_argument 0`

Definition at line 154 of file main.c.

6.8.1.11 `#define optional_argument 2`

Definition at line 156 of file main.c.

6.8.1.12 `#define required_argument 1`

Definition at line 155 of file main.c.

6.8.1.13 `#define SAMPLE_MAX_FILENAME 50`

Max filename size

Definition at line 122 of file main.c.

6.8.2 Typedef Documentation

6.8.2.1 `typedef void(* T_DpIIProfileFunc)(T_idtDrvHdlr hdlr, T_idtDpIIInstance nDpII)`

Function typedef for DPLL profile provisioning.

Definition at line 58 of file main.c.

6.8.2.2 `typedef void(* T_SampleConfigDescrFunc)(void)`

Function typedef for sample configuration descriptions.

Definition at line 69 of file main.c.

6.8.2.3 typedef void(* T_SampleConfigFunc)(T_idtDrvHdlr hdlr)

Function typedef for sample configurations.

Definition at line 64 of file main.c.

6.8.3 Enumeration Type Documentation

6.8.3.1 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 82 of file main.c.

6.8.4 Function Documentation

6.8.4.1 void IDTSample_DumpRegistersRgf (char * filename)

6.8.4.2 int main (int argc, char * argv[])

Definition at line 622 of file main.c.

6.8.4.3 void printHelp (void)

Definition at line 238 of file main.c.

6.8.4.4 int processCmdLineArguments (int argc, char * argv[], globalArgs_t * gblArgs)

Definition at line 422 of file main.c.

6.8.5 Variable Documentation

6.8.5.1 globalArgs_t globalArgs

Global command line argument variable

Definition at line 166 of file main.c.

6.8.5.2 const char* IDTHIApi_Frequency2String_c[OutFreq_MAX]

Definition at line 77 of file outputFreqString.c.

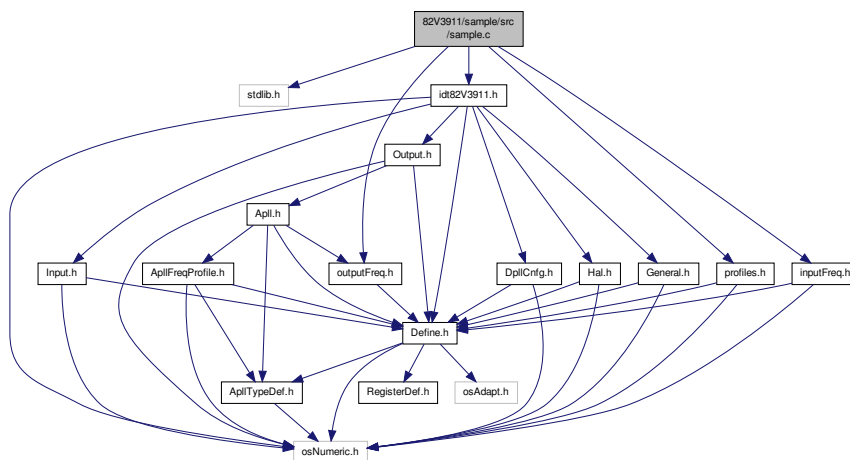
6.8.5.3 int simDebugFlag

Definition at line 48 of file ReadWrite_sim.c.

6.9 82V3911/sample/src/sample.c File Reference

```
#include <stdlib.h>
#include "idt82V3911.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_ConfigureSampleConfig6](#) (T_idtDrvHdlr hdlr)
Sample configuration 6.
- void [IDTSample_ConfigureSampleConfig7](#) (T_idtDrvHdlr hdlr)
Sample configuration 7.

6.9.1 Function Documentation

6.9.1.1 void IDTSample_ConfigureSampleConfig1 (T_idtDrvHdlr *hdlr*)

Sample configuration 1.

Sample Configuration 1 (see header file for description)

Definition at line 54 of file sample.c.

6.9.1.2 void IDTSample_ConfigureSampleConfig2 (T_idtDrvHdlr *hdlr*)

Sample configuration 2.

Sample Configuration 2 (see header file for description)

Definition at line 130 of file sample.c.

6.9.1.3 void IDTSample_ConfigureSampleConfig3 (T_idtDrvHdlr *hdlr*)

Sample configuration 3.

Sample Configuration 3 (see header file for description)

Definition at line 221 of file sample.c.

6.9.1.4 void IDTSample_ConfigureSampleConfig4 (T_idtDrvHdlr *hdlr*)

Sample configuration 4.

Sample Configuration 4 (see header file for description)

Definition at line 309 of file sample.c.

6.9.1.5 void IDTSample_ConfigureSampleConfig5 (T_idtDrvHdlr *hdlr*)

Sample configuration 5.

Sample Configuration 5 (see header file for description)

Definition at line 416 of file sample.c.

6.9.1.6 void IDTSample_ConfigureSampleConfig6 (T_idtDrvHdlr *hdlr*)

Sample configuration 6.

Sample Configuration 6 (see header file for description)

Definition at line 488 of file sample.c.

6.9.1.7 void IDTSample_ConfigureSampleConfig7 (T_idtDrvHdlr *hdlr*)

Sample configuration 7.

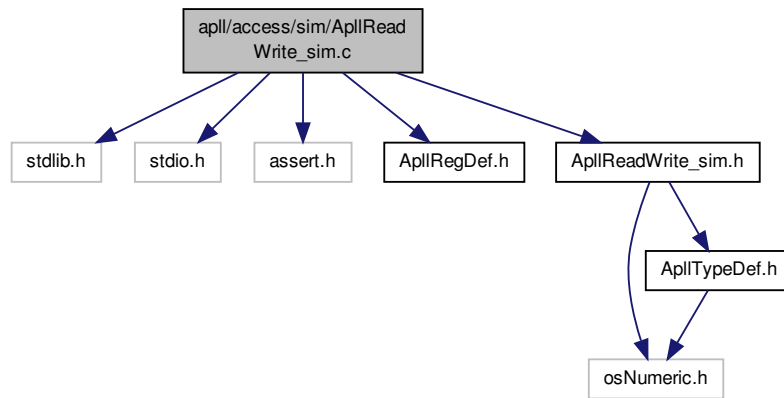
Sample Configuration 7 (see header file for description)

Definition at line 562 of file sample.c.

6.10 apII/access/sim/ApIIReadWrite_sim.c File Reference

```
#include <stdlib.h>
```

```
#include <stdio.h>
#include <assert.h>
#include "ApIRegDef.h"
#include "ApIReadWrite_sim.h"
Include dependency graph for ApIReadWrite_sim.c:
```



Macros

- `#define SIM_REG_MAP_SIZE (0x11)`

Functions

- void `ApIInit_Sim` (void *usrData)
- void `ApIDelInit_Sim` (void *usrData)
- T_osUInt8 `ApIRead_Sim` (void *usrData, T_osUInt8 deviceAddr, T_idtApIRegAddr nAddr)
- void `ApIWrite_Sim` (void *usrData, T_osUInt8 deviceAddr, T_idtApIRegAddr nAddr, T_idtApIRegVal data)

Variables

- T_idtApIRegVal `simulatedRegMapApI0` [SIM_REG_MAP_SIZE]
- T_idtApIRegVal `simulatedRegMapApI1` [SIM_REG_MAP_SIZE]

6.10.1 Macro Definition Documentation

6.10.1.1 `#define SIM_REG_MAP_SIZE (0x11)`

Definition at line 41 of file `ApIReadWrite_sim.c`.

6.10.2 Function Documentation

6.10.2.1 void `ApIDelInit_Sim` (void * *usrData*)

Definition at line 62 of file `ApIReadWrite_sim.c`.

6.10.2.2 void ApllInit_Sim (void * *usrData*)

Definition at line 57 of file ApllReadWrite_sim.c.

6.10.2.3 T_osUInt8 ApllRead_Sim (void * *usrData*, T_osUInt8 *deviceAddr*, T_idtApllRegAddr *nAddr*)

Definition at line 67 of file ApllReadWrite_sim.c.

6.10.2.4 void ApllWrite_Sim (void * *usrData*, T_osUInt8 *deviceAddr*, T_idtApllRegAddr *nAddr*, T_idtApllRegVal *data*)

Definition at line 82 of file ApllReadWrite_sim.c.

6.10.3 Variable Documentation**6.10.3.1 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.

6.10.3.2 T_idtApllRegVal simulatedRegMapApll1[SIM_REG_MAP_SIZE]

Initial value:

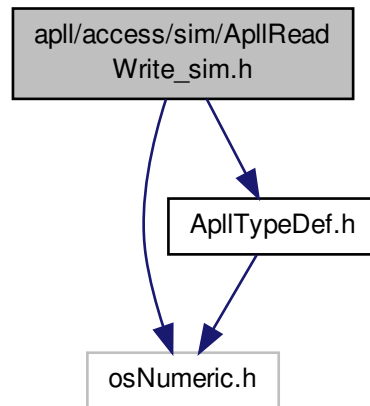
```
= {
    0x00, 0x01, 0x0c, 0x35, 0x0c,
    0x35, 0x0c, 0x35, 0x0c, 0x35,
    0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00
}
```

Definition at line 50 of file ApllReadWrite_sim.c.

6.11 apll/access/sim/ApllReadWrite_sim.h File Reference

```
#include "osNumeric.h"
#include "ApllTypeDef.h"
```

Include dependency graph for `ApllReadWrite_sim.h`:



Functions

- `T_osUInt8 ApllRead_Sim` (`void *usrData`, `T_osUInt8 deviceAddr`, `T_idtApllRegAddr nAddr`)
- `void ApllWrite_Sim` (`void *usrData`, `T_osUInt8 deviceAddr`, `T_idtApllRegAddr nAddr`, `T_idtApllRegVal data`)
- `void ApllInit_Sim` (`void *userData`)
- `void ApllDeInit_Sim` (`void *userData`)

6.11.1 Function Documentation

6.11.1.1 `void ApllDeInit_Sim ( void * userData )`

Definition at line 62 of file `ApllReadWrite_sim.c`.

6.11.1.2 `void ApllInit_Sim ( void * userData )`

Definition at line 57 of file `ApllReadWrite_sim.c`.

6.11.1.3 `T_osUInt8 ApllRead_Sim ( void * usrData, T_osUInt8 deviceAddr, T_idtApllRegAddr nAddr )`

Definition at line 67 of file `ApllReadWrite_sim.c`.

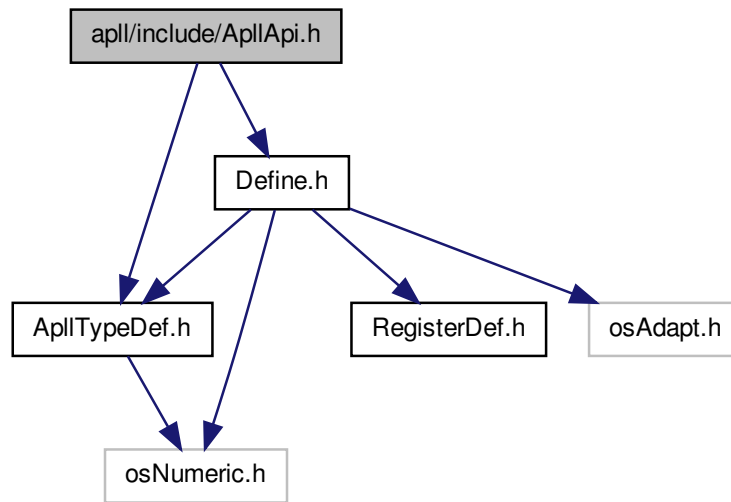
6.11.1.4 `void ApllWrite_Sim ( void * usrData, T_osUInt8 deviceAddr, T_idtApllRegAddr nAddr, T_idtApllRegVal data )`

Definition at line 82 of file `ApllReadWrite_sim.c`.

6.12 `apll/include/ApllApi.h` File Reference

```
#include "Define.h"
#include "ApllTypeDef.h"
```


Include dependency graph for ApllApi.h:



Functions

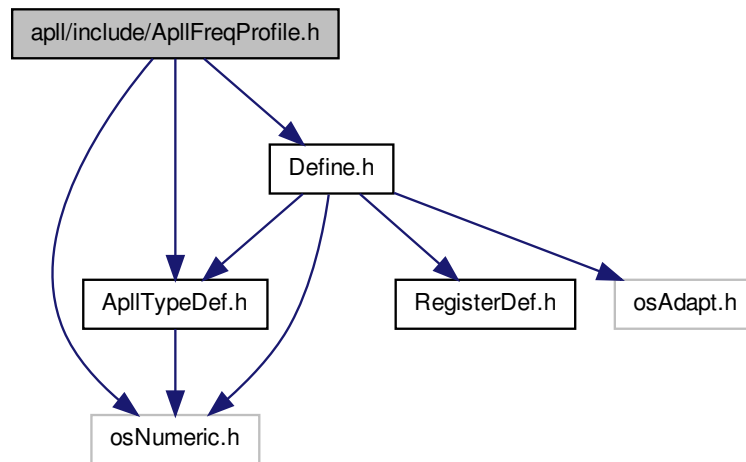
- [T_idtApllConfigSel](#) [IDTApll_GetConfigSel](#) (T_idtDrvHdlr hdlr, [T_idtApllId](#) appllInstId)
Function to retrieve APLL configuration.
- void [IDTApll_SetConfigSel](#) (T_idtDrvHdlr hdlr, [T_idtApllId](#) appllInstId, [T_idtApllConfigSel](#) appllConfig)
Function to provision APLL based on input 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.

6.13 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.

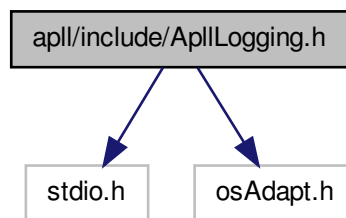
- const `T_idtApIIFreqMapping` * `IDTAplI_GetConfigListByXtalType` (`T_idtApIIXtalType` `apIIXtalFreq`)
Function to retrieve output configuration given the oscillator type (frequency)
- `T_osBool` `IDTAplI_NullFreqTableEntry` (const `T_idtApIIFreqMapping` *`apIIFreqMapping`)
Function to check whether 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.

6.14 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)

6.15 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_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 }

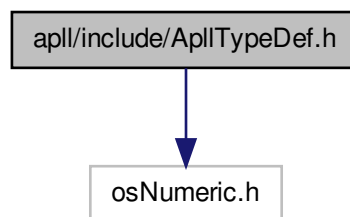
```

Enumerated type listing register offsets and bit field constants.

6.16 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` }

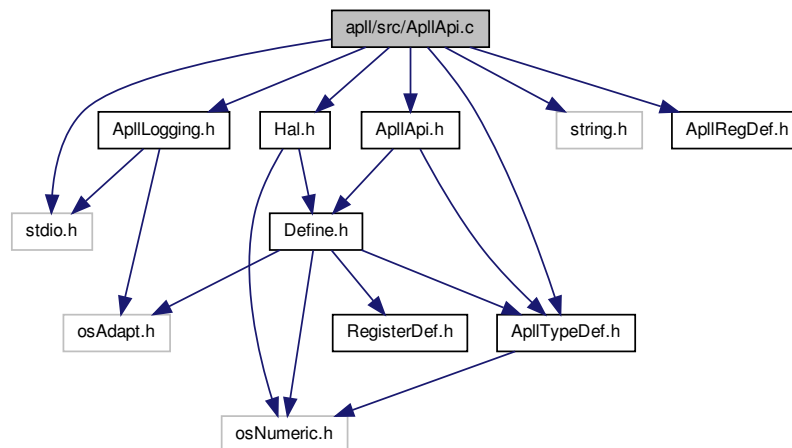
Enumerated type listing Apll bypass status/setting.

- enum `T_idtApllShutoff` { `APLL_SHUTOFF_NORMAL_OP` = 0, `APLL_SHUTOFF_SHUTOFF` = 1, `APLL_SHUTOFF_MAX` }

Enumerated type listing Apll shutoff status/setting.

6.17 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_idtApplConfigSel IDTApl_GetConfigSel (T_idtDrvHdlr hdlr, T_idtApplId apllInstId)`
Function to retrieve APLL configuration.
- `void IDTApl_SetConfigSel (T_idtDrvHdlr hdlr, T_idtApplId apllInstId, T_idtApplConfigSel apllConfig)`
Function to provision APLL based on input configuration.
- `T_idtApplInputClkSrc IDTApl_GetInputClkSrc (T_idtDrvHdlr hdlr, T_idtApplId apllInstId)`
Function to retrieve selected APLL input frequency source (external/internal)
- `void IDTApl_SetInputClkSrc (T_idtDrvHdlr hdlr, T_idtApplId apllInstId, T_idtApplInputClkSrc inputClk)`
Function to provision APLL input frequency source (external/internal)
- `T_idtApplDividerValue IDTApl_GetPreDiv (T_idtDrvHdlr hdlr, T_idtApplId apllInstId, T_idtApplConfigSel apllConfig)`
Function to retrieve pre- (input) divider value.
- `void IDTApl_GetPreDivRange (T_idtDrvHdlr hdlr, T_idtApplDividerValue *minDiv, T_idtApplDividerValue *maxDiv)`
Function to retrieve pre- (input) divisor range.
- `void IDTApl_SetPreDiv (T_idtDrvHdlr hdlr, T_idtApplId apllInstId, T_idtApplConfigSel apllConfig, T_idtApplDividerValue div)`
Function to provision pre- (input) divider value.
- `void IDTApl_GetFeedbackDivRange (T_idtDrvHdlr hdlr, T_idtApplDividerValue *minDiv, T_idtApplDividerValue *maxDiv)`
Function to retrieve feedback (oscillator) divisor range.
- `T_idtApplDividerValue IDTApl_GetFeedbackDiv (T_idtDrvHdlr hdlr, T_idtApplId apllInstId, T_idtApplConfigSel apllConfig)`
Function to retrieve feedback (oscillator) divider value.
- `void IDTApl_SetFeedbackDiv (T_idtDrvHdlr hdlr, T_idtApplId apllInstId, T_idtApplConfigSel apllConfig, T_idtApplDividerValue div)`
Function to provision feedback (oscillator) divider value.
- `T_idtApplQaDiv IDTApl_GetQaDiv (T_idtDrvHdlr hdlr, T_idtApplId apllInstId, T_idtApplConfigSel config)`
Function to retrieve output port A divisor.
- `void IDTApl_SetQaDiv (T_idtDrvHdlr hdlr, T_idtApplId apllInstId, T_idtApplConfigSel config, T_idtApplQaDiv div)`
Function to provision output port A divisor.
- `T_idtApplQaDiv IDTApl_GetQbDiv (T_idtDrvHdlr hdlr, T_idtApplId apllInstId, T_idtApplConfigSel config)`
Function to retrieve output port B divisor.
- `void IDTApl_SetQbDiv (T_idtDrvHdlr hdlr, T_idtApplId apllInstId, T_idtApplConfigSel config, T_idtApplQbDiv 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.*

6.17.1 Macro Definition Documentation

6.17.1.1 #define APLL1_REG_HIGH_BIT (0)

Definition at line 61 of file ApllApi.c.

6.17.1.2 #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.

6.17.1.3 #define APLL2_REG_HIGH_BIT (1)

Definition at line 62 of file ApIApi.c.

6.17.1.4 #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.

6.17.1.5 #define APLL_DIVIDER_LSB(val) ((val)&0x00FF)

Definition at line 45 of file ApIApi.c.

6.17.1.6 #define APLL_DIVIDER_MSB(val) (((val)&0x7F00)>>8)

Definition at line 46 of file ApIApi.c.

6.17.1.7 #define APLL_DIVIDER_VAL(lsb, msb) ((msb)<<8|(lsb))

Definition at line 47 of file ApIApi.c.

6.17.1.8 #define APLL_FB_DIV_LSB(val) APLL_DIVIDER_LSB(val)

Definition at line 52 of file ApIApi.c.

6.17.1.9 #define APLL_FB_DIV_MAX (0x7FFF)

Definition at line 59 of file ApIApi.c.

6.17.1.10 #define APLL_FB_DIV_MIN (0x0005)

Definition at line 58 of file ApIApi.c.

6.17.1.11 #define APLL_FB_DIV_MSB(val) APLL_DIVIDER_MSB(val)

Definition at line 53 of file ApIApi.c.

6.17.1.12 #define APLL_FB_DIV_VAL(lsb, msb) APLL_DIVIDER_VAL(lsb, msb)

Definition at line 54 of file ApIApi.c.

6.17.1.13 `#define APLL_PRE_DIV_LSB( val ) APLL_DIVIDER_LSB(val)`

Definition at line 49 of file ApllApi.c.

6.17.1.14 `#define APLL_PRE_DIV_MAX (0x7FFF)`

Definition at line 57 of file ApllApi.c.

6.17.1.15 `#define APLL_PRE_DIV_MIN (0x0004)`

Definition at line 56 of file ApllApi.c.

6.17.1.16 `#define APLL_PRE_DIV_MSB( val ) APLL_DIVIDER_MSB(val)`

Definition at line 50 of file ApllApi.c.

6.17.1.17 `#define APLL_PRE_DIV_VAL( lsb, msb ) APLL_DIVIDER_VAL(lsb, msb)`

Definition at line 51 of file ApllApi.c.

6.17.1.18 `#define APLL_REG_HIGH_BIT_OFFSET (7)`

Definition at line 63 of file ApllApi.c.

6.17.1.19 `#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.

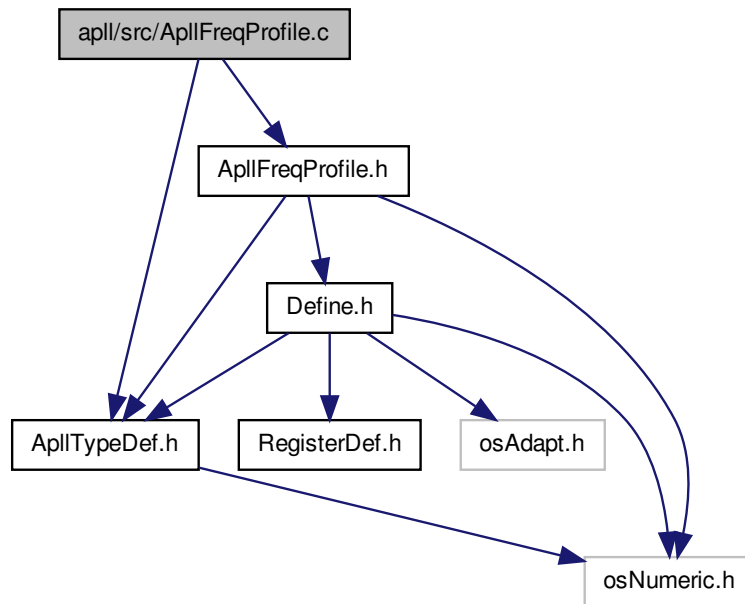
6.17.1.20 `#define INVALID_XTAL_SEL_VALUE (0xFF)`

Definition at line 89 of file ApllApi.c.

6.18 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_VCXO_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_idtApllFreqMapping` * `IDTApll_GetCurrentApllFreqConfig` (`T_idtApllId` apllId, `T_idtApllOutput` output)

Function to retrieve the current APLL frequency mapping configuration for the output.

6.18.1 Macro Definition Documentation

6.18.1.1 #define APLL_FB_DIV_ETH (3124)

Definition at line 45 of file `ApllFreqProfile.c`.

6.18.1.2 #define APLL_FB_DIV_ETH_FEC (3125)

Definition at line 47 of file `ApllFreqProfile.c`.

6.18.1.3 #define APLL_FB_DIV_SONET (3128)

Definition at line 46 of file `ApllFreqProfile.c`.

6.18.1.4 #define APLL_OUTPUT_DIV_1 (1)

Definition at line 50 of file `ApllFreqProfile.c`.

6.18.1.5 #define APLL_OUTPUT_DIV_2 (2)

Definition at line 51 of file `ApllFreqProfile.c`.

6.18.1.6 #define APLL_OUTPUT_DIV_25 (25)

Definition at line 55 of file `ApllFreqProfile.c`.

6.18.1.7 #define APLL_OUTPUT_DIV_4 (4)

Definition at line 52 of file `ApllFreqProfile.c`.

6.18.1.8 #define APLL_OUTPUT_DIV_5 (5)

Definition at line 53 of file `ApllFreqProfile.c`.

6.18.1.9 #define APLL_OUTPUT_DIV_8 (8)

Definition at line 54 of file `ApllFreqProfile.c`.

6.18.1.10 #define APLL_PRE_DIV_ETH (19525)

Definition at line 40 of file `ApllFreqProfile.c`.

6.18.1.11 #define APLL_PRE_DIV_ETH_FEC (15625)

Definition at line 42 of file `ApllFreqProfile.c`.

6.18.1.12 #define APLL_PRE_DIV_SONET (9775)

Definition at line 41 of file ApIIFreqProfile.c.

6.18.1.13 #define MAX_CONFIG_COMBO_PER_FREQ (1)

Definition at line 57 of file ApIIFreqProfile.c.

6.18.1.14 #define MAX_CONFIG_COMBO_PER_VC XO_TYPE (6)

Definition at line 58 of file ApIIFreqProfile.c.

6.18.1.15 #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              \
}
```

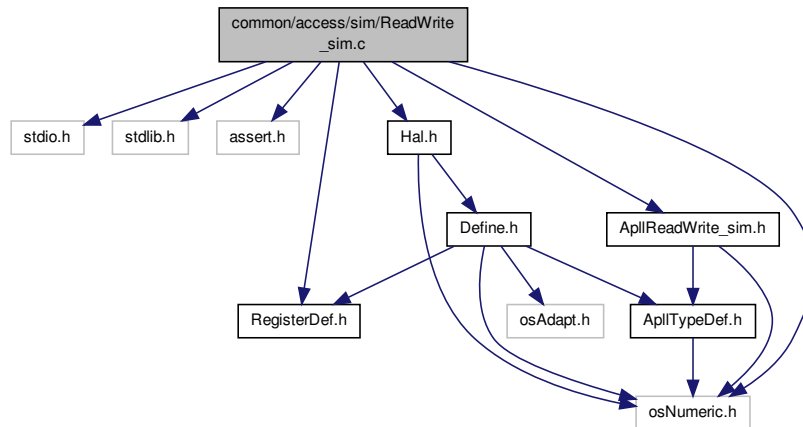
Utility marco for initializing NULL APLL configuration entry.

Definition at line 63 of file ApIIFreqProfile.c.

6.19 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` []

6.19.1 Macro Definition Documentation

6.19.1.1 `#define DRV_MEMMAP_SIZE 0x80`

Definition at line 45 of file ReadWrite_sim.c.

6.19.1.2 `#define NONE_APLL_REG_SPACE_FLAG 0x01`

Definition at line 46 of file ReadWrite_sim.c.

6.19.2 Function Documentation

6.19.2.1 void Delnit_Sim (void * *usrData*)

Definition at line 162 of file ReadWrite_sim.c.

6.19.2.2 void Init_Sim (void * *usrData*)

Definition at line 157 of file ReadWrite_sim.c.

6.19.2.3 T_uint8 RDByte_Sim (void * *usrData*, T_uint8 *devAddr*, T_uint8 *nAddr*)

Definition at line 167 of file ReadWrite_sim.c.

6.19.2.4 void WRByte_Sim (void * *usrData*, T_uint8 *devAddr*, T_uint8 *nAddr*, T_uint8 *nData*)

Definition at line 222 of file ReadWrite_sim.c.

6.19.3 Variable Documentation

6.19.3.1 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.

6.19.3.2 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.

6.19.3.3 int simDebugFlag = 0

Definition at line 48 of file ReadWrite_sim.c.

6.19.3.4 T_osUInt8 simulatedMemMap[2][DRV_MEMMAP_SIZE]

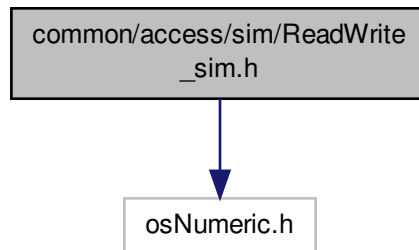
Reset values for device

Definition at line 51 of file ReadWrite_sim.c.

6.20 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)

6.20.1 Function Documentation

6.20.1.1 void DelInit_Sim (void * *usrData*)

Definition at line 162 of file ReadWrite_sim.c.

6.20.1.2 void Init_Sim (void * *usrData*)

Definition at line 157 of file ReadWrite_sim.c.

6.20.1.3 T_osUInt8 RDByte_Sim (void * *usrData*, T_osUInt8 *devAddr*, T_osUInt8 *nAddr*)

Definition at line 167 of file ReadWrite_sim.c.

6.20.1.4 void WRByte_Sim (void * usrData, T_osUInt8 devAddr, T_osUInt8 nAddr, T_osUInt8 nData)

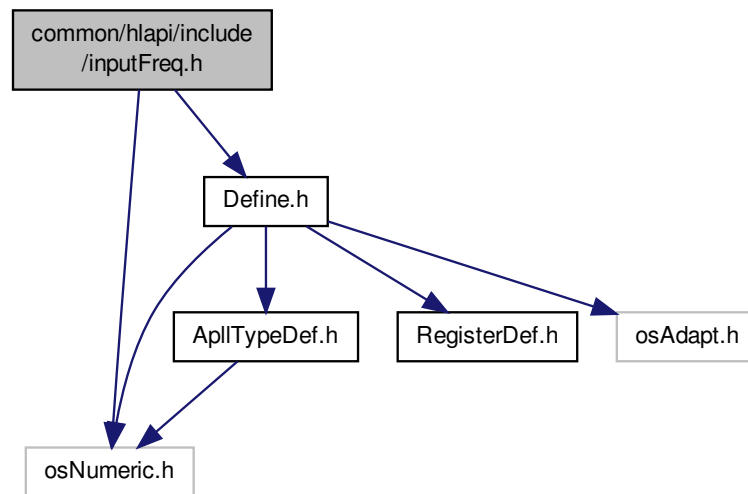
Definition at line 222 of file ReadWrite_sim.c.

6.21 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.

6.21.1 Function Documentation

6.21.1.1 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.

6.21.1.2 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.

6.21.1.3 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.

6.21.1.4 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

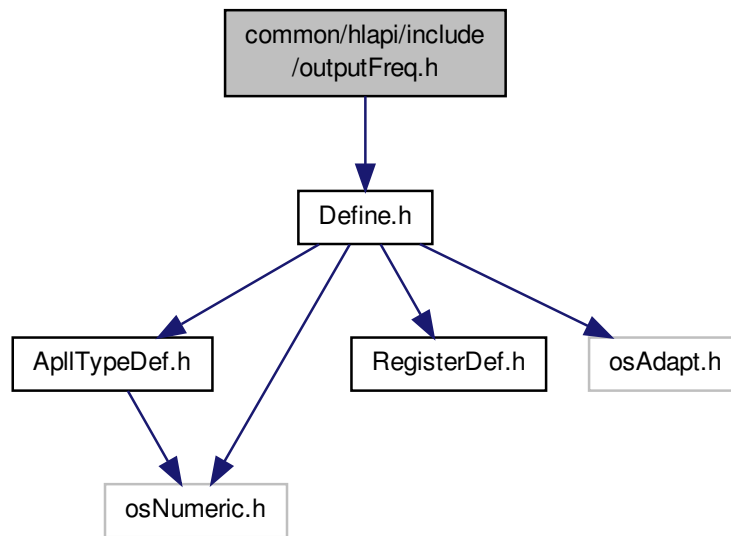
<i>hdlr</i>	Device driver handle
-------------	----------------------

Definition at line 470 of file inputFreq.c.

6.22 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_idtDpllInstance` dpIIInstance, `T_sonetGigeMode` sonetGigeMode, `T_idtApIIConfigSel` apIIConfSel, `T_idtApIIOutputSigType` apIIOutputSigType, `T_idtApIIInputClkSrc` apIIClkSource)

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)

6.22.1 Enumeration Type Documentation

6.22.1.1 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.

6.22.1.2 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.

6.22.2 Function Documentation

6.22.2.1 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.

6.22.2.2 int IDTHIApi_ConfigureOutput (T_idtDrvHdlr *hdlr*, T_osUInt8 *nOutN*, outputFrequency_t *nOutFreq*, T_idtDpllInstance *dpllInstance*, T_sonetGigeMode *sonetGigeMode*, T_idtApllConfigSel *apllConfSel*, T_idtApllOutputSigType *apllOutputSigType*, T_idtApllInputClkSrc *apllClkSource*)

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,

<i>dpllInstance</i>	<i>sonetGigeMode</i>	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 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 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 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, OR if you want to program backup APLL configuration
<i>apllOutputSigType</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.

6.22.2.3 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.

6.22.2.4 void IDTHIApi_PrintInput (T_idtDrvHdlr *hdlr*)

Definition at line 948 of file outputFreqString.c.

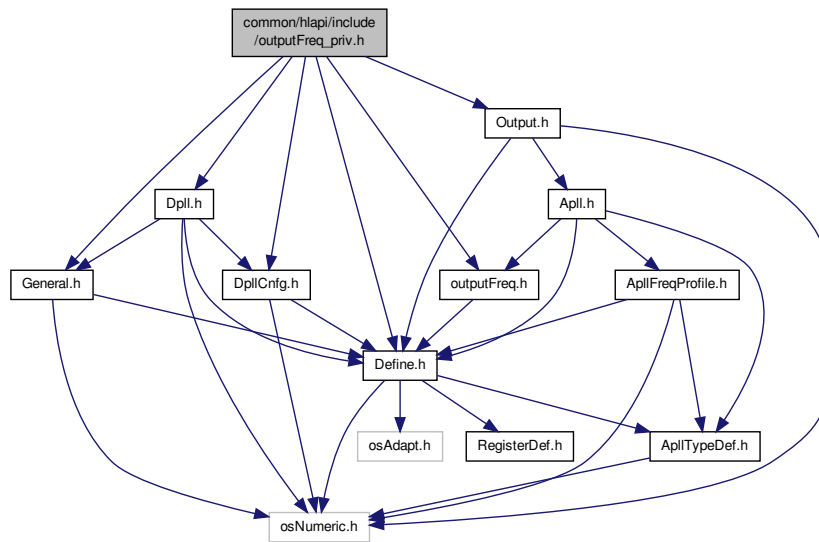
6.22.2.5 void IDTHIApi_PrintOutput (T_idtDrvHdlr *hdlr*)

Definition at line 1011 of file outputFreqString.c.

6.23 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`]

6.23.1 Macro Definition Documentation

6.23.1.1 `#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`.

6.23.1.2 `#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`.

6.23.2 Typedef Documentation

6.23.2.1 `typedef int validFreq_t[OutFreq_MAX]`

Array type for storing frequency usage.

Definition at line 104 of file `outputFreq_priv.h`.

6.23.3 Enumeration Type Documentation

6.23.3.1 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.

6.23.4 Function Documentation

6.23.4.1 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.

6.23.4.2 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.

6.23.4.3 int IDTHIApi.IsFreqArrayInFreqList (const outputFrequency_t * frequencies, validFreq_t validFrequencies)

6.23.4.4 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.

6.23.4.5 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.

6.23.4.6 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.

6.23.4.7 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.

6.23.4.8 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.

6.23.4.9 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.

6.23.4.10 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.

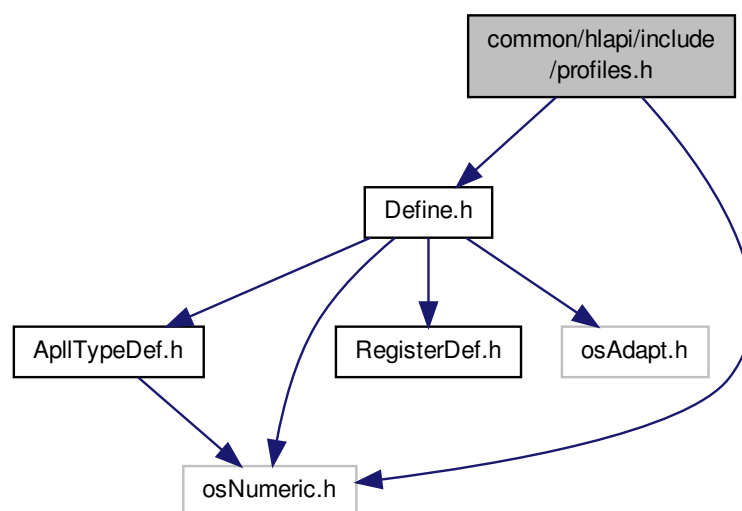
6.23.5 Variable Documentation

6.23.5.1 const char* IDTHIApi_Frequency2String_c[OutFreq_MAX]

Definition at line 77 of file outputFreqString.c.

6.24 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.

6.24.1 Function Documentation

6.24.1.1 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.

6.24.1.2 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.

6.24.1.3 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.

6.24.1.4 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.

6.24.1.5 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.

6.24.1.6 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.

6.24.1.7 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.

6.24.1.8 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.

6.24.1.9 void IDTHIApi_wideband (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

DPLL configuration for wideband.

Parameters

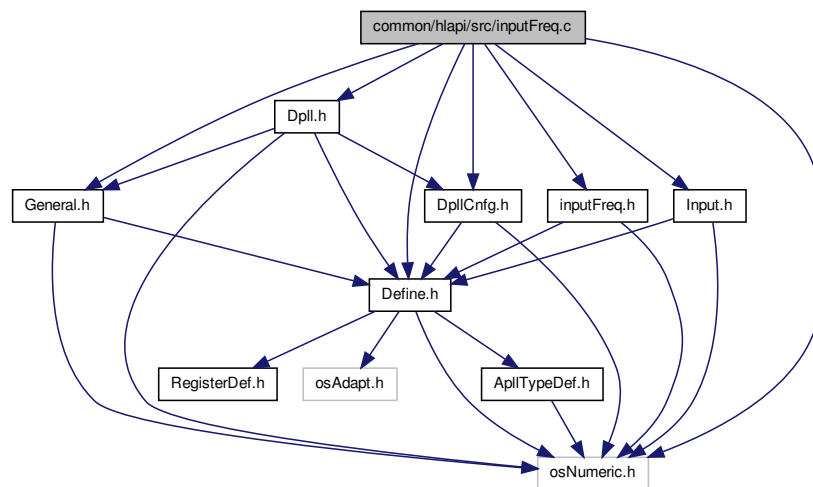
<i>hdlr</i>	Device driver handler
<i>nDpll</i>	DPLL instance

Definition at line 523 of file profiles.c.

6.25 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_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.

6.25.1 Function Documentation

6.25.1.1 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.

6.25.1.2 void IDTHIApi_ConfigureInputAmi (T_idtDrvHdlr hdlr, T_osUInt8 nInputNo, T_idtAmiInputFrequencyBitFieldValue 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 T_idtAmiInputFrequencyBitFieldValue

Definition at line 222 of file inputFreq.c.

6.25.1.3 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.

6.25.1.4 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.

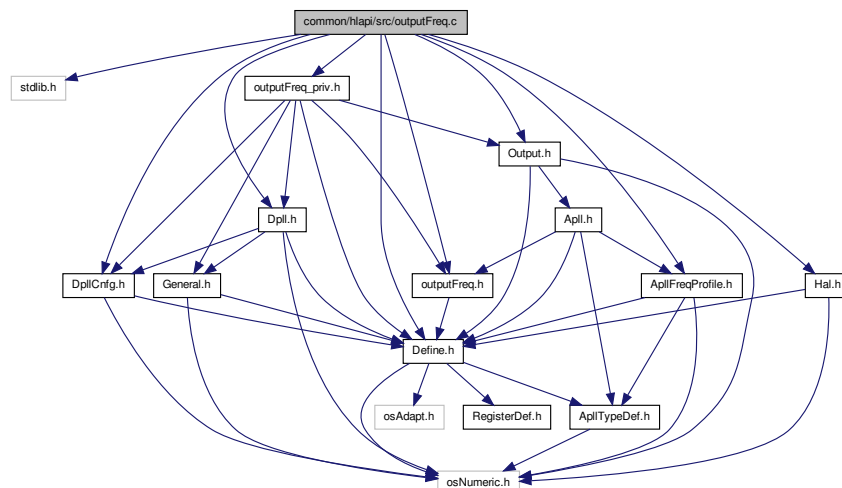
Parameters

<i>hdlr</i>	Device driver handle
-------------	----------------------

Definition at line 470 of file inputFreq.c.

6.26 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](#) dpllInstance, [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.

6.26.1 Macro Definition Documentation

6.26.1.1 #define MAX_SUPPORTED_FREQUENCIES 9

Definition at line 53 of file outputFreq.c.

6.26.1.2 #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.

6.26.2 Function Documentation

6.26.2.1 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.

6.26.2.2 int IDTHIApi_ConfigureOutput (T_idtDrvHdlr hdlr, T_osUInt8 nOutN, [outputFrequency_t](#) nOutFreq, [T_idtDpllInstance](#) dpllInstance, [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 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,

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 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 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 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.

6.26.2.3 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.

6.26.2.4 void IDTHIApi_RetrieveOutputPathMuxFromHardware (T_idtDrvHdlr *hdlr*, T_IDTPathConfiguration * *pathConfig_p*)

Function to retrieve output path configuration from hardware.

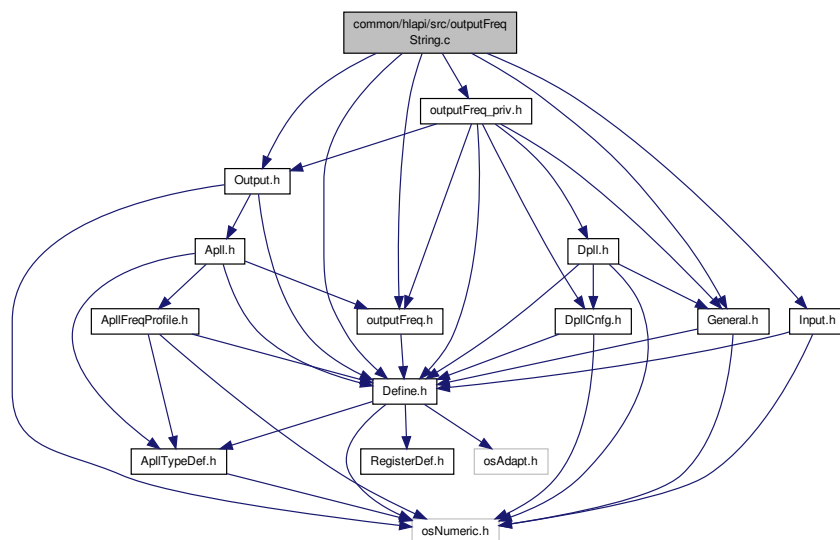
Parameters

<i>hdlr</i>	Device driver handler
<i>pathConfig_p</i>	Output path configuration

Definition at line 282 of file outputFreq.c.

6.27 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](#)]

6.27.1 Enumeration Type Documentation

6.27.1.1 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.

6.27.2 Function Documentation

6.27.2.1 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.

6.27.2.2 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.

6.27.2.3 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.

6.27.2.4 void IDTHIApi_PrintInput (T_idtDrvHdlr hdlr)

Definition at line 948 of file outputFreqString.c.

6.27.2.5 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.

6.27.2.6 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.

6.27.2.7 void IDTHIApi_PrintOutput (T_idtDrvHdlr *hdlr*)

Definition at line 1011 of file outputFreqString.c.

6.27.2.8 void IDTHIApi_PrintOutputHwConfig (networkType_t *network*, T_osUInt8 *nOutN*, T_osUInt8 *internalPort*, T_osUInt8 *isEnabled*, T_osUInt8 *hasAppl*, T_idtApIldividerValue *apIldivVal*, 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.

6.27.2.9 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.

6.27.3 Variable Documentation

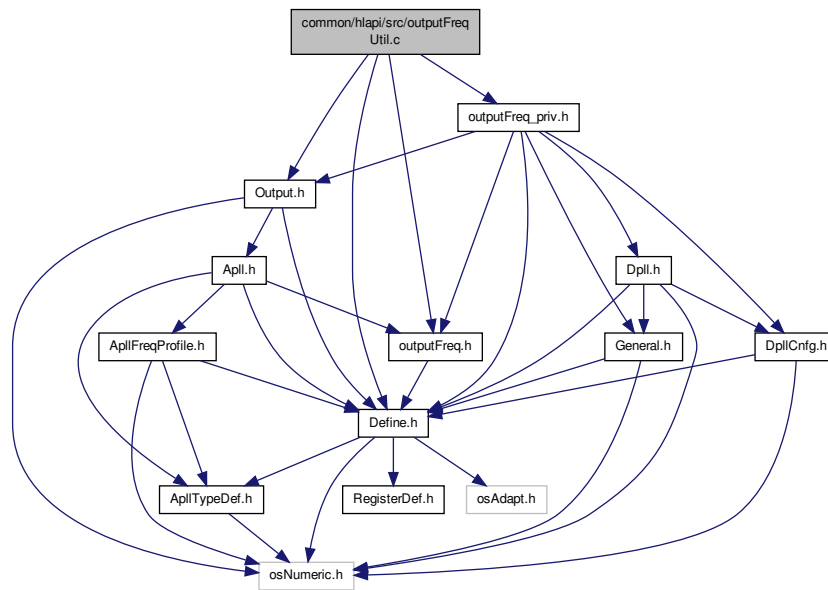
6.27.3.1 const char* IDTHIApi_Frequency2String_c[OutFreq_MAX]

Definition at line 77 of file outputFreqString.c.

6.28 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.

6.28.1 Function Documentation

6.28.1.1 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.

6.28.1.2 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.

6.28.1.3 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.

6.28.1.4 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

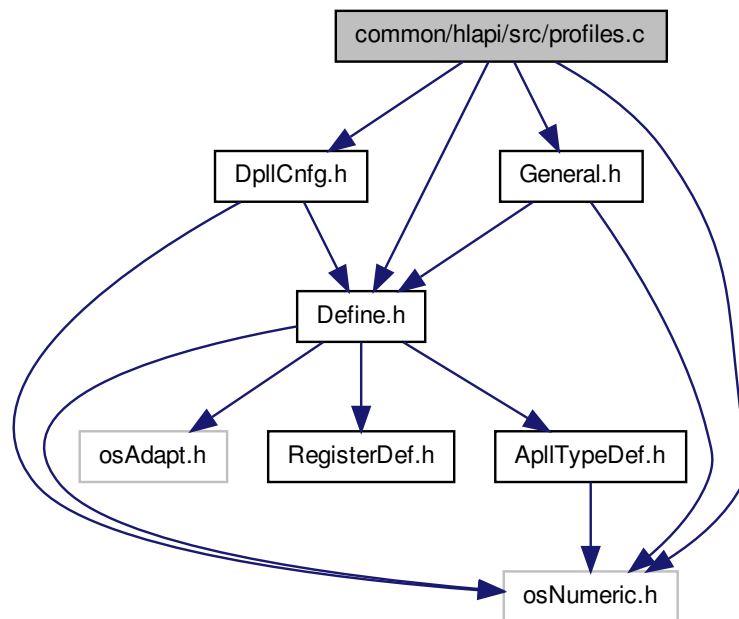
1-Frequency is in list, 0-Frequency is not in list

Definition at line 124 of file outputFreqUtil.c.

6.29 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.

6.29.1 Detailed Description

This sample file contains DPLL configurations used to conform to various standards.

Definition in file [profiles.c](#).

6.29.2 Function Documentation

6.29.2.1 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.

6.29.2.2 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.

6.29.2.3 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.

6.29.2.4 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.

6.29.2.5 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.

6.29.2.6 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.

6.29.2.7 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.

6.29.2.8 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.

6.29.2.9 void IDTHIApi_wideband (T_idtDrvHdlr *hdlr*, T_idtDpllInstance *nDpll*)

DPLL configuration for wideband.

Parameters

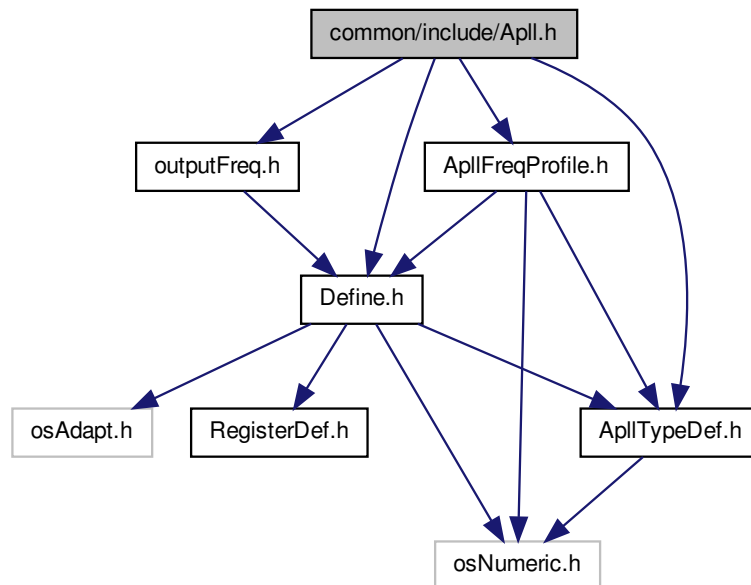
<i>hdlr</i>	Device driver handler
<i>nDpll</i>	DPLL instance

Definition at line 523 of file profiles.c.

6.30 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)

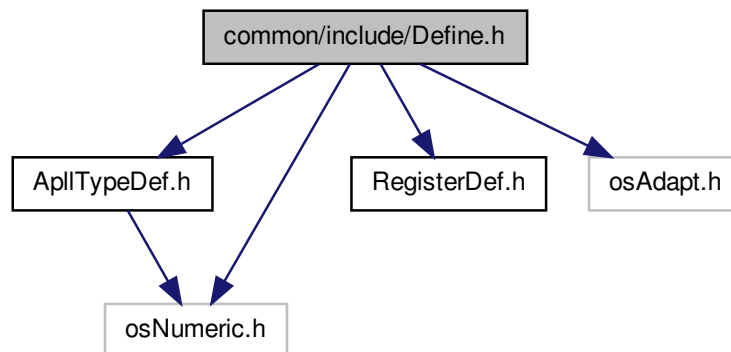
Checks if there is input XTAL information is valid for the device.
- `T_idtApllXtal` * `IDTApll_GetSupportedXtal` (const `T_idtDrvDeviceConfig` *deviceConfig, `T_osUInt8` *numberOfXtal)

Return supported xtal configuration by the device.

6.31 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.

6.31.1 Macro Definition Documentation

6.31.1.1 #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.

6.31.1.2 #define IDT_DRV_MAX_APLL_XTAL 4

Define maximum number of APLL XTALs per device.

Definition at line 110 of file Define.h.

6.31.1.3 #define IDT_DRV_MAX_INPUT 14

Define maximum number of supported inputs.

Definition at line 104 of file Define.h.

6.31.1.4 #define IDT_DRV_MAX_OUTPUT 11

Define maximum number of supported outputs.

Definition at line 107 of file Define.h.

6.31.1.5 #define IDT_DRV_MAX_XTAL_PER_APLL 2

Define maximum number of XTALs per APLL.

Definition at line 113 of file Define.h.

6.31.1.6 #define MPU_I2C_MODE 0x00

Definition at line 62 of file Define.h.

6.31.1.7 #define MPU_SERIAL_MODE 0x01

Definition at line 63 of file Define.h.

6.31.1.8 #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.

6.31.2 Typedef Documentation**6.31.2.1 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.

6.31.2.2 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.

6.31.2.3 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.

6.31.2.4 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.

6.31.2.5 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.

6.31.2.6 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.

6.31.2.7 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.

6.31.3 Enumeration Type Documentation**6.31.3.1 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.

6.31.3.2 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.

6.31.3.3 enum T_idtDpllType

Enumerated type for selecting DPLLs (T0 or T4)

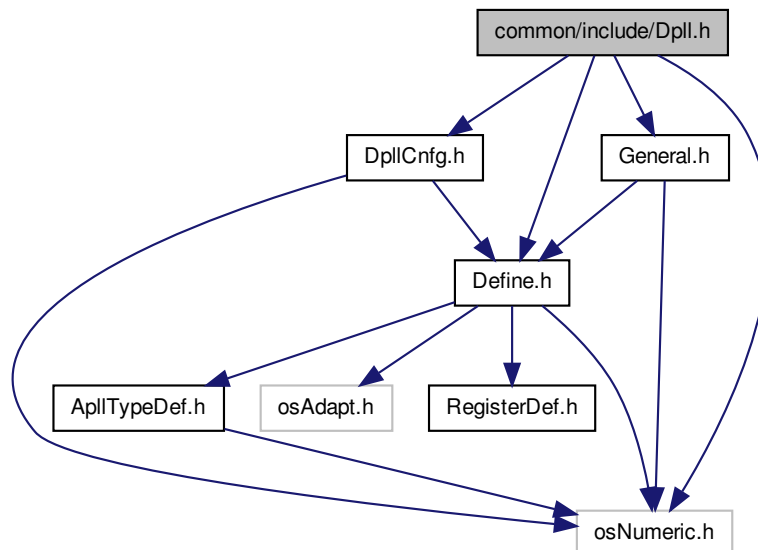
Enumerator

Dpll_T0 Select T0 DPLL
Dpll_T4 Select T4 DPLL
Dpll_MAX

Definition at line 53 of file Define.h.

6.32 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.

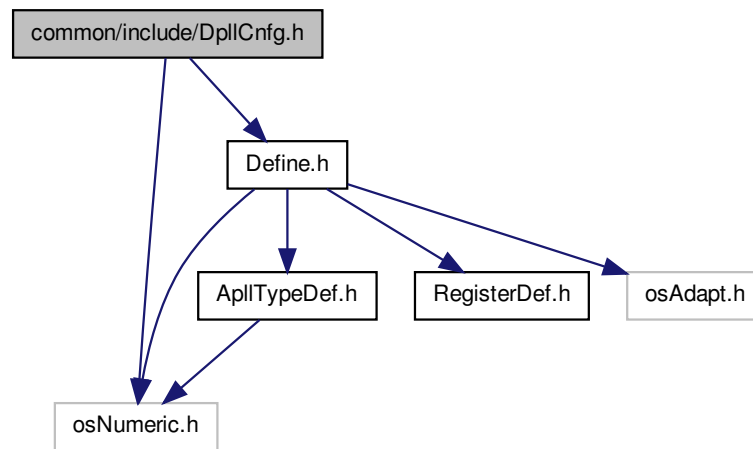
- const `T_idtSonetGigEthSelector IDTDpll_sonetGigEthPathConfig2Bitfield [Dpll_MAX][SonetGigEthOutput - MAX]`

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

6.33 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`, `SonetGigEth-`
`Output_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.

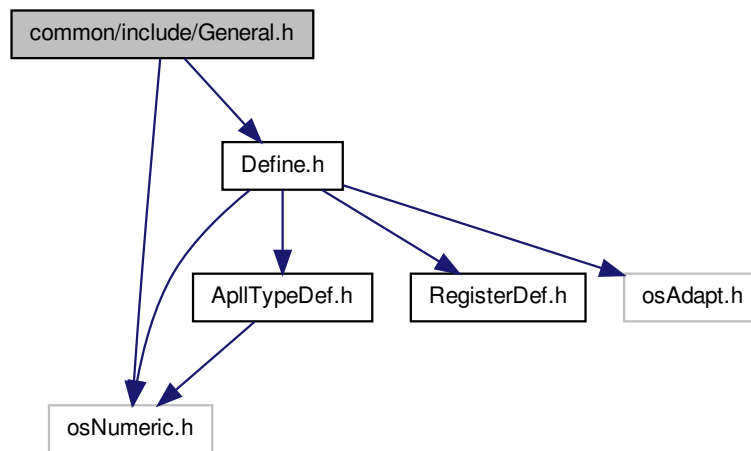
- void IDTDpll_GetPpsPhase (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_idtDpllPpsPhasePostion *nPhase_p, T_idtDpllPpsPulseWidth *nPulse_p)

Get PPS phase and pulse for DPLL.

6.34 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

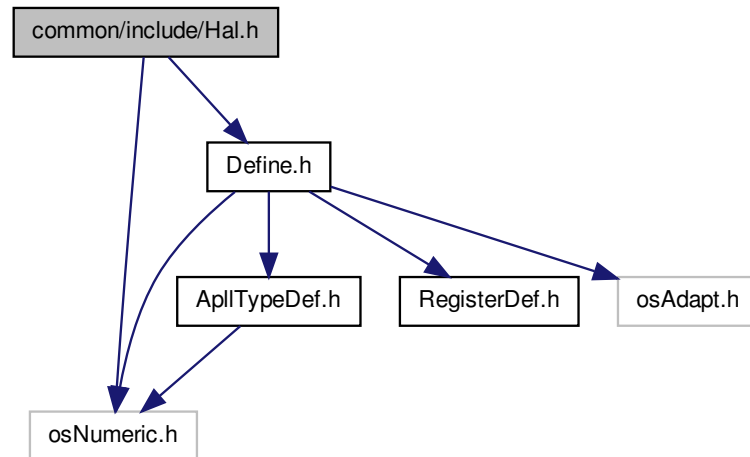
- enum [networkType_t](#) { [Network_SDH](#) = 0, [Network_SONET](#) = 1, [Network_MAX](#) }
 - enum [phaseTimeoutMultiplier_t](#) { [phaseTimeoutMultiplier_2](#) = 0, [phaseTimeoutMultiplier_4](#) = 1, [phaseTimeoutMultiplier_8](#) = 2, [phaseTimeoutMultiplier_16](#) = 3, [phaseTimeoutMultiplier_MAX](#) }
- 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.

6.35 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.

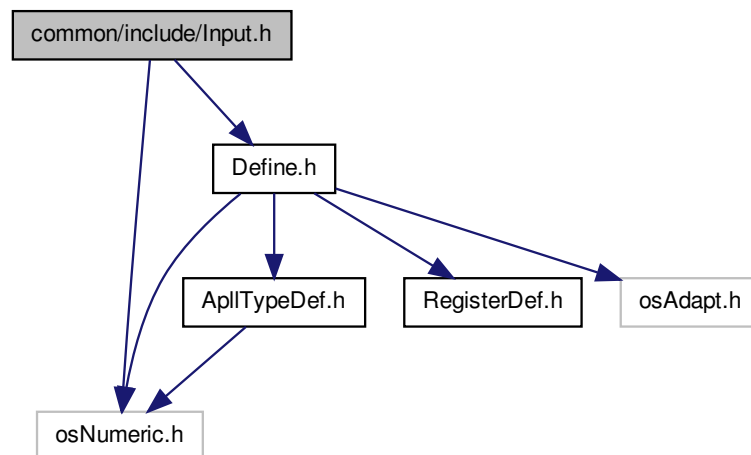
Functions

- 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.

6.36 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)

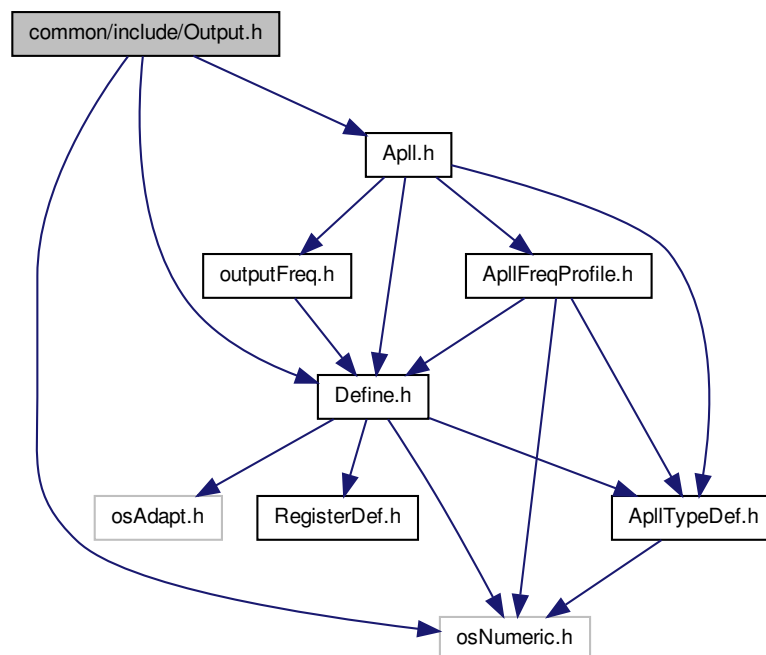
Function to provision enable mode for input external sync.
- [T_externalSyncEnable IDTInput_GetSyncEnable](#) (T_idtDrvHdlr hdlr)

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.

6.37 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_OUTPORT_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.

6.38 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_PEC-
L_LVDS_SIZE = 1,
BF_OUT7_PEC-
L_LVDS_MASK = 0x02, BF_OUT7_PEC-
L_LVDS_LSB = 1, BF_OUT6_PEC-
L_LVDS_SIZE = 1, BF_OUT6_PEC-
L_LVDS_MASK = 0x01,
BF_OUT6_PEC-
L_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.

6.38.1 Detailed Description

This header file contains register and bitfield information for accessing device.

See Also

Datasheet for detailed register descriptions.

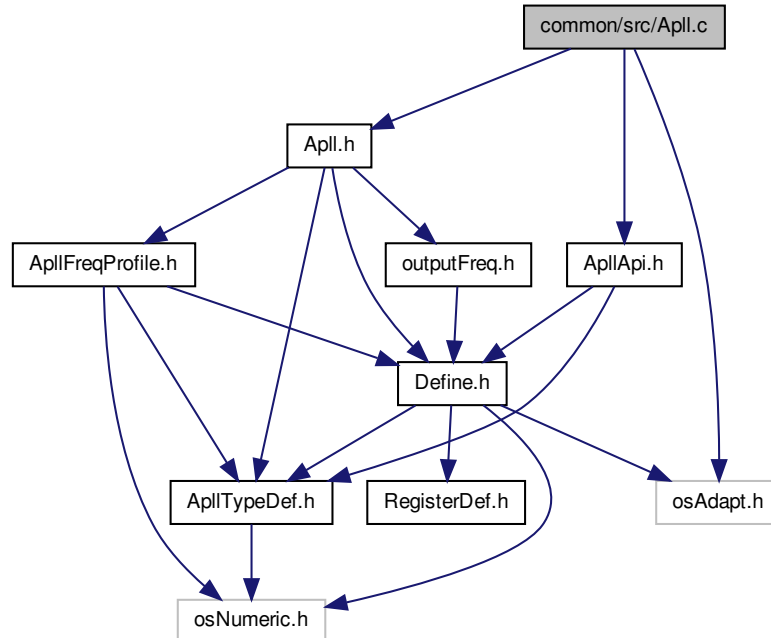
Definition in file [RegisterDef.h](#).

6.39 common/src/Apll.c File Reference

```

#include "Apl1.h"
#include "Apl1Api.h"
#include "osAdapt.h"
Include dependency graph for Apll.c:

```



Functions

- void [IDTApl1_SetApl1ConfigPerOutput](#) (T_idtDrvHdlr hdlr, T_idtApl1Id ap1lInstId, T_idtApl1Output output, T_idtApl1ConfigPerOutput *ap1lConfigPerOutput)

Set APLL per output configuration.

- void `IDTApll_GetApllConfigPerOutput` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllInstId, `T_idtApllOutput` output, `T_idtApllConfigPerOutput` *apllConfigPerOutput)

Get APLL per output configuration.

- void `IDTApll_SetApllConfig` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllInstId, `T_idtApllConfig` *apllConfig)

Set APLL full configuration.

- void `IDTApll_GetApllConfig` (`T_idtDrvHdlr` hdlr, `T_idtApllId` apllInstId, `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` apllInst)

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` apllInst, `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.

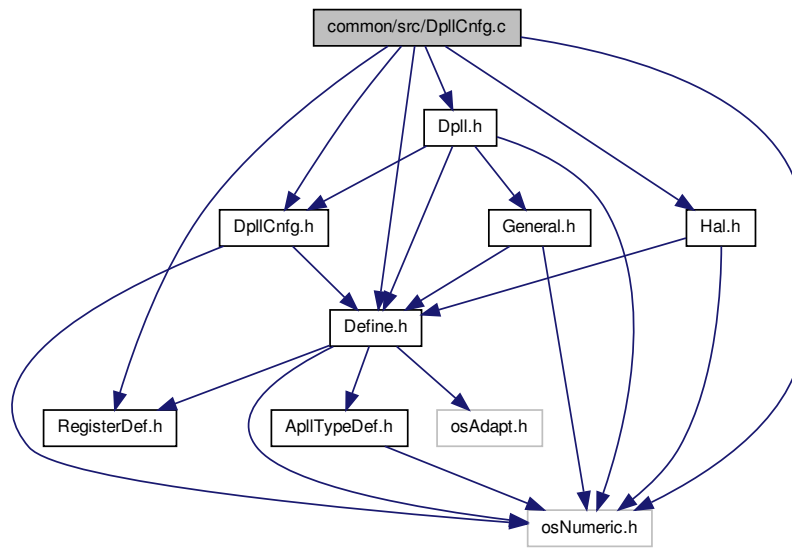
- `T_idtApllXtal` * `IDTApll_GetSupportedXtal` (const `T_idtDrvDeviceConfig` *deviceConfig, `T_osUInt8` *numberOfXtal)

Return supported xtal configuration by the device.

6.40 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 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.

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.

6.40.1 Macro Definition Documentation

6.40.1.1 #define INVALID_REGISTER_OFFSET ((T_osUInt8)(-1))

Tag used to indicating invalid register offset.

Definition at line 46 of file `DpllCnfg.c`.

6.40.2 Variable Documentation

6.40.2.1 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
    }
}
```

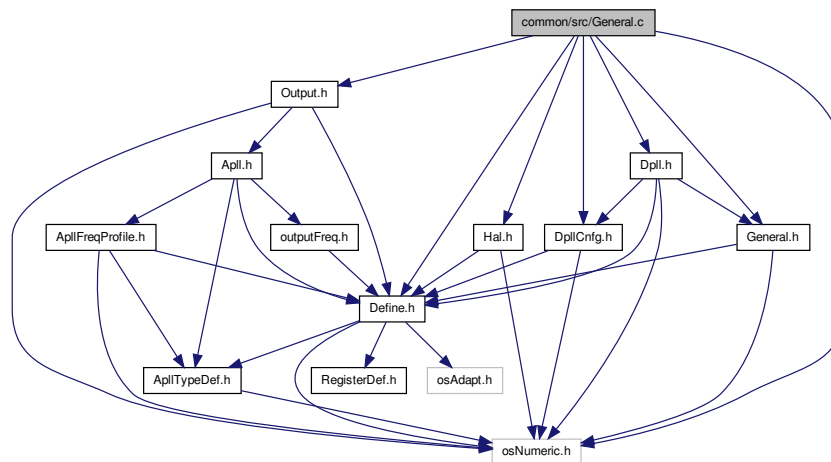
Structure used to determine which register to access for bandwidth limiting.

Definition at line 75 of file `DpllCnfg.c`.

6.41 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:



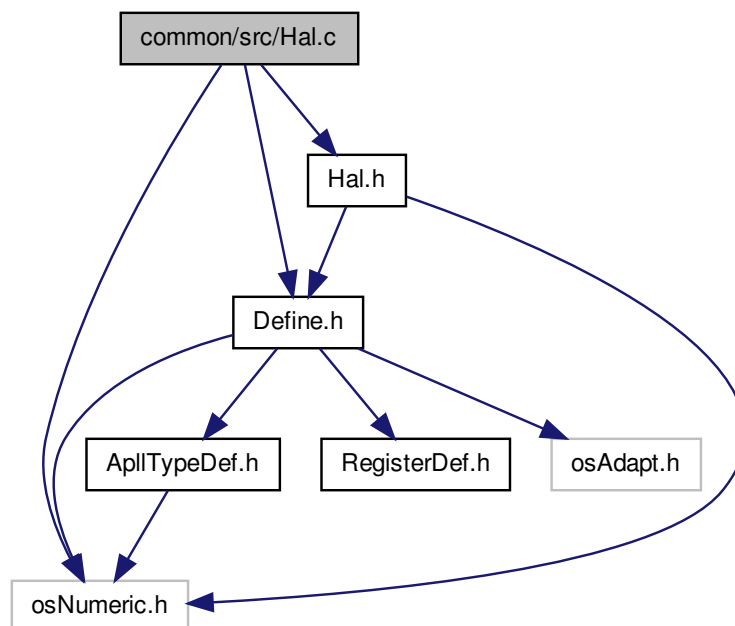
Functions

- 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.

6.42 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.

6.42.1 Macro Definition Documentation

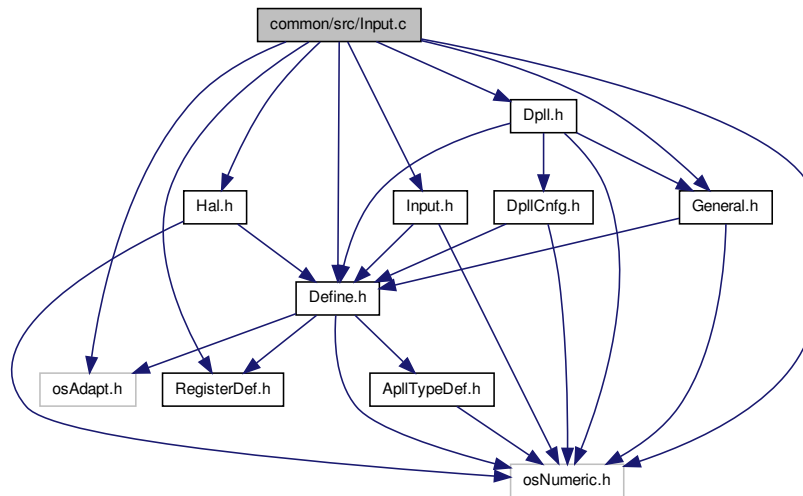
6.42.1.1 `#define DEBUG_IDT_HAL`

Definition at line 41 of file Hal.c.

6.43 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)

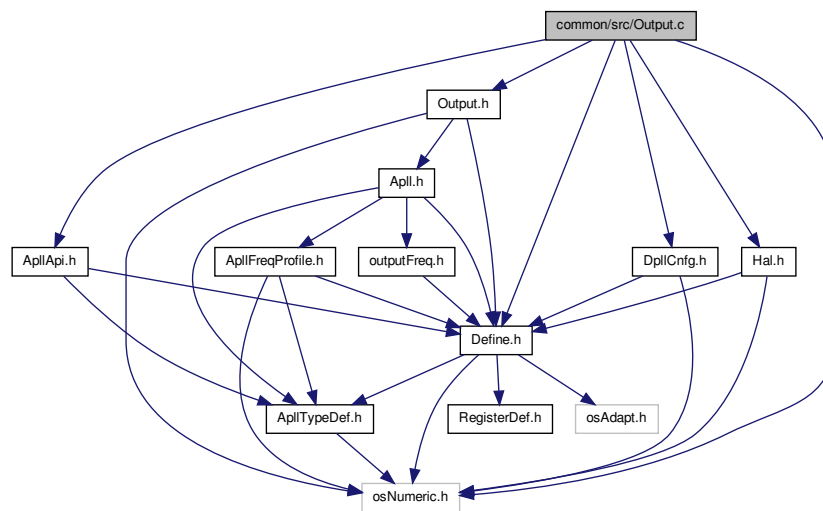
Provision AMI signal for input port.

- [T_idtAmiInputFrequencyBitFieldValue](#) [IDTInput_GetAmiInputFrequency](#) (T_idtDrvHdlr hdlr, T_osUInt8 nInputNo)

Function to retrieve AMI input frequency for input port.

6.44 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.

6.44.1 Enumeration Type Documentation

6.44.1.1 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.

6.44.1.2 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

82V3911.c
 IDTGeneral_InitDevice, [177](#)
 IDTGeneral_InitDriver, [177](#)
 idt82V3911Config, [178](#)
82V3911/dev/include/idt82V3911.h, [175](#)
82V3911/dev/src/82V3911.c, [176](#)
82V3911/sample/include/access.h, [178](#)
82V3911/sample/include/loadstore.h, [181](#)
82V3911/sample/include/sample.h, [182](#)
82V3911/sample/src/access.c, [185](#)
82V3911/sample/src/loadstore.c, [188](#)
82V3911/sample/src/main.c, [189](#)
82V3911/sample/src/sample.c, [194](#)

APLL_ACCESS_I2C
 ApI Function Module, [124](#)
APLL_ACCESS_MAX
 ApI Function Module, [124](#)
APLL_ACCESS_SIM
 ApI Function Module, [124](#)
APLL_BYPASS_BYPASS
 ApI Function Module, [124](#)
APLL_BYPASS_MAX
 ApI Function Module, [124](#)
APLL_BYPASS_NORMAL_OP
 ApI Function Module, [124](#)
APLL_CLK_SEL_EXTERNAL
 ApI Function Module, [125](#)
APLL_CLK_SEL_INTERNAL
 ApI Function Module, [125](#)
APLL_CLK_SEL_MAX
 ApI Function Module, [125](#)
APLL_CONFIG0
 ApI Function Module, [124](#)
APLL_CONFIG1
 ApI Function Module, [124](#)
APLL_CONFIG_MAX
 ApI Function Module, [124](#)
APLL_FREQ_SEL_DOUBLE
 ApI Function Module, [125](#)
APLL_FREQ_SEL_MAX
 ApI Function Module, [125](#)
APLL_FREQ_SEL_SINGLE
 ApI Function Module, [125](#)
APLL_IN_MR_SYNC
 ApI Function Module, [126](#)
APLL_INPUT_FREQ_125000KHz
 ApI Function Module, [125](#)
APLL_INPUT_FREQ_128906_DOT_25KHz
 ApI Function Module, [125](#)

APLL_INPUT_FREQ_155520KHz
 ApI Function Module, [125](#)
APLL_INPUT_FREQ_156250KHz
 ApI Function Module, [125](#)
APLL_INPUT_FREQ_19440KHz
 ApI Function Module, [125](#)
APLL_INPUT_FREQ_25000KHz
 ApI Function Module, [125](#)
APLL_INPUT_FREQ_25781_DOT_25KHz
 ApI Function Module, [125](#)
APLL_INPUT_FREQ_312500KHz
 ApI Function Module, [125](#)
APLL_INPUT_FREQ_77760KHz
 ApI Function Module, [125](#)
APLL_INPUT_FREQ_8KHz
 ApI Function Module, [125](#)
APLL_INPUT_FREQ_MAX
 ApI Function Module, [125](#)
APLL_INST_1
 ApI Function Module, [125](#)
APLL_INST_2
 ApI Function Module, [125](#)
APLL_INST_MAX
 ApI Function Module, [125](#)
APLL_MR_SYNC_MAX
 ApI Function Module, [126](#)
APLL_OUT_MAX
 ApI Function Module, [127](#)
APLL_OUT_MR_SYNC
 ApI Function Module, [126](#)
APLL_OUT_QA_ONLY
 ApI Function Module, [127](#)
APLL_OUT_QA_OR_QB
 ApI Function Module, [127](#)
APLL_OUT_QB_ONLY
 ApI Function Module, [127](#)
APLL_OUT_SIG_LVDS
 ApI Function Module, [127](#)
APLL_OUT_SIG_LVPECL
 ApI Function Module, [127](#)
APLL_OUT_SIG_MAX
 ApI Function Module, [127](#)
APLL_OUTPUT_DISABLE
 ApI Function Module, [126](#)
APLL_OUTPUT_EN_MAX
 ApI 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, [207](#)
- APLL2_REG_HIGH_BIT
 - ApIApi.c, [207](#)
- APLL_CONFIGURED
 - Define.h, [247](#)
- APLL_DIVIDER_LSB
 - ApIApi.c, [208](#)
- APLL_DIVIDER_MSB
 - ApIApi.c, [208](#)
- APLL_DIVIDER_VAL
 - ApIApi.c, [208](#)
- APLL_FB_DIV_ETH
 - ApIFreqProfile.c, [211](#)
- APLL_FB_DIV_LSB
 - ApIApi.c, [208](#)
- APLL_FB_DIV_MAX
 - ApIApi.c, [208](#)
- APLL_FB_DIV_MIN
 - ApIApi.c, [208](#)
- APLL_FB_DIV_MSB
 - ApIApi.c, [208](#)
- APLL_FB_DIV_SONET
 - ApIFreqProfile.c, [211](#)
- APLL_FB_DIV_VAL

- ApIiApi.c, 208
- APLL_OUTPUT_DIV_1
 - ApIIFreqProfile.c, 211
- APLL_OUTPUT_DIV_2
 - ApIIFreqProfile.c, 211
- APLL_OUTPUT_DIV_25
 - ApIIFreqProfile.c, 211
- APLL_OUTPUT_DIV_4
 - ApIIFreqProfile.c, 211
- APLL_OUTPUT_DIV_5
 - ApIIFreqProfile.c, 211
- APLL_OUTPUT_DIV_8
 - ApIIFreqProfile.c, 211
- APLL_PRE_DIV_ETH
 - ApIIFreqProfile.c, 211
- APLL_PRE_DIV_LSB
 - ApIiApi.c, 208
- APLL_PRE_DIV_MAX
 - ApIiApi.c, 209
- APLL_PRE_DIV_MIN
 - ApIiApi.c, 209
- APLL_PRE_DIV_MSB
 - ApIiApi.c, 209
- APLL_PRE_DIV_VAL
 - ApIiApi.c, 209
- APLL_REG_OFFSET
 - ApIiApi.c, 209
- access.c
 - Delnit_SimWrapper, 186
 - I2cAccessMethods, 188
 - IDTSample_GetLogVerbosity, 186
 - IDTSample_InitDriverI2c, 186
 - IDTSample_InitDriverSimulator, 186
 - IDTSample_ResetXtalList, 186
 - IDTSample_SetLogVerbosity, 187
 - Init_SimWrapper, 187
 - ReadByte_SimWrapper, 187
 - SimAccessMethods, 188
 - WriteByte_SimWrapper, 187
- 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
- ApIi Function Module, 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, 195
- apll/access/sim/ApllReadWrite_sim.h, 197
- apll/include/ApllApi.h, 198
- apll/include/ApllFreqProfile.h, 200
- apll/include/ApllLogging.h, 202
- apll/include/ApllRegDef.h, 202
- apll/include/ApllTypeDef.h, 203
- apll/src/ApllApi.c, 205
- apll/src/ApllFreqProfile.c, 209
- ApllApi.c
 - APLL1_REG_HIGH_BIT, 207
 - APLL2_REG_HIGH_BIT, 207
 - APLL_DIVIDER_LSB, 208
 - APLL_DIVIDER_MSB, 208

- APLL_DIVIDER_VAL, [208](#)
- APLL_FB_DIV_LSB, [208](#)
- APLL_FB_DIV_MAX, [208](#)
- APLL_FB_DIV_MIN, [208](#)
- APLL_FB_DIV_MSB, [208](#)
- APLL_FB_DIV_VAL, [208](#)
- APLL_PRE_DIV_LSB, [208](#)
- APLL_PRE_DIV_MAX, [209](#)
- APLL_PRE_DIV_MIN, [209](#)
- APLL_PRE_DIV_MSB, [209](#)
- APLL_PRE_DIV_VAL, [209](#)
- APLL_REG_OFFSET, [209](#)
- apllCfg
 - T_idtApllConfig, [158](#)
 - T_idtApllConfigPerOutput, [159](#)
- ApllDelnit_Sim
 - ApllReadWrite_sim.c, [196](#)
 - ApllReadWrite_sim.h, [198](#)
- apllDivVal_m
 - T_apllOutputFrequencyEntry, [157](#)
- apllFbDivider
 - T_idtApllFreqMapping, [161](#)
- ApllFreqProfile.c
 - APLL_FB_DIV_ETH, [211](#)
 - APLL_OUTPUT_DIV_1, [211](#)
 - APLL_OUTPUT_DIV_2, [211](#)
 - APLL_OUTPUT_DIV_25, [211](#)
 - APLL_OUTPUT_DIV_4, [211](#)
 - APLL_OUTPUT_DIV_5, [211](#)
 - APLL_OUTPUT_DIV_8, [211](#)
 - APLL_PRE_DIV_ETH, [211](#)
- apllI2CTransData_mp
 - drvHdlr_s, [168](#)
- ApllInit_Sim
 - ApllReadWrite_sim.c, [196](#)
 - ApllReadWrite_sim.h, [198](#)
- apllInput
 - T_idtOutputApllConfig, [171](#)
- apllInputFreq
 - T_idtApllFreqMapping, [161](#)
- apllInst
 - T_idtOutputApllConfig, [171](#)
- apllOutput
 - T_idtOutputApllConfig, [171](#)
- apllOutputDivider
 - T_idtApllFreqMapping, [161](#)
- apllOutputFeq
 - T_idtApllFreqMapping, [161](#)
- apllOutputs
 - T_idtApllFreqMapping, [161](#)
- apllPreDivider
 - T_idtApllFreqMapping, [161](#)
- ApllRead_Sim
 - ApllReadWrite_sim.c, [197](#)
 - ApllReadWrite_sim.h, [198](#)
- ApllReadWrite_sim.c
 - ApllDelnit_Sim, [196](#)
 - ApllInit_Sim, [196](#)
 - ApllRead_Sim, [197](#)
 - ApllWrite_Sim, [197](#)
 - simulatedRegMapAppl0, [197](#)
 - simulatedRegMapAppl1, [197](#)
- ApllReadWrite_sim.h
 - ApllDelnit_Sim, [198](#)
 - ApllInit_Sim, [198](#)
 - ApllRead_Sim, [198](#)
 - ApllWrite_Sim, [198](#)
- ApllWrite_Sim
 - ApllReadWrite_sim.c, [197](#)
 - ApllReadWrite_sim.h, [198](#)
- apllXtalFreq
 - T_idtApllFreqMapping, [161](#)
 - T_idtApllXtal, [162](#)
- apllXtalld
 - 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](#)

- 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, [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, [107](#)

- 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](#)

- 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](#)
- 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](#)

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](#)
- 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](#)

- 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, [105](#)

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_HOLDOVER_LSB
 Register definitions, [109](#)
 BF_MAN_HOLDOVER_MASK
 Register definitions, [109](#)
 BF_MAN_HOLDOVER_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](#)

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](#)
- 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, [224](#)
- 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, 212
- common/access/sim/ReadWrite_sim.h, 215
- common/hlapi/include/inputFreq.h, 216
- common/hlapi/include/outputFreq.h, 217
- common/hlapi/include/outputFreq_priv.h, 222
- common/hlapi/include/profiles.h, 227
- common/hlapi/src/inputFreq.c, 230
- common/hlapi/src/outputFreq.c, 232
- common/hlapi/src/outputFreqString.c, 235
- common/hlapi/src/outputFreqUtil.c, 238
- common/hlapi/src/profiles.c, 240
- common/include/Apll.h, 243
- common/include/Define.h, 245
- common/include/Dpll.h, 251
- common/include/DpllCnfg.h, 252
- common/include/General.h, 258
- common/include/Hal.h, 261
- common/include/Input.h, 262
- common/include/Output.h, 265
- common/include/RegisterDef.h, 267
- common/src/Apll.c, 275
- common/src/DpllCnfg.c, 276
- common/src/General.c, 281
- common/src/Hal.c, 283
- common/src/Input.c, 284
- common/src/Output.c, 287
- 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, [251](#)
- DPLL_INSTANCE_2
 - Define.h, [251](#)
- DPLL_INSTANCE_MAX
 - Define.h, [251](#)
- 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, [284](#)
- 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_GetLcpCtrl, 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_SetLcpCtrl, 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, 213
- 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_idtApplConfig, [158](#)
 - T_idtApplConfigPerOutput, [159](#)
- Dco_Mode
 - DPLL Function Module, [18](#)
- dcoMode_m
 - globalArgs_t, [149](#)
- dcoModeSupport_ma
 - T_idtDrvDeviceConfig, [165](#)
- Delnit_Sim
 - ReadWrite_sim.c, [214](#)
 - ReadWrite_sim.h, [215](#)
- Delnit_SimWrapper
 - access.c, [186](#)
- decayRateReg_m
 - bucketReg_t, [147](#)
- Define.h
 - DPLL_INSTANCE_1, [251](#)
 - DPLL_INSTANCE_2, [251](#)
 - DPLL_INSTANCE_MAX, [251](#)
 - Dpll_MAX, [251](#)
 - Dpll_T0, [251](#)
 - Dpll_T4, [251](#)
 - IDT_82V33930, [250](#)
 - IDT_82V3910, [250](#)
 - IDT_82V3911, [250](#)
 - IDT_8V89316, [250](#)
 - IDT_8V89317, [250](#)
 - IDT_DEVICE_MAX, [250](#)
- Define.h
 - APLL_CONFIGURED, [247](#)
 - IDT_DRV_MAX_INPUT, [247](#)
 - IDT_DRV_MAX_OUTPUT, [248](#)
 - MPU_I2C_MODE, [248](#)
 - MPU_SERIAL_MODE, [248](#)
 - NULL_APLL_CONFIG, [248](#)
 - T_idtAccessDelnitFunc, [248](#)
 - T_idtAccessInitFunc, [248](#)
 - T_idtDeviceType, [250](#)
 - T_idtDpllInstance, [250](#)
 - T_idtDpllType, [251](#)
 - T_idtI2CaddrTransFunc, [249](#)
 - T_idtInputFreqValid, [249](#)
 - T_idtOutputFreqValid, [249](#)
 - T_idtReadFunc, [250](#)
 - T_idtWriteFunc, [250](#)
- 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, [251](#)
- Dpll_T0
 - Define.h, [251](#)
- Dpll_T4
 - Define.h, [251](#)
- dppllBandwidth_0p1Hz
 - DPLL Function Module, [17](#)
- dppllBandwidth_0p3Hz
 - DPLL Function Module, [17](#)
- dppllBandwidth_0p5mHz
 - DPLL Function Module, [17](#)
- dppllBandwidth_0p6Hz
 - DPLL Function Module, [17](#)
- dppllBandwidth_15mHz
 - DPLL Function Module, [17](#)
- dppllBandwidth_18Hz
 - DPLL Function Module, [17](#)
- dppllBandwidth_1mHz
 - DPLL Function Module, [17](#)
- dppllBandwidth_1p2Hz
 - DPLL Function Module, [17](#)
- dppllBandwidth_2mHz
 - DPLL Function Module, [17](#)
- dppllBandwidth_2p5Hz
 - DPLL Function Module, [17](#)
- dppllBandwidth_30mHz
 - DPLL Function Module, [17](#)
- dppllBandwidth_35Hz
 - DPLL Function Module, [17](#)
- dppllBandwidth_4Hz
 - DPLL Function Module, [17](#)
- dppllBandwidth_4mHz
 - DPLL Function Module, [17](#)
- dppllBandwidth_560Hz
 - DPLL Function Module, [17](#)
- dppllBandwidth_60mHz
 - DPLL Function Module, [17](#)
- dppllBandwidth_70Hz
 - DPLL Function Module, [17](#)
- dppllBandwidth_8Hz
 - DPLL Function Module, [17](#)
- dppllBandwidth_8mHz
 - DPLL Function Module, [17](#)
- dppllBandwidth_MAX
 - DPLL Function Module, [17](#)
- dppllBandwidthLimit_MAX
 - DPLL Function Module, [16](#)
- dppllBandwidthLimitAcquire
 - DPLL Function Module, [16](#)
- dppllBandwidthLimitLocked
 - DPLL Function Module, [16](#)
- dppllBandwidthLimitStart
 - DPLL Function Module, [16](#)
- DpllOperMode_MAX

- DPLL Function Module, 18
- DpllProfile_1PPS
 - main.c, 193
- DpllProfile_G813Option1
 - main.c, 193
- DpllProfile_G813Option2
 - main.c, 193
- DpllProfile_G8262Option1
 - main.c, 193
- DpllProfile_G8262Option2
 - main.c, 193
- DpllProfile_GR1244
 - main.c, 193
- DpllProfile_GR253
 - main.c, 193
- DpllProfile_MAX
 - main.c, 193
- DpllProfile_SMC
 - main.c, 193
- dpll_m
 - globalArgs_t, 149
- DpllCnfg.c
 - IDTDpll_bandwidthLimitRegisters_a, 281
- dpllI2CTransData_mp
 - drvHdlr_s, 168
- dpllOverlayRegisters
 - ReadWrite_sim.c, 214
- 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, 193
- 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, 284
- 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, 188
- i2cAddr
 - drvHdlr_s, 168
- i2cAddrTransFunc_m
 - drvHdlr_s, 168
- IDT_82V33930
 - Define.h, 250
- IDT_82V3910
 - Define.h, 250
- IDT_82V3911
 - Define.h, 250
- IDT_8V89316
 - Define.h, 250
- IDT_8V89317
 - Define.h, 250
- IDT_DEVICE_MAX
 - Define.h, 250
- IDT_APLL_API_TRACE
 - ApIi Function Module, 123
- IDT_DRV_MAX_INPUT
 - Define.h, 247
- IDT_DRV_MAX_OUTPUT
 - Define.h, 248
- IDT_HAL_USE_MACRO
 - Hardware Abstraction Layer Module, 56
- IDTAplI_ApIiInput2DpllOutput
 - ApIi Function Module, 129
- IDTAplI_DpllOutput2ApIiOutput
 - ApIi Function Module, 129
- IDTAplI_GetApIiConfig
 - ApIi Function Module, 129
- IDTAplI_GetApIiConfigPerOutput
 - ApIi Function Module, 129
- IDTAplI_GetApIiFreqTableEntry
 - ApIi Function Module, 129
- IDTAplI_GetBypass
 - ApIi Function Module, 130
- IDTAplI_GetConfigListByFreq
 - ApIi Function Module, 130
- IDTAplI_GetConfigListByXtalType
 - ApIi Function Module, 130
- IDTAplI_GetConfigSel
 - ApIi Function Module, 131
- IDTAplI_GetCurrentApIiFreqConfig
 - ApIi Function Module, 131
- IDTAplI_GetFeedbackDiv
 - ApIi Function Module, 131
- IDTAplI_GetFeedbackDivRange
 - ApIi Function Module, 132
- IDTAplI_GetFreqDoublerSel
 - ApIi Function Module, 132
- IDTAplI_GetInputClkSrc
 - ApIi Function Module, 132
- IDTAplI_GetMasterResetSync
 - ApIi Function Module, 133
- IDTAplI_GetNonActiveApIiConfig
 - ApIi 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
 - DpllCnfg.c, [281](#)
- 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
 - 82V3911.c, [177](#)
 - idt82V3911.h, [176](#)
- IDTGeneral_InitDeviceCommon
 - General Function Module, [53](#)
- IDTGeneral_InitDriver
 - 82V3911.c, [177](#)
 - idt82V3911.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, [225](#)
 - outputFreqUtil.c, [239](#)
- IDTHIApi_ClearProvisioning
 - outputFreq.c, [233](#)
 - outputFreq.h, [220](#)
- IDTHIApi_ConfigureInput1PPS
 - inputFreq.c, [231](#)
 - inputFreq.h, [216](#)
- IDTHIApi_ConfigureInputAmi
 - inputFreq.c, [231](#)
 - inputFreq.h, [217](#)
- IDTHIApi_ConfigureInputFrequency
 - inputFreq.c, [231](#)
 - inputFreq.h, [217](#)
- IDTHIApi_ConfigureOutput
 - outputFreq.c, [233](#)
 - outputFreq.h, [221](#)
- IDTHIApi_ConvertFromFreqListToFreqArray
 - outputFreq_priv.h, [225](#)
 - outputFreqUtil.c, [239](#)
- IDTHIApi_DisableAllInputPriorities
 - inputFreq.c, [231](#)
 - inputFreq.h, [217](#)
- IDTHIApi_DisableAllOutputs
 - outputFreq.c, [234](#)
 - outputFreq.h, [222](#)
- IDTHIApi_Frequency2String_c
 - main.c, [193](#)
 - outputFreq_priv.h, [227](#)
 - outputFreqString.c, [238](#)
- IDTHIApi_IsFreqArrayInFreqList
 - outputFreq_priv.h, [225](#)
 - outputFreqUtil.c, [240](#)
- IDTHIApi_IsFrequencyInFreqList
 - outputFreq_priv.h, [225](#)
 - outputFreqUtil.c, [240](#)
- IDTHIApi_PrintAvailFrequencies
 - outputFreq_priv.h, [226](#)
 - outputFreqString.c, [237](#)
- IDTHIApi_PrintConfiguration
 - outputFreq_priv.h, [226](#)
 - outputFreqString.c, [237](#)
- IDTHIApi_PrintFrequencyList
 - outputFreq_priv.h, [226](#)
 - outputFreqString.c, [237](#)
- IDTHIApi_PrintInput
 - outputFreq.h, [222](#)
 - outputFreqString.c, [237](#)
- IDTHIApi_PrintInputHwConfig
 - outputFreqString.c, [237](#)
- IDTHIApi_PrintOption
 - outputFreq_priv.h, [226](#)
 - outputFreqString.c, [237](#)
- IDTHIApi_PrintOutput
 - outputFreq.h, [222](#)
 - outputFreqString.c, [237](#)
- IDTHIApi_PrintOutputHwConfig

- outputFreq_priv.h, [226](#)
- outputFreqString.c, [238](#)
- IDTHIApi_PrintSelectedOutputPath
 - outputFreq_priv.h, [227](#)
 - outputFreqString.c, [238](#)
- IDTHIApi_RetrieveOutputPathMuxFromHardware
 - outputFreq.c, [234](#)
- IDTHIApi_g813Option1
 - profiles.c, [242](#)
 - profiles.h, [228](#)
- IDTHIApi_g813Option2
 - profiles.c, [242](#)
 - profiles.h, [228](#)
- IDTHIApi_g8262EecOption1
 - profiles.c, [242](#)
 - profiles.h, [228](#)
- IDTHIApi_g8262EecOption2
 - profiles.c, [242](#)
 - profiles.h, [229](#)
- IDTHIApi_gr1244Stratum3
 - profiles.c, [242](#)
 - profiles.h, [229](#)
- IDTHIApi_gr253SonetStratum3
 - profiles.c, [242](#)
 - profiles.h, [229](#)
- IDTHIApi_pps
 - profiles.c, [243](#)
 - profiles.h, [229](#)
- IDTHIApi_smc
 - profiles.c, [243](#)
 - profiles.h, [229](#)
- IDTHIApi_wideband
 - profiles.c, [243](#)
 - profiles.h, [230](#)
- 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_GetOutputpInterface
 - 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_SetOutputInterface
 - Output Function Module, [83](#)
- IDTOutput_SetOutputConfig
 - Output Function Module, [83](#)
- IDTOutput_SetOutputEnable
 - Output Function Module, [83](#)
- IDTOutput_SetOutputInvert
 - Output Function Module, [84](#)
- IDTSample_ConfigureSampleConfig1
 - sample.c, [194](#)
 - sample.h, [182](#)
- IDTSample_ConfigureSampleConfig2
 - sample.c, [195](#)
 - sample.h, [183](#)
- IDTSample_ConfigureSampleConfig3
 - sample.c, [195](#)
 - sample.h, [183](#)
- IDTSample_ConfigureSampleConfig4
 - sample.c, [195](#)
 - sample.h, [183](#)
- IDTSample_ConfigureSampleConfig5
 - sample.c, [195](#)
 - sample.h, [184](#)
- IDTSample_ConfigureSampleConfig6
 - sample.c, [195](#)
 - sample.h, [184](#)
- IDTSample_ConfigureSampleConfig7
 - sample.c, [195](#)
 - sample.h, [184](#)
- IDTSample_DumpRegistersRgf
 - main.c, [193](#)
- IDTSample_GetLogVerbosity
 - access.c, [186](#)
 - access.h, [179](#)
- IDTSample_InitDriverI2c
 - access.c, [186](#)
 - access.h, [179](#)
- IDTSample_InitDriverSimulator
 - access.c, [186](#)
 - access.h, [180](#)
- IDTSample_LoadRgfFile
 - loadstore.c, [189](#)
 - loadstore.h, [181](#)
- IDTSample_ResetXtalList
 - access.c, [186](#)
 - access.h, [180](#)
- IDTSample_SaveRgfFile
 - loadstore.c, [189](#)
- loadstore.h, [181](#)
- IDTSample_SetLogVerbosity
 - access.c, [187](#)
 - access.h, [180](#)
- INT_Active_High
 - General Function Module, [48](#)
- INT_Active_Low
 - General Function Module, [49](#)
- idt82V3911.h
 - IDTGeneral_InitDevice, [176](#)
 - IDTGeneral_InitDriver, [176](#)
- idt82V3911Config
 - 82V3911.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, [214](#)
 - ReadWrite_sim.h, [215](#)
- Init_SimWrapper
 - access.c, [187](#)
- 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, 231
 - IDTHIApi_ConfigureInputAmi, 231
 - IDTHIApi_ConfigureInputFrequency, 231
 - IDTHIApi_DisableAllInputPriorities, 231
- inputFreq.h
 - IDTHIApi_ConfigureInput1PPS, 216
 - IDTHIApi_ConfigureInputAmi, 217
 - IDTHIApi_ConfigureInputFrequency, 217
 - IDTHIApi_DisableAllInputPriorities, 217
- inputFreq_m
 - globalArgs_t, 149
- inputFreqValidFunc_m
 - T_idtDrvDeviceConfig, 165
- instance_m
 - pathSelectorConfig_t, 152
- isSim_m
 - globalArgs_t, 149
- loadstore.c
 - IDTSample_LoadRgfFile, 189
 - IDTSample_SaveRgfFile, 189
- loadstore.h
 - IDTSample_LoadRgfFile, 181
 - IDTSample_SaveRgfFile, 181
- lowerThreshReg_m
 - bucketReg_t, 147
- MODE_DpllProfile
 - main.c, 191
- MODE_Input
 - main.c, 191
- MODE_Load
 - main.c, 191
- MODE_None
 - main.c, 191
- MODE_Output
 - main.c, 191
- MODE_Read
 - main.c, 191
- MODE_Sample
 - main.c, 192
- MODE_Save
 - main.c, 192
- MODE_Write
 - main.c, 192
- MPU_I2C_MODE
 - Define.h, 248
- MPU_SERIAL_MODE
 - Define.h, 248
- main
 - main.c, 193
- main.c
 - DpllProfile_1PPS, 193
 - DpllProfile_G813Option1, 193
 - DpllProfile_G813Option2, 193
 - DpllProfile_G8262Option1, 193
 - DpllProfile_G8262Option2, 193
 - DpllProfile_GR1244, 193
 - DpllProfile_GR253, 193
 - DpllProfile_MAX, 193
 - DpllProfile_SMC, 193
- main.c
 - globalArgs, 193
 - IDTHIApi_Frequency2String_c, 193
 - IDTSample_DumpRegistersRgf, 193
 - MODE_DpllProfile, 191
 - MODE_Input, 191
 - MODE_Load, 191
 - MODE_None, 191
 - MODE_Output, 191
 - MODE_Read, 191
 - MODE_Sample, 192
 - MODE_Save, 192
 - MODE_Write, 192
 - main, 193
 - no_argument, 192
 - optional_argument, 192
 - printHelp, 193
 - processCmdLineArguments, 193
 - required_argument, 192
 - SAMPLE_MAX_FILENAME, 192
 - simDebugFlag, 193
 - T_DpllProfile, 193
 - T_DpllProfileFunc, 192
 - T_SampleConfigDescrFunc, 192
 - T_SampleConfigFunc, 192
- 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, 248
- 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, 192

- numberOfApll_m
 - T_idtDrvDeviceConfig, [166](#)
- numberOfApllXtal_m
 - drvHdlr_s, [169](#)
- numberOfSonetGigEthPath_m
 - T_idtDrvDeviceConfig, [166](#)
- numberOfXtal
 - globalArgs_t, [150](#)
- OUTPUTIF_AMI
 - Output Function Module, [80](#)
- OUTPUTIF_AMI_MASK
 - Output.c, [288](#)
- OUTPUTIF_CMOS
 - Output Function Module, [80](#)
- OUTPUTIF_CMOS_MASK
 - Output.c, [288](#)
- OUTPUTIF_LVDS
 - Output Function Module, [80](#)
- OUTPUTIF_LVDS_MASK
 - Output.c, [288](#)
- OUTPUTIF_MAX
 - Output Function Module, [80](#)
- OUTPUTIF_PECL
 - Output Function Module, [80](#)
- OUTPUTIF_PECL_MASK
 - Output.c, [288](#)
- OUTMUX_ENTRY
 - outputFreq.c, [233](#)
- 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, [192](#)
- OutFreq_1
 - outputFreq.h, [219](#)
- OutFreq_10000k
 - outputFreq.h, [220](#)
- OutFreq_12288k
 - outputFreq.h, [219](#)
- OutFreq_12352k
 - outputFreq.h, [219](#)
- OutFreq_124416k
 - outputFreq.h, [220](#)
- OutFreq_125000k
 - outputFreq.h, [220](#)
- OutFreq_125000k_66_64
 - outputFreq.h, [220](#)
- OutFreq_13000k
 - outputFreq.h, [219](#)
- OutFreq_15360k
 - outputFreq.h, [219](#)
- OutFreq_1544k
 - outputFreq.h, [219](#)
- OutFreq_155520k
 - outputFreq.h, [220](#)
- OutFreq_156250k
 - outputFreq.h, [220](#)
- OutFreq_156250k_66_64
 - outputFreq.h, [220](#)
- OutFreq_16384k
 - outputFreq.h, [219](#)
- OutFreq_18528k
 - outputFreq.h, [219](#)
- OutFreq_19440k
 - outputFreq.h, [220](#)
- OutFreq_20000k
 - outputFreq.h, [219](#)
- OutFreq_2048k
 - outputFreq.h, [219](#)
- OutFreq_24576k
 - outputFreq.h, [219](#)
- OutFreq_24704k
 - outputFreq.h, [219](#)
- OutFreq_24883_DOT_2k
 - outputFreq.h, [220](#)
- OutFreq_25000k
 - outputFreq.h, [220](#)
- OutFreq_25000k_66_64
 - outputFreq.h, [220](#)
- OutFreq_25920k
 - outputFreq.h, [220](#)
- OutFreq_26000k
 - outputFreq.h, [219](#)
- OutFreq_2k
 - outputFreq.h, [219](#)
- OutFreq_30720k
 - outputFreq.h, [219](#)
- OutFreq_3088k
 - outputFreq.h, [219](#)
- OutFreq_311040k
 - outputFreq.h, [220](#)
- OutFreq_312500k
 - outputFreq.h, [220](#)
- OutFreq_312500k_66_64
 - outputFreq.h, [220](#)
- OutFreq_32768k
 - outputFreq.h, [219](#)
- OutFreq_34368k
 - outputFreq.h, [219](#)
- OutFreq_37056k
 - outputFreq.h, [219](#)
- OutFreq_3840k
 - outputFreq.h, [219](#)
- OutFreq_38880k
 - outputFreq.h, [220](#)

- OutFreq_400
 - outputFreq.h, [219](#)
- OutFreq_40000k
 - outputFreq.h, [219](#)
- OutFreq_4096k
 - outputFreq.h, [219](#)
- OutFreq_44736k
 - outputFreq.h, [219](#)
- OutFreq_4632k
 - outputFreq.h, [219](#)
- OutFreq_5000k
 - outputFreq.h, [220](#)
- OutFreq_51840k
 - outputFreq.h, [220](#)
- OutFreq_6144k
 - outputFreq.h, [219](#)
- OutFreq_6176k
 - outputFreq.h, [219](#)
- OutFreq_622080k
 - outputFreq.h, [220](#)
- OutFreq_625000k
 - outputFreq.h, [220](#)
- OutFreq_625000k_66_64
 - outputFreq.h, [220](#)
- OutFreq_6480k
 - outputFreq.h, [220](#)
- OutFreq_64k
 - outputFreq.h, [219](#)
- OutFreq_7680k
 - outputFreq.h, [219](#)
- OutFreq_77760k
 - outputFreq.h, [220](#)
- OutFreq_78125k
 - outputFreq.h, [220](#)
- OutFreq_78125k_66_64
 - outputFreq.h, [220](#)
- OutFreq_8192k
 - outputFreq.h, [219](#)
- OutFreq_8k
 - outputFreq.h, [219](#)
- OutFreq_9264k
 - outputFreq.h, [219](#)
- OutFreq_Invalid
 - outputFreq.h, [219](#)
- OutFreq_MAX
 - outputFreq.h, [220](#)
- outFreqLookupIdx_Dpll12E1
 - outputFreqString.c, [236](#)
- outFreqLookupIdx_Dpll16E1
 - outputFreqString.c, [236](#)
- outFreqLookupIdx_Dpll16T1
 - outputFreqString.c, [236](#)
- outFreqLookupIdx_Dpll24T1
 - outputFreqString.c, [236](#)
- outFreqLookupIdx_Dpll77p76
 - outputFreqString.c, [236](#)
- outFreqLookupIdx_DpllE3
 - outputFreqString.c, [236](#)
- outFreqLookupIdx_DpllEth
 - outputFreqString.c, [236](#)
- outFreqLookupIdx_DpllGps
 - outputFreqString.c, [236](#)
- outFreqLookupIdx_DpllGsm
 - outputFreqString.c, [236](#)
- outFreqLookupIdx_DpllObsai
 - outputFreqString.c, [236](#)
- outFreqLookupIdx_DpllT3
 - outputFreqString.c, [236](#)
- outFreqLookupIdx_MAX
 - outputFreqString.c, [236](#)
- outFreqLookupIdx_SonetGigEthEth
 - outputFreqString.c, [236](#)
- outFreqLookupIdx_SonetGigEthFec
 - outputFreqString.c, [236](#)
- outFreqLookupIdx_SonetGigEthSonet
 - outputFreqString.c, [236](#)
- 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_GetOutputpulInterface, [81](#)
 - IDTOutput_GetOutputConfig, [82](#)
 - IDTOutput_GetOutputEnable, [82](#)
 - IDTOutput_GetOutputInvert, [82](#)
 - IDTOutput_SetFrameSync, [82](#)
 - IDTOutput_SetFrameSyncPulseEdge, [83](#)
 - IDTOutput_SetOutputpulInterface, [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, [288](#)
 - OUTPUTIF_CMOS_MASK, [288](#)
 - OUTPUTIF_LVDS_MASK, [288](#)
 - OUTPUTIF_PECL_MASK, [288](#)
 - OutputPathBfVal_MAX, [289](#)
 - OutputPathBfVal_NULL, [289](#)
 - OutputPathBfVal_T0Dpll_12E1_GPS_E3_T3, [289](#)
 - OutputPathBfVal_T0Dpll_16E1_16T1, [289](#)
 - OutputPathBfVal_T0Dpll_7776kHz, [289](#)
 - OutputPathBfVal_T0Dpll_Eth, [289](#)
 - OutputPathBfVal_T0Dpll_GSM_OBSAI_16E1_16-T1, [289](#)
 - OutputPathBfVal_T0SonetGigEth, [289](#)
 - OutputPathBfVal_T4Dpll_12E1_24T1_E3_T3, [289](#)
 - OutputPathBfVal_T4Dpll_16E1_16T1, [289](#)
 - OutputPathBfVal_T4Dpll_7776kHz, [289](#)
 - OutputPathBfVal_T4Dpll_Eth, [289](#)
 - OutputPathBfVal_T4Dpll_GSM_GPS_16E1_16T1, [289](#)
 - OutputPathBfVal_T4SonetGigEth, [289](#)
- Output.c
 - T_idtOutputPathBfVal, [288](#)
- outputFreq.h
 - OutFreq_1, [219](#)
 - OutFreq_10000k, [220](#)
 - OutFreq_12288k, [219](#)
 - OutFreq_12352k, [219](#)
 - OutFreq_124416k, [220](#)
 - OutFreq_125000k, [220](#)
 - OutFreq_125000k_66_64, [220](#)
 - OutFreq_13000k, [219](#)
 - OutFreq_15360k, [219](#)
 - OutFreq_1544k, [219](#)
 - OutFreq_155520k, [220](#)
 - OutFreq_156250k, [220](#)
 - OutFreq_156250k_66_64, [220](#)
 - OutFreq_16384k, [219](#)
 - OutFreq_18528k, [219](#)
 - OutFreq_19440k, [220](#)
 - OutFreq_20000k, [219](#)
 - OutFreq_2048k, [219](#)
 - OutFreq_24576k, [219](#)
 - OutFreq_24704k, [219](#)
 - OutFreq_24883_DOT_2k, [220](#)
 - OutFreq_25000k, [220](#)
 - OutFreq_25000k_66_64, [220](#)
 - OutFreq_25920k, [220](#)
 - OutFreq_26000k, [219](#)
 - OutFreq_2k, [219](#)
 - OutFreq_30720k, [219](#)
 - OutFreq_3088k, [219](#)
 - OutFreq_311040k, [220](#)
 - OutFreq_312500k, [220](#)
 - OutFreq_312500k_66_64, [220](#)
 - OutFreq_32768k, [219](#)
 - OutFreq_34368k, [219](#)
 - OutFreq_37056k, [219](#)
 - OutFreq_3840k, [219](#)
 - OutFreq_38880k, [220](#)
 - OutFreq_400, [219](#)
 - OutFreq_40000k, [219](#)
 - OutFreq_4096k, [219](#)
 - OutFreq_44736k, [219](#)
 - OutFreq_4632k, [219](#)
 - OutFreq_5000k, [220](#)
 - OutFreq_51840k, [220](#)
 - OutFreq_6144k, [219](#)
 - OutFreq_6176k, [219](#)
 - OutFreq_622080k, [220](#)
 - OutFreq_625000k, [220](#)
 - OutFreq_625000k_66_64, [220](#)
 - OutFreq_6480k, [220](#)
 - OutFreq_64k, [219](#)
 - OutFreq_7680k, [219](#)
 - OutFreq_77760k, [220](#)
 - OutFreq_78125k, [220](#)
 - OutFreq_78125k_66_64, [220](#)
 - OutFreq_8192k, [219](#)
 - OutFreq_8k, [219](#)
 - OutFreq_9264k, [219](#)
 - OutFreq_Invalid, [219](#)
 - OutFreq_MAX, [220](#)
 - sonetGigMode_AllFromDpll1, [220](#)
 - sonetGigMode_AllFromDpll2, [220](#)
 - sonetGigMode_MAX, [220](#)
 - sonetGigMode_Matching, [220](#)
- outputFreq_priv.h
 - Selector_Any, [225](#)
 - Selector_MAX, [225](#)
 - Selector_Network, [225](#)
 - Selector_SonetGigEth, [225](#)
 - Selector_T0DpllSelA, [225](#)
 - Selector_T0DpllSelB, [225](#)
 - Selector_T4DpllSelA, [225](#)
 - Selector_T4DpllSelB, [225](#)
- outputFreqString.c
 - outFreqLookupIdx_Dpll12E1, [236](#)
 - outFreqLookupIdx_Dpll16E1, [236](#)
 - outFreqLookupIdx_Dpll16T1, [236](#)
 - outFreqLookupIdx_Dpll24T1, [236](#)
 - outFreqLookupIdx_Dpll77p76, [236](#)
 - outFreqLookupIdx_DpllE3, [236](#)
 - outFreqLookupIdx_DpllEth, [236](#)
 - outFreqLookupIdx_DpllGps, [236](#)
 - outFreqLookupIdx_DpllGsm, [236](#)
 - outFreqLookupIdx_DpllObsai, [236](#)
 - outFreqLookupIdx_DpllT3, [236](#)
 - outFreqLookupIdx_MAX, [236](#)
 - outFreqLookupIdx_SonetGigEthEth, [236](#)
 - outFreqLookupIdx_SonetGigEthFec, [236](#)
 - outFreqLookupIdx_SonetGigEthSonet, [236](#)
- 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, [289](#)
- OutputPathBfVal_NULL
 - Output.c, [289](#)
- OutputPathBfVal_T0Dpll_12E1_GPS_E3_T3
 - Output.c, [289](#)
- OutputPathBfVal_T0Dpll_16E1_16T1
 - Output.c, [289](#)
- OutputPathBfVal_T0Dpll_7776kHz
 - Output.c, [289](#)
- OutputPathBfVal_T0Dpll_Eth
 - Output.c, [289](#)
- OutputPathBfVal_T0Dpll_GSM_OBSAI_16E1_16T1
 - Output.c, [289](#)
- OutputPathBfVal_T0SonetGigEth
 - Output.c, [289](#)
- OutputPathBfVal_T4Dpll_12E1_24T1_E3_T3
 - Output.c, [289](#)
- OutputPathBfVal_T4Dpll_16E1_16T1
 - Output.c, [289](#)
- OutputPathBfVal_T4Dpll_7776kHz
 - Output.c, [289](#)
- OutputPathBfVal_T4Dpll_Eth
 - Output.c, [289](#)
- OutputPathBfVal_T4Dpll_GSM_GPS_16E1_16T1
 - Output.c, [289](#)
- OutputPathBfVal_T4SonetGigEth
 - Output.c, [289](#)
- outputConfig
 - T_idtApIConfig, [158](#)
- outputDiv
 - T_idtApIConfig, [158](#)
 - T_idtApIConfigPerOutput, [160](#)
- outputDivConfig0
 - T_idtApIConfig::outputDiv_s::qaDiv_s, [155](#)
 - T_idtApIConfig::outputDiv_s::qbDiv_s, [156](#)
- outputDivConfig1
 - T_idtApIConfig::outputDiv_s::qaDiv_s, [155](#)
 - T_idtApIConfig::outputDiv_s::qbDiv_s, [156](#)
- outputDivider_m
 - pathSelectorConfig_t, [152](#)
- outputExt2IntXlate_ma
 - T_idtDrvDeviceConfig, [166](#)
- outputFreq.c
 - IDTHIApi_ClearProvisioning, [233](#)
 - IDTHIApi_ConfigureOutput, [233](#)
 - IDTHIApi_DisableAllOutputs, [234](#)
 - IDTHIApi_RetrieveOutputPathMuxFromHardware, [234](#)
 - OUTMUX_ENTRY, [233](#)
- outputFreq.h
 - IDTHIApi_ClearProvisioning, [220](#)
 - IDTHIApi_ConfigureOutput, [221](#)
 - IDTHIApi_DisableAllOutputs, [222](#)
 - IDTHIApi_PrintInput, [222](#)
 - IDTHIApi_PrintOutput, [222](#)
 - outputFrequency_t, [219](#)
 - T_sonetGigeMode, [220](#)
- outputFreq_m
 - T_idtSonetGigEthBitfield2PathConfig, [173](#)
- outputFreq_priv.h
 - CHECK_BIT, [224](#)
 - IDTHIApi_ApplyFrequencies, [225](#)
 - IDTHIApi_ConvertFromFreqListToFreqArray, [225](#)
 - IDTHIApi_Frequency2String_c, [227](#)
 - IDTHIApi_IsFreqArrayInFreqList, [225](#)
 - IDTHIApi_IsFrequencyInFreqList, [225](#)
 - IDTHIApi_PrintAvailFrequencies, [226](#)
 - IDTHIApi_PrintConfiguration, [226](#)
 - IDTHIApi_PrintFrequencyList, [226](#)
 - IDTHIApi_PrintOption, [226](#)
 - IDTHIApi_PrintOutputHwConfig, [226](#)
 - IDTHIApi_PrintSelectedOutputPath, [227](#)
 - pathSelectors_t, [225](#)
 - SET_BIT, [224](#)
 - validFreq_t, [224](#)
- outputFreqString.c
 - IDTHIApi_Frequency2String_c, [238](#)
 - IDTHIApi_PrintAvailFrequencies, [237](#)
 - IDTHIApi_PrintConfiguration, [237](#)
 - IDTHIApi_PrintFrequencyList, [237](#)
 - IDTHIApi_PrintInput, [237](#)
 - IDTHIApi_PrintInputHwConfig, [237](#)
 - IDTHIApi_PrintOption, [237](#)
 - IDTHIApi_PrintOutput, [237](#)
 - IDTHIApi_PrintOutputHwConfig, [238](#)
 - IDTHIApi_PrintSelectedOutputPath, [238](#)
 - T_outFreqLookupIdx, [236](#)
- outputFreqUtil.c
 - IDTHIApi_ApplyFrequencies, [239](#)
 - IDTHIApi_ConvertFromFreqListToFreqArray, [239](#)
 - IDTHIApi_IsFreqArrayInFreqList, [240](#)
 - IDTHIApi_IsFrequencyInFreqList, [240](#)
- outputFreqValidFunc_m
 - T_idtDrvDeviceConfig, [166](#)
- outputFrequency_t
 - outputFreq.h, [219](#)
- outputPathSelector_m

- pathSelectorConfig_t, 152
- outputSigType
 - T_idtApplConfig::outputConfig_s::qaConfig_s, 154
 - T_idtApplConfig::outputConfig_s::qbConfig_s, 156
- page1Registers
 - ReadWrite_sim.c, 214
- 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, 225
- 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
- populatedAppl_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_idtApplConfig, 158
 - T_idtApplConfigPerOutput, 160
- preDivConfig0
 - T_idtApplConfig::preDiv_s, 154
- preDivConfig1
 - T_idtApplConfig::preDiv_s, 154
- printHelp
 - main.c, 193
- processCmdLineArguments
 - main.c, 193
- profiles.c
 - IDTHIApi_g813Option1, 242
 - IDTHIApi_g813Option2, 242
 - IDTHIApi_g8262EecOption1, 242
 - IDTHIApi_g8262EecOption2, 242
 - IDTHIApi_gr1244Stratum3, 242
 - IDTHIApi_gr253SonetStratum3, 242
 - IDTHIApi_pps, 243
 - IDTHIApi_smc, 243
 - IDTHIApi_wideband, 243
- profiles.h
 - IDTHIApi_g813Option1, 228
 - IDTHIApi_g813Option2, 228
 - IDTHIApi_g8262EecOption1, 228
 - IDTHIApi_g8262EecOption2, 229
 - IDTHIApi_gr1244Stratum3, 229
 - IDTHIApi_gr253SonetStratum3, 229
 - IDTHIApi_pps, 229
 - IDTHIApi_smc, 229
 - IDTHIApi_wideband, 230
- qaConfig
 - T_idtApplConfig::outputConfig_s, 151
- qaDiv
 - T_idtApplConfig::outputDiv_s, 152

- qbConfig
 - T_idtApplConfig::outputConfig_s, [151](#)
- qbDiv
 - T_idtApplConfig::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_ODBSELA0
 - Register definitions, [115](#)
- REG_ODBSELA1
 - 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](#)
- RDByte_Sim
 - ReadWrite_sim.c, [214](#)
 - ReadWrite_sim.h, [215](#)
- ReadByte_SimWrapper
 - access.c, [187](#)
- readFunc_m
 - T_idtDrvAccess, [164](#)
- ReadWrite_sim.c
 - DRV_MEMMAP_SIZE, [213](#)
 - Delnit_Sim, [214](#)
 - dpIIOverlayRegisters, [214](#)
 - Init_Sim, [214](#)
 - page1Registers, [214](#)
 - RDByte_Sim, [214](#)
 - simDebugFlag, [214](#)
 - simulatedMemMap, [215](#)
 - WRByte_Sim, [214](#)
- ReadWrite_sim.h
 - Delnit_Sim, [215](#)
 - Init_Sim, [215](#)
 - RDByte_Sim, [215](#)
 - WRByte_Sim, [215](#)
- 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_LIMT_A_LSB, 110
- BF_DPLL_FREQ_HARD_LIMT_A_MASK, 110
- BF_DPLL_FREQ_HARD_LIMT_A_SIZE, 110
- BF_DPLL_FREQ_HARD_LIMT_B_LSB, 110
- BF_DPLL_FREQ_HARD_LIMT_B_MASK, 110
- BF_DPLL_FREQ_HARD_LIMT_B_SIZE, 110
- BF_DPLL_FREQ_SOFT_LIMT_LSB, 110
- BF_DPLL_FREQ_SOFT_LIMT_MASK, 110
- BF_DPLL_FREQ_SOFT_LIMT_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_LIMT_EN_LSB, 109
- BF_FINE_PH_LOS_LIMT_EN_MASK, 109
- BF_FINE_PH_LOS_LIMT_EN_SIZE, 109
- BF_FREQ_LIMT_PH_LOS_LSB, 110
- BF_FREQ_LIMT_PH_LOS_MASK, 110
- BF_FREQ_LIMT_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_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_ODBSLA0_LSB, 116
 BF_ODBSLA0_MASK, 116
 BF_ODBSLA0_SIZE, 116
 BF_ODBSLA1_LSB, 116
 BF_ODBSLA1_MASK, 116
 BF_ODBSLA1_SIZE, 116
 BF_ODBSLB0_LSB, 116
 BF_ODBSLB0_MASK, 116
 BF_ODBSLB0_SIZE, 116
 BF_ODBSLB1_LSB, 116
 BF_ODBSLB1_MASK, 116
 BF_ODBSLB1_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_PECCL_LVDS_LSB, 98
- BF_OUT6_PECCL_LVDS_MASK, 98
- BF_OUT6_PECCL_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_PECCL_LVDS_LSB, 98
- BF_OUT7_PECCL_LVDS_MASK, 98
- BF_OUT7_PECCL_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_LIMT_LSB, 109
- BF_PH_LOS_COARSE_LIMT_MASK, 109
- BF_PH_LOS_COARSE_LIMT_SIZE, 109
- BF_PH_LOS_FINE_LIMT_LSB, 109
- BF_PH_LOS_FINE_LIMT_MASK, 109
- BF_PH_LOS_FINE_LIMT_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_HOLD OVER_FREQ_HIGH_CNFG, 96
- REG_HOLD OVER_FREQ_LOW_CNFG, 96
- REG_HOLD OVER_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_VC XO_SEL, 115
- REGMAP_APLL_MAX, 117
- REGMAP_MAX, 115
- required_argument
 - main.c, 192
- 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, 192
- SET_BIT

- outputFreq_priv.h, [224](#)
- SIM_REG_MAP_SIZE
 - ApIIRedWrite_sim.c, [196](#)
- sample.c
 - IDTSample_ConfigureSampleConfig1, [194](#)
 - IDTSample_ConfigureSampleConfig2, [195](#)
 - IDTSample_ConfigureSampleConfig3, [195](#)
 - IDTSample_ConfigureSampleConfig4, [195](#)
 - IDTSample_ConfigureSampleConfig5, [195](#)
 - IDTSample_ConfigureSampleConfig6, [195](#)
 - IDTSample_ConfigureSampleConfig7, [195](#)
- sample.h
 - IDTSample_ConfigureSampleConfig1, [182](#)
 - IDTSample_ConfigureSampleConfig2, [183](#)
 - IDTSample_ConfigureSampleConfig3, [183](#)
 - IDTSample_ConfigureSampleConfig4, [183](#)
 - IDTSample_ConfigureSampleConfig5, [184](#)
 - IDTSample_ConfigureSampleConfig6, [184](#)
 - IDTSample_ConfigureSampleConfig7, [184](#)
- sampleConfig_m
 - globalArgs_t, [150](#)
- Selector_Any
 - outputFreq_priv.h, [225](#)
- Selector_MAX
 - outputFreq_priv.h, [225](#)
- Selector_Network
 - outputFreq_priv.h, [225](#)
- Selector_SonetGigEth
 - outputFreq_priv.h, [225](#)
- Selector_T0DpllSelA
 - outputFreq_priv.h, [225](#)
- Selector_T0DpllSelB
 - outputFreq_priv.h, [225](#)
- Selector_T4DpllSelA
 - outputFreq_priv.h, [225](#)
- Selector_T4DpllSelB
 - outputFreq_priv.h, [225](#)
- selector_m
 - pathSelectorConfig_t, [152](#)
- shutOff
 - T_idtApIIRedWrite_sim.c, [158](#)
- sigType
 - T_idtApIIRedWrite_sim.c, [160](#)
- SimAccessMethods
 - access.c, [188](#)
- simDebugFlag
 - main.c, [193](#)
 - ReadWrite_sim.c, [214](#)
- simulatedMemMap
 - ReadWrite_sim.c, [215](#)
- simulatedRegMapApIIRedWrite_sim.c, [197](#)
- simulatedRegMapApIIRedWrite_sim.c, [197](#)
- size_m
 - pathSelectorIndex_t, [153](#)
- SonetGigEthOutput_10GbE
 - DPLL Function Module, [21](#)
- SonetGigEthOutput_MAX
 - DPLL Function Module, [21](#)
- SonetGigEthOutput_Sonet
 - DPLL Function Module, [21](#)
- SonetGigEthSel_MAX
 - DPLL Private header file, [8](#)
- SonetGigEthSel_T0Dpll10GbE
 - DPLL Private header file, [8](#)
- SonetGigEthSel_T0Dpll10GbE_6664
 - DPLL Private header file, [8](#)
- SonetGigEthSel_T0DpllSonet
 - DPLL Private header file, [8](#)
- SonetGigEthSel_T4Dpll10GbE
 - DPLL Private header file, [8](#)
- SonetGigEthSel_T4Dpll10GbE_6664
 - DPLL Private header file, [8](#)
- SonetGigEthSel_T4DpllSonet
 - DPLL Private header file, [8](#)
- sonetGigeMode_AllFromDpll1
 - outputFreq.h, [220](#)
- sonetGigeMode_AllFromDpll2
 - outputFreq.h, [220](#)
- sonetGigeMode_MAX
 - outputFreq.h, [220](#)
- sonetGigeMode_Matching
 - outputFreq.h, [220](#)
- 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](#)
- T0DpllSelA_16E1
 - DPLL Private header file, [8](#)
- T0DpllSelA_16T1
 - DPLL Private header file, [8](#)
- T0DpllSelA_GSM
 - DPLL Private header file, [8](#)
- T0DpllSelA_MAX
 - DPLL Private header file, [8](#)
- T0DpllSelA_OBSAI
 - DPLL Private header file, [8](#)
- T0DpllSelB_12E1
 - DPLL Private header file, [9](#)
- T0DpllSelB_E3
 - DPLL Private header file, [9](#)
- T0DpllSelB_GPS
 - DPLL Private header file, [9](#)
- T0DpllSelB_MAX
 - DPLL Private header file, [9](#)
- T0DpllSelB_T3
 - DPLL Private header file, [9](#)

- t0PllMode_m
 - T_idtDrvDeviceConfig, [167](#)
- T0SelA_m
 - T_IDTPathConfiguration, [172](#)
- T0SelB_m
 - T_IDTPathConfiguration, [172](#)
- T4DpllSelA_16E1
 - DPLL Private header file, [9](#)
- T4DpllSelA_16T1
 - DPLL Private header file, [9](#)
- T4DpllSelA_GPS
 - DPLL Private header file, [9](#)
- T4DpllSelA_GSM
 - DPLL Private header file, [9](#)
- T4DpllSelA_MAX
 - DPLL Private header file, [9](#)
- T4DpllSelB_12E1
 - DPLL Private header file, [9](#)
- T4DpllSelB_24T1
 - DPLL Private header file, [9](#)
- T4DpllSelB_E3
 - DPLL Private header file, [9](#)
- T4DpllSelB_MAX
 - DPLL Private header file, [9](#)
- T4DpllSelB_T3
 - DPLL Private header file, [9](#)
- T4SelA_m
 - T_IDTPathConfiguration, [173](#)
- T4SelB_m
 - T_IDTPathConfiguration, [173](#)
- T_DpllProfile
 - main.c, [193](#)
- T_DpllProfileFunc
 - main.c, [192](#)
- 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, [192](#)
- T_SampleConfigEntry, [174](#)
 - configDescription_m, [174](#)
 - configFunction_m, [174](#)
- T_SampleConfigFunc
 - main.c, [192](#)
- 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, [248](#)
- T_idtAccessInitFunc
 - Define.h, [248](#)
- 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](#)
 - applFbDivider, [161](#)
 - applInputFreq, [161](#)
 - applOutputDivider, [161](#)
 - applOutputFreq, [161](#)
 - applOutputs, [161](#)
 - applPreDivider, [161](#)
 - applXtalFreq, [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_idtApplOutputSupportedType
 - Appl Function Module, [127](#)
- T_idtApplQaDiv
 - Appl Function Module, [127](#)
- T_idtApplQbDiv
 - Appl Function Module, [127](#)
- T_idtApplRegAddr
 - Appl Function Module, [123](#)
- T_idtApplRegMask
 - Appl Function Module, [123](#)
- T_idtApplRegVal
 - Appl Function Module, [124](#)
- T_idtApplShutoff
 - Appl Function Module, [128](#)
- T_idtApplXtal, [161](#)
 - applXtalFreq, [162](#)
 - applXtalId, [162](#)
- T_idtApplXtalId
 - Appl Function Module, [128](#)
- T_idtApplXtalList, [162](#)
 - xtal1Appl1, [163](#)
 - xtal1Appl1Freq, [163](#)
 - xtal2Appl2, [163](#)
 - xtal2Appl2Freq, [163](#)
 - xtal3Appl1, [163](#)
 - xtal3Appl1Freq, [163](#)
 - xtal4Appl2, [163](#)
 - xtal4Appl2Freq, [163](#)
- T_idtApplXtalType
 - Appl 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, [250](#)
- T_idtDpllBandwidth
 - DPLL Function Module, [17](#)
- T_idtDpllHoldover
 - DPLL Function Module, [17](#)
- T_idtDpllInstance
 - Define.h, [250](#)
- 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, [251](#)
- 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](#)
 - maxNumXtalPerAppl_m, [166](#)
 - numberOfAppl_m, [166](#)
 - numberOfSonetGigEthPath_m, [166](#)
 - outPutApplConfig, [166](#)
 - outSyntXlate_ma, [166](#)
 - outputExt2IntXlate_ma, [166](#)
 - outputFreqValidFunc_m, [166](#)
 - phaseSlopeLimiting_ma, [166](#)
 - pllExt2IntXlate_ma, [166](#)
 - pllInt2ExtXlate_ma, [167](#)
 - populatedAppl_ma, [167](#)
 - supportedXtalFreq_ma, [167](#)
 - supportedXtalId_ma, [167](#)

- t0PllMode_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, 249
- T_idtI2CtransData, 170
 - bitLocation, 171
 - bitValue, 171
- T_idtInputDividerMux
 - Input Function Module, 65
- T_idtInputFreqValid
 - Define.h, 249
- 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, 249
- T_idtOutputApplConfig, 171
 - apllInput, 171
 - apllInst, 171
 - apllOutput, 171
 - extOutput, 172
- T_idtOutputPathBfVal
 - Output.c, 288
- T_idtPhaseSlope
 - DPLL Function Module, 20
- T_idtReadFunc
 - Define.h, 250
- 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, 250
- T_outFreqLookupIdx
 - outputFreqString.c, 236
- T_outputInterface
 - Output Function Module, 80
- T_outputPathType
 - Output Function Module, 80
- T_sonetGigeMode
 - outputFreq.h, 220
- 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, 224
- value_m
 - pathSelectorConfig_t, 153
- WRByte_Sim
 - ReadWrite_sim.c, 214
 - ReadWrite_sim.h, 215
- WriteByte_SimWrapper
 - access.c, 187
- writeFunc_m
 - T_idtDrvAccess, 164
- xtal
 - T_idtApplConfig, 158
 - T_idtApplConfigPerOutput, 160
- xtal1Appl1
 - T_idtApplXtalList, 163
- xtal1Appl1Freq
 - T_idtApplXtalList, 163
- xtal2Appl2
 - T_idtApplXtalList, 163
- xtal2Appl2Freq
 - T_idtApplXtalList, 163
- xtal3Appl1
 - T_idtApplXtalList, 163
- xtal3Appl1Freq
 - T_idtApplXtalList, 163
- xtal4Appl2
 - T_idtApplXtalList, 163
- xtal4Appl2Freq
 - T_idtApplXtalList, 163
- xtalList

drvHdlr_s, [169](#)
globalArgs_t, [150](#)