82V3910 WANPLL – Application Programmer's Interface Reference Manual

Version 1.1

September 23, 2013

Version	Date	Description
1.0	May 23, 2013	Initial release.
1.1	September 23, 2013	- Add support for APLL pre-divider value 1. - Update Ethernet FEC frequency configuration.

Contents

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

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[DpII_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 { dpllBandwidthLimitStart, dpllBandwidthLimitAcquire, dpllBandwidthLimit-Locked, 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_PreLocked2 = 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

hdlr	Device driver handle
ext	External DPLL instance number (0..N)
internal	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_0pt5sec 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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	DPLL instance
stage	Bandwidth stage
nBandWidth_p	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
nDamping_p	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	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_osUint16 IDTDpll_GetDpllHardAlarmLimit (T_idtDrvHdlr *hdlr,* T_idtDpllInstance *nDpll*)

Get the limitation of Hard alarm of DPLL.

Parameters

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	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

hdlr	Device driver handler
nDpll	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

hdlr	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

hdlr	Device driver handler
accepting_p	Accepting threshold (ppm) is a multiple of frequency monitoring factor value. Accepting Threshold = (AcceptingValue + 1)∗(FreqMonFactor)
rejecting_p	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

hdlr	Device driver handler
nEnable_p	• 0= Disable the hitless switching feature; • 1= Enable the hitless switching feature;
nFrozen_p	• 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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	DPLL instance
nMode_p	• Holdover_Instantaneous = automatic instantaneous mode • Holdover_Slow_Average = automatic slow average • Holdover_Fast_Average = Automatic fast average • Holdover_Manual = Manual
nReadMode_p	• 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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	DPLL instance

nEnable_p	• 0= Disable the coarse phase detector • 1= Enable the coarse phase detector
nWide_p	• 0= Wide range disable • 1= Wide range enable
nMultiPhase_p	• 0= Limit to +/- 1UI • 1= Limit to PH_LOS_COARSE_LIMT register
nMultiPh8kEn_p	• 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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	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

hdlr	Device driver handler
nDpll	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

hdlr	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

hdlr	Device driver handler
nEnableFreq_p	Enable (1) / Disable (0) frequency locking
nEnablePhase_p	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

hdlr	Device driver handler
nDpll	DPLL instance
nPhase_p	Phase position

See Also

T_idtDpllPpsPhasePostion

Parameters

nPulse_p	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

hdlr	Device driver handler
accepting_p	Accepting threshold (ppm) is a multiple of frequency monitoring factor value. Accepting Threshold = (AcceptingValue + 1)∗(FreqMonFactor)
rejecting_p	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

hdlr	Device driver handler
instance	Sonet/GigE path instance 0..1
inputDpll_p	Provisioned DPLL instance of Sonet/GigE path
outputFreq_p	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	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

hdlr	Device driver handler
nDpll	DPLL instance
nEnable	• 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

hdlr	Device driver handler
nDpll	DPLL instance
mode	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

hdlr	Device driver handler
nDpll	DPLL instance
stage	Bandwidth stage
nBandWidth	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
nDamping	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

hdlr	Device driver handler
nDpll	DPLL instance
nDisable	• 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

hdlr	Device driver handler
nDpll	DPLL instance
nDisable	• 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

hdlr	Device driver handler
nDpll	DPLL instance
nEnable	• 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

hdlr	Device driver handler

nEnable	• 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

hdlr	Device driver handler
nDpll	DPLL instance
nLimit	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

hdlr	Device driver handler
nDpll	DPLL instance
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 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

hdlr	Device driver handler
nDpll	DPLL instance
path	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

hdlr	Device driver handler
nDpll	DPLL instance
path	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

hdlr	Device driver handler
nDpll	DPLL instance
nDisable	• 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

hdlr	Device driver handler
nDpll	DPLL instance
nDisable	• 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

hdlr	Device driver handler
nDpll	DPLL instance
nLimit	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

hdlr	Device driver handler
nEnable	• 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

hdlr	Device driver handler
nDpll	DPLL instance
nInputNo	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

hdlr	Device driver handler
freqMonFactor	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

hdlr	Device driver handler
accepting	Accepting threshold (ppm) is a multiple of frequency monitoring factor value. Accepting Threshold = (AcceptingValue + 1)∗(FreqMonFactor)
rejecting	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

hdlr	Device driver handler
nEnable	• 0= Disable the hitless switching feature; • 1= Enable the hitless switching feature;
nFrozen	• 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

hdlr	Device driver handler
nDpll	DPLL instance
pptOffset	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

hdlr	Device driver handler
nDpll	DPLL instance
nMode	• Holdover_Instantaneous = automatic instantaneous mode • Holdover_Slow_Average = automatic slow average • Holdover_Fast_Average = Automatic fast average • Holdover_Manual = Manual
nReadMode	• 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 *nIcp*)

Set charge pump control.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance
nIcp	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

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

nMultiPh8kEn	• 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

hdlr	Device driver handler
nDpll	DPLL instance
nLimit	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

hdlr	Device driver handler
nDpll	DPLL instance
nEnable	• 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

hdlr	Device driver handler
nDpll	DPLL instance
nLimit	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

hdlr	Device driver handler
nOffset	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

hdlr	Device driver handler
nDpll	DPLL instance
nPhaseSlope	• 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

hdlr	Device driver handler
nEnable	• 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

hdlr	Device driver handler
nEnableFreq	Enable (1) / Disable (0) frequency locking
nEnablePhase	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

hdlr	Device driver handler
nDpll	DPLL instance
nPhase	Phase position

See Also

T_idtDpllPpsPhasePostion

Parameters

nPulse	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

hdlr	Device driver handler
accepting	Accepting threshold (ppm) is a multiple of frequency monitoring factor value. Accepting Threshold = (AcceptingValue + 1)∗(FreqMonFactor)

rejecting	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

hdlr	Device driver handler
instance	Sonet/GigE path instance 0..1
inputDpll	Dpll instance
outputFreq	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

hdlr	Device driver handler
nDpll	DPLL instance
nMode	• 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 }

  Enumerated type listing types of selectable network type.
- enum phaseTimeoutMultiplier_t {
  phaseTimeoutMultiplier_2 = 0, phaseTimeoutMultiplier_4 = 1, phaseTimeoutMultiplier_8 = 2, phaseTimeoutMultiplier_16 = 3,
  phaseTimeoutMultiplier_MAX }

  Enumerated type listing phase alarm timeout multipliers.

Functions

- T_osUint16 IDTGeneral_GetDeviceID (T_idtDrvHdlr hdlr)

  Get ID number of the device.
- void IDTGeneral_SetOscFreqOffset (T_idtDrvHdlr hdlr, long pptOffset)

  Set compensation to the oscillator offset.
- long IDTGeneral_GetOscFreqOffset (T_idtDrvHdlr hdlr)

  Get compensated oscillator offset.
- void IDTGeneral_SetNetworkType (T_idtDrvHdlr hdlr, networkType_t nType)

  Set the network type, SDH or SONET.
- networkType_t IDTGeneral_GetNetworkType (T_idtDrvHdlr hdlr)

  Get the network type, SDH or SONET.
- void IDTGeneral_SetRevertiveMode (T_idtDrvHdlr hdlr, T_osUint8 nEnable)

  Set the device running in Revertive mode or not.
- T_osUint8 IDTGeneral_GetRevertiveMode (T_idtDrvHdlr hdlr)

  Get the device running in Revertive mode or not.

- void IDTGeneral_SetTimeoutValue (T_idtDrvHdlr hdlr, phaseTimeoutMultiplier_t nMultiFactor, T_osUint8 nTimeout)

  Set the value of time out of phase alarm Timeout(second)= nMultiFactor ∗ nTimeout.
- void IDTGeneral_GetTimeoutValue (T_idtDrvHdlr hdlr, phaseTimeoutMultiplier_t ∗nMultiFactor_p, T_osUint8 ∗nTimeout_p)

  Get the value of time out of phase alarm Timeout(second)= nMultiFactor ∗ nTimeout.
- void IDTGeneral_SetOscActiveEdge (T_idtDrvHdlr hdlr, T_osUint8 nEdge)

  Set the activate edge of main oscillator.
- T_osUint8 IDTGeneral_GetOscActiveEdge (T_idtDrvHdlr hdlr)

  Get the activate edge of main oscillator.
- void IDTGeneral_SetProtectMode (T_idtDrvHdlr hdlr)

  Enable protected register access mode.
- void IDTGeneral_SetSingleUnprotectMode (T_idtDrvHdlr hdlr)

  Set single access register access mode.
- void IDTGeneral_SetFullyUnprotectMode (T_idtDrvHdlr hdlr)

  Set fully un-protected register access mode.
- void IDTGeneral_SetFlagOnTDO (T_idtDrvHdlr hdlr, T_osUint8 nEnable)

  Set TDO pin to indicate the fail of selected clock.
- T_osUint8 IDTGeneral_GetFlagOnTDO (T_idtDrvHdlr hdlr)

  Get TDO pin to indicate the fail of selected clock.
- void IDTGeneral_SetUltraFastSwitch (T_idtDrvHdlr hdlr, T_osUint8 nEnable)

  Set Ultra-fast-switch enable or not.
- T_osUint8 IDTGeneral_GetUltraFastSwitch (T_idtDrvHdlr hdlr)

  Get Ultra-fast-switch enable or not.
- void IDTGeneral_SetHardAlarm (T_idtDrvHdlr hdlr, T_osUint8 nEnable)

  Set Hard-alarm enable when the input exceeds the threshold.
- T_osUint8 IDTGeneral_GetHardAlarm (T_idtDrvHdlr hdlr)

  Get Hard-alarm enable when the input exceeds the threshold.
- T_osUint8 IDTGeneral_i2cTranslate (T_osUint8 i2cAddr, void ∗transData)

  I2C address translation function used internally by the driver.
- void IDTGeneral_InitDeviceCommon (T_idtDrvHdlr hdlr)
- T_idtDrvHdlr IDTGeneral_InitDriverCommon (const char ∗name, T_idtDrvAccess drvAccess, T_idtDeviceType deviceType, const T_idtDrvDeviceConfig ∗deviceConfig, int useHighLevelApi)
- void IDTGeneral_DeInitDriver (T_idtDrvHdlr hdlr)

  De-initialize driver handler.

4.3.1 Detailed Description

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

4.3.2 Macro Definition Documentation

4.3.2.1 #define Activate_Edge_Falling 1

Definition at line 85 of file General.h.

4.3.2.2 #define Activate_Edge_Rising 0

Definition at line 84 of file General.h.

4.3.2.3 #define INT_Active_High 1

Definition at line 89 of file General.h.

4.3.2.4 #define INT_Active_Low 0

Definition at line 88 of file General.h.

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

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

Definition at line 97 of file General.h.

4.3.2.6 #define NOMINAL_MAX_PPT 94893

Maximum oscillator offset value.

Definition at line 100 of file General.h.

4.3.2.7 #define NOMINAL_MIN_PPT -94893

Minimum oscillator offset value.

Definition at line 103 of file General.h.

4.3.2.8 #define NOMINAL_PPT_TO_2COMPL(*ppt*) (T_osUint32)(((ppt)*10)/884)

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

Definition at line 93 of file General.h.

4.3.3 Enumeration Type Documentation

4.3.3.1 enum networkType_t

Enumerated type listing types of selectable network type.

Enumerator

Network_SDH SDH network

Network_SONET SONET network

Network_MAX

Definition at line 58 of file General.h.

4.3.3.2 enum phaseTimeoutMultiplier_t

Enumerated type listing phase alarm timeout multipliers.

Enumerator

phaseTimeoutMultiplier_2 2x Multiplier

phaseTimeoutMultiplier_4 4x Multiplier

phaseTimeoutMultiplier_8 8x Multiplier

phaseTimeoutMultiplier_16 18x Multiplier

phaseTimeoutMultiplier_MAX

Definition at line 66 of file General.h.

4.3.4 Function Documentation

4.3.4.1 void IDTGeneral_DeInitDriver (T_idtDrvHdlr *hdlr*)

De-initialize driver handler.

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

Parameters

hdlr	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

hdlr	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

hdlr	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

hdlr	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

hdlr	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

hdlr	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

hdlr	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

hdlr	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

hdlr	Device driver handler
nMultiFactor_p	• phaseTimeoutMultiplier_2 - Multiple 2 • phaseTimeoutMultiplier_4 - Multiple 4 • phaseTimeoutMultiplier_8 - Multiple 8 • phaseTimeoutMultiplier_16 - Multiple 16
nTimeout_p	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

hdlr	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

i2cAddr	I2C slave address
transData	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

hdlr	Device driver handler
nEnable	• 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

hdlr	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

hdlr	Device driver handler
nEnable	• 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

hdlr	Device driver handler
nType	• 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

hdlr	Device driver handler
nEdge	• 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

hdlr	Device driver handler
pptOffset	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

hdlr	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

hdlr	Device driver handler
nEnable	• 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

hdlr	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

hdlr	Device driver handler
nMultiFactor	• phaseTimeoutMultiplier_2 - Multiple 2 • phaseTimeoutMultiplier_4 - Multiple 4 • phaseTimeoutMultiplier_8 - Multiple 8 • phaseTimeoutMultiplier_16 - Multiple 16
nTimeout	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

hdlr	Device driver handler
nEnable	• 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

regVal	register data value.
mask	Bitfield mask
lsb	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

regVal	register data value.
mask	Bitfield mask
lsb	Least significant bit
data	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

hdlr	Device driver handler
offset	Register address offset

Function to read from device.

Parameters

hdlr	Device driver handler
offset	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

hdlr	Device driver handler
offset	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

hdlr	Device driver handler
offset	Register address offset
mask	Bitfield mask
lsb	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

hdlr	Device driver handler
offset	Register address offset
mask	Bitfield mask
lsb	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

hdlr	Device driver handler
offset	Register address offset
data	Data to write

Define function to write to device.

Parameters

hdlr	Device driver handler
offset	Register address offset
data	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 apll address space.

Parameters

hdlr	Device driver handler
offset	Register address offset
data	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

hdlr	Device driver handler
offset	Register address offset
mask	Bitfield mask
lsb	Least significant bit
data	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

hdlr	Device driver handler
offset	Register address offset
mask	Bitfield mask
lsb	Least significant bit
data	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_idtHighFreqencyDivisor { 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_idtHighFreqencyDivisor nUsage)

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

- T_idtHighFreqencyDivisor 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_idtInputFrequencyBitfieldValue 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

hdlr	Device driver handle
ext	External input port number

Returns

a positive number - popluated, 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

hdlr	Device driver handle
ext	External input port number
internal	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 loacking 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_idtHighFreqencyDivisor

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

hdlr	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

hdlr	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

hdlr	Device driver handler
nInputNo	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

hdlr	Device driver handler
nInputNo	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

hdlr	Device driver handler
nBucket	the bucket number to provision 0-3
nUpper_p	Upper threshold of Leaky Bucket
nLower_p	Lower threshold of Leaky Bucket
nSize_p	Bucket size of Leaky Bucket
nDecay_p	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

hdlr	Device driver handler
nInputNo	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

hdlr	Device driver handler
nInputNo	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

hdlr	Device driver handler
nInputNo	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

hdlr	Device driver handler
nInputNo	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

hdlr	Device driver handler
nInputNo	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

hdlr	Device driver handler
nInputNo	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_idtHighFreqencyDivisor IDTInput_GetInputHFdivider (T_idtDrvHdlr *hdlr*, T_osUint8 *nInputNo*)

Get high frequency configuration divider.

Parameters

hdlr	Device driver handler
nInputNo	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

hdlr	Device driver handler
nInputNo	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

hdlr	Device driver handler
nInputNo	Input port number (consult data sheet for valid port numbers)
nDpll	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

hdlr	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

hdlr	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

hdlr	Device driver handler

Returns

Edge to syncrhonize 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

hdlr	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

hdlr	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

hdlr	Device driver handler
exSyncN	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

hdlr	Device driver handler
inFreqBitValue	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

hdlr	Device driver handler
nInputNo	the specified port
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 329 of file Input.c.

4.5.4.23 void IDTInput_SetAmiInputFrequency (T_idtDrvHdlr *hdlr,* T_osUint8 *nInputNo,* T_idtAmiInputFrequencyBitFieldValue *amiSignal*)

Provision AMI signal for input port.

Parameters

hdlr	Device driver handler
nInputNo	the specified input port number
amiSignal	AMI signal • AmiFreq_64kHzPlus8kHz= 64 kHz + 8 kHz signal • AmiFreq_64kHzPlus400Hz= 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

hdlr	Device driver handler
nBucket	the bucket number to provision 0-3
nUpper	Upper threshold of Leaky Bucket
nLower	Lower threshold of Leaky Bucket
nSize	Bucket size of Leaky Bucket
nDecay	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

hdlr	Device driver handler
nInputNo	the input port number to which the pre-divider
nFactor	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

hdlr	Device driver handler
nInputNo	number of the input clock
enable	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

hdlr	Device driver handler
nInputNo	the specified input port number
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 255 of file Input.c.

4.5.4.28 void IDTInput_SetInputHFdivider (T_idtDrvHdlr *hdlr,* T_osUint8 *nInputNo,* T_idtHighFreqencyDivisor *nUsage*)

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

Parameters

hdlr	Device driver handler
nInputNo	Input port number
nUsage	• 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

hdlr	Device driver handler
nInputNo	the specified input port number
nDivider	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

hdlr	Device driver handler
nInputNo	Input port number (consult data sheet for valid port numbers)
nDpll	DPLL instance
nPriority	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

hdlr	Device driver handler
range	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

hdlr	Device driver handler
bypass	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

hdlr	Device driver handler
edge	Edge to syncrhonize 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

hdlr	Device driver handler
enable	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

hdlr	Device driver handler
syncFreq	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

hdlr	Device driver handler
exSyncN	External sync (1-EXT SYNC1, 2-EX SYNC2)
sampling	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_output-PathType 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_SetOuputpuInterface (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_GetOuputpuInterface (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_os-Uint8 *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.

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

hdlr	Device driver handle
ext	External output port number

Returns

a positive number - popluated, 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

hdlr	Device driver handle
ext	External output port number
internal	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

hdlr	Device driver handler

Definition at line 1030 of file Output.c.

4.6.4.2 void IDTOutput_DisableApllInput (T_idtDrvHdlr *hdlr*, T_idtOutputApllConfig *outputApllConfig*)

Function disables APLL input.

Only valid for output path has APLL configured

Parameters

hdlr	Device driver handler
outputApllConfig	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

hdlr	Device driver handler
frameSync	Select between FR and MFR frame syncs
enable_p	Enable frame sync
invert_p	Invert output frame sync (0-noninverted, 1-invert)
pulse_p	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

hdlr	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_GetOuputpuInterface (T_idtDrvHdlr *hdlr*, T_osUint8 *nOutN*)

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

Parameters

hdlr	Device driver handler
nOutN	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_idtApllConfigPerOutput * *apllConfig*)

Get the configuration of Output port.

Parameters

hdlr	Device driver handler
nOutN	Output port that want to be used
nDpll_p	DPLL instance
nPath_p	Select a specified path as input of output divider
nFactor_p	Dividen factor of a certain output divider. Look up the table in datasheet
apllConfig	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

hdlr	Device driver handler
nOutN	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

hdlr	Device driver handler
nOutN	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

hdlr	Device driver handler
frameSync	Select between FR and MFR frame syncs
enable	Enable frame sync
invert	Invert output frame sync (0-noninverted, 1-invert)
pulse	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

hdlr	Device driver handler
isRisingEdge	Frame sync pulse triggered by rising edge.

Definition at line 962 of file Output.c.

4.6.4.11 void IDTOutput_SetOuputpuInterface (T_idtDrvHdlr *hdlr,* T_osUint8 *nOutN,* T_outputInterface *outIf*)

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

Parameters

hdlr	Device driver handler
nOutN	Output port number
outIf	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

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

hdlr	Device driver handler
nOutN	Output port that want to be used
nEnable	• 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

hdlr	Device driver handler
nOutN	Output port that want to be used
invert	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_LOW_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_CNFG = 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_DIVN_LOW_CNFG = 0x24,
  REG_PRE_DIVN_HIGH_CNFG = 0x25, REG_IN1_IN2_SEL_PRIORITY_CNFG = 0x26, REG_IN3_IN4_SEL_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_IN11_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_BUCKET_SIZE_0_CNFG = 0x33, REG_DECAY_RATE_0_CNFG = 0x34,
  REG_UPPER_THRESHOLD_1_CNFG = 0x35, REG_LOWER_THRESHOLD_1_CNFG = 0x36, REG_BUCKET_SIZE_1_CNFG = 0x37, REG_DECAY_RATE_1_CNFG = 0x38,
  REG_UPPER_THRESHOLD_2_CNFG = 0x39, REG_LOWER_THRESHOLD_2_CNFG = 0x3A, REG_BUCKET_SIZE_2_CNFG = 0x3B, REG_DECAY_RATE_2_CNFG = 0x3C,
  REG_UPPER_THRESHOLD_3_CNFG = 0x3D, REG_LOWER_THRESHOLD_3_CNFG = 0x3E, REG_BUCKET_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_STS = 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_CNFG = 0x53, REG_T4_OPERATION_MODE_CNFG = 0x54,
  REG_T0_DPLL_APLL_PATH_CNFG = 0x55, REG_T0_DPLL_START_BW_DAMPING_CNFG = 0x56, REG_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_HOLDOVER_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, REG_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_NOMINAL_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_SEL_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_MULTI_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_SIZE = 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_SONET_SDH_SIZE = 1,
BF_IN_SONET_SDH_MASK = 0x04, BF_IN_SONET_SDH_LSB = 2, BF_MASTER_SLAVE_SIZE = 1, BF_MASTER_SLAVE_MASK = 0x02,
BF_MASTER_SLAVE_LSB = 1, BF_REVERTIVE_MODE_SIZE = 1, BF_REVERTIVE_MODE_MASK = 0x01, BF_REVERTIVE_MODE_LSB = 0,
BF_OSC_EDGE_SIZE = 1, BF_OSC_EDGE_MASK = 0x04, BF_OSC_EDGE_LSB = 2, BF_OUT7_PECL_LVDS_SIZE = 1,
BF_OUT7_PECL_LVDS_MASK = 0x02, BF_OUT7_PECL_LVDS_LSB = 1, BF_OUT6_PECL_LVDS_SIZE = 1, BF_OUT6_PECL_LVDS_MASK = 0x01,
BF_OUT6_PECL_LVDS_LSB = 0, BF_FREQ_MON_CLK_SIZE = 1, BF_FREQ_MON_CLK_MASK = 0x80, BF_FREQ_MON_CLK_LSB = 7,
BF_LOS_FLAG_TO_TDO_SIZE = 1, BF_LOS_FLAG_TO_TDO_MASK = 0x40, BF_LOS_FLAG_TO_TDO_LSB = 6, BF_ULTR_FAST_SW_SIZE = 1,
BF_ULTR_FAST_SW_MASK = 0x20, BF_ULTR_FAST_SW_LSB = 5, BF_EXT_SW_SIZE = 1, BF_EXT_SW_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_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_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, 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_HARD_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_HARD_

D_ALARM_MASK = 0x40, BF_IN8_FREQ_HARD_ALARM_LSB = 6,
BF_IN8_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN8_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN8_NO_-
ACTIVITY_ALARM_LSB = 5, BF_IN8_PH_LOCK_ALARM_SIZE = 1,
BF_IN8_PH_LOCK_ALARM_MASK = 0x10, BF_IN8_PH_LOCK_ALARM_LSB = 4, BF_IN7_FREQ_SOFT-
_ALARM_SIZE = 1, BF_IN7_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN7_FREQ_SOFT_ALARM_LSB = 3, BF_IN7_FREQ_HARD_ALARM_SIZE = 1, BF_IN7_FREQ_HAR-
D_ALARM_MASK = 0x04, BF_IN7_FREQ_HARD_ALARM_LSB = 2,
BF_IN7_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN7_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN7_NO_-
ACTIVITY_ALARM_LSB = 1, BF_IN7_PH_LOCK_ALARM_SIZE = 1,
BF_IN7_PH_LOCK_ALARM_MASK = 0x01, BF_IN7_PH_LOCK_ALARM_LSB = 0, BF_IN10_FREQ_SOF-
T_ALARM_SIZE = 1, BF_IN10_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN10_FREQ_SOFT_ALARM_LSB = 7, BF_IN10_FREQ_HARD_ALARM_SIZE = 1, BF_IN10_FREQ_H-
ARD_ALARM_MASK = 0x40, BF_IN10_FREQ_HARD_ALARM_LSB = 6,
BF_IN10_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN10_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN10_-
NO_ACTIVITY_ALARM_LSB = 5, BF_IN10_PH_LOCK_ALARM_SIZE = 1,
BF_IN10_PH_LOCK_ALARM_MASK = 0x10, BF_IN10_PH_LOCK_ALARM_LSB = 4, BF_IN9_FREQ_SO-
FT_ALARM_SIZE = 1, BF_IN9_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN9_FREQ_SOFT_ALARM_LSB = 3, BF_IN9_FREQ_HARD_ALARM_SIZE = 1, BF_IN9_FREQ_HAR-
D_ALARM_MASK = 0x04, BF_IN9_FREQ_HARD_ALARM_LSB = 2,
BF_IN9_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN9_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN9_NO_-
ACTIVITY_ALARM_LSB = 1, BF_IN9_PH_LOCK_ALARM_SIZE = 1,
BF_IN9_PH_LOCK_ALARM_MASK = 0x01, BF_IN9_PH_LOCK_ALARM_LSB = 0, BF_IN12_FREQ_SOF-
T_ALARM_SIZE = 1, BF_IN12_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN12_FREQ_SOFT_ALARM_LSB = 7, BF_IN12_FREQ_HARD_ALARM_SIZE = 1, BF_IN12_FREQ_H-
ARD_ALARM_MASK = 0x40, BF_IN12_FREQ_HARD_ALARM_LSB = 6,
BF_IN12_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN12_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN12_-
NO_ACTIVITY_ALARM_LSB = 5, BF_IN12_PH_LOCK_ALARM_SIZE = 1,
BF_IN12_PH_LOCK_ALARM_MASK = 0x10, BF_IN12_PH_LOCK_ALARM_LSB = 4, BF_IN11_FREQ_S-
OFT_ALARM_SIZE = 1, BF_IN11_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN11_FREQ_SOFT_ALARM_LSB = 3, BF_IN11_FREQ_HARD_ALARM_SIZE = 1, BF_IN11_FREQ_H-
ARD_ALARM_MASK = 0x04, BF_IN11_FREQ_HARD_ALARM_LSB = 2,
BF_IN11_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN11_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN11_-
NO_ACTIVITY_ALARM_LSB = 1, BF_IN11_PH_LOCK_ALARM_SIZE = 1,
BF_IN11_PH_LOCK_ALARM_MASK = 0x01, BF_IN11_PH_LOCK_ALARM_LSB = 0, BF_IN14_FREQ_S-
OFT_ALARM_SIZE = 1, BF_IN14_FREQ_SOFT_ALARM_MASK = 0x80,
BF_IN14_FREQ_SOFT_ALARM_LSB = 7, BF_IN14_FREQ_HARD_ALARM_SIZE = 1, BF_IN14_FREQ_H-
ARD_ALARM_MASK = 0x40, BF_IN14_FREQ_HARD_ALARM_LSB = 6,
BF_IN14_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN14_NO_ACTIVITY_ALARM_MASK = 0x20, BF_IN14_-
NO_ACTIVITY_ALARM_LSB = 5, BF_IN14_PH_LOCK_ALARM_SIZE = 1,
BF_IN14_PH_LOCK_ALARM_MASK = 0x10, BF_IN14_PH_LOCK_ALARM_LSB = 4, BF_IN13_FREQ_S-
OFT_ALARM_SIZE = 1, BF_IN13_FREQ_SOFT_ALARM_MASK = 0x08,
BF_IN13_FREQ_SOFT_ALARM_LSB = 3, BF_IN13_FREQ_HARD_ALARM_SIZE = 1, BF_IN13_FREQ_H-
ARD_ALARM_MASK = 0x04, BF_IN13_FREQ_HARD_ALARM_LSB = 2,
BF_IN13_NO_ACTIVITY_ALARM_SIZE = 1, BF_IN13_NO_ACTIVITY_ALARM_MASK = 0x02, BF_IN13_-
NO_ACTIVITY_ALARM_LSB = 1, BF_IN13_PH_LOCK_ALARM_SIZE = 1,
BF_IN13_PH_LOCK_ALARM_MASK = 0x01, BF_IN13_PH_LOCK_ALARM_LSB = 0, BF_HIGHEST_PRI-
ORITY_VALIDATED_SIZE = 4, BF_HIGHEST_PRIORITY_VALIDATED_MASK = 0xF0,
BF_HIGHEST_PRIORITY_VALIDATED_LSB = 4, BF_CURRENTLY_SELECTED_INPUTS_SIZE = 4, BF_-
CURRENTLY_SELECTED_INPUTS_MASK = 0x0F, BF_CURRENTLY_SELECTED_INPUTS_LSB = 0,
BF_THIRD_HIGHEST_PRIORITY_VALIDATED_SIZE = 4, BF_THIRD_HIGHEST_PRIORITY_VALIDATE-
D_MASK = 0xF0, BF_THIRD_HIGHEST_PRIORITY_VALIDATED_LSB = 4, BF_SECOND_HIGHEST_PRI-
ORITY_VALIDATED_SIZE = 4,
BF_SECOND_HIGHEST_PRIORITY_VALIDATED_MASK = 0x0F, BF_SECOND_HIGHEST_PRIORITY_V-
ALIDATED_LSB = 0, BF_T0_INPUT_SEL_SIZE = 4, BF_T0_INPUT_SEL_MASK = 0x0F,
BF_T0_INPUT_SEL_LSB = 0, BF_T4_LOCK_T0_SIZE = 1, BF_T4_LOCK_T0_MASK = 0x40, BF_T4_LOC-
K_T0_LSB = 6,
BF_T0_FOR_T4_SIZE = 1, BF_T0_FOR_T4_MASK = 0x20, BF_T0_FOR_T4_LSB = 5, BF_T4_TEST_T0_-

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

ASK = 0xFF, BF_HOLDOVER_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_IN_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_SYNC_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_PULSE_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_ICP_CTRL_CODE_MASK = 0x1F,
BF_ICP_CTRL_CODE_LSB = 0, REGMAP_MAX }

Enumerated type listing register offsers 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_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 offsers 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 offsers 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_INPU-T_VALID2_CNFG

BF_IN14_MASK

BF_IN14_LSB

BF_IN13_SIZE INTERRUPTS2_STS INTERRUPTS2_MASK_CNFG INPUT_VALID2_STS REMOTE_INPU-T_VALID2_CNFG

BF_IN13_MASK

BF_IN13_LSB

BF_IN12_SIZE INTERRUPTS2_STS INTERRUPTS2_MASK_CNFG INPUT_VALID2_STS REMOTE_INPU-T_VALID2_CNFG

BF_IN12_MASK

BF_IN12_LSB

BF_IN11_SIZE INTERRUPTS2_STS INTERRUPTS2_MASK_CNFG INPUT_VALID2_STS REMOTE_INPU-T_VALID2_CNFG

BF_IN11_MASK

BF_IN11_LSB

BF_IN10_SIZE INTERRUPTS2_STS INTERRUPTS2_MASK_CNFG INPUT_VALID2_STS REMOTE_INPU-T_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 IN10_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_CNFG 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_HOLDOVER_FREQ_B_MASK

BF_HOLDOVER_FREQ_B_LSB

BF_HOLDOVER_FREQ_C_SIZE HOLDOVER_FREQ_HIGH_CNFG

BF_HOLDOVER_FREQ_C_MASK

BF_HOLDOVER_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 offsers 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 ApllRegDef.h.

4.8 Apll Function Module

Functions in this group are related to APLL provisioning.

Data Structures

- struct T_idtApllConfigPerOutput

  Structure containing APLL per output configuration.
- struct T_idtApllConfig

  Structure containing APLL full configuration.
- struct T_idtApllFreqMapping

  Type definition for mapping output frequency to APLL configuration.
- struct T_idtApllXtal

  Structure containing APLL crystal id and frequency information.

Macros

- #define IDT_GET_OUTPUT_APLL_CONFIG(hdlr, output, outputApll)

  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_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_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.
- 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.

Functions

- void IDTApll_SetApllConfigPerOutput (T_idtDrvHdlr hdlr, T_idtApllId apllId, T_idtApllOutput output, T_idtApllConfigPerOutput *apllConfigPerOutput)

  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 confiugred 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.

- T_idtApllConfigSel IDTApll_GetConfigSel (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve APLL configuration.

- void IDTApll_SetConfigSel (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig)

  Function to provision APLL based on input configuration.

- T_idtApllInputClkSrc IDTApll_GetInputClkSrc (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve selected APLL input frequency source (external/internal)

- void IDTApll_SetInputClkSrc (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllInputClkSrc inputClk)

  Function to provision APLL input frequency source (external/internal)

- T_idtApllDividerValue IDTApll_GetPreDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig)

Function to retrieve pre- (input) divider value.

- void IDTApll_GetPreDivRange (T_idtDrvHdlr hdlr, T_idtApllDividerValue *minDiv, T_idtApllDividerValue *maxDiv)

  Function to retrieve pre- (input) divisor range.
- void IDTApll_SetPreDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig, T_idtApll-DividerValue div)

  Function to provision pre- (input) divider value.
- T_idtApllDividerValue IDTApll_GetFeedbackDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig)

  Function to retrieve feedback (oscillator) divider value.
- void IDTApll_GetFeedbackDivRange (T_idtDrvHdlr hdlr, T_idtApllDividerValue *minDiv, T_idtApllDivider-Value *maxDiv)

  Function to retrieve feedback (oscillator) divisor range.
- void IDTApll_SetFeedbackDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig, T_idt-ApllDividerValue div)

  Function to provision feedback (oscillator) divider value.
- T_idtApllQaDiv IDTApll_GetQaDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig)

  Function to retrieve output port A divisor.
- void IDTApll_SetQaDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig, T_idtApllQa-Div div)

  Function to provision output port A divisor.
- T_idtApllQaDiv IDTApll_GetQbDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig)

  Function to retrieve output port B divisor.
- void IDTApll_SetQbDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig, T_idtApllQb-Div div)

  Function to provision output port B divisor.
- T_idtApllFreqDoublerSel IDTApll_GetFreqDoublerSel (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve whether feedback frequency doubling is enabled.
- void IDTApll_SetFreqDoublerSel (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllFreqDoublerSel doubler)

  Function to enable/disable feedback frequency doubling.
- T_idtApllXtalId IDTApll_GetXtalId (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve oscillator currently used.
- void IDTApll_SetXtalId (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllXtalId xtal)

  Function to select oscillator.
- T_idtApllOutputSigType IDTApll_GetOutputSigType (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllOutput output)

  Function to retrieve output port signal type.
- void IDTApll_SetOutputSigType (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllOutput output, T_idtApll-OutputSigType outputSigType)

  Function to provision output port signal type.
- T_idtApllOutputEn IDTApll_GetOutputEnState (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllOutput output)

  Function to retrieve output port status (enable/disable)
- void IDTApll_SetOutputEnState (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllOutput output, T_idtApll-OutputEn enable)

  Function to control (enable/disable) output port.
- T_idtApllMasterResetSync IDTApll_GetMasterResetSync (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve reset status of device.
- void IDTApll_SetMasterResetSync (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllMasterResetSync re-set)

  Function to reset APLL.
- T_idtApllBypass IDTApll_GetBypass (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

Function to retrieve bypass status of device.

- void IDTApll_SetBypass (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllBypass bypass)

  Function to set APLL bypass setting.
- T_idtApllShutoff IDTApll_GetShutoff (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve shutoff status of device.
- void IDTApll_SetShutoff (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllShutoff shutoff)

  Function to set APLL shutoff setting.
- const T_idtApllFreqMapping * IDTApll_GetConfigListByFreq (T_idtApllOutputFreqType apllOutputFreq)

  Function to retrieve output configuration given an output frequency.
- const T_idtApllFreqMapping * IDTApll_GetConfigListByXtalType (T_idtApllXtalType apllXtalFreq)

  Function to retrieve output configuration gvien the oscilator type (frequency)
- T_osBool IDTApll_NullFreqTableEntry (const T_idtApllFreqMapping *apllFreqMapping)

  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.

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 ApllLogging.h.

4.8.2.5 #define IDT_APLL_API_TRACE(*fmt*) OS_TRACE(fmt)

Definition at line 72 of file ApllLogging.h.

4.8.2.6 #define IDT_GET_OUTPUT_APLL_CONFIG(*hdlr, output, outputApll*)

Value:

```
do {                                \
            OS_ASSERT((output)<=IDT_DRV_MAX_OUTPUT);    \
            OS_ASSERT((output)>0);              \
            outputApll = hdlr->deviceConfig_m->outPutApllConfig[(output)]; \
    } while(0)
```

Utility macro to translate output port to APLL instance.

Parameters

hdlr	Device driver handle
output	External output port number
outputApll	APLL instance

Definition at line 60 of file Apll.h.

4.8.2.7 #define INVALID_DIVIDER_VALUE (0xFFFF)

Define constant indicating invalid divider value.

Definition at line 49 of file ApllTypeDef.h.

4.8.3 Typedef Documentation

4.8.3.1 typedef T_osUint16 T_idtApllDividerValue

Type define for APLL divider value.

Definition at line 224 of file ApllTypeDef.h.

4.8.3.2 typedef T_osUint8 T_idtApllRegAddr

Type define for APLL register address.

Definition at line 229 of file ApllTypeDef.h.

4.8.3.3 typedef T_osUint8 T_idtApllRegMask

Type define for APLL register value mask.

Definition at line 234 of file ApllTypeDef.h.

4.8.3.4 typedef T_osUint8 T_idtApllRegVal

Type define for APLL register value.

Definition at line 239 of file ApllTypeDef.h.

4.8.4 Enumeration Type Documentation

4.8.4.1 enum T_idtApllAccessType

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 ApllTypeDef.h.

4.8.4.2 enum T_idtApllBypass

Enumerated type listing Apll bypass status/setting.

Enumerator

APLL_BYPASS_NORMAL_OP Apll in normal operation

APLL_BYPASS_BYPASS Apll bypass VCXO and connect selected clock input to output A and B

APLL_BYPASS_MAX

Definition at line 204 of file ApllTypeDef.h.

4.8.4.3 enum T_idtApllConfigSel

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 ApllTypeDef.h.

4.8.4.4 enum T_idtApllFreqDoublerSel

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 ApllTypeDef.h.

4.8.4.5 enum T_idtApllId

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 ApllTypeDef.h.

4.8.4.6 enum T_idtApllInputClkSrc

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 ApllTypeDef.h.

4.8.4.7 enum T_idtApllInputFreqType

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 ApllFreqProfile.h.

4.8.4.8 enum T_idtApllMasterResetSync

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 ApllTypeDef.h.

4.8.4.9 enum T_idtApllOutput

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 ApllTypeDef.h.

4.8.4.10 enum T_idtApllOutputEn

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 ApllTypeDef.h.

4.8.4.11 enum T_idtApllOutputFreqType

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 ApllFreqProfile.h.

4.8.4.12 enum T_idtApllOutputSigType

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 ApllTypeDef.h.

4.8.4.13 enum T_idtApllOutputSupportedType

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 ApllTypeDef.h.

4.8.4.14 enum T_idtApllQaDiv

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 ApllTypeDef.h.

4.8.4.15 enum T_idtApllQbDiv

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 ApllTypeDef.h.

4.8.4.16 enum T_idtApllShutoff

Enumerated type listing Apll shutoff status/setting.

Enumerator

APLL_SHUTOFF_NORMAL_OP Apll in normal operation
APLL_SHUTOFF_SHUTOFF VCXO current is shut off and Outputs are squelched
APLL_SHUTOFF_MAX

Definition at line 214 of file ApllTypeDef.h.

4.8.4.17 enum T_idtApllXtalId

Enumerated type listing crystals.

Enumerator

APLL_XTAL_1_APLL1 XTAL1 connects to APPL1
APLL_XTAL_2_APLL2 XTAL2 connects to APPL2
APLL_XTAL_3_APLL1 XTAL3 connects to APPL1
APLL_XTAL_4_APLL2 XTAL4 connects to APPL2
APLL_XTAL_MAX

Definition at line 171 of file ApllTypeDef.h.

4.8.4.18 enum T_idtApllXtalType

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 ApllTypeDef.h.

4.8.5 Function Documentation

4.8.5.1 outputFrequency_t IDTApll_ApllInput2DpllOutput (T_idtApllInputFreqType *apllInputFreq*)

Translate APLL input frequency to DPLL output frequency.

Parameters

apllInputFreq	APLL input frequency

Returns

DPLL output frequency

Definition at line 474 of file Apll.c.

4.8.5.2 T_idtApllOutputFreqType IDTApll_DpllOutput2ApllOutput (outputFrequency_t *dpllOutput*)

Translate DPLL output frequency to APLL output frequency.

Parameters

dpllOutput	DPLL output frequency

Returns

APLL output frequency

Definition at line 533 of file Apll.c.

4.8.5.3 void IDTApll_GetApllConfig (T_idtDrvHdlr *hdlr,* T_idtApllId *apllInstId,* T_idtApllConfig * *apllConfig*)

Get APLL full configuration.

Parameters

hdlr	Device driver handler
apllInstId	APLL instance
apllConfig	APLL configuration

Definition at line 416 of file Apll.c.

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

Get APLL per output configuration.

Parameters

hdlr	Device driver handler
apllInstId	APLL instance
output	Output port
apllConfigPer-Output	APLL per port configuration

Definition at line 318 of file Apll.c.

4.8.5.5 const T_idtApIIFreqMapping* IDTApll_GetApIIFreqTableEntry (T_idtDrvHdlr *hdlr,* T_idtOutputApIIConfig *outputApll,* outputFrequency_t *nOutFreq*)

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

Parameters

hdlr	Device driver handler
outputApll	Output APLL configuration
nOutFreq	Output frequency

Returns

APLL frequency configuration

Definition at line 627 of file Apll.c.

4.8.5.6 T_idtApIIBypass IDTApll_GetBypass (T_idtDrvHdlr *hdlr,* T_idtApIId *apllInstId*)

Function to retrieve bypass status of device.

Parameters

hdlr	Device driver handler
apllInstId	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 ApllApi.c.

4.8.5.7 const T_idtApIIFreqMapping* IDTApll_GetConfigListByFreq (T_idtApIIOutputFreqType *apllOutputFreq*)

Function to retrieve output configuration given an output frequency.

Parameters

apllOutputFreq	Output frequency

Returns

Output configuration

Definition at line 545 of file ApllFreqProfile.c.

4.8.5.8 const T_idtApIIFreqMapping* IDTApll_GetConfigListByXtalType (T_idtApIIXtalType *apllVcxoFreq*)

Function to retrieve output configuration gvien the oscilator type (frequency)

Parameters

apllVcxoFreq	Oscilator type (frequency)

Returns

Output configuration

Definition at line 564 of file ApllFreqProfile.c.

4.8.5.9 T_idtApIIConfigSel IDTApII_GetConfigSel (T_idtDrvHdlr *hdlr*, T_idtApIIId *apllInstId*)

Function to retrieve APLL configuration.

Parameters

hdlr	Device driver handler
apllInstId	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 ApllApi.c.

4.8.5.10 const T_idtApIIFreqMapping* IDTApII_GetCurrentApIIFreqConfig (T_idtApIIId *apllId*, T_idtApIIOutput *output*)

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

Parameters

apllId	APLL instance
output	APLL output

Returns

APLL frequency mapping configuration

Definition at line 604 of file ApllFreqProfile.c.

4.8.5.11 T_idtApIIDividerValue IDTApII_GetFeedbackDiv (T_idtDrvHdlr *hdlr*, T_idtApIIId *apllInstId*, T_idtApIIConfigSel *apllConfig*)

Function to retrieve feedback (oscillator) divider value.

Parameters

hdlr	Device driver handler
apllInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
apllConfig	APLL configuration • APLL_CONFIG0 - configuration 0 • APLL_CONFIG1 - configuration 1

Returns

Divisor value

Definition at line 525 of file ApllApi.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

hdlr	Device driver handler
minDiv	minimum divisor
maxDiv	maximum divisor

Definition at line 502 of file ApllApi.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

hdlr	Device driver handler
apllInstId	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 ApllApi.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

hdlr	Device driver handler
apllInstId	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_idtApIIMasterResetSync IDTApII_GetMasterResetSync (T_idtDrvHdlr *hdlr*, T_idtApIIId *apllInstId*)

Function to retrieve reset status of device.

Parameters

hdlr	Device driver handler
apllInstId	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_idtApIIConfigSel IDTApII_GetNonActiveApIIConfig (T_idtDrvHdlr *hdlr*, T_idtApIIId *apllInst*)

Get non-active APLL configuration.

Parameters

hdlr	Device driver handler
apllInst	APLL instance

Returns

Non-active APLL configuration

Definition at line 706 of file Apll.c.

4.8.5.17 T_idtApllOutputEn IDTApll_GetOutputEnState (T_idtDrvHdlr *hdlr*, T_idtApllId *apllInstId*, T_idtApllOutput *output*)

Function to retrieve output port status (enable/disable)

Parameters

hdlr	Device driver handler
apllInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
output	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 ApllApi.c.

4.8.5.18 T_idtApllOutputSigType IDTApll_GetOutputSigType (T_idtDrvHdlr *hdlr*, T_idtApllId *apllInstId*, T_idtApllOutput *output*)

Function to retrieve output port signal type.

Parameters

hdlr	Device driver handler
apllInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
output	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 ApllApi.c.

4.8.5.19 T_idtApIIDividerValue IDTApII_GetPreDiv (T_idtDrvHdlr *hdlr,* T_idtApIIId *apIIInstId,* T_idtApIIConfigSel *apIIConfig*)

Function to retrieve pre- (input) divider value.

Parameters

hdlr	Device driver handler
apIIInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
apIIConfig	APLL configuration • APLL_CONFIG0 - configuration 0 • APLL_CONFIG1 - configuration 1

Returns

Divisor value

Definition at line 356 of file ApllApi.c.

4.8.5.20 void IDTApII_GetPreDivRange (T_idtDrvHdlr *hdlr,* T_idtApIIDividerValue * *minDiv,* T_idtApIIDividerValue * *maxDiv*)

Function to retrieve pre- (input) divisor range.

Parameters

hdlr	Device driver handler
minDiv	minimum divisor
maxDiv	maximum divisor

Note

Divisor "1" is also a valid divisor

Definition at line 415 of file ApllApi.c.

4.8.5.21 T_idtApIIQaDiv IDTApII_GetQaDiv (T_idtDrvHdlr *hdlr,* T_idtApIIId *apIIInstId,* T_idtApIIConfigSel *config*)

Function to retrieve output port A divisor.

Parameters

hdlr	Device driver handler
apIIInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2

config	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 ApllApi.c.

4.8.5.22 T_idtApllQaDiv IDTApll_GetQbDiv (T_idtDrvHdlr *hdlr*, T_idtApllId *apllInstId*, T_idtApllConfigSel *config*)

Function to retrieve output port B divisor.

Parameters

hdlr	Device driver handler
apllInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
config	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 ApllApi.c.

4.8.5.23 T_idtApllShutoff IDTApll_GetShutoff (T_idtDrvHdlr *hdlr*, T_idtApllId *apllInstId*)

Function to retrieve shutoff status of device.

Parameters

hdlr	Device driver handler
apllInstId	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 ApllApi.c.

4.8.5.24 T_idtApllXtal* IDTApll_GetSupportedXtal (const T_idtDrvDeviceConfig * *deviceConfig,* T_osUint8 * *numberOfXtal*)

Return supported xtal configuration by the device.

Parameters

deviceConfig	APLL device configuration
numberOfXtal	Number of xtal configuration in the returned list

Returns

The list of suppoted XTAL configuration

Caller needs to free the list after finishing using it.

Definition at line 899 of file Apll.c.

4.8.5.25 T_idtApllXtalId IDTApll_GetXtalId (T_idtDrvHdlr *hdlr,* T_idtApllId *apllInstId*)

Function to retrieve oscillator currently used.

Parameters

hdlr	Device driver handler
apllInstId	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 ApllApi.c.

4.8.5.26 T_idtApllXtalId IDTApll_GetXtalIdFromXtalFreq (T_idtDrvHdlr *hdlr*, T_idtApllId *apllInst*, T_idtApllXtalType *apllVcxoFreq*)

Get APLL configured crystal ID from crystal frequency.

Parameters

hdlr	Device driver handler
apllInst	APLL instance
apllVcxoFreq	Crytal frequency

Returns

Crytal ID

Definition at line 753 of file Apll.c.

4.8.5.27 T_osBool IDTApll_NullFreqTableEntry (const T_idtApllFreqMapping * *apllFreqConfig*)

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

Parameters

apllFreqConfig	APLL frequency mapping configuration

Returns

E_osTrue - not configured, E_osFalse - configured

Definition at line 583 of file ApllFreqProfile.c.

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

Set APLL full configuration.

Parameters

hdlr	Device driver handler
apllInstId	APLL instance
apllConfig	APLL configuration

Definition at line 357 of file Apll.c.

4.8.5.29 void IDTApll_SetApllConfigPerOutput (T_idtDrvHdlr *hdlr*, T_idtApllId *apllInstId*, T_idtApllOutput *output*, T_idtApllConfigPerOutput * *apllConfigPerOutput*)

Set APLL per output configuration.

Parameters

hdlr	Device driver handler
apllInstId	APLL instance
output	Output port
apllConfigPer-Output	APLL per port configuration

Definition at line 272 of file Apll.c.

4.8.5.30 void IDTApll_SetBypass (T_idtDrvHdlr *hdlr,* T_idtApllId *apllInstId,* T_idtApllBypass *bypass*)

Function to set APLL bypass setting.

Parameters

hdlr	Device driver handler
apllInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
bypass	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 ApllApi.c.

4.8.5.31 void IDTApll_SetConfigSel (T_idtDrvHdlr *hdlr,* T_idtApllId *apllInstId,* T_idtApllConfigSel *apllConfig*)

Function to provision APLL based on input configuration.

Parameters

hdlr	Device driver handler
apllInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
apllConfig	APLL configuration • APLL_CONFIG0 - configuration 0 • APLL_CONFIG1 - configuration 1

Definition at line 250 of file ApllApi.c.

4.8.5.32 void IDTApll_SetFeedbackDiv (T_idtDrvHdlr *hdlr,* T_idtApllId *apllInstId,* T_idtApllConfigSel *apllConfig,* T_idtApllDividerValue *div*)

Function to provision feedback (oscillator) divider value.

Parameters

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

Definition at line 588 of file ApllApi.c.

4.8.5.33 void IDTApll_SetFreqDoublerSel (T_idtDrvHdlr *hdlr,* T_idtApllId *apllInstId,* T_idtApllFreqDoublerSel *doubler*)

Function to enable/disable feedback frequency doubling.

Parameters

hdlr	Device driver handler
apllInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
doubler	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 ApllApi.c.

4.8.5.34 void IDTApll_SetInputClkSrc (T_idtDrvHdlr *hdlr,* T_idtApllId *apllInstId,* T_idtApllInputClkSrc *inputClk*)

Function to provision APLL input frequency source (external/internal)

Parameters

hdlr	Device driver handler
apllInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
inputClk	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 ApllApi.c.

4.8.5.35 void IDTApII_SetMasterResetSync (T_idtDrvHdlr *hdlr*, T_idtApIIId *apIIInstId*, T_idtApIIMasterResetSync *reset*)

Function to reset APLL.

Parameters

hdlr	Device driver handler
apIIInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
reset	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 ApIIApi.c.

4.8.5.36 void IDTApII_SetOutputEnState (T_idtDrvHdlr *hdlr*, T_idtApIIId *apIIInstId*, T_idtApIIOutput *output*, T_idtApIIOutputEn *enable*)

Function to control (enable/disable) output port.

Parameters

hdlr	Device driver handler
apIIInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
output	Output port • APLL_OUTPUT_QA - Output port A • APLL_OUTPUT_QB - Output port B
enable	Enable /Disable output • APLL_OUTPUT_DISABLE - Disable output • APLL_OUTPUT_ENABLE - Enable output

Definition at line 1199 of file ApIIApi.c.

4.8.5.37 void IDTApII_SetOutputSigType (T_idtDrvHdlr *hdlr*, T_idtApIIId *apIIInstId*, T_idtApIIOutput *output*, T_idtApIIOutputSigType *outputSigType*)

Function to provision output port signal type.

Parameters

hdlr	Device driver handler
apIIInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2

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

Definition at line 1088 of file ApllApi.c.

4.8.5.38 void IDTApll_SetPreDiv (T_idtDrvHdlr *hdlr,* T_idtApllId *apllInstId,* T_idtApllConfigSel *apllConfig,* T_idtApllDividerValue *div*)

Function to provision pre- (input) divider value.

Parameters

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

Definition at line 438 of file ApllApi.c.

4.8.5.39 void IDTApll_SetQaDiv (T_idtDrvHdlr *hdlr,* T_idtApllId *apllInstId,* T_idtApllConfigSel *config,* T_idtApllQaDiv *div*)

Function to provision output port A divisor.

Parameters

hdlr	Device driver handler
apllInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
config	APLL configuration • APLL_CONFIG0 - configuration 0 • APLL_CONFIG1 - configuration 1
div	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 ApllApi.c.

4.8.5.40 void IDTApll_SetQbDiv (T_idtDrvHdlr *hdlr*, T_idtApIIId *apllInstId*, T_idtApIIConfigSel *config*, T_idtApIIQbDiv *div*)

Function to provision output port B divisor.

Parameters

hdlr	Device driver handler
apllInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
config	APLL configuration • APLL_CONFIG0 - configuration 0 • APLL_CONFIG1 - configuration 1
div	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 ApllApi.c.

4.8.5.41 void IDTApll_SetShutoff (T_idtDrvHdlr *hdlr*, T_idtApIIId *apllInstId*, T_idtApIIShutoff *shutoff*)

Function to set APLL shutoff setting.

Parameters

hdlr	Device driver handler
apllInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
shutoff	Value to set shutoff • APLL_SHUTOFF_NORMAL_OP - Normal operation • APLL_SHUTOFF_SHUTOFF - VCXO shutoff

Definition at line 1390 of file ApllApi.c.

4.8.5.42 void IDTApll_SetXtalId (T_idtDrvHdlr *hdlr*, T_idtApllId *apllInstId*, T_idtApllXtalId *xtal*)

Function to select oscillator.

Parameters

hdlr	Device driver handler
apllInstId	APLL instance • APLL_INST_1 - APLL #1 • APLL_INST_2 - APLL #2
xtal	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 ApllApi.c.

4.8.5.43 void IDTApll_Switch2NonActiveApllConfig (T_idtDrvHdlr *hdlr*, T_idtApllId *apllId*)

Switch to non-active APLL configuration.

Parameters

hdlr	Device driver handler
apllId	APLL instance

Definition at line 733 of file Apll.c.

4.8.5.44 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.

Parameters

deviceConfig	APLL device configuration
xtalList	List of XTAL information for the device
numberOfXtal	Number of XTAL in the xtalList

Returns

TRUE, if input XTAL information is valid; FALSE, otherwise

Definition at line 834 of file Apll.c.

4.8.5.45 T_osBool lDTApll_XtalConfigured (T_idtDrvHdlr *hdlr,* T_idtApllId *apllId*)

Checks if there is any XTAL confiugred for the APLL.

Parameters

hdlr	Device driver handler
apllId	APLL instance

Returns

TRUE, if there is at least one XTAL configured; FALSE, otherwise

Definition at line 790 of file Apll.c.

Chapter 5

Data Structure Documentation

5.1 bucketReg_t Struct Reference

Structure used to store leaky bucket register offsets.

Data Fields

- T_osUint8 upperThreshReg_m
- T_osUint8 lowerThreshReg_m
- T_osUint8 bucketSizeReg_m
- T_osUint8 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_osUint8 bucketSizeReg_m

Bucket size register offset

Definition at line 51 of file Input.c.

5.1.2.2 T_osUint8 decayRateReg_m

Decay rate register offset

Definition at line 52 of file Input.c.

5.1.2.3 T_osUint8 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_idtApllConfig::fbDiv_s Struct Reference

`#include <Apll.h>`

Data Fields

- T_idtApllDividerValue fbDivConfig0
- T_idtApllDividerValue fbDivConfig1

5.2.1 Detailed Description

Definition at line 95 of file Apll.h.

5.2.2 Field Documentation

5.2.2.1 T_idtApllDividerValue fbDivConfig0

APLL feedback divider value for configuration 0

Definition at line 97 of file Apll.h.

5.2.2.2 T_idtApllDividerValue fbDivConfig1

APLL feedback divider value for configuration 1

Definition at line 98 of file Apll.h.

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

- common/include/Apll.h

5.3 globalArgs_t Struct Reference

Structure to define program command line arguments.

Data Fields

- T_osUint32 modes_m
- char filename_ma [SAMPLE_MAX_FILENAME]
- T_osUint32 inputFreq_m [IDT_DRV_MAX_INPUT]
- outputFrequency_t outFreq_m [IDT_DRV_MAX_OUTPUT]
- T_idtDpllType dpll_m [IDT_DRV_MAX_OUTPUT]

- T_osUint8 isSim_m
- T_osUint8 dcoMode_m
- long offset_m
- T_osUint8 numberOfXtal
- T_idtApllXtal * xtalList
- T_DpllProfile dpllProfile_ma [DPLL_INSTANCE_MAX]
- T_osUint8 sampleConfig_m
- T_osUint8 regOffset_m
- T_osUint8 regValue_m

5.3.1 Detailed Description

Structure to define program command line arguments.

Definition at line 127 of file main.c.

5.3.2 Field Documentation

5.3.2.1 T_osUint8 dcoMode_m

```
Enable/Disable
```

DCO mode

Definition at line 143 of file main.c.

5.3.2.2 T_idtDpllType dpll_m[IDT_DRV_MAX_OUTPUT]

Select DPLL T0 / T4

Definition at line 138 of file main.c.

5.3.2.3 T_DpllProfile dpllProfile_ma[DPLL_INSTANCE_MAX]

Dpll Profile

Definition at line 148 of file main.c.

5.3.2.4 char filename_ma[SAMPLE_MAX_FILENAME]

```
File name for
```

load/store

Definition at line 130 of file main.c.

5.3.2.5 T_osUint32 inputFreq_m[IDT_DRV_MAX_INPUT]

Input freqency (kHz)

Definition at line 132 of file main.c.

5.3.2.6 T_osUint8 isSim_m

```
Switch to
```

similated mode

Definition at line 141 of file main.c.

5.3.2.7 T_osUint32 modes_m

```
Operation modes
```

for prgram

Definition at line 128 of file main.c.

5.3.2.8 T_osUint8 numberOfXtal

Number APLL external Xtals

Definition at line 146 of file main.c.

5.3.2.9 long offset_m

DCO Offset

Definition at line 145 of file main.c.

5.3.2.10 outputFrequency_t outFreq_m[IDT_DRV_MAX_OUTPUT]

Output frequency (enum)

Definition at line 135 of file main.c.

5.3.2.11 T_osUint8 regOffset_m

register offset

Definition at line 151 of file main.c.

5.3.2.12 T_osUint8 regValue_m

register value

Definition at line 152 of file main.c.

5.3.2.13 T_osUint8 sampleConfig_m

Sample configuration

Definition at line 150 of file main.c.

5.3.2.14 T_idtApllXtal* xtalList

APLL external Xtal list

Definition at line 147 of file main.c.

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

- 82V3910/sample/src/main.c

5.4 T_idtApllConfig::outputConfig_s Struct Reference

`#include <Apll.h>`

Data Structures

- struct qaConfig_s
- struct qbConfig_s

Data Fields

- struct T_idtApllConfig::outputConfig_s::qaConfig_s qaConfig
- struct T_idtApllConfig::outputConfig_s::qbConfig_s qbConfig

5.4.1 Detailed Description

Definition at line 118 of file Apll.h.

5.4.2 Field Documentation

5.4.2.1 struct T_idtApllConfig::outputConfig_s::qaConfig_s qaConfig

5.4.2.2 struct T_idtApllConfig::outputConfig_s::qbConfig_s qbConfig

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

- common/include/Apll.h

5.5 T_idtApllConfig::outputDiv_s Struct Reference

`#include <Apll.h>`

Data Structures

- struct qaDiv_s
- struct qbDiv_s

Data Fields

- struct T_idtApllConfig::outputDiv_s::qaDiv_s qaDiv
- struct T_idtApllConfig::outputDiv_s::qbDiv_s qbDiv

5.5.1 Detailed Description

Definition at line 100 of file Apll.h.

5.5.2 Field Documentation

5.5.2.1 struct T_idtApllConfig::outputDiv_s::qaDiv_s qaDiv

5.5.2.2 struct T_idtApllConfig::outputDiv_s::qbDiv_s qbDiv

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

- common/include/Apll.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_idtApllConfig::preDiv_s Struct Reference

`#include <Apll.h>`

Data Fields

- T_idtApllDividerValue preDivConfig0
- T_idtApllDividerValue preDivConfig1

5.8.1 Detailed Description

Definition at line 90 of file Apll.h.

5.8.2 Field Documentation

5.8.2.1 T_idtApllDividerValue preDivConfig0

APLL pre-divider value for configuration 0

Definition at line 92 of file Apll.h.

5.8.2.2 T_idtApllDividerValue preDivConfig1

APLL pre-divider value for configuration 1

Definition at line 93 of file Apll.h.

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

- common/include/Apll.h

5.9 T_idtApllConfig::outputConfig_s::qaConfig_s Struct Reference

```
#include <Apll.h>
```

Data Fields

- T_idtApllOutputSigType outputSigType
- T_idtApllOutputEn enOutput

5.9.1 Detailed Description

Definition at line 120 of file Apll.h.

5.9.2 Field Documentation

5.9.2.1 T_idtApllOutputEn enOutput

APLL QA output enable

Definition at line 123 of file Apll.h.

5.9.2.2 T_idtApllOutputSigType outputSigType

APLL QA output signal type

Definition at line 122 of file Apll.h.

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

- common/include/Apll.h

5.10 T_idtApllConfig::outputDiv_s::qaDiv_s Struct Reference

`#include <Apll.h>`

Data Fields

- T_idtApllDividerValue outputDivConfig0
- T_idtApllDividerValue outputDivConfig1

5.10.1 Detailed Description

Definition at line 102 of file Apll.h.

5.10.2 Field Documentation

5.10.2.1 T_idtApllDividerValue outputDivConfig0

APLL QA output divider value for configuration 0

Definition at line 104 of file Apll.h.

5.10.2.2 T_idtApllDividerValue outputDivConfig1

APLL QB output divider value for configuration 1

Definition at line 105 of file Apll.h.

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

- common/include/Apll.h

5.11 T_idtApllConfig::outputConfig_s::qbConfig_s Struct Reference

`#include <Apll.h>`

Data Fields

- T_idtApllOutputSigType outputSigType
- T_idtApllOutputEn enOutput

5.11.1 Detailed Description

Definition at line 125 of file Apll.h.

5.11.2 Field Documentation

5.11.2.1 T_idtApllOutputEn enOutput

APLL QB output enable

Definition at line 128 of file Apll.h.

5.11.2.2 T_idtApllOutputSigType outputSigType

APLL QB output signal type

Definition at line 127 of file Apll.h.

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

- common/include/Apll.h

5.12 T_idtApllConfig::outputDiv_s::qbDiv_s Struct Reference

`#include <Apll.h>`

Data Fields

- T_idtApllDividerValue outputDivConfig0
- T_idtApllDividerValue outputDivConfig1

5.12.1 Detailed Description

Definition at line 107 of file Apll.h.

5.12.2 Field Documentation

5.12.2.1 T_idtApllDividerValue outputDivConfig0

APLL QB output divider value for configuration 0

Definition at line 109 of file Apll.h.

5.12.2.2 T_idtApllDividerValue outputDivConfig1

APLL QB output divider value for configuration 1

Definition at line 110 of file Apll.h.

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

- common/include/Apll.h

5.13 T_apllOutputFrequencyEntry Struct Reference

Structure used to translate APLL configuration and output port frequency.

Data Fields

- T_idtApllDividerValue apllDivVal_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_idtApllDividerValue apllDivVal_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_idtApllConfig Struct Reference

Structure containing APLL full configuration.

`#include <Apll.h>`

Data Structures

- struct fbDiv_s
- struct outputConfig_s
- struct outputDiv_s
- struct preDiv_s

Data Fields

- T_idtApllConfigSel apllCfg
- T_idtApllInputClkSrc inputClk
- struct T_idtApllConfig::preDiv_s preDiv
- struct T_idtApllConfig::fbDiv_s fbDiv
- struct T_idtApllConfig::outputDiv_s outputDiv
- T_idtApllFreqDoublerSel dbleSel
- T_idtApllMasterResetSync mrSync
- T_idtApllBypass byPass
- T_idtApllShutoff shutOff
- T_idtApllXtalId xtal
- struct T_idtApllConfig::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_idtApllConfigSel apllCfg

APLL stored register configuration selection

Definition at line 88 of file Apll.h.

5.14.2.2 T_idtApllBypass byPass

APLL bypass configuration

Definition at line 115 of file Apll.h.

5.14.2.3 T_idtApllFreqDoublerSel dbleSel

APLL frequency doubler in feedback path

Definition at line 113 of file Apll.h.

5.14.2.4 struct T_idtApllConfig::fbDiv_s fbDiv

5.14.2.5 T_idtApllInputClkSrc inputClk

APLL sourced clock selection

Definition at line 89 of file Apll.h.

5.14.2.6 T_idtApllMasterResetSync mrSync

APLL Master reset sync

Definition at line 114 of file Apll.h.

5.14.2.7 struct T_idtApllConfig::outputConfig_s outputConfig

5.14.2.8 struct T_idtApllConfig::outputDiv_s outputDiv

5.14.2.9 struct T_idtApllConfig::preDiv_s preDiv

5.14.2.10 T_idtApllShutoff shutOff

APLL shut off mode

Definition at line 116 of file Apll.h.

5.14.2.11 T_idtApllXtalId xtal

APLL crystal selection

Definition at line 117 of file Apll.h.

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

- common/include/Apll.h

5.15 T_idtApllConfigPerOutput Struct Reference

Structure containing APLL per output configuration.

`#include <Apll.h>`

Data Fields

- T_idtApllConfigSel apllCfg
- T_idtApllInputClkSrc inputClk
- T_idtApllDividerValue preDiv
- T_idtApllDividerValue fbDiv
- T_idtApllDividerValue outputDiv
- T_idtApllFreqDoublerSel dbleSel
- T_idtApllXtalId xtal
- T_idtApllOutputSigType sigType
- T_idtApllOutputEn enOutput

5.15.1 Detailed Description

Structure containing APLL per output configuration.

Definition at line 70 of file Apll.h.

5.15.2 Field Documentation

5.15.2.1 T_idtApllConfigSel apllCfg

APLL stored register configuration selection

Definition at line 72 of file Apll.h.

5.15.2.2 T_idtApllFreqDoublerSel dbleSel

APLL frequency doubler in feedback path

Definition at line 77 of file Apll.h.

5.15.2.3 T_idtApllOutputEn enOutput

APLL output enable

Definition at line 80 of file Apll.h.

5.15.2.4 T_idtApllDividerValue fbDiv

APLL feedback divider value

Definition at line 75 of file Apll.h.

5.15.2.5 T_idtApllInputClkSrc inputClk

APLL sourced clock selection

Definition at line 73 of file Apll.h.

5.15.2.6 T_idtApllDividerValue outputDiv

APLL output divider value

Definition at line 76 of file Apll.h.

5.15.2.7 T_idtApllDividerValue preDiv

APLL pre-divider value

Definition at line 74 of file Apll.h.

5.15.2.8 T_idtApllOutputSigType sigType

APLL output signal configuration

Definition at line 79 of file Apll.h.

5.15.2.9 T_idtApllXtalId xtal

APLL crystal selection

Definition at line 78 of file Apll.h.

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

- common/include/Apll.h

5.16 T_idtApllFreqMapping Struct Reference

Type definition for mapping output frequency to APLL configuration.

`#include <ApllFreqProfile.h>`

Data Fields

- T_idtApllInputFreqType apllInputFreq
- T_idtApllDividerValue apllPreDivider
- T_idtApllDividerValue apllFbDivider
- T_idtApllXtalType apllXtalFreq
- T_idtApllDividerValue apllOutputDivider
- T_idtApllOutputFreqType apllOutputFeq
- T_idtApllOutputSupportedType apllOutputs

5.16.1 Detailed Description

Type definition for mapping output frequency to APLL configuration.

Definition at line 100 of file ApllFreqProfile.h.

5.16.2 Field Documentation

5.16.2.1 T_idtApllDividerValue apllFbDivider

Feedback (oscillator) divider

Definition at line 104 of file ApllFreqProfile.h.

5.16.2.2 T_idtApllInputFreqType apllInputFreq

Input frequency

Definition at line 102 of file ApllFreqProfile.h.

5.16.2.3 T_idtApllDividerValue apllOutputDivider

Output divider

Definition at line 106 of file ApllFreqProfile.h.

5.16.2.4 T_idtApllOutputFreqType apllOutputFeq

Output frequency type

Definition at line 107 of file ApllFreqProfile.h.

5.16.2.5 T_idtApllOutputSupportedType apllOutputs

Output supported (A or B or both)

Definition at line 108 of file ApllFreqProfile.h.

5.16.2.6 T_idtApllDividerValue apllPreDivider

Pre- (input) divider

Definition at line 103 of file ApllFreqProfile.h.

5.16.2.7 T_idtApllXtalType apllXtalFreq

Oscillator type

Definition at line 105 of file ApllFreqProfile.h.

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

- apll/include/ApllFreqProfile.h

5.17 T_idtApllXtal Struct Reference

Structure containing APLL crystal id and frequency information.

`#include <ApllTypeDef.h>`

Data Fields

- T_idtApllXtalId apllXtalId
- T_idtApllXtalType apllXtalFreq

5.17.1 Detailed Description

Structure containing APLL crystal id and frequency information.

Definition at line 244 of file ApllTypeDef.h.

5.17.2 Field Documentation

5.17.2.1 T_idtApllXtalType apllXtalFreq

APLL crystal frequency

Definition at line 247 of file ApllTypeDef.h.

5.17.2.2 T_idtApllXtalId apllXtalId

APLL crystal ID

Definition at line 246 of file ApllTypeDef.h.

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

- apll/include/ApllTypeDef.h

5.18 T_idtApllXtalList Struct Reference

Wrapper structure containing APLL crystal id and frequency information.

`#include <access.h>`

Data Fields

- int xtal1Apll1
- T_idtApllXtalType xtal1Apll1Freq
- int xtal3Apll1
- T_idtApllXtalType xtal3Apll1Freq
- int xtal2Apll2
- T_idtApllXtalType xtal2Apll2Freq
- int xtal4Apll2
- T_idtApllXtalType xtal4Apll2Freq

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 xtal1Apll1

APLL1 XTAL1 flag

Definition at line 52 of file access.h.

5.18.2.2 T_idtApIIXtalType xtal1Apll1Freq

APLL1 XTAL1 Frequency

Definition at line 53 of file access.h.

5.18.2.3 int xtal2Apll2

APLL2 XTAL2 flag

Definition at line 56 of file access.h.

5.18.2.4 T_idtApIIXtalType xtal2Apll2Freq

APLL2 XTAL2 Frequency

Definition at line 57 of file access.h.

5.18.2.5 int xtal3Apll1

APLL1 XTAL3 flag

Definition at line 54 of file access.h.

5.18.2.6 T_idtApIIXtalType xtal3Apll1Freq

APLL1 XTAL3 Frequency

Definition at line 55 of file access.h.

5.18.2.7 int xtal4Apll2

APLL2 XTAL4 flag

Definition at line 58 of file access.h.

5.18.2.8 T_idtApIIXtalType xtal4Apll2Freq

APLL2 XTAL4 Frequency

Definition at line 59 of file access.h.

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

- 82V3910/sample/include/access.h

5.19 T_idtDrvAccess Struct Reference

Initialization structure detailing device access information for device driver.

`#include <Define.h>`

Data Fields

- T_idtAccessInitFunc initFunc_m
- T_idtAccessDeInitFunc deinitFunc_m
- T_idtReadFunc readFunc_m
- T_idtWriteFunc writeFunc_m
- void * userData_m

5.19.1 Detailed Description

Initialization structure detailing device access information for device driver.

Definition at line 193 of file Define.h.

5.19.2 Field Documentation

5.19.2.1 T_idtAccessDeInitFunc deinitFunc_m

Access deinitialization

Definition at line 196 of file Define.h.

5.19.2.2 T_idtAccessInitFunc initFunc_m

Access initialization

Definition at line 195 of file Define.h.

5.19.2.3 T_idtReadFunc readFunc_m

Read function

Definition at line 197 of file Define.h.

5.19.2.4 void* userData_m

Extra user data

Definition at line 199 of file Define.h.

5.19.2.5 T_idtWriteFunc writeFunc_m

Write function

Definition at line 198 of file Define.h.

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

- common/include/Define.h

5.20 T_idtDrvDeviceConfig Struct Reference

Device type/variant information.

`#include <Define.h>`

Data Fields

- T_osUint8 inputExt2IntXlate_ma [IDT_DRV_MAX_INPUT+1]
- T_osUint8 inSyncXlate_ma [2]
- T_osUint8 outputExt2IntXlate_ma [IDT_DRV_MAX_OUTPUT+1]
- T_idtOutputApllConfig outPutApllConfig [IDT_DRV_MAX_OUTPUT+1]
- T_osUint8 outSyntXlate_ma [2]
- T_osUint8 numberOfSonetGigEthPath_m
- T_osUint8 numberOfApll_m
- T_osUint8 populatedApll_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 maxNumXtalPerApll_m
- T_osUint8 supportedXtalId_ma [APLL_XTAL_MAX]
- T_osUint8 supportedXtalFreq_ma [APLL_XTAL_MAX][APLL_XTAL_FREQ_MAX]
- T_idtInputFreqValid inputFreqValidFunc_m
- T_idtOuputFreqValid 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 maxNumXtalPerApll_m

Maximum number of XTALs per APLL

Definition at line 234 of file Define.h.

5.20.2.6 T_osUint8 numberOfApll_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_idtOutputApllConfig outPutApllConfig[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 populatedApll_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 numberOfApllXtal_m
- T_idtApllXtal xtalList [IDT_DRV_MAX_APLL_XTAL]
- struct S_IDTPathConfiguration * currOutPathCfg_mp

- void * dpllI2CTransData_mp
- void * apllI2CTransData_mp
- T_idtI2CaddrTransFunc i2cAddrTransFunc_m
- T_osUint8 i2cAddr

5.21.1 Detailed Description

Device driver handler.

All device driver functions have driver handlers as an input parameter. This handler allows the device driver to support an unlimited number devices.

Definition at line 255 of file Define.h.

5.21.2 Field Documentation

5.21.2.1 void* apllI2CTransData_mp

Helper data for translating I2C address mapping to apll register space

Definition at line 265 of file Define.h.

5.21.2.2 struct S_IDTPathConfiguration* currOutPathCfg_mp

Current output path configuration - used by high level API code. Basic driver doesn't require this.

Definition at line 263 of file Define.h.

5.21.2.3 const T_idtDrvDeviceConfig* deviceConfig_m

Device configuration

Definition at line 260 of file Define.h.

5.21.2.4 T_idtDeviceType deviceType_m

Device Type

Definition at line 258 of file Define.h.

5.21.2.5 void* dpllI2CTransData_mp

Helper data for translating I2C address mapping to dpll register space

Definition at line 264 of file Define.h.

5.21.2.6 T_idtDrvAccess drvAccess_m

Device access functions

Definition at line 259 of file Define.h.

5.21.2.7 T_osUint8 i2cAddr

7-bit unshifted i2c slave address for PLL

Definition at line 267 of file Define.h.

5.21.2.8 T_idtI2CaddrTransFunc i2cAddrTransFunc_m

Function to translate I2C address

Definition at line 266 of file Define.h.

5.21.2.9 const char* name_m

Name of driver instance

Definition at line 257 of file Define.h.

5.21.2.10 T_osUint8 numberOfApllXtal_m

Number of APLL XTALs

Definition at line 261 of file Define.h.

5.21.2.11 T_idtApllXtal xtalList[IDT_DRV_MAX_APLL_XTAL]

List of APLL XTALs

Definition at line 262 of file Define.h.

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

- common/include/Define.h

5.22 T_IDTFreq2bfVal Struct Reference

Structure to contain translation information for frequency to bitfield value.

Data Fields

- T_osUint32 freq_m
- T_idtInputFrequencyBitfieldValue bfVal_m

5.22.1 Detailed Description

Structure to contain translation information for frequency to bitfield value.

Definition at line 50 of file inputFreq.c.

5.22.2 Field Documentation

5.22.2.1 T_idtInputFrequencyBitfieldValue bfVal_m

Bitfield value

Definition at line 52 of file inputFreq.c.

5.22.2.2 T_osUint32 freq_m

Frequency (in Hz)

Definition at line 51 of file inputFreq.c.

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

- common/hlapi/src/inputFreq.c

5.23 T_idtHfDivEntry Struct Reference

Structure to use as high frequency (HF) divider information.

Data Fields

- T_osUint8 divValue_m
- T_idtHighFreqencyDivisor 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_idtHighFreqencyDivisor 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_idtOutputApllConfig Struct Reference

Structure containing APLL inforatmion.

`#include <Define.h>`

Data Fields

- T_idtApllId apllInst
- T_osUint8 apllInput
- T_idtApllOutput apllOutput
- 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 apllInput

APLL input port number

Definition at line 208 of file Define.h.

5.25.2.2 T_idtApllId apllInst

APLL instanace ID

Definition at line 207 of file Define.h.

5.25.2.3 T_idtApllOutput apllOutput

APLL output port number

Definition at line 209 of file Define.h.

5.25.2.4 T_osUint8 extOutput

External output port number

Definition at line 210 of file Define.h.

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

- common/include/Define.h

5.26 T_IDTPathConfiguration Struct Reference

Structure containing output path configuration.

`#include <Dpll.h>`

Data Fields

- networkType_t networkType_m
- T_idtDpllOutputSelectorA T0SelA_m
- T_idtDpllOutputSelectorB T0SelB_m
- T_idtDpllOutputSelectorA T4SelA_m
- T_idtDpllOutputSelectorB T4SelB_m
- T_idtSonetGigEthOutput sonetGigEthSel_ma [SonetGigEthSel_NUMMAX]

5.26.1 Detailed Description

Structure containing output path configuration.

Definition at line 114 of file Dpll.h.

5.26.2 Field Documentation

5.26.2.1 networkType_t networkType_m

Network type

Definition at line 115 of file Dpll.h.

5.26.2.2 T_idtSonetGigEthOutput sonetGigEthSel_ma[SonetGigEthSel_NUMMAX]

Sonet/GigE Output Path Selector

Definition at line 122 of file Dpll.h.

5.26.2.3 T_idtDpllOutputSelectorA T0SelA_m

T0 Output Path Selector A

Definition at line 116 of file Dpll.h.

5.26.2.4 T_idtDpllOutputSelectorB T0SelB_m

T0 Output Path Selector B

Definition at line 117 of file Dpll.h.

5.26.2.5 T_idtDpllOutputSelectorA T4SelA_m

T4 Output Path Selector A

Definition at line 118 of file Dpll.h.

5.26.2.6 T_idtDpllOutputSelectorB T4SelB_m

T4 Output Path Selector B

Definition at line 119 of file Dpll.h.

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

- common/include/Dpll.h

5.27 T_idtSonetGigEthBitfield2PathConfig Struct Reference

Structure Translate from Sonet/GigE bitfield to path configuration.

Data Fields

- T_idtDpllInstance inputDpll_m
- T_idtSonetGigEthOutput outputFreq_m

5.27.1 Detailed Description

Structure Translate from Sonet/GigE bitfield to path configuration.

Definition at line 51 of file DpllCnfg.c.

5.27.2 Field Documentation

5.27.2.1 T_idtDpllInstance inputDpll_m

input DPLL type

Definition at line 53 of file DpllCnfg.c.

5.27.2.2 T_idtSonetGigEthOutput outputFreq_m

output frequency

Definition at line 54 of file DpllCnfg.c.

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

- common/src/DpllCnfg.c

5.28 T_SampleConfigEntry Struct Reference

Sample configuration information.

Data Fields

- T_SampleConfigFunc configFunction_m
- T_SampleConfigDescrFunc configDescription_m

5.28.1 Detailed Description

Sample configuration information.

Definition at line 75 of file main.c.

5.28.2 Field Documentation

5.28.2.1 T_SampleConfigDescrFunc configDescription_m

Function to output description

Definition at line 77 of file main.c.

5.28.2.2 T_SampleConfigFunc configFunction_m

Function to configure

Definition at line 76 of file main.c.

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

- 82V3910/sample/src/main.c

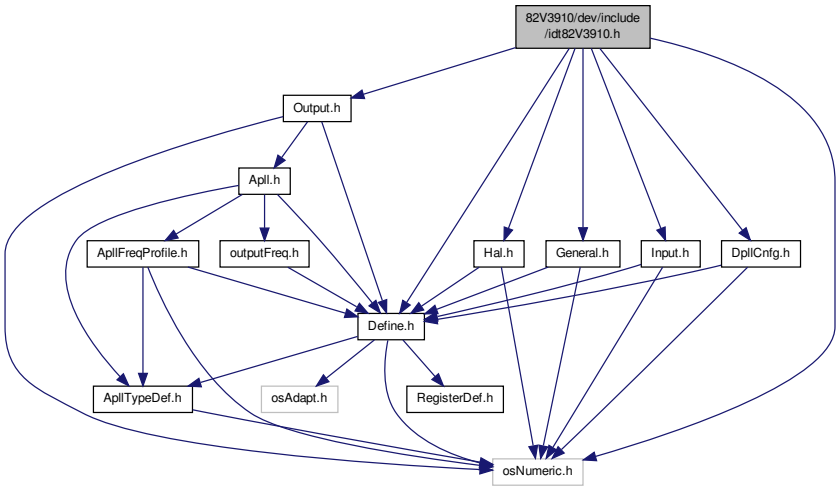
Chapter 6

File Documentation

6.1 82V3910/dev/include/idt82V3910.h File Reference

```
#include "General.h"
#include "Define.h"
#include "Input.h"
#include "Output.h"
#include "DpllCnfg.h"
#include "Hal.h"
#include "osNumeric.h"
```

Include dependency graph for idt82V3910.h:



Functions

- void IDTGeneral_InitDevice (T_idtDrvHdlr hdlr)

  Initialize the device to its power on defaults.

- T_idtDrvHdlr IDTGeneral_InitDriver (const char *name, T_idtDrvAccess drvAccess, T_idtDeviceType deviceType, T_idtApllXtal *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 intialize device registers to know states. For "warm" restarts skip this function.

Parameters

hdlr	Device driver handle

Definition at line 101 of file 82V3910.c.

6.1.1.2 T_idtDrvHdlr IDTGeneral_InitDriver (const char * *name*, T_idtDrvAccess *drvAccess*, T_idtDeviceType *deviceType*, T_idtApllXtal * *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

name	Name of device (ex. "myDev")
drvAccess	Device access data.

See Also

drvAccess_t structure information

Parameters

deviceType	What supported device
xtalList	A list of crystal(s) connected to APLL(s), set this parameter to NULL if there is no XTAL connected to APLL
numberOfXtal	Number of crystal(s) in the xtalList, set this parameter to 0 if xtaList is NULL
i2cAddr	7-bit unmodified i2c slave address
useHighLevelApi	0-Don't use high level API, 1-high level API enabled.

Returns

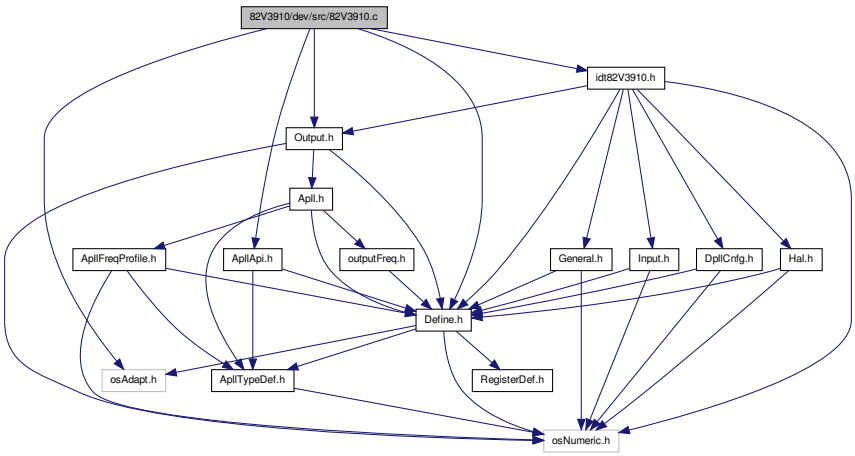
Device driver handler (used in all APIs)

Definition at line 167 of file 82V3910.c.

6.2 82V3910/dev/src/82V3910.c File Reference

```
#include "osAdapt.h"
#include "idt82V3910.h"
#include "Define.h"
#include "ApllApi.h"
#include "Output.h"
```

Include dependency graph for 82V3910.c:



Functions

- void IDTGeneral_InitDevice (T_idtDrvHdlr hdlr)

  Initialize the device to its power on defaults.
- T_idtDrvHdlr IDTGeneral_InitDriver (const char ∗name, T_idtDrvAccess drvAccess, T_idtDeviceType deviceType, T_idtApllXtal ∗xtalList, T_osUint8 numberOfXtal, T_osUint8 i2cAddr, int useHighLevelApi)

  Initialize device driver.

Variables

- const T_idtDrvDeviceConfig idt82V3910Config

  Specific device profile.

6.2.1 Function Documentation

6.2.1.1 void IDTGeneral_InitDevice (T_idtDrvHdlr *hdlr*)

Initialize the device to its power on defaults.

This function is optionally called. This function is required only after power on to intialize device registers to know states. For "warm" restarts skip this function.

Parameters

hdlr	Device driver handle

Definition at line 101 of file 82V3910.c.

6.2.1.2 T_idtDrvHdlr IDTGeneral_InitDriver (const char ∗ *name*, T_idtDrvAccess *drvAccess*, T_idtDeviceType *deviceType*, T_idtApllXtal ∗ *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

name	Name of device (ex. "myDev")
drvAccess	Device access data.

See Also

drvAccess_t structure information

Parameters

deviceType	What supported device
xtalList	A list of crystal(s) connected to APLL(s), set this parameter to NULL if there is no XTAL connected to APLL
numberOfXtal	Number of crystal(s) in the xtalList, set this parameter to 0 if xtaList is NULL
i2cAddr	7-bit unmodified i2c slave address
useHighLevelApi	0-Don't use high level API, 1-high level API enabled.

Returns

Device driver handler (used in all APIs)

Definition at line 167 of file 82V3910.c.

6.2.2 Variable Documentation

6.2.2.1 const T_idtDrvDeviceConfig idt82V3910Config

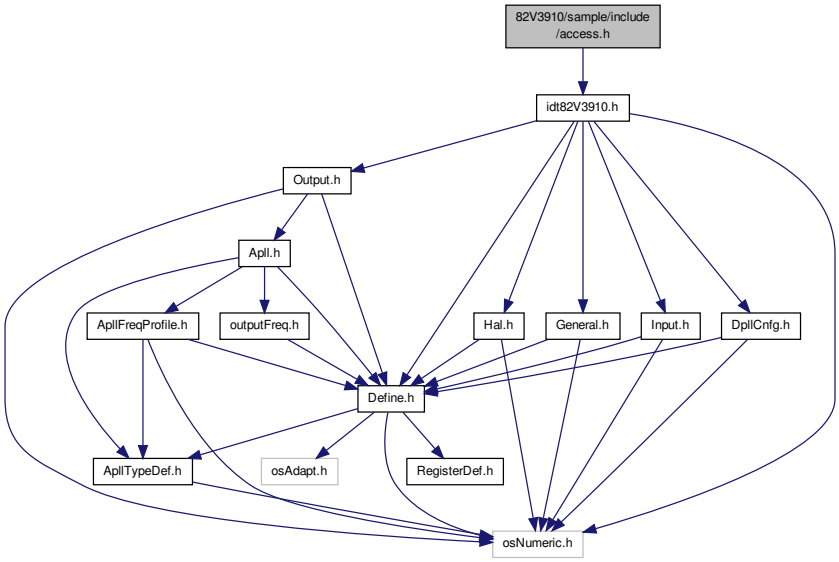
Specific device profile.

Definition at line 50 of file 82V3910.c.

6.3 82V3910/sample/include/access.h File Reference

```
#include "idt82V3910.h"
```

Include dependency graph for access.h:



Data Structures

- struct T_idtApllXtalList

  Wrapper structure containing APLL crystal id and frequency information.

Functions

- T_idtDrvHdlr IDTSample_InitDriverI2c (T_idtApllXtalList *vcxoList, T_osUint8 coreI2CAddr)

  IDTSample_InitDriverI2c Utility function to initialize device driver for I2C access.
- T_idtDrvHdlr IDTSample_InitDriverSimulator (T_idtApllXtalList *vcxoList, T_osUint8 coreI2CAddr)

  Utility function to initialize device driver for simulator access.
- void IDTSample_ResetXtalList (T_idtApllXtalList *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_idtApllXtalList * *xtalList*, T_osUint8 *coreI2CAddr*)

IDTSample_InitDriverI2c Utility function to initialize device driver for I2C access.

This function is used for TCL extension library

Parameters

xtalList	List of XTAL
coreI2CAddr	Device I2C address

Returns

Device driver handle

Definition at line 161 of file access.c.

6.3.1.3 T_idtDrvHdlr IDTSample_InitDriverSimulator (T_idtApllXtalList * *xtalList*, T_osUint8 *coreI2CAddr*)

Utility function to initialize device driver for simulator access.

This function is used for TCL extension library

Parameters

xtalList	List of XTAL
coreI2CAddr	Device I2C address

Returns

Device driver handle

Definition at line 229 of file access.c.

6.3.1.4 void IDTSample_ResetXtalList (T_idtApllXtalList * *xtalList*)

Utility function to reset APLL XTAL list to unconfigured value.

This function is used for TCL extension library

Parameters

xtalList	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

level	Log verbosity level

Definition at line 316 of file access.c.

6.4 82V3910/sample/include/loadstore.h File Reference

Functions

- void IDTSample_SaveRgfFile (T_idtDrvHdlr hdlr, char *filename)

  Function to output register contents of a device to standard IO or to a file.
- void IDTSample_LoadRgfFile (T_idtDrvHdlr hdlr, char *filename)

  Function to read in a file and program device registers.

6.4.1 Function Documentation

6.4.1.1 void IDTSample_LoadRgfFile (T_idtDrvHdlr *hdlr*, char * *filename*)

Function to read in a file and program device registers.

Parameters

hdlr	Device driver handler
filename	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

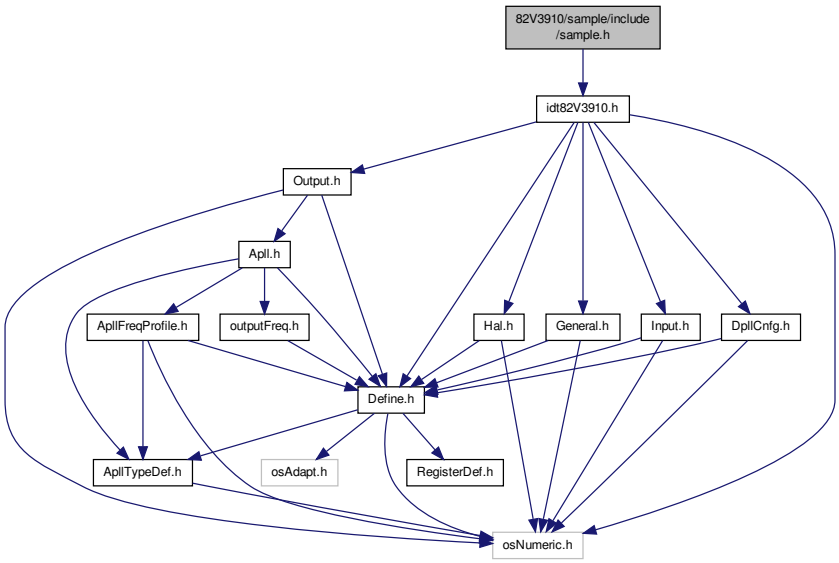
hdlr	Device driver handler
filename	Output file name, NULL for standard IO

Definition at line 94 of file loadstore.c.

6.5 82V3910/sample/include/sample.h File Reference

```
#include "idt82V3910.h"
```

Include dependency graph for sample.h:



Functions

- void IDTSample_ConfigureSampleConfig1 (T_idtDrvHdlr hdlr)

  Sample configuration 1.
- void IDTSample_ConfigureSampleConfig2 (T_idtDrvHdlr hdlr)

  Sample configuration 2.
- void IDTSample_ConfigureSampleConfig3 (T_idtDrvHdlr hdlr)

  Sample configuration 3.
- void IDTSample_ConfigureSampleConfig4 (T_idtDrvHdlr hdlr)

  Sample configuration 4.
- void IDTSample_ConfigureSampleConfig5 (T_idtDrvHdlr hdlr)

  Sample configuration 5.
- void IDTSample_Configure1Pps (T_idtDrvHdlr hdlr)

  Sample configuration 1PPS.
- void IDTSample_Configure1PpsHoldover (T_idtDrvHdlr hdlr)

  Sample configuration for 1PPS holdover issue.
- void IDTSample_ConfigureDcoMode (T_idtDrvHdlr hdlr)

  Function to configure device for DCO mode.
- void IDTSample_SetCombinedDcoOffset (T_idtDrvHdlr hdlr, long pptOffset)

  Function to set combined DCO offset value. It uses both DCO (holdover) and nominal offsets.
- long IDTSample_GetCombinedDcoOffset (T_idtDrvHdlr hdlr)

  Function to get combined DCO offset value. It uses both DCO (holdover) and nominal offsets.

6.5.1 Function Documentation

6.5.1.1 void IDTSample_Configure1Pps (T_idtDrvHdlr *hdlr*)

Sample configuration 1PPS.

```
IN3 (1PPS) -> T0 -> OUT3 (1PPS)
```

Parameters

hdlr	Device driver handler

Definition at line 420 of file sample.c.

6.5.1.2 void IDTSample_Configure1PpsHoldover (T_idtDrvHdlr *hdlr*)

Sample configuration for 1PPS holdover issue.

Parameters

hdlr	Device driver handler

Definition at line 449 of file sample.c.

6.5.1.3 void IDTSample_ConfigureDcoMode (T_idtDrvHdlr *hdlr*)

Function to configure device for DCO mode.

Parameters

hdlr	Device driver handler

Definition at line 480 of file sample.c.

6.5.1.4 void IDTSample_ConfigureSampleConfig1 (T_idtDrvHdlr *hdlr*)

Sample configuration 1.

```
IN3 (25 MHz) -> T0 +-> OUT1 (25 MHz)
                   +-> OUT6 (156.25 MHz)
```

Parameters

hdlr	Device driver handler

Definition at line 46 of file sample.c.

6.5.1.5 void IDTSample_ConfigureSampleConfig2 (T_idtDrvHdlr *hdlr*)

Sample configuration 2.

```
IN3 (25 MHz) -> T0 +-> OUT1 (25 MHz)
                   +-> OUT3 (8 kHz)
                   +-> OUT5 (125 MHz)
```

Parameters

hdlr	Device driver handler

Definition at line 98 of file sample.c.

6.5.1.6 void IDTSample_ConfigureSampleConfig3 (T_idtDrvHdlr *hdlr*)

Sample configuration 3.

```
IN3 (25 MHz) -> T0 +-> OUT1 (25 MHz)
                   +-> OUT3 (10 MHz)
                   +-> OUT5 (1 PPS)
```

Parameters

hdlr	Device driver handler

Definition at line 154 of file sample.c.

6.5.1.7 void IDTSample_ConfigureSampleConfig4 (T_idtDrvHdlr *hdlr*)

Sample configuration 4.

```
Inputs: IN3 (10MHz), IN4 (1.544MHz), IN5 (25MHz), IN6 (8KHz), IN7 (8KHz),
        IN8 (156.25MHz), IN9 (156.25MHz), IN10 (25MHz), IN11 (25MHz)
Outputs: OUT1 (1.544MHz), OUT3 (10MHz), OUT4 (25MHz), OUT7 (156.25MHz),
         OUT10 (156.25MHz), OUT11 (156.25MHz)
T0 Profile: G.8262 Option 2
T4 Profile: Default
XTAL1/2: 25MHz
```

Parameters

hdlr	Device driver handler

Definition at line 226 of file sample.c.

6.5.1.8 void IDTSample_ConfigureSampleConfig5 (T_idtDrvHdlr *hdlr*)

Sample configuration 5.

```
Carbon copy of IDTSample_ConfigureSampleConfig4,
except there is no input and DPLL is in free-run
mode (default).
```

Parameters

hdlr	Device driver handler

Definition at line 339 of file sample.c.

6.5.1.9 long IDTSample_GetCombinedDcoOffset (T_idtDrvHdlr *hdlr*)

Function to get combined DCO offset value. It uses both DCO (holdover) and nominal offsets.

Parameters

hdlr	Device driver handler

Returns

Parts per trillion offset

Definition at line 542 of file sample.c.

6.5.1.10 void IDTSample_SetCombinedDcoOffset (T_idtDrvHdlr *hdlr*, long *pptOffset*)

Function to set combined DCO offset value. It uses both DCO (holdover) and nominal offsets.

Parameters

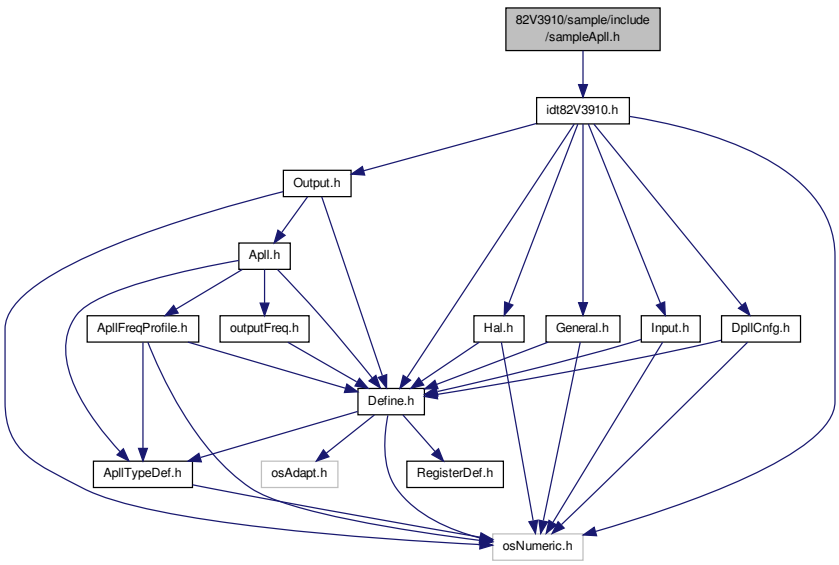
hdlr	Device driver handler
pptOffset	Parts per trillion offset

Definition at line 492 of file sample.c.

6.6 82V3910/sample/include/sampleApll.h File Reference

```
#include "idt82V3910.h"
```

Include dependency graph for sampleApll.h:



Functions

- void IDTSample_ConfigureSampleConfigOutput6_156_dot_25MHz (T_idtDrvHdlr hdlr)

  Sample configuration for 156.25MHz output at port 6.
- void IDTSample_ConfigureSampleConfigureOutput7_77_dot_76MHz (T_idtDrvHdlr hdlr)

  Sample configuration for 77.76 MHz output at port 7.

- void IDTSample_ConfigureSampleConfigureOutput10_25_MHz (T_idtDrvHdlr hdlr)

  Sample configuration for 25MHz output at port 10.
- void IDTSample_SyncE_WAN (T_idtDrvHdlr hdlr)

  Sample configuration for SyncE WAN application.
- void IDTSample_Sonet (T_idtDrvHdlr hdlr)

  Sample configuration for SONET application.
- void IDTSample_DualMode (T_idtDrvHdlr hdlr)

  Sample configuration for dual mode application.
- void IDTSample_SyncE_Sonet (T_idtDrvHdlr hdlr)

  Sample configuration for SyncE SONET application.
- void IDTSample_SyncE_LAN (T_idtDrvHdlr hdlr)

  Sample configuration for SyncE LAN application.
- void IDTSample_GPS_1PPS (T_idtDrvHdlr hdlr)

  Sample configuration for GPS 1PPS application.
- void IDTSample_GPS (T_idtDrvHdlr hdlr)

  Sample configuration for GPS application.

6.6.1 Function Documentation

6.6.1.1 void IDTSample_ConfigureSampleConfigOutput6_156_dot_25MHz (T_idtDrvHdlr *hdlr*)

Sample configuration for 156.25MHz output at port 6.

```
IN3 (25 MHz) -> T0 +-> OUT6 (Ethernet, 156.25 MHz)
```

Parameters

hdlr	Device driver handler

Definition at line 47 of file sampleApll.c.

6.6.1.2 void IDTSample_ConfigureSampleConfigureOutput10_25_MHz (T_idtDrvHdlr *hdlr*)

Sample configuration for 25MHz output at port 10.

```
IN3 (25 MHz) -> T0 +-> OUT10 (Ethernet, 25 MHz)
```

Parameters

hdlr	Device driver handler

Definition at line 125 of file sampleApll.c.

6.6.1.3 void IDTSample_ConfigureSampleConfigureOutput7_77_dot_76MHz (T_idtDrvHdlr *hdlr*)

Sample configuration for 77.76 MHz output at port 7.

```
IN3 (25 MHz) -> T0 +-> OUT7 (SONET, 77.76 MHz)
```

Parameters

hdlr	Device driver handler

Definition at line 86 of file sampleApll.c.

6.6.1.4 void IDTSample_DualMode (T_idtDrvHdlr *hdlr*)

Sample configuration for dual mode application.

```
Inputs: IN3 (19.44MHz), IN5 (155.52MHz), IN14 (1.544MHz)
T0 Profile: ST3 SONET
T4 Profile: default
Select XTAL3: done automatically by IDTHlApi_ConfigureOutput function
              based on the output frequency. To manually select XTAL,
              call SetJaXtalId function.
Outputs: OUT1 (19.44MHz), OUT6 (155.52MHz), OUT7 (77.76MHz), OUT9 (1.544MHz)
T0: IN3, IN5, IN14 -> OUT1, OUT6, OUT7 (IN14 is highest priority)
T4: IN3, IN5 -> OUT9
APLL1 XTAL1 at 25MHz & XTAL3 at 24.8832MHz
```

Parameters

hdlr	Device driver handler

Definition at line 410 of file sampleApll.c.

6.6.1.5 void IDTSample_GPS (T_idtDrvHdlr *hdlr*)

Sample configuration for GPS application.

```
Inputs: IN3 (10MHz), IN4 (10MHz), EX_SYNC1 (1PPS), EX_SYNC2 (1PPS)
T0 Profile: BITS / SSU (Wideband) - default
Outputs: OUT1 (25MHz), OUT2 (19.44MHz), OUT3 (10MHz), OUT4 (1PPS),
         OUT6 (25MHz), OUT7 (156.25MHz), OUT9 (1.544MHz), OUT10 (155.52MHz),
         OUT11 (77.76MHz), FRSYNC_8K_1PPS (1PPS),  MFRSYNC_2K_1PPS (1PPS)
APLL1 XTAL1 =  25MHz & APLL2 XTAL2 = 24.8832MHz
```

Parameters

hdlr	Device driver handler

Definition at line 1013 of file sampleApll.c.

6.6.1.6 void IDTSample_GPS_1PPS (T_idtDrvHdlr *hdlr*)

Sample configuration for GPS 1PPS application.

```
Inputs: IN3 (1PPS), IN5 (1PPS)
T0 Profile: GPS (15mHz)
Outputs: OUT1 (25MHz), OUT2 (19.44MHz), OUT3 (10MHz), OUT4 (1PPS),
         OUT6 (25MHz), OUT7 (156.25MHz), OUT9 (2.048MHz),
         OUT10 (155.52MHz), OUT11 (77.76MHz)
APLL1 XTAL1 at 25MHz & APLL2 XTAL2 at 24.8832MHz
```

Parameters

hdlr	Device driver handler

Definition at line 835 of file sampleApll.c.

6.6.1.7 void IDTSample_Sonet (T_idtDrvHdlr *hdlr*)

Sample configuration for SONET application.

```
Inputs: IN3 (19.44MHz), IN5 (155.52MHz), IN14 (1.544MHz)
T0 Profile: ST3 SONET
T4 Profile: default
Outputs: OUT1 (19.44MHz), OUT9 (1.544MHz), OUT10 (155.52MHz)
T0: IN3, IN5, IN14 -> OUT1, OUT10 (IN14 is highest priority)
T4: IN3, IN5 -> OUT9
APLL2 XTAL2 at 24.8832MHz
```

Parameters

hdlr	Device driver handler

Definition at line 294 of file sampleApll.c.

6.6.1.8 void IDTSample_SyncE_LAN (T_idtDrvHdlr *hdlr*)

Sample configuration for SyncE LAN application.

```
Inputs: IN3 (25MHz), IN14 (2.048MHz)
T0 Profile: G.8262 Option 2
T4 Profile: BITS / SSU (Wideband)
Outputs: OUT1 (25MHz), OUT10 (25.78125), OUT11 (161.1328125MHz), OUT9 (2.048MHz)
T0: IN3, IN14 -> OUT1, OUT10, OUT11 (IN14 is highest priority)
T4: IN3 -> OUT9
APLL2 XTAL4 at 25.78125MHz
```

Parameters

hdlr	Device driver handler

Definition at line 720 of file sampleApll.c.

6.6.1.9 void IDTSample_SyncE_Sonet (T_idtDrvHdlr *hdlr*)

Sample configuration for SyncE SONET application.

```
Inputs: IN3 (25MHz) , IN5 (156.25MHz), IN6 (155.52MHz), IN7 (19.44MHz), IN14 (1.544MHz)
T0 Profile: G.8262 Option 2
T4 Profile: BITS / SSU (Wideband) - default
Outputs: OUT1 (25MHz), OUT2 (19.44MHz), OUT6 (25MHz), OUT7 (156.25MHz),
         OUT9 (1.544MHz), OUT10 (155.52MHz), OUT11 (77.76MHz)
T0: IN3, IN5, IN6, IN7, IN14 -> OUT1, OUT2, OUT6, OUT7, OUT10, OUT11 (IN14 is highest priority)
T4: IN3, IN5, IN6, IN7 -> OUT9
APLL1 XTAL1 at 25MHz & APLL2 XTAL2 at 24.8832MHz
```

Parameters

hdlr	Device driver handler

Definition at line 534 of file sampleApll.c.

6.6.1.10 void IDTSample_SyncE_WAN (T_idtDrvHdlr *hdlr*)

Sample configuration for SyncE WAN application.

```
Inputs: IN3 (25MHz), IN5 (156.25MHz), IN14 (1.544MHz)
T0 Profile: G.8262 Option 2
T4 Profile: Default
Outputs: OUT1 (25MHz), OUT6 (25MHz), OUT7 (156.25MHz), OUT9 (1.544MHz)
T0: IN3, IN5, IN14 -> OUT1, OUT10, OUT11 (IN14 has highest priority)
T4: IN3, IN5 -> OUT9
APLL1 XTAL1 at 25MHz
```

Parameters

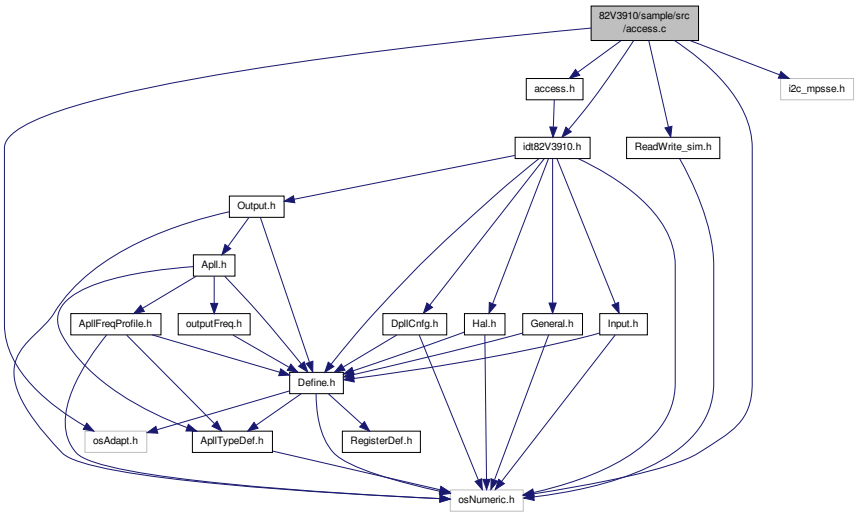
hdlr	Device driver handler

Definition at line 170 of file sampleApll.c.

6.7 82V3910/sample/src/access.c File Reference

```
#include "osNumeric.h"
#include "osAdapt.h"
#include "idt82V3910.h"
#include "access.h"
#include "ReadWrite_sim.h"
#include "i2c_mpsse.h"
```

Include dependency graph for access.c:



Functions

- void Init_SimWrapper (void ∗usrData)

  Function to wrap both common and APLL HAL initialization functions.
- void DeInit_SimWrapper (void ∗usrData)

  Function to wrap both common and APLL HAL deinitialization functions.
- T_osUint8 ReadByte_SimWrapper (void ∗usrData, T_osUint8 devAddr, T_osUint8 nAddr)

  Function to wrap both common and APLL HAL read register functions.
- void WriteByte_SimWrapper (void ∗usrData, T_osUint8 devAddr, T_osUint8 nAddr, T_osUint8 nData)

  Function to wrap both common and APLL HAL write register functions.
- T_idtDrvHdlr IDTSample_InitDriverI2c (T_idtApllXtalList ∗xtalList, T_osUint8 coreI2CAddr)

IDTSample_InitDriverI2c Utility function to initialize device driver for I2C access.

- T_idtDrvHdlr IDTSample_InitDriverSimulator (T_idtApllXtalList *xtalList, T_osUint8 coreI2CAddr)

  Utility function to initialize device driver for simulator access.
- void IDTSample_ResetXtalList (T_idtApllXtalList *xtalList)

  Utility function to reset APLL XTAL list to unconfigured value.
- void IDTSample_SetLogVerbosity (int level)

  Utility function to change log verbosity level access.
- int IDTSample_GetLogVerbosity (void)

  Utility function to get current log verbosity level.

Variables

- const T_idtDrvAccess I2cAccessMethods
- const T_idtDrvAccess SimAccessMethods

6.7.1 Function Documentation

6.7.1.1 void DeInit_SimWrapper (void * *usrData*)

Function to wrap both common and APLL HAL deinitialization functions.

Parameters

usrData	User data optionally used for de-initialization

Definition at line 116 of file access.c.

6.7.1.2 int IDTSample_GetLogVerbosity (void)

Utility function to get current log verbosity level.

This function is used for TCL extension library

Definition at line 326 of file access.c.

6.7.1.3 T_idtDrvHdlr IDTSample_InitDriverI2c (T_idtApllXtalList * *xtalList*, T_osUint8 *coreI2CAddr*)

IDTSample_InitDriverI2c Utility function to initialize device driver for I2C access.

This function is used for TCL extension library

Parameters

xtalList	List of XTAL
coreI2CAddr	Device I2C address

Returns

Device driver handle

Definition at line 161 of file access.c.

6.7.1.4 T_idtDrvHdlr IDTSample_InitDriverSimulator (T_idtApllXtalList * *xtalList*, T_osUint8 *coreI2CAddr*)

Utility function to initialize device driver for simulator access.

This function is used for TCL extension library

Parameters

xtalList	List of XTAL
coreI2CAddr	Device I2C address

Returns

Device driver handle

Definition at line 229 of file access.c.

6.7.1.5 void IDTSample_ResetXtalList (T_idtApllXtalList * *xtalList*)

Utility function to reset APLL XTAL list to unconfigured value.

This function is used for TCL extension library

Parameters

xtalList	List of XTAL

Definition at line 296 of file access.c.

6.7.1.6 void IDTSample_SetLogVerbosity (int *level*)

Utility function to change log verbosity level access.

This function is used for TCL extension library

Parameters

level	Log verbosity level

Definition at line 316 of file access.c.

6.7.1.7 void Init_SimWrapper (void * *usrData*)

Function to wrap both common and APLL HAL initialization functions.

Parameters

usrData	User data optionally used for initialization.

Definition at line 106 of file access.c.

6.7.1.8 T_osUint8 ReadByte_SimWrapper (void * *usrData,* T_osUint8 *devAddr,* T_osUint8 *nAddr*)

Function to wrap both common and APLL HAL read register functions.

Parameters

usrData	User data optionally used for reading registers
devAddr	Device Address (I2C addr)
nAddr	Register offset

Returns

Read register

Definition at line 129 of file access.c.

6.7.1.9 void WriteByte_SimWrapper (void * *usrData*, T_osUint8 *devAddr*, T_osUint8 *nAddr*, T_osUint8 *nData*)

Function to wrap both common and APLL HAL write register functions.

Parameters

usrData	User data optionally used for writing registers
devAddr	Device Address (I2C addr)
nAddr	Register offset
nData	Register value to write

Definition at line 144 of file access.c.

6.7.2 Variable Documentation

6.7.2.1 const T_idtDrvAccess I2cAccessMethods

Initial value:

```
= {
    i2c_init,
    i2c_deinit,
    i2c_read_byte,
    i2c_write_byte,
    NULL
}
```

Definition at line 79 of file access.c.

6.7.2.2 const T_idtDrvAccess SimAccessMethods

Initial value:

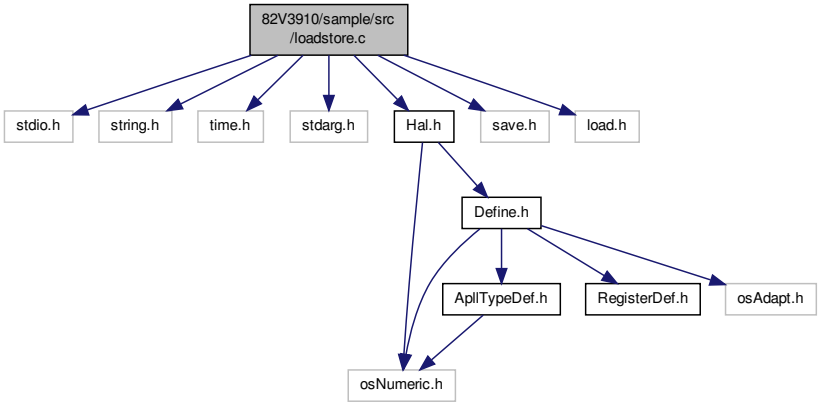
```
= {
    Init_SimWrapper,
    DeInit_SimWrapper,
    ReadByte_SimWrapper,
    WriteByte_SimWrapper,
    NULL
}
```

Definition at line 89 of file access.c.

6.8 82V3910/sample/src/loadstore.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <time.h>
#include <stdarg.h>
#include "Hal.h"
#include "save.h"
#include "load.h"
```

Include dependency graph for loadstore.c:



Functions

- void IDTSample_SaveRgfFile (T_idtDrvHdlr hdlr, char ∗filename)

  Function to output register contents of a device to standard IO or to a file.
- void IDTSample_LoadRgfFile (T_idtDrvHdlr hdlr, char ∗filename)

  Function to read in a file and program device registers.

6.8.1 Function Documentation

6.8.1.1 void IDTSample_LoadRgfFile (T_idtDrvHdlr *hdlr,* char ∗ *filename*)

Function to read in a file and program device registers.

Parameters

hdlr	Device driver handler
filename	Input file name to load register settings from.

Definition at line 124 of file loadstore.c.

6.8.1.2 void IDTSample_SaveRgfFile (T_idtDrvHdlr *hdlr,* char ∗ *filename*)

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

Parameters

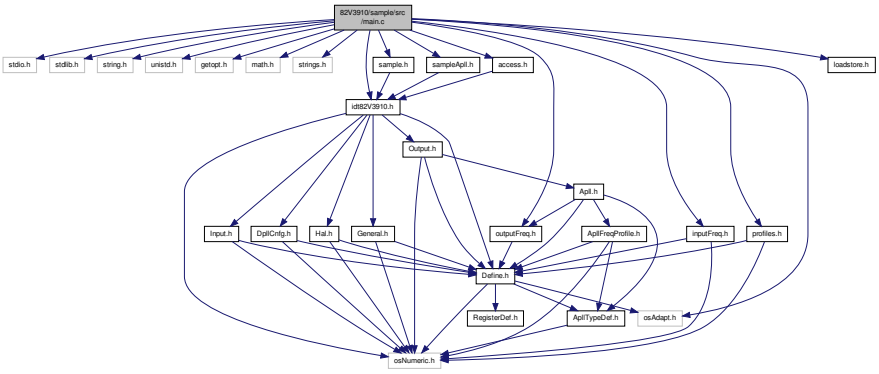
hdlr	Device driver handler
filename	Output file name, NULL for standard IO

Definition at line 94 of file loadstore.c.

6.9 82V3910/sample/src/main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <getopt.h>
#include <math.h>
#include <strings.h>
#include "idt82V3910.h"
#include "osAdapt.h"
#include "sample.h"
#include "sampleApll.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 ∗ IDTHlApi_Frequency2String_c [OutFreq_MAX]

6.9.1 Macro Definition Documentation

6.9.1.1 #define MODE_DpllProfile (1<<4)

Provision DPLL profile mode

Definition at line 111 of file main.c.

6.9.1.2 #define MODE_Input (1<<2)

Provision input port mode

Definition at line 105 of file main.c.

6.9.1.3 #define MODE_Load (1<<0)

Load from file mode

Definition at line 99 of file main.c.

6.9.1.4 #define MODE_None (0)

No-op mode

Definition at line 96 of file main.c.

6.9.1.5 #define MODE_Output (1<<3)

Provision output port mode

Definition at line 108 of file main.c.

6.9.1.6 #define MODE_Read (1<<6)

Read register mode

Definition at line 117 of file main.c.

6.9.1.7 #define MODE_Sample (1<<5)

Sample configuration mode

Definition at line 114 of file main.c.

6.9.1.8 #define MODE_Save (1<<1)

Save to file mode

Definition at line 102 of file main.c.

6.9.1.9 #define MODE_Write (1<<7)

Read register mode

Definition at line 120 of file main.c.

6.9.1.10 #define no_argument 0

Definition at line 155 of file main.c.

6.9.1.11 #define optional_argument 2

Definition at line 157 of file main.c.

6.9.1.12 #define required_argument 1

Definition at line 156 of file main.c.

6.9.1.13 #define SAMPLE_MAX_FILENAME 50

Max filename size

Definition at line 123 of file main.c.

6.9.2 Typedef Documentation

6.9.2.1 typedef void(* T_DpllProfileFunc)(T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

Function typedef for DPLL profile provisioning.

Definition at line 59 of file main.c.

6.9.2.2 typedef void(* T_SampleConfigDescrFunc)(void)

Function typedef for sample configuration descriptions.

Definition at line 70 of file main.c.

6.9.2.3 typedef void(* T_SampleConfigFunc)(T_idtDrvHdlr hdlr)

Function typedef for sample configurations.

Definition at line 65 of file main.c.

6.9.3 Enumeration Type Documentation

6.9.3.1 enum T_DpllProfile

List of possible DPLL profiles.

Enumerator

DpllProfile_GR253 GR-253 (SONET) Stratum 3 DPLL Profile

DpllProfile_GR1244 GR-1244 Stratum 3 DPLL Profile

DpllProfile_SMC SMC DPLL Profile

DpllProfile_G8262Option1 G-8262 Option 1 DPLL Profile

DpllProfile_G8262Option2 G-8262 Option 2 DPLL Profile

DpllProfile_G813Option1 G-813 Option 1 DPLL Profile

DpllProfile_G813Option2 G-813 Option 2 DPLL Profile

DpllProfile_1PPS 1 PPS DPLL Profile

DpllProfile_MAX

Definition at line 83 of file main.c.

6.9.4 Function Documentation

6.9.4.1 void IDTSample_DumpRegistersRgf (char * *filename*)

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

Definition at line 641 of file main.c.

6.9.4.3 void printHelp (void)

Definition at line 257 of file main.c.

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

Definition at line 441 of file main.c.

6.9.5 Variable Documentation

6.9.5.1 globalArgs_t globalArgs

Global command line argument variable

Definition at line 167 of file main.c.

6.9.5.2 const char* IDTHlApi_Frequency2String_c[OutFreq_MAX]

Definition at line 77 of file outputFreqString.c.

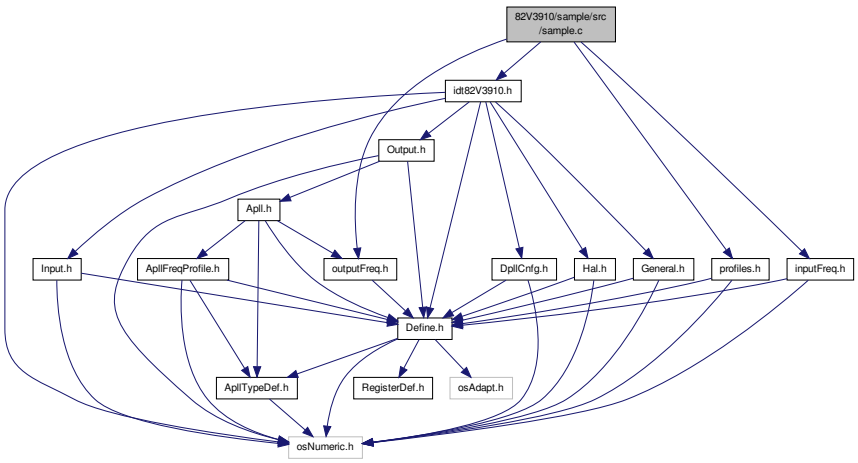
6.9.5.3 int simDebugFlag

Definition at line 48 of file ReadWrite_sim.c.

6.10 82V3910/sample/src/sample.c File Reference

```
#include "idt82V3910.h"
#include "profiles.h"
#include "inputFreq.h"
#include "outputFreq.h"
```

Include dependency graph for sample.c:



Functions

- void IDTSample_ConfigureSampleConfig1 (T_idtDrvHdlr hdlr)

  Sample configuration 1.
- void IDTSample_ConfigureSampleConfig2 (T_idtDrvHdlr hdlr)

  Sample configuration 2.

- void IDTSample_ConfigureSampleConfig3 (T_idtDrvHdlr hdlr)

  Sample configuration 3.
- void IDTSample_ConfigureSampleConfig4 (T_idtDrvHdlr hdlr)

  Sample configuration 4.
- void IDTSample_ConfigureSampleConfig5 (T_idtDrvHdlr hdlr)

  Sample configuration 5.
- void IDTSample_Configure1Pps (T_idtDrvHdlr hdlr)

  Sample configuration 1PPS.
- void IDTSample_Configure1PpsHoldover (T_idtDrvHdlr hdlr)

  Sample configuration for 1PPS holdover issue.
- void IDTSample_ConfigureDcoMode (T_idtDrvHdlr hdlr)

  Function to configure device for DCO mode.
- void IDTSample_SetCombinedDcoOffset (T_idtDrvHdlr hdlr, long pptOffset)

  Function to set combined DCO offset value. It uses both DCO (holdover) and nominal offsets.
- long IDTSample_GetCombinedDcoOffset (T_idtDrvHdlr hdlr)

  Function to get combined DCO offset value. It uses both DCO (holdover) and nominal offsets.

6.10.1 Function Documentation

6.10.1.1 void IDTSample_Configure1Pps (T_idtDrvHdlr *hdlr*)

Sample configuration 1PPS.

```
IN3 (1PPS) -> T0 -> OUT3 (1PPS)
```

Parameters

hdlr	Device driver handler

Definition at line 420 of file sample.c.

6.10.1.2 void IDTSample_Configure1PpsHoldover (T_idtDrvHdlr *hdlr*)

Sample configuration for 1PPS holdover issue.

Parameters

hdlr	Device driver handler

Definition at line 449 of file sample.c.

6.10.1.3 void IDTSample_ConfigureDcoMode (T_idtDrvHdlr *hdlr*)

Function to configure device for DCO mode.

Parameters

hdlr	Device driver handler

Definition at line 480 of file sample.c.

6.10.1.4 void IDTSample_ConfigureSampleConfig1 (T_idtDrvHdlr *hdlr*)

Sample configuration 1.

```
IN3 (25 MHz) -> T0 +-> OUT1 (25 MHz)
                   +-> OUT6 (156.25 MHz)
```

Parameters

hdlr	Device driver handler

Definition at line 46 of file sample.c.

6.10.1.5 void IDTSample_ConfigureSampleConfig2 (T_idtDrvHdlr *hdlr*)

Sample configuration 2.

```
IN3 (25 MHz) -> T0 +-> OUT1 (25 MHz)
                   +-> OUT3 (8 kHz)
                   +-> OUT5 (125 MHz)
```

Parameters

hdlr	Device driver handler

Definition at line 98 of file sample.c.

6.10.1.6 void IDTSample_ConfigureSampleConfig3 (T_idtDrvHdlr *hdlr*)

Sample configuration 3.

```
IN3 (25 MHz) -> T0 +-> OUT1 (25 MHz)
                   +-> OUT3 (10 MHz)
                   +-> OUT5 (1 PPS)
```

Parameters

hdlr	Device driver handler

Definition at line 154 of file sample.c.

6.10.1.7 void IDTSample_ConfigureSampleConfig4 (T_idtDrvHdlr *hdlr*)

Sample configuration 4.

```
Inputs: IN3 (10MHz), IN4 (1.544MHz), IN5 (25MHz), IN6 (8KHz), IN7 (8KHz),
        IN8 (156.25MHz), IN9 (156.25MHz), IN10 (25MHz), IN11 (25MHz)
Outputs: OUT1 (1.544MHz), OUT3 (10MHz), OUT4 (25MHz), OUT7 (156.25MHz),
         OUT10 (156.25MHz), OUT11 (156.25MHz)
T0 Profile: G.8262 Option 2
T4 Profile: Default
XTAL1/2: 25MHz
```

Parameters

hdlr	Device driver handler

Definition at line 226 of file sample.c.

6.10.1.8 void IDTSample_ConfigureSampleConfig5 (T_idtDrvHdlr *hdlr*)

Sample configuration 5.

```
Carbon copy of IDTSample_ConfigureSampleConfig4,
except there is no input and DPLL is in free-run
mode (default).
```

Parameters

hdlr	Device driver handler

Definition at line 339 of file sample.c.

6.10.1.9 long IDTSample_GetCombinedDcoOffset (T_idtDrvHdlr *hdlr*)

Function to get combined DCO offset value. It uses both DCO (holdover) and nominal offsets.

Parameters

hdlr	Device driver handler

Returns

Parts per trillion offset

Definition at line 542 of file sample.c.

6.10.1.10 void IDTSample_SetCombinedDcoOffset (T_idtDrvHdlr *hdlr,* long *pptOffset*)

Function to set combined DCO offset value. It uses both DCO (holdover) and nominal offsets.

Parameters

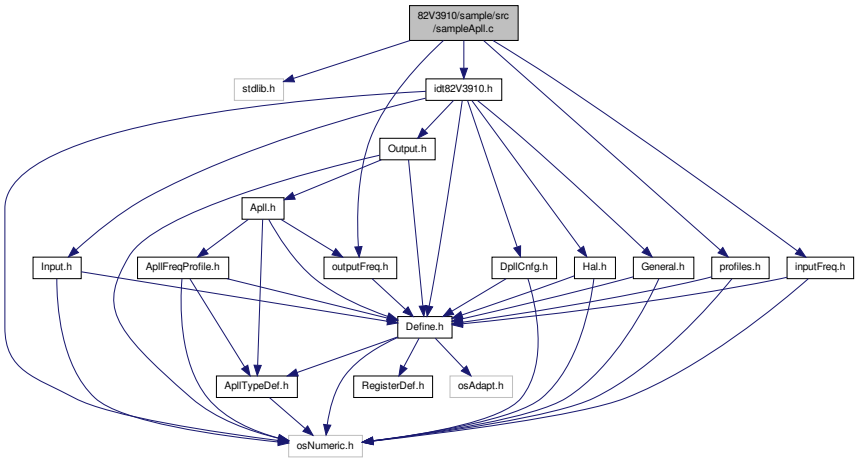
hdlr	Device driver handler
pptOffset	Parts per trillion offset

Definition at line 492 of file sample.c.

6.11 82V3910/sample/src/sampleApll.c File Reference

```
#include <stdlib.h>
#include "idt82V3910.h"
#include "profiles.h"
#include "inputFreq.h"
#include "outputFreq.h"
```

Include dependency graph for sampleApll.c:



Functions

- void IDTSample_ConfigureSampleConfigOutput6_156_dot_25MHz (T_idtDrvHdlr hdlr)

  Sample configuration for 156.25MHz output at port 6.
- void IDTSample_ConfigureSampleConfigureOutput7_77_dot_76MHz (T_idtDrvHdlr hdlr)

  Sample configuration for 77.76 MHz output at port 7.
- void IDTSample_ConfigureSampleConfigureOutput10_25_MHz (T_idtDrvHdlr hdlr)

  Sample configuration for 25MHz output at port 10.
- void IDTSample_SyncE_WAN (T_idtDrvHdlr hdlr)

  Sample configuration for SyncE WAN application.
- void IDTSample_Sonet (T_idtDrvHdlr hdlr)

  Sample configuration for SONET application.
- void IDTSample_DualMode (T_idtDrvHdlr hdlr)

  Sample configuration for dual mode application.
- void IDTSample_SyncE_Sonet (T_idtDrvHdlr hdlr)

  Sample configuration for SyncE SONET application.
- void IDTSample_SyncE_LAN (T_idtDrvHdlr hdlr)

  Sample configuration for SyncE LAN application.
- void IDTSample_GPS_1PPS (T_idtDrvHdlr hdlr)

  Sample configuration for GPS 1PPS application.
- void IDTSample_GPS (T_idtDrvHdlr hdlr)

  Sample configuration for GPS application.

6.11.1 Function Documentation

6.11.1.1 void IDTSample_ConfigureSampleConfigOutput6_156_dot_25MHz (T_idtDrvHdlr *hdlr*)

Sample configuration for 156.25MHz output at port 6.

```
IN3 (25 MHz) -> T0 +-> OUT6 (Ethernet, 156.25 MHz)
```

Parameters

hdlr	Device driver handler

Definition at line 47 of file sampleApll.c.

6.11.1.2 void IDTSample_ConfigureSampleConfigureOutput10_25_MHz (T_idtDrvHdlr *hdlr*)

Sample configuration for 25MHz output at port 10.

```
IN3 (25 MHz) -> T0 +-> OUT10 (Ethernet, 25 MHz)
```

Parameters

hdlr	Device driver handler

Definition at line 125 of file sampleApll.c.

6.11.1.3 void IDTSample_ConfigureSampleConfigureOutput7_77_dot_76MHz (T_idtDrvHdlr *hdlr*)

Sample configuration for 77.76 MHz output at port 7.

```
IN3 (25 MHz) -> T0 +-> OUT7 (SONET, 77.76 MHz)
```

Parameters

hdlr	Device driver handler

Definition at line 86 of file sampleApll.c.

6.11.1.4 void IDTSample_DualMode (T_idtDrvHdlr *hdlr*)

Sample configuration for dual mode application.

```
Inputs: IN3 (19.44MHz), IN5 (155.52MHz), IN14 (1.544MHz)
T0 Profile: ST3 SONET
T4 Profile: default
Select XTAL3: done automatically by IDTHlApi_ConfigureOutput function
              based on the output frequency. To manually select XTAL,
              call SetJaXtalId function.
Outputs: OUT1 (19.44MHz), OUT6 (155.52MHz), OUT7 (77.76MHz), OUT9 (1.544MHz)
T0: IN3, IN5, IN14 -> OUT1, OUT6, OUT7 (IN14 is highest priority)
T4: IN3, IN5 -> OUT9
APLL1 XTAL1 at 25MHz & XTAL3 at 24.8832MHz
```

Parameters

hdlr	Device driver handler

Definition at line 410 of file sampleApll.c.

6.11.1.5 void IDTSample_GPS (T_idtDrvHdlr *hdlr*)

Sample configuration for GPS application.

```
Inputs: IN3 (10MHz), IN4 (10MHz), EX_SYNC1 (1PPS), EX_SYNC2 (1PPS)
```

```
T0 Profile: BITS / SSU (Wideband) - default
Outputs: OUT1 (25MHz), OUT2 (19.44MHz), OUT3 (10MHz), OUT4 (1PPS),
         OUT6 (25MHz), OUT7 (156.25MHz), OUT9 (1.544MHz), OUT10 (155.52MHz),
         OUT11 (77.76MHz), FRSYNC_8K_1PPS (1PPS),  MFRSYNC_2K_1PPS (1PPS)
APLL1 XTAL1 =  25MHz & APLL2 XTAL2 = 24.8832MHz
```

Parameters

hdlr	Device driver handler

Definition at line 1013 of file sampleApll.c.

6.11.1.6 void IDTSample_GPS_1PPS (T_idtDrvHdlr *hdlr*)

Sample configuration for GPS 1PPS application.

```
Inputs: IN3 (1PPS), IN5 (1PPS)
T0 Profile: GPS (15mHz)
Outputs: OUT1 (25MHz), OUT2 (19.44MHz), OUT3 (10MHz), OUT4 (1PPS),
         OUT6 (25MHz), OUT7 (156.25MHz), OUT9 (2.048MHz),
         OUT10 (155.52MHz), OUT11 (77.76MHz)
APLL1 XTAL1 at 25MHz & APLL2 XTAL2 at 24.8832MHz
```

Parameters

hdlr	Device driver handler

Definition at line 835 of file sampleApll.c.

6.11.1.7 void IDTSample_Sonet (T_idtDrvHdlr *hdlr*)

Sample configuration for SONET application.

```
Inputs: IN3 (19.44MHz), IN5 (155.52MHz), IN14 (1.544MHz)
T0 Profile: ST3 SONET
T4 Profile: default
Outputs: OUT1 (19.44MHz), OUT9 (1.544MHz), OUT10 (155.52MHz)
T0: IN3, IN5, IN14 -> OUT1, OUT10 (IN14 is highest priority)
T4: IN3, IN5 -> OUT9
APLL2 XTAL2 at 24.8832MHz
```

Parameters

hdlr	Device driver handler

Definition at line 294 of file sampleApll.c.

6.11.1.8 void IDTSample_SyncE_LAN (T_idtDrvHdlr *hdlr*)

Sample configuration for SyncE LAN application.

```
Inputs: IN3 (25MHz), IN14 (2.048MHz)
T0 Profile: G.8262 Option 2
T4 Profile: BITS / SSU (Wideband)
Outputs: OUT1 (25MHz), OUT10 (25.78125), OUT11 (161.1328125MHz), OUT9 (2.048MHz)
T0: IN3, IN14 -> OUT1, OUT10, OUT11 (IN14 is highest priority)
T4: IN3 -> OUT9
APLL2 XTAL4 at 25.78125MHz
```

Parameters

hdlr	Device driver handler

Definition at line 720 of file sampleApll.c.

6.11.1.9 void IDTSample_SyncE_Sonet (T_idtDrvHdlr *hdlr*)

Sample configuration for SyncE SONET application.

```
Inputs: IN3 (25MHz) , IN5 (156.25MHz), IN6 (155.52MHz), IN7 (19.44MHz), IN14 (1.544MHz)
T0 Profile: G.8262 Option 2
T4 Profile: BITS / SSU (Wideband) - default
Outputs: OUT1 (25MHz), OUT2 (19.44MHz), OUT6 (25MHz), OUT7 (156.25MHz),
         OUT9 (1.544MHz), OUT10 (155.52MHz), OUT11 (77.76MHz)
T0: IN3, IN5, IN6, IN7, IN14 -> OUT1, OUT2, OUT6, OUT7, OUT10, OUT11 (IN14 is highest priority)
T4: IN3, IN5, IN6, IN7 -> OUT9
APLL1 XTAL1 at 25MHz & APLL2 XTAL2 at 24.8832MHz
```

Parameters

hdlr	Device driver handler

Definition at line 534 of file sampleApll.c.

6.11.1.10 void IDTSample_SyncE_WAN (T_idtDrvHdlr *hdlr*)

Sample configuration for SyncE WAN application.

```
Inputs: IN3 (25MHz), IN5 (156.25MHz), IN14 (1.544MHz)
T0 Profile: G.8262 Option 2
T4 Profile: Default
Outputs: OUT1 (25MHz), OUT6 (25MHz), OUT7 (156.25MHz), OUT9 (1.544MHz)
T0: IN3, IN5, IN14 -> OUT1, OUT10, OUT11 (IN14 has highest priority)
T4: IN3, IN5 -> OUT9
APLL1 XTAL1 at 25MHz
```

Parameters

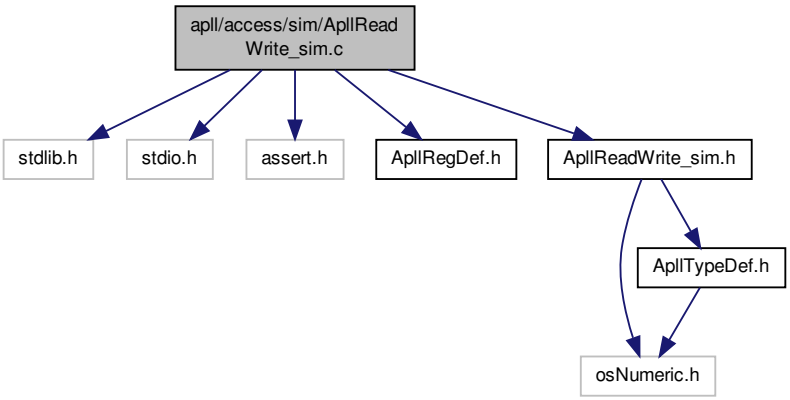
hdlr	Device driver handler

Definition at line 170 of file sampleApll.c.

6.12 apll/access/sim/ApllReadWrite_sim.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <assert.h>
#include "ApllRegDef.h"
#include "ApllReadWrite_sim.h"
```

Include dependency graph for ApllReadWrite_sim.c:



Macros

- #define SIM_REG_MAP_SIZE (0x11)

Functions

- void ApllInit_Sim (void ∗usrData)
- void ApllDeInit_Sim (void ∗usrData)
- 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)

Variables

- T_idtApllRegVal simulatedRegMapApll0 [SIM_REG_MAP_SIZE]
- T_idtApllRegVal simulatedRegMapApll1 [SIM_REG_MAP_SIZE]

6.12.1 Macro Definition Documentation

6.12.1.1 #define SIM_REG_MAP_SIZE (0x11)

Definition at line 41 of file ApllReadWrite_sim.c.

6.12.2 Function Documentation

6.12.2.1 void ApllDeInit_Sim (void ∗ *usrData*)

Definition at line 62 of file ApllReadWrite_sim.c.

6.12.2.2 void ApllInit_Sim (void ∗ *usrData*)

Definition at line 57 of file ApllReadWrite_sim.c.

6.12.2.3 T_osUint8 ApllRead_Sim (void * *usrData,* T_osUint8 *deviceAddr,* T_idtApllRegAddr *nAddr*)

Definition at line 67 of file ApllReadWrite_sim.c.

6.12.2.4 void ApllWrite_Sim (void * *usrData,* T_osUint8 *deviceAddr,* T_idtApllRegAddr *nAddr,* T_idtApllRegVal *data*)

Definition at line 82 of file ApllReadWrite_sim.c.

6.12.3 Variable Documentation

6.12.3.1 T_idtApllRegVal simulatedRegMapApll0[SIM_REG_MAP_SIZE]

Initial value:

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

Definition at line 43 of file ApllReadWrite_sim.c.

6.12.3.2 T_idtApllRegVal simulatedRegMapApll1[SIM_REG_MAP_SIZE]

Initial value:

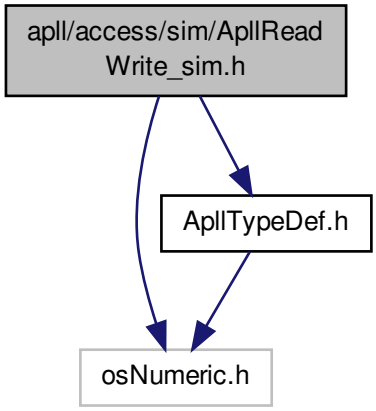
```
= {
    0x00, 0x01, 0x0c, 0x35, 0x0c,
    0x35, 0x0c, 0x35, 0x0c, 0x35,
    0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00
}
```

Definition at line 50 of file ApllReadWrite_sim.c.

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

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

Include dependency graph for ApllReadWrite_sim.h:



Functions

- T_osUint8 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)

6.13.1 Function Documentation

6.13.1.1 void ApllDeInit_Sim (void ∗ *userData*)

Definition at line 62 of file ApllReadWrite_sim.c.

6.13.1.2 void ApllInit_Sim (void ∗ *userData*)

Definition at line 57 of file ApllReadWrite_sim.c.

6.13.1.3 T_osUint8 ApllRead_Sim (void ∗ *usrData,* T_osUint8 *deviceAddr,* T_idtApllRegAddr *nAddr*)

Definition at line 67 of file ApllReadWrite_sim.c.

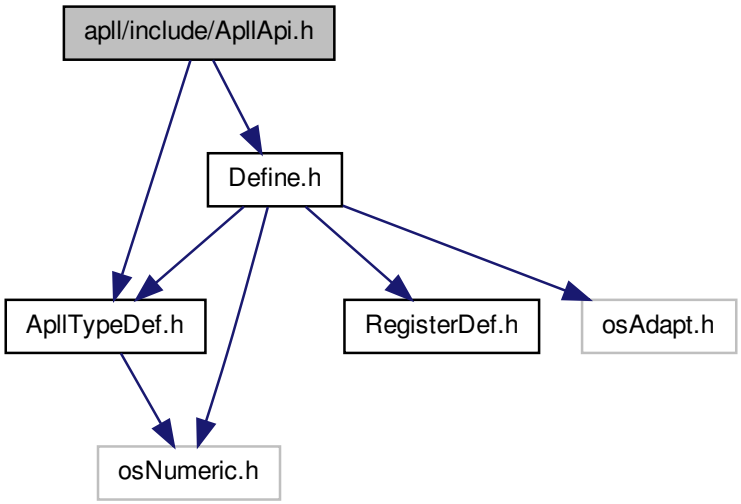
6.13.1.4 void ApllWrite_Sim (void ∗ *usrData,* T_osUint8 *deviceAddr,* T_idtApllRegAddr *nAddr,* T_idtApllRegVal *data*)

Definition at line 82 of file ApllReadWrite_sim.c.

6.14 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 apllInstId)

  Function to retrieve APLL configuration.
- void IDTApll_SetConfigSel (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig)

  Function to provision APLL based on input configuration.
- T_idtApllInputClkSrc IDTApll_GetInputClkSrc (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve selected APLL input frequency source (external/internal)
- void IDTApll_SetInputClkSrc (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllInputClkSrc inputClk)

  Function to provision APLL input frequency source (external/internal)
- T_idtApllDividerValue IDTApll_GetPreDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig)

  Function to retrieve pre- (input) divider value.
- void IDTApll_GetPreDivRange (T_idtDrvHdlr hdlr, T_idtApllDividerValue *minDiv, T_idtApllDividerValue *maxDiv)

  Function to retrieve pre- (input) divisor range.
- void IDTApll_SetPreDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig, T_idtApllDividerValue div)

  Function to provision pre- (input) divider value.
- T_idtApllDividerValue IDTApll_GetFeedbackDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig)

  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 apllInstId, T_idtApllConfigSel apllConfig, T_idtApllDividerValue div)

  Function to provision feedback (oscillator) divider value.

- T_idtApllQaDiv IDTApll_GetQaDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig)

  Function to retrieve output port A divisor.
- void IDTApll_SetQaDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig, T_idtApllQaDiv div)

  Function to provision output port A divisor.
- T_idtApllQaDiv IDTApll_GetQbDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig)

  Function to retrieve output port B divisor.
- void IDTApll_SetQbDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig, T_idtApllQbDiv div)

  Function to provision output port B divisor.
- T_idtApllFreqDoublerSel IDTApll_GetFreqDoublerSel (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve whether feedback frequency doubling is enabled.
- void IDTApll_SetFreqDoublerSel (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllFreqDoublerSel doubler)

  Function to enable/disable feedback frequency doubling.
- T_idtApllXtalId IDTApll_GetXtalId (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve oscillator currently used.
- void IDTApll_SetXtalId (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllXtalId xtal)

  Function to select oscillator.
- T_idtApllOutputSigType IDTApll_GetOutputSigType (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllOutput output)

  Function to retrieve output port signal type.
- void IDTApll_SetOutputSigType (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllOutput output, T_idtApllOutputSigType outputSigType)

  Function to provision output port signal type.
- T_idtApllOutputEn IDTApll_GetOutputEnState (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllOutput output)

  Function to retrieve output port status (enable/disable)
- void IDTApll_SetOutputEnState (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllOutput output, T_idtApllOutputEn enable)

  Function to control (enable/disable) output port.
- T_idtApllMasterResetSync IDTApll_GetMasterResetSync (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve reset status of device.
- void IDTApll_SetMasterResetSync (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllMasterResetSync reset)

  Function to reset APLL.
- T_idtApllBypass IDTApll_GetBypass (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve bypass status of device.
- void IDTApll_SetBypass (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllBypass bypass)

  Function to set APLL bypass setting.
- T_idtApllShutoff IDTApll_GetShutoff (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

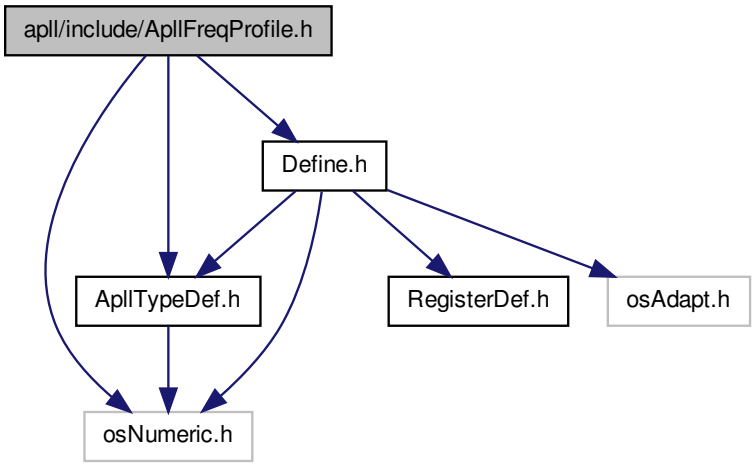
  Function to retrieve shutoff status of device.
- void IDTApll_SetShutoff (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllShutoff shutoff)

  Function to set APLL shutoff setting.

6.15 apll/include/ApllFreqProfile.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "ApllTypeDef.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_idtApllFreqMapping * IDTApll_GetConfigListByXtalType (T_idtApllXtalType apllXtalFreq)

  Function to retrieve output configuration gvien the oscilator type (frequency)
- T_osBool IDTApll_NullFreqTableEntry (const T_idtApllFreqMapping *apllFreqMapping)

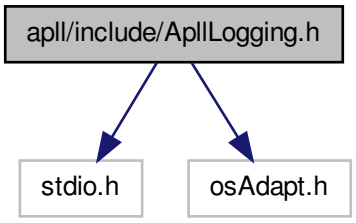
  Function to check whether the input APLL frequency mapping is NULL (not configured)
- const T_idtApllFreqMapping * IDTApll_GetCurrentApllFreqConfig (T_idtApllId apllId, T_idtApllOutput output)

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

6.16 apll/include/ApllLogging.h File Reference

```
#include <stdio.h>
#include "osAdapt.h"
```

Include dependency graph for ApllLogging.h:



Macros

- #define IDT_APLL_API_LOG_HEADER
- #define IDT_APLL_API_LOG_ERR(fmt)
- #define IDT_APLL_API_LOG_INFO(fmt)
- #define IDT_APLL_API_LOG_DEBUG(fmt)
- #define IDT_APLL_API_TRACE(fmt) OS_TRACE(fmt)

6.17 apll/include/ApllRegDef.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-

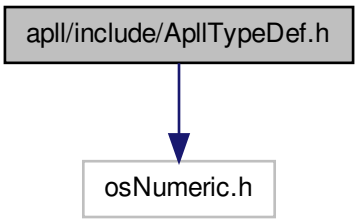
_LSB_LSB = 0,
BF_PDSEL1_MSB_SIZE = 7, BF_PDSEL1_MSB_MASK = 0x7F, BF_PDSEL1_MSB_LSB = 0, BF_M0_LSB_SIZE = 8,
BF_M0_LSB_MASK = 0xFF, BF_M0_LSB_LSB = 0, BF_M0_MSB_SIZE = 7, BF_M0_MSB_MASK = 0x7F,
BF_M0_MSB_LSB = 0, BF_M1_LSB_SIZE = 8, BF_M1_LSB_MASK = 0xFF, BF_M1_LSB_LSB = 0,
BF_M1_MSB_SIZE = 7, BF_M1_MSB_MASK = 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 offsers and bit field constants.

6.18 apll/include/ApllTypeDef.h File Reference

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

Include dependency graph for ApllTypeDef.h:



Data Structures

- struct T_idtApllXtal

  Structure containing APLL crystal id and frequency information.

Macros

- #define INVALID_DIVIDER_VALUE (0xFFFF)

  Define constant indicating invalid divider value.

Typedefs

- typedef T_osUint16 T_idtApllDividerValue

  Type define for APLL divider value.
- typedef T_osUint8 T_idtApllRegAddr

  Type define for APLL register address.
- typedef T_osUint8 T_idtApllRegMask

  Type define for APLL register value mask.
- typedef T_osUint8 T_idtApllRegVal

  Type define for APLL register value.

Enumerations

- enum T_idtApllId { APLL_INST_1, APLL_INST_2, APLL_INST_MAX }

  Enumerated type listing APLL instance within device.
- enum T_idtApllAccessType { APLL_ACCESS_I2C, APLL_ACCESS_SIM, APLL_ACCESS_MAX }

  Enumerated type listing supported access methods.
- enum T_idtApllConfigSel { APLL_CONFIG0 = 0, APLL_CONFIG1 = 1, APLL_CONFIG_MAX }

  Enumerated type listing possible configuration for each APLL.
- enum T_idtApllInputClkSrc { APLL_CLK_SEL_EXTERNAL, APLL_CLK_SEL_INTERNAL, APLL_CLK_SEL_MAX }

  Enumerated type listing input ports.
- enum T_idtApllOutput { APLL_OUTPUT_QA, APLL_OUTPUT_QB, APLL_OUTPUT_MAX }

  Enumerated type listing output ports.
- enum T_idtApllOutputEn { APLL_OUTPUT_DISABLE, APLL_OUTPUT_ENABLE, APLL_OUTPUT_EN_MAX }

  Enumerated type listing output port status (enable/disable)
- enum T_idtApllOutputSigType { APLL_OUT_SIG_LVPECL, APLL_OUT_SIG_LVDS, APLL_OUT_SIG_MAX }

  Enumerated type listing output signal type.
- enum T_idtApllOutputSupportedType { APLL_OUT_QA_ONLY, APLL_OUT_QB_ONLY, APLL_OUT_QA_OR_QB, APLL_OUT_MAX }

  Enumerated type listing frequency supported by the output.
- enum T_idtApllQaDiv {
  APLL_QA_ODSEL_DIV_25 = 0, APLL_QA_ODSEL_DIV_5 = 1, APLL_QA_ODSEL_DIV_4 = 2, APLL_QA_ODSEL_DIV_2 = 3,
  APLL_QA_ODSEL_DIV_1 = 4, APLL_QA_ODSEL_DIV_MAX }

  Enumerated type listing output port A divider values.
- enum T_idtApllQbDiv {
  APLL_QB_ODSEL_DIV_8 = 0, APLL_QB_ODSEL_DIV_5 = 1, APLL_QB_ODSEL_DIV_4 = 2, APLL_QB_ODSEL_DIV_2 = 3,
  APLL_QB_ODSEL_DIV_1 = 4, APLL_QB_ODSEL_DIV_MAX }

  Enumerated type listing output port B divider values.
- enum T_idtApllFreqDoublerSel { APLL_FREQ_SEL_SINGLE = 0, APLL_FREQ_SEL_DOUBLE = 1, APLL_FREQ_SEL_MAX }

  Enumerated type listing feedback frequency doubling values.
- enum T_idtApllXtalId {
  APLL_XTAL_1_APLL1, APLL_XTAL_2_APLL2, APLL_XTAL_3_APLL1, APLL_XTAL_4_APLL2,
  APLL_XTAL_MAX }

  Enumerated type listing crystals.
- enum T_idtApllXtalType { APLL_XTAL_FREQ_25000KHz, APLL_XTAL_FREQ_25781_DOT_25KHz, APLL_XTAL_FREQ_24883_DOT_2KHz, APLL_XTAL_FREQ_MAX }

Enumerated type listing support crystal frequencies.

- enum T_idtApllMasterResetSync { APLL_OUT_MR_SYNC = 0, APLL_IN_MR_SYNC = 1, APLL_MR_SYNC_MAX }

  Enumerated type listing Device master reset sync status.

- enum T_idtApllBypass { APLL_BYPASS_NORMAL_OP = 0, APLL_BYPASS_BYPASS = 1, APLL_BYPASS_MAX }

  Enumerated type listing Apll bypass status/setting.

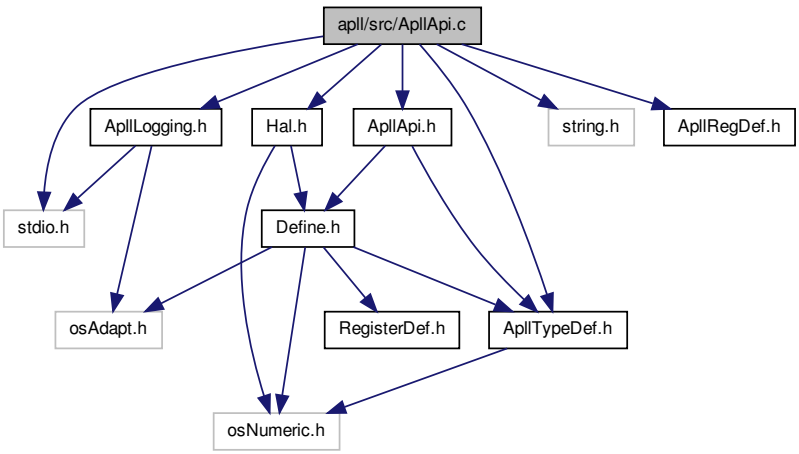
- enum T_idtApllShutoff { APLL_SHUTOFF_NORMAL_OP = 0, APLL_SHUTOFF_SHUTOFF = 1, APLL_SHUTOFF_MAX }

  Enumerated type listing Apll shutoff status/setting.

6.19 apll/src/ApllApi.c File Reference

```
#include <stdio.h>
#include <string.h>
#include "Hal.h"
#include "ApllRegDef.h"
#include "ApllTypeDef.h"
#include "ApllApi.h"
#include "ApllLogging.h"
```

Include dependency graph for ApllApi.c:



Macros

- #define APLL_DIVIDER_LSB(val) ((val)&0x00FF)
- #define APLL_DIVIDER_MSB(val) (((val)&0x7F00)>>8)
- #define APLL_DIVIDER_VAL(lsb, msb) ((msb)<<8|(lsb))
- #define APLL_PRE_DIV_LSB(val) APLL_DIVIDER_LSB(val)
- #define APLL_PRE_DIV_MSB(val) APLL_DIVIDER_MSB(val)
- #define APLL_PRE_DIV_VAL(lsb, msb) APLL_DIVIDER_VAL(lsb, msb)
- #define APLL_FB_DIV_LSB(val) APLL_DIVIDER_LSB(val)
- #define APLL_FB_DIV_MSB(val) APLL_DIVIDER_MSB(val)
- #define APLL_FB_DIV_VAL(lsb, msb) APLL_DIVIDER_VAL(lsb, msb)
- #define APLL_PRE_DIV_MIN (0x0004)

- #define APLL_PRE_DIV_MAX (0x7FFF)
- #define APLL_FB_DIV_MIN (0x0005)
- #define APLL_FB_DIV_MAX (0x7FFF)
- #define APLL1_REG_HIGH_BIT (0)
- #define APLL2_REG_HIGH_BIT (1)
- #define APLL_REG_HIGH_BIT_OFFSET (7)
- #define APLL_REG_OFFSET(apllInstId, base)

  Utility macro to translate APLL register address.
- #define APLL1_REG_VALID_OFFSET(offset) ((offset >= REG_CONTROL) && (offset <= REG_CONFIG_QB))

  Utility macro to sanity check APLL #1 register address.
- #define APLL2_REG_VALID_OFFSET(offset)

  Utility macro to translate APLL #2 register address.
- #define INVALID_XTAL_SEL_VALUE (0xFF)

Functions

- T_idtApllConfigSel IDTApll_GetConfigSel (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve APLL configuration.
- void IDTApll_SetConfigSel (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig)

  Function to provision APLL based on input configuration.
- T_idtApllInputClkSrc IDTApll_GetInputClkSrc (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve selected APLL input frequency source (external/internal)
- void IDTApll_SetInputClkSrc (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllInputClkSrc inputClk)

  Function to provision APLL input frequency source (external/internal)
- T_idtApllDividerValue IDTApll_GetPreDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig)

  Function to retrieve pre- (input) divider value.
- void IDTApll_GetPreDivRange (T_idtDrvHdlr hdlr, T_idtApllDividerValue *minDiv, T_idtApllDividerValue *maxDiv)

  Function to retrieve pre- (input) divisor range.
- void IDTApll_SetPreDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig, T_idtApllDividerValue div)

  Function to provision pre- (input) divider value.
- void IDTApll_GetFeedbackDivRange (T_idtDrvHdlr hdlr, T_idtApllDividerValue *minDiv, T_idtApllDividerValue *maxDiv)

  Function to retrieve feedback (oscillator) divisor range.
- T_idtApllDividerValue IDTApll_GetFeedbackDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig)

  Function to retrieve feedback (oscillator) divider value.
- void IDTApll_SetFeedbackDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel apllConfig, T_idtApllDividerValue div)

  Function to provision feedback (oscillator) divider value.
- T_idtApllQaDiv IDTApll_GetQaDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel config)

  Function to retrieve output port A divisor.
- void IDTApll_SetQaDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel config, T_idtApllQaDiv div)

  Function to provision output port A divisor.
- T_idtApllQaDiv IDTApll_GetQbDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel config)

  Function to retrieve output port B divisor.
- void IDTApll_SetQbDiv (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllConfigSel config, T_idtApllQbDiv div)

Function to provision output port B divisor.

- T_idtApllFreqDoublerSel IDTApll_GetFreqDoublerSel (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve whether feedback frequency doubling is enabled.
- void IDTApll_SetFreqDoublerSel (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllFreqDoublerSel doubler)

  Function to enable/disable feedback frequency doubling.
- T_idtApllXtalId IDTApll_GetXtalId (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve oscillator currently used.
- void IDTApll_SetXtalId (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllXtalId xtal)

  Function to select oscillator.
- T_idtApllOutputSigType IDTApll_GetOutputSigType (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllOutput output)

  Function to retrieve output port signal type.
- void IDTApll_SetOutputSigType (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllOutput output, T_idtApllOutputSigType outputSigType)

  Function to provision output port signal type.
- T_idtApllOutputEn IDTApll_GetOutputEnState (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllOutput output)

  Function to retrieve output port status (enable/disable)
- void IDTApll_SetOutputEnState (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllOutput output, T_idtApllOutputEn enable)

  Function to control (enable/disable) output port.
- void IDTApll_SetMasterResetSync (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllMasterResetSync reset)

  Function to reset APLL.
- T_idtApllMasterResetSync IDTApll_GetMasterResetSync (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve reset status of device.
- void IDTApll_SetBypass (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllBypass bypass)

  Function to set APLL bypass setting.
- T_idtApllBypass IDTApll_GetBypass (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve bypass status of device.
- void IDTApll_SetShutoff (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllShutoff shutoff)

  Function to set APLL shutoff setting.
- T_idtApllShutoff IDTApll_GetShutoff (T_idtDrvHdlr hdlr, T_idtApllId apllInstId)

  Function to retrieve shutoff status of device.

6.19.1 Macro Definition Documentation

6.19.1.1 #define APLL1_REG_HIGH_BIT (0)

Definition at line 61 of file ApllApi.c.

6.19.1.2 #define APLL1_REG_VALID_OFFSET(*offset*) ((offset >= REG_CONTROL) && (offset <= REG_CONFIG_QB))

Utility macro to sanity check APLL #1 register address.

Parameters

offset	APLL register address

Definition at line 78 of file ApllApi.c.

6.19.1.3 #define APLL2_REG_HIGH_BIT (1)

Definition at line 62 of file ApllApi.c.

6.19.1.4 #define APLL2_REG_VALID_OFFSET(*offset*)

Value:

```
(offset >= APLL_REG_OFFSET(APLL_INST_2, REG_CONTROL)) && \
        (offset <= APLL_REG_OFFSET(APLL_INST_2,
      REG_CONFIG_QB))
```

Utility macro to translate APLL #2 register address.

Parameters

offset	APLL register base address

Definition at line 85 of file ApllApi.c.

6.19.1.5 #define APLL_DIVIDER_LSB(*val*) ((val)&0x00FF)

Definition at line 45 of file ApllApi.c.

6.19.1.6 #define APLL_DIVIDER_MSB(*val*) (((val)&0x7F00)>>8)

Definition at line 46 of file ApllApi.c.

6.19.1.7 #define APLL_DIVIDER_VAL(*lsb, msb*) ((msb)<<8|(lsb))

Definition at line 47 of file ApllApi.c.

6.19.1.8 #define APLL_FB_DIV_LSB(*val*) APLL_DIVIDER_LSB(val)

Definition at line 52 of file ApllApi.c.

6.19.1.9 #define APLL_FB_DIV_MAX (0x7FFF)

Definition at line 59 of file ApllApi.c.

6.19.1.10 #define APLL_FB_DIV_MIN (0x0005)

Definition at line 58 of file ApllApi.c.

6.19.1.11 #define APLL_FB_DIV_MSB(*val*) APLL_DIVIDER_MSB(val)

Definition at line 53 of file ApllApi.c.

6.19.1.12 #define APLL_FB_DIV_VAL(*lsb, msb*) APLL_DIVIDER_VAL(lsb, msb)

Definition at line 54 of file ApllApi.c.

6.19.1.13 #define APLL_PRE_DIV_LSB(*val*) APLL_DIVIDER_LSB(val)

Definition at line 49 of file ApllApi.c.

6.19.1.14 #define APLL_PRE_DIV_MAX (0x7FFF)

Definition at line 57 of file ApllApi.c.

6.19.1.15 #define APLL_PRE_DIV_MIN (0x0004)

Definition at line 56 of file ApllApi.c.

6.19.1.16 #define APLL_PRE_DIV_MSB(*val*) APLL_DIVIDER_MSB(val)

Definition at line 50 of file ApllApi.c.

6.19.1.17 #define APLL_PRE_DIV_VAL(*lsb, msb*) APLL_DIVIDER_VAL(lsb, msb)

Definition at line 51 of file ApllApi.c.

6.19.1.18 #define APLL_REG_HIGH_BIT_OFFSET (7)

Definition at line 63 of file ApllApi.c.

6.19.1.19 #define APLL_REG_OFFSET(*apllInstId, base*)

Value:

```
(apllInstId == APLL_INST_1) ? (base) : \
    (base|(APLL2_REG_HIGH_BIT<<APLL_REG_HIGH_BIT_OFFSET))
```

Utility macro to translate APLL register address.

Parameters

apllInstId	APLL instance
base	APLL register base address

Definition at line 70 of file ApllApi.c.

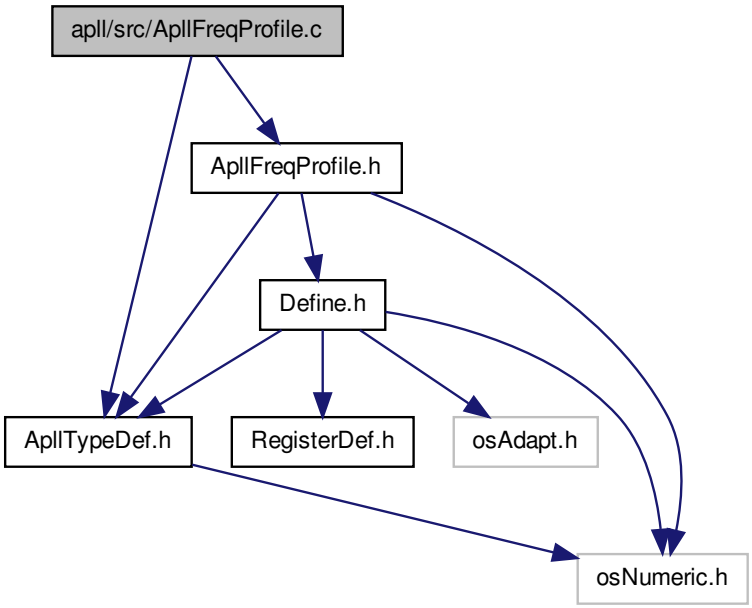
6.19.1.20 #define INVALID_XTAL_SEL_VALUE (0xFF)

Definition at line 89 of file ApllApi.c.

6.20 apll/src/ApllFreqProfile.c File Reference

```
#include "ApllTypeDef.h"
#include "ApllFreqProfile.h"
```

Include dependency graph for ApllFreqProfile.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_idtApllFreqMapping * IDTApll_GetConfigListByFreq (T_idtApllOutputFreqType apllOutputFreq)

  Function to retrieve output configuration given an output frequency.
- const T_idtApllFreqMapping * IDTApll_GetConfigListByXtalType (T_idtApllXtalType apllVcxoFreq)

  Function to retrieve output configuration gvien the oscilator type (frequency)
- T_osBool IDTApll_NullFreqTableEntry (const T_idtApllFreqMapping *apllFreqConfig)

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.

6.20.1 Macro Definition Documentation

6.20.1.1 #define APLL_FB_DIV_ETH (3124)

Definition at line 45 of file ApllFreqProfile.c.

6.20.1.2 #define APLL_FB_DIV_ETH_FEC (3125)

Definition at line 47 of file ApllFreqProfile.c.

6.20.1.3 #define APLL_FB_DIV_SONET (3128)

Definition at line 46 of file ApllFreqProfile.c.

6.20.1.4 #define APLL_OUTPUT_DIV_1 (1)

Definition at line 50 of file ApllFreqProfile.c.

6.20.1.5 #define APLL_OUTPUT_DIV_2 (2)

Definition at line 51 of file ApllFreqProfile.c.

6.20.1.6 #define APLL_OUTPUT_DIV_25 (25)

Definition at line 55 of file ApllFreqProfile.c.

6.20.1.7 #define APLL_OUTPUT_DIV_4 (4)

Definition at line 52 of file ApllFreqProfile.c.

6.20.1.8 #define APLL_OUTPUT_DIV_5 (5)

Definition at line 53 of file ApllFreqProfile.c.

6.20.1.9 #define APLL_OUTPUT_DIV_8 (8)

Definition at line 54 of file ApllFreqProfile.c.

6.20.1.10 #define APLL_PRE_DIV_ETH (19525)

Definition at line 40 of file ApllFreqProfile.c.

6.20.1.11 #define APLL_PRE_DIV_ETH_FEC (15625)

Definition at line 42 of file ApllFreqProfile.c.

6.20.1.12 #define APLL_PRE_DIV_SONET (9775)

Definition at line 41 of file ApllFreqProfile.c.

6.20.1.13 #define MAX_CONFIG_COMBO_PER_FREQ (1)

Definition at line 57 of file ApllFreqProfile.c.

6.20.1.14 #define MAX_CONFIG_COMBO_PER_VCXO_TYPE (6)

Definition at line 58 of file ApllFreqProfile.c.

6.20.1.15 #define NULL_APLL_FREQ_TABLE_ENTRY

Value:

```
{                                                       \
        APLL_INPUT_FREQ_MAX,                    \
        0,                                              \
        0,                                              \
        APLL_XTAL_FREQ_MAX,                             \
        0,                                              \
        APLL_OUTPUT_FREQ_MAX,                   \
        APLL_OUT_MAX                                    \
}
```

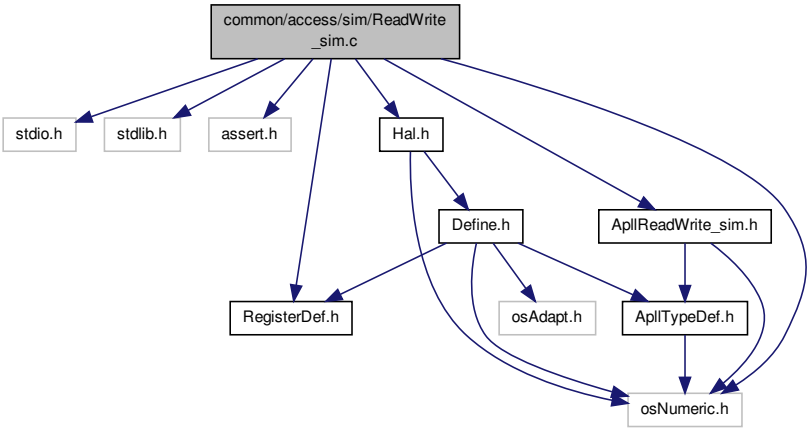
Utility marco for initializing NULL APLL configuration entry.

Definition at line 63 of file ApllFreqProfile.c.

6.21 common/access/sim/ReadWrite_sim.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <assert.h>
#include "RegisterDef.h"
#include "osNumeric.h"
#include "Hal.h"
#include "ApllReadWrite_sim.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 dpllOverlayRegisters []
- T_osUint8 page1Registers []

6.21.1 Macro Definition Documentation

6.21.1.1 #define DRV_MEMMAP_SIZE 0x80

Definition at line 45 of file ReadWrite_sim.c.

6.21.1.2 #define NONE_APLL_REG_SPACE_FLAG 0x01

Definition at line 46 of file ReadWrite_sim.c.

6.21.2 Function Documentation

6.21.2.1 void DeInit_Sim (void * *usrData*)

Definition at line 162 of file ReadWrite_sim.c.

6.21.2.2 void Init_Sim (void * *usrData*)

Definition at line 157 of file ReadWrite_sim.c.

6.21.2.3 T_osUint8 RDByte_Sim (void * *usrData,* T_osUint8 *devAddr,* T_osUint8 *nAddr*)

Definition at line 167 of file ReadWrite_sim.c.

6.21.2.4 void WRByte_Sim (void * *usrData,* T_osUint8 *devAddr,* T_osUint8 *nAddr,* T_osUint8 *nData*)

Definition at line 222 of file ReadWrite_sim.c.

6.21.3 Variable Documentation

6.21.3.1 T_osUint8 dpllOverlayRegisters[ ]

Initial value:

```
= {
  REG_IN1_IN2_SEL_PRIORITY_CNFG,
  REG_IN3_IN4_SEL_PRIORITY_CNFG,
  REG_IN5_IN6_SEL_PRIORITY_CNFG,
  REG_IN7_IN8_SEL_PRIORITY_CNFG,
  REG_IN9_IN10_SEL_PRIORITY_CNFG,
  REG_IN11_IN12_SEL_PRIORITY_CNFG,
  REG_IN13_IN14_SEL_PRIORITY_CNFG,
  REG_PRIORITY_TABLE1_STS,
  REG_PRIORITY_TABLE2_STS,
  REG_PHASE_LOSS_COARSE_LIMIT_CNFG,
  REG_PHASE_LOSS_FINE_LIMIT_CNFG,
  REG_HOLDOVER_FREQ_LOW_CNFG,
  REG_HOLDOVER_FREQ_MID_CNFG,
  REG_HOLDOVER_FREQ_HIGH_CNFG,
  REG_CURRENT_FREQ_LOW_STS,
  REG_CURRENT_FREQ_MID_STS,
  REG_CURRENT_FREQ_HIGH_STS,
  REG_CURRENT_DPLL_PHASE_LOW_STS,
  REG_CURRENT_DPLL_PHASE_HIGH_STS

}
```

Definition at line 90 of file ReadWrite_sim.c.

6.21.3.2 T_osUint8 page1Registers[ ]

Initial value:

```
= {
  REG_DFS_OFF_CNFG,
  REG_T0_PPS_CNFG,
  REG_PH_SLOPE_CNFG,
  REG_T0_ICP_CTRL_CNFG,
  REG_T4_ICP_CTRL_CNFG,
  REG_T4_PPS_CNFG
}
```

Definition at line 139 of file ReadWrite_sim.c.

6.21.3.3 int simDebugFlag = 0

Definition at line 48 of file ReadWrite_sim.c.

6.21.3.4 T_osUint8 simulatedMemMap[2][DRV_MEMMAP_SIZE]

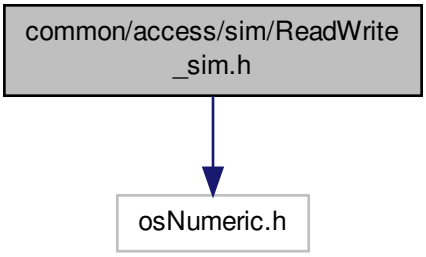
Reset values for device

Definition at line 51 of file ReadWrite_sim.c.

6.22 common/access/sim/ReadWrite_sim.h File Reference

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

Include dependency graph for ReadWrite_sim.h:



Functions

- void Init_Sim (void *usrData)
- void 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)

6.22.1 Function Documentation

6.22.1.1 void DeInit_Sim (void * *usrData*)

Definition at line 162 of file ReadWrite_sim.c.

6.22.1.2 void Init_Sim (void * *usrData*)

Definition at line 157 of file ReadWrite_sim.c.

6.22.1.3 T_osUint8 RDByte_Sim (void * *usrData,* T_osUint8 *devAddr,* T_osUint8 *nAddr*)

Definition at line 167 of file ReadWrite_sim.c.

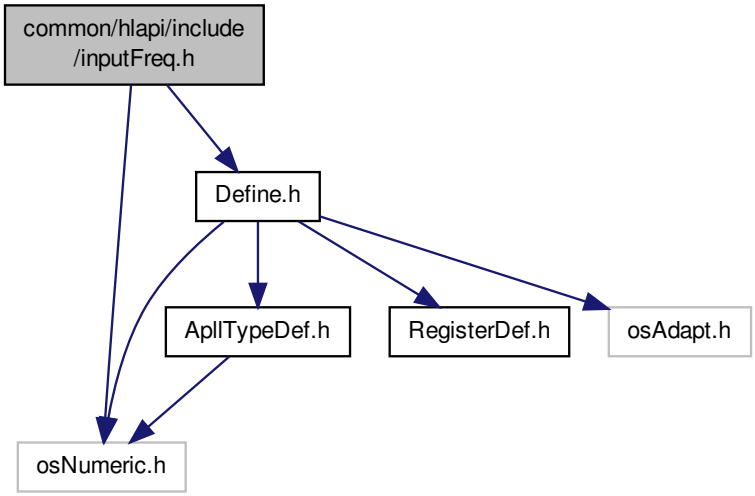
6.22.1.4 void WRByte_Sim (void * *usrData,* T_osUint8 *devAddr,* T_osUint8 *nAddr,* T_osUint8 *nData*)

Definition at line 222 of file ReadWrite_sim.c.

6.23 common/hlapi/include/inputFreq.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
```

Include dependency graph for inputFreq.h:



Functions

- void IDTHlApi_ConfigureInput1PPS (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Function to provision input port for 1PPS (1 Hz)
- void IDTHlApi_ConfigureInputAmi (T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_idtAmiInputFrequencyBitFieldValue amiSignal)

  Function to provision input AMI port.
- void IDTHlApi_ConfigureInputFrequency (T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_osUint32 nFrequency)

  Function to provision input port.
- void IDTHlApi_DisableAllInputPriorities (T_idtDrvHdlr hdlr)

  Function to disable all inputs priorities.

6.23.1 Function Documentation

6.23.1.1 void IDTHlApi_ConfigureInput1PPS (T_idtDrvHdlr *hdlr,* T_osUint8 *nInputNo*)

Function to provision input port for 1PPS (1 Hz)

Provision input for 1PPS (1 Hz)

Parameters

hdlr	driver handler
nInputNo	input port number

Definition at line 180 of file inputFreq.c.

6.23.1.2 void IDTHlApi_ConfigureInputAmi (T_idtDrvHdlr *hdlr,* T_osUint8 *nInputNo,* T_idtAmiInputFrequencyBitFieldValue *amiSignal*)

Function to provision input AMI port.

Provision AMI input

Parameters

hdlr	driver handler
nInputNo	input port number
amiSignal	Selected AMI signal /see T_idtAmiInputFrequencyBitFieldValue

Definition at line 222 of file inputFreq.c.

6.23.1.3 void IDTHlApi_ConfigureInputFrequency (T_idtDrvHdlr *hdlr,* T_osUint8 *nInputNo,* T_osUint32 *nFrequency*)

Function to provision input port.

Provision input port specify input freqency

Parameters

hdlr	driver handler
nInputNo	input port number
nFrequency	input frequency (in kHz)

Definition at line 357 of file inputFreq.c.

6.23.1.4 void IDTHlApi_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 priorites tells DPLLs to ignore all inputs.

Use this function when a custom priority is desired. This functions will clear all input priorites first. Input priorities can then be set by calling IDTInput_SetPriority function.

Parameters

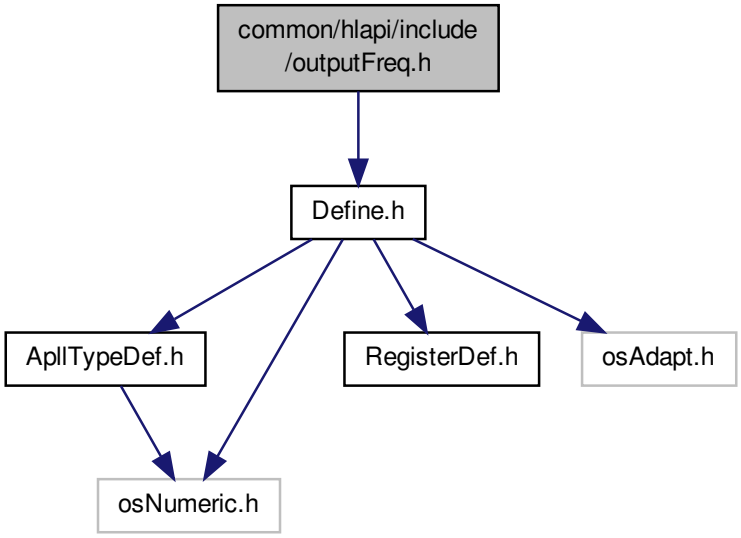
hdlr	Device driver handle

Definition at line 470 of file inputFreq.c.

6.24 common/hlapi/include/outputFreq.h File Reference

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

Include dependency graph for outputFreq.h:



Enumerations

- enum outputFrequency_t {
  OutFreq_Invalid, OutFreq_64k, OutFreq_8k, OutFreq_2k,
  OutFreq_400, OutFreq_1, OutFreq_24576k, OutFreq_12288k,
  OutFreq_6144k, OutFreq_4096k, OutFreq_2048k, OutFreq_32768k,
  OutFreq_16384k, OutFreq_8192k, OutFreq_24704k, OutFreq_12352k,
  OutFreq_6176k, OutFreq_3088k, OutFreq_1544k, OutFreq_34368k,
  OutFreq_44736k, OutFreq_26000k, OutFreq_13000k, OutFreq_30720k,
  OutFreq_15360k, OutFreq_7680k, OutFreq_3840k, OutFreq_37056k,
  OutFreq_18528k, OutFreq_9264k, OutFreq_4632k, OutFreq_40000k,
  OutFreq_20000k, OutFreq_10000k, OutFreq_5000k, OutFreq_622080k,
  OutFreq_625000k, OutFreq_625000k_66_64, OutFreq_311040k, OutFreq_312500k,
  OutFreq_312500k_66_64, OutFreq_155520k, OutFreq_156250k, OutFreq_156250k_66_64,
  OutFreq_77760k, OutFreq_51840k, OutFreq_38880k, OutFreq_25920k,
  OutFreq_125000k, OutFreq_125000k_66_64, OutFreq_19440k, OutFreq_25000k,
  OutFreq_25000k_66_64, OutFreq_6480k, OutFreq_24883_DOT_2k, OutFreq_78125k,
  OutFreq_78125k_66_64, OutFreq_124416k, OutFreq_MAX }

  Enumerated type listing output frequencies.
- enum T_sonetGigeMode { sonetGigeMode_Matching, sonetGigeMode_AllFromDpll1, sonetGigeMode_AllFromDpll2, sonetGigeMode_MAX }

  Enumerated type listing sonetGige modes.

Functions

- int IDTHlApi_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.

- void IDTHlApi_DisableAllOutputs (T_idtDrvHdlr hdlr)

  Function to disable all outputs.
- void IDTHlApi_ClearProvisioning (T_idtDrvHdlr hdlr)

  Function to clear saved provisioning used by high level API.
- void IDTHlApi_PrintOutput (T_idtDrvHdlr hdlr)
- void IDTHlApi_PrintInput (T_idtDrvHdlr hdlr)

6.24.1 Enumeration Type Documentation

6.24.1.1 enum outputFrequency_t

Enumerated type listing output frequencies.

Enumerator

OutFreq_Invalid Output frequency invalid

OutFreq_64k Output frequency is 64 kHz

OutFreq_8k Output frequency is 8 kHz

OutFreq_2k Output frequency is 2 kHz

OutFreq_400 Output frequency is 400 Hz

OutFreq_1 Output frequency is 1 Hz (1 PPS)

OutFreq_24576k Output frequency is 24.576 MHz

OutFreq_12288k Output frequency is 12.288 MHz

OutFreq_6144k Output frequency is 6.144 MHz

OutFreq_4096k Output frequency is 4.096 MHz

OutFreq_2048k Output frequency is 2.048 MHz

OutFreq_32768k Output frequency is 32.768 MHz

OutFreq_16384k Output frequency is 16.384MHz

OutFreq_8192k Output frequency is 8.192 MHz

OutFreq_24704k Output frequency is 24.704 MHz

OutFreq_12352k Output frequency is 12.352 MHz

OutFreq_6176k Output frequency is 6.176 MHz

OutFreq_3088k Output frequency is 3.088 MHz

OutFreq_1544k Output frequency is 1.544 MHz

OutFreq_34368k Output frequency is 34.368 MHz

OutFreq_44736k Output frequency is 44.736 MHz

OutFreq_26000k Output frequency is 26 MHz

OutFreq_13000k Output frequency is 13 MHz

OutFreq_30720k Output frequency is 30.72 MHz

OutFreq_15360k Output frequency is 15.36 MHz

OutFreq_7680k Output frequency is 7.68 MHz

OutFreq_3840k Output frequency is 3.84 MHz

OutFreq_37056k Output frequency is 37.056 MHz

OutFreq_18528k Output frequency is 18.528 MHz

OutFreq_9264k Output frequency is 9.264 MHz

OutFreq_4632k Output frequency is 4.632 MHz

OutFreq_40000k Output frequency is 40 MHz

OutFreq_20000k Output frequency is 20 MHz
OutFreq_10000k Output frequency is 10 MHz
OutFreq_5000k Output frequency is 5 MHz
OutFreq_622080k Output frequency is 622.08 MHz
OutFreq_625000k Output frequency is 625 MHz
OutFreq_625000k_66_64 Output frequency is 625∗(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∗(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∗(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∗(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∗(66/64) MHz
OutFreq_6480k Output frequency is 6.48 MHz
OutFreq_24883_DOT_2k Output frequency is 24.8832 MHz
OutFreq_78125k Output frequency is 78.125 MHz
OutFreq_78125k_66_64 Output frequency is 80.56640625 MHz
OutFreq_124416k Output frequency is 124.416 MHz
OutFreq_MAX

Definition at line 46 of file outputFreq.h.

6.24.1.2 enum T_sonetGigeMode

Enumerated type listing sonetGige modes.

Enumerator

sonetGigeMode_Matching DPLL #1 to APLL #1 and DPLL #2 to APLL #2
sonetGigeMode_AllFromDpll1 DPLL #1 to APLL #1 and DPLL #1 to APLL #2
sonetGigeMode_AllFromDpll2 DPLL #2 to APLL #1 and DPLL #2 to APLL #2
sonetGigeMode_MAX

Definition at line 113 of file outputFreq.h.

6.24.2 Function Documentation

6.24.2.1 void IDTHlApi_ClearProvisioning (T_idtDrvHdlr *hdlr*)

Function to clear saved provisioning used by high level API.

Parameters

hdlr	Device driver handle

Definition at line 1464 of file outputFreq.c.

6.24.2.2 int IDTHlApi_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 freqency and DPLL to use

Parameters

hdlr	driver handler
nOutN	Output port (1-11)
nOutFreq	Output frequency /see outputFrequency_t for list of supported frequencies
dpllInstance	Dpll Instance (DPLL_INSTANCE_1, DPLL_INSTANCE_2, specify DPLL_INSTANCE_MAX for no specific DPLL)
sonetGigeMode	sonetGige mux configuration.

See Also

T_sonetGigeMode This parameter and dpllInstance determine what ouptut path(s) software may use to generate specified output frequency. The following table illustrates the available paths,

dpllInstance	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

apllConfSel	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
apllOutputSig-Type	APLL output signal type, set to APLL_OUT_SIG_MAX if output port does not go through APLL

apllClkSource	APLL clock source (internal or external), set to APLL_CLK_SEL_MAX if output port does not go through APLL

Returns

0-OK, 1-Error, unable to configure output path mux

Definition at line 1261 of file outputFreq.c.

6.24.2.3 void IDTHIApi_DisableAllOutputs (T_idtDrvHdlr *hdlr*)

Function to disable all outputs.

Parameters

hdlr	driver handler

Definition at line 1443 of file outputFreq.c.

6.24.2.4 void IDTHIApi_PrintInput (T_idtDrvHdlr *hdlr*)

Definition at line 948 of file outputFreqString.c.

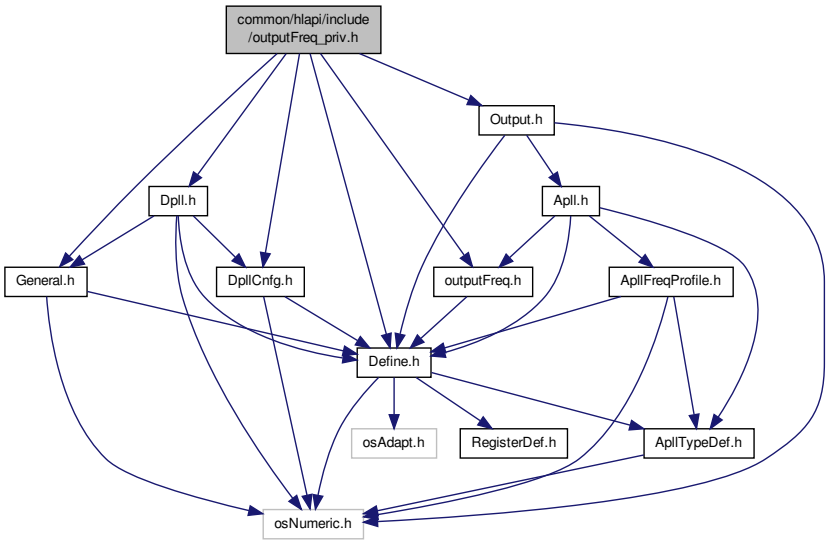
6.24.2.5 void IDTHIApi_PrintOutput (T_idtDrvHdlr *hdlr*)

Definition at line 1011 of file outputFreqString.c.

6.25 common/hlapi/include/outputFreq_priv.h File Reference

```
#include "Define.h"
#include "General.h"
#include "DpllCnfg.h"
#include "Output.h"
#include "Dpll.h"
#include "outputFreq.h"
```

Include dependency graph for outputFreq_priv.h:



Data Structures

- struct pathSelectorConfig_t

  Structure containing select configuration. Used to translate between frequency and output path configuration.
- struct pathSelectorIndex_t

  Structure to index to path selector configs.

Macros

- #define SET_BIT(array, bit, set) array[bit] = set

  set a bit within a bit array
- #define CHECK_BIT(array, bit) (array[bit] != OutFreq_Invalid)

  check a bit within a bit array

Typedefs

- typedef int validFreq_t [OutFreq_MAX]

  Array type for storing frequency usage.

Enumerations

- enum pathSelectors_t {
  Selector_Network, Selector_T0DpllSelA, Selector_T0DpllSelB, Selector_T4DpllSelA,
  Selector_T4DpllSelB, Selector_SonetGigEth, Selector_Any, Selector_MAX }

  Enumerated type listing output path selectors.

Functions

- void IDTHlApi_ConvertFromFreqListToFreqArray (outputFrequency_t frequencies[], validFreq_t validFrequencies)

  Convert from a frequency list to frequency array.
- void IDTHlApi_ApplyFrequencies (const outputFrequency_t *frequencies, validFreq_t validFrequencies, int value)

  Function to set status of frequency in frequency array.
- int IDTHlApi_IsFreqArrayInFreqList (const outputFrequency_t *frequencies, validFreq_t validFrequencies)
- int IDTHlApi_IsFrequencyInFreqList (outputFrequency_t frequency, const outputFrequency_t validFrequencies[])

  Function to verify that frequency is within frequency list.
- void IDTHlApi_PrintAvailFrequencies (validFreq_t frequencyArray)

  Function to print frequency array in string format.
- void IDTHlApi_PrintFrequencyList (const outputFrequency_t frequencies[])

  Function to print frequency list in string format.
- void IDTHlApi_PrintConfiguration (T_IDTPathConfiguration pathConfig)

  Function to print output path configuration to a string format.
- void IDTHlApi_PrintOption (pathSelectorConfig_t option, T_osUint8 printPort)

  Function to print output path configuration option to string format.
- void IDTHlApi_PrintSelectedOutputPath (T_outputPathType nPath)

  Function to display selected output port path.
- void IDTHlApi_PrintOutputHwConfig (networkType_t network, T_osUint8 nOutN, T_osUint8 internalPort, T_osUint8 isEnabled, T_osUint8 hasApll, T_idtApllDividerValue apllDivVal, T_idtApllXtalType xtalType, T_osUint8 outDiv, T_outputPathType outPath, T_idtDpllInstance dpllInstOrSonetGigEthInst, T_idtDpllInstance sonetGigEthInput, T_idtSonetGigEthOutput sonetGigEthMode, T_idtDpllOutputSelectorA dpllSelA, T_idtDpllOutputSelectorB dpllSelB)

Variables

- const char * IDTHlApi_Frequency2String_c [OutFreq_MAX]

6.25.1 Macro Definition Documentation

6.25.1.1 #define CHECK_BIT(*array, bit*) (array[bit] != OutFreq_Invalid)

check a bit within a bit array

Definition at line 89 of file outputFreq_priv.h.

6.25.1.2 #define SET_BIT(*array, bit, set*) array[bit] = set

set a bit within a bit array

Definition at line 85 of file outputFreq_priv.h.

6.25.2 Typedef Documentation

6.25.2.1 typedef int validFreq_t[OutFreq_MAX]

Array type for storing frequency usage.

Definition at line 104 of file outputFreq_priv.h.

6.25.3 Enumeration Type Documentation

6.25.3.1 enum pathSelectors_t

Enumerated type listing output path selectors.

Enumerator

Selector_Network Network selector (SDH/SONET)

Selector_T0DpllSelA T0 Dpll selector A (16E1/16T1/GSM/OBSAI)

Selector_T0DpllSelB T0 Dpll selector B (12E1/GPS/E3/T3)

Selector_T4DpllSelA T4 Dpll selector A (16E1/16T1/GSM/GPS)

Selector_T4DpllSelB T4 Dpll selector B (12E1/24T1/E3/T3)

Selector_SonetGigEth APLL selector (T0 622/T4 622/T0 625/ T4 625/T0 625(66/64)/ T4 625(66/64))

Selector_Any For 77.76 / Eth always available frequencies

Selector_MAX

Definition at line 53 of file outputFreq_priv.h.

6.25.4 Function Documentation

6.25.4.1 void IDTHlApi_ApplyFrequencies (const outputFrequency_t * *frequencies,* validFreq_t *validFrequencies,* int *value*)

Function to set status of frequency in frequency array.

Parameters

frequencies	Constant frequency list
validFrequencies	
value	Status to set for frequency array

Definition at line 79 of file outputFreqUtil.c.

6.25.4.2 void IDTHlApi_ConvertFromFreqListToFreqArray (outputFrequency_t *frequencies[],* validFreq_t *validFrequencies*)

Convert from a frequency list to frequency array.

Parameters

frequencies	Frequency list to convert
validFrequencies	Output frequency array

Definition at line 57 of file outputFreqUtil.c.

6.25.4.3 int IDTHlApi_IsFreqArrayInFreqList (const outputFrequency_t * *frequencies,* validFreq_t *validFrequencies*)

6.25.4.4 int IDTHlApi_IsFrequencyInFreqList (outputFrequency_t *frequency,* const outputFrequency_t *validFrequencies[]*)

Function to verify that frequency is within frequency list.

Parameters

frequency	Frequency to check
validFrequencies	Frequency list to check against

Returns

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

Definition at line 124 of file outputFreqUtil.c.

6.25.4.5 void IDTHlApi_PrintAvailFrequencies (validFreq_t *frequencyArray*)

Function to print frequency array in string format.

Parameters

frequencyArray	Frequency array

Definition at line 317 of file outputFreqString.c.

6.25.4.6 void IDTHlApi_PrintConfiguration (T_IDTPathConfiguration *pathConfig*)

Function to print output path configuration to a string format.

Parameters

pathConfig	Output path config

Definition at line 358 of file outputFreqString.c.

6.25.4.7 void IDTHlApi_PrintFrequencyList (const outputFrequency_t *frequencies[]*)

Function to print frequency list in string format.

Parameters

frequencies	Frequency list

Definition at line 339 of file outputFreqString.c.

6.25.4.8 void IDTHlApi_PrintOption (pathSelectorConfig_t *option,* T_osUint8 *printPort*)

Function to print output path configuration option to string format.

Parameters

option	Output path configuration option.
printPort	Flag to indicate whether port information is also displayed

Definition at line 387 of file outputFreqString.c.

6.25.4.9 void IDTHlApi_PrintOutputHwConfig (networkType_t *network*, T_osUint8 *nOutN*, T_osUint8 *internalPort*, T_osUint8 *isEnabled*, T_osUint8 *hasApll*, T_idtApllDividerValue *apllDivVal*, T_idtApllXtalType *xtalType*, T_osUint8 *outDiv*, T_outputPathType *outPath*, T_idtDpllInstance *dpllInstOrSonetGigEthInst*, T_idtDpllInstance *sonetGigEthInput*, T_idtSonetGigEthOutput *sonetGigEthMode*, T_idtDpllOutputSelectorA *dpllSelA*, T_idtDpllOutputSelectorB *dpllSelB*)

Definition at line 581 of file outputFreqString.c.

6.25.4.10 void IDTHlApi_PrintSelectedOutputPath (T_outputPathType *nPath*)

Function to display selected output port path.

Parameters

nPath	output port path

Definition at line 443 of file outputFreqString.c.

6.25.5 Variable Documentation

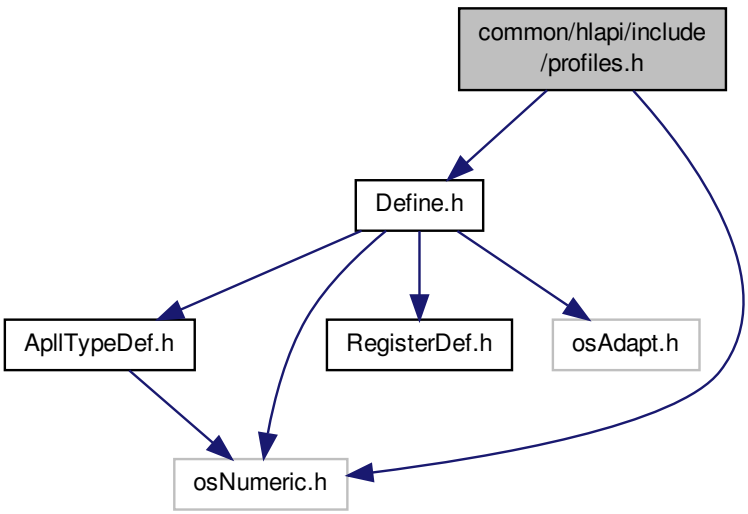
6.25.5.1 const char* IDTHlApi_Frequency2String_c[OutFreq_MAX]

Definition at line 77 of file outputFreqString.c.

6.26 common/hlapi/include/profiles.h File Reference

```
#include "Define.h"
#include "osNumeric.h"
```

Include dependency graph for profiles.h:



Functions

- void IDTHlApi_gr253SonetStratum3 (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  DPLL configuration for GR-253 (SONET) Stratum 3.
- void IDTHlApi_gr1244Stratum3 (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  DPLL configuration for GR-1244 Stratum 3.
- void IDTHlApi_smc (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  DPLL configuration for SMC.
- void IDTHlApi_g8262EecOption1 (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  DPLL configuration for G-8262 Option 1.
- void IDTHlApi_g8262EecOption2 (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  DPLL configuration for G-8262 Option 2.
- void IDTHlApi_g813Option1 (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  DPLL configuration for G-813 Option 1.
- void IDTHlApi_g813Option2 (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  DPLL configuration for G-813 Option 2.
- void IDTHlApi_pps (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  DPLL configuration for 1 pulse per second (PPS)
- void IDTHlApi_wideband (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  DPLL configuration for wideband.

6.26.1 Function Documentation

6.26.1.1 void IDTHlApi_g813Option1 (T_idtDrvHdlr *hdlr*, T_idtDpllInstance *nDpll*)

DPLL configuration for G-813 Option 1.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 396 of file profiles.c.

6.26.1.2 void IDTHlApi_g813Option2 (T_idtDrvHdlr *hdlr*, T_idtDpllInstance *nDpll*)

DPLL configuration for G-813 Option 2.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 408 of file profiles.c.

6.26.1.3 void IDTHlApi_g8262EecOption1 (T_idtDrvHdlr *hdlr*, T_idtDpllInstance *nDpll*)

DPLL configuration for G-8262 Option 1.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 258 of file profiles.c.

6.26.1.4 void IDTHlApi_g8262EecOption2 (T_idtDrvHdlr *hdlr*, T_idtDpllInstance *nDpll*)

DPLL configuration for G-8262 Option 2.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 327 of file profiles.c.

6.26.1.5 void IDTHlApi_gr1244Stratum3 (T_idtDrvHdlr *hdlr*, T_idtDpllInstance *nDpll*)

DPLL configuration for GR-1244 Stratum 3.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 120 of file profiles.c.

6.26.1.6 void IDTHlApi_gr253SonetStratum3 (T_idtDrvHdlr *hdlr*, T_idtDpllInstance *nDpll*)

DPLL configuration for GR-253 (SONET) Stratum 3.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 51 of file profiles.c.

6.26.1.7 void IDTHlApi_pps (T_idtDrvHdlr *hdlr*, T_idtDpllInstance *nDpll*)

DPLL configuration for 1 pulse per second (PPS)

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 421 of file profiles.c.

6.26.1.8 void IDTHlApi_smc (T_idtDrvHdlr *hdlr*, T_idtDpllInstance *nDpll*)

DPLL configuration for SMC.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 189 of file profiles.c.

6.26.1.9 void IDTHlApi_wideband (T_idtDrvHdlr *hdlr*, T_idtDpllInstance *nDpll*)

DPLL configuration for wideband.

Parameters

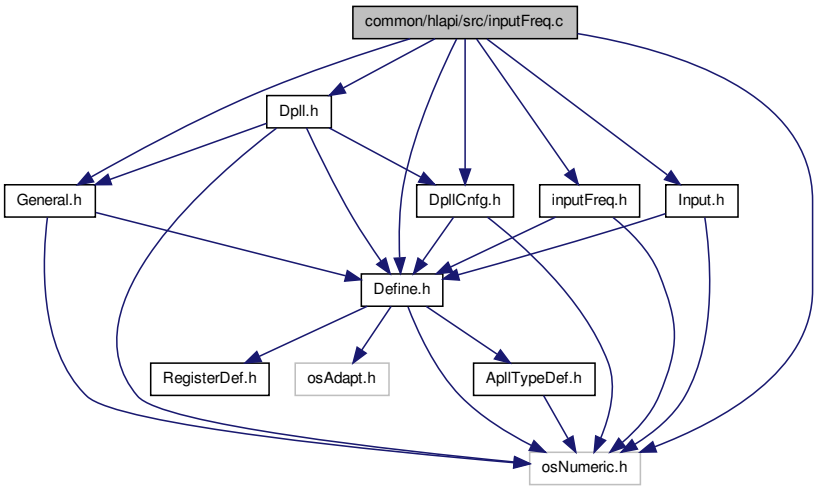
hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 523 of file profiles.c.

6.27 common/hlapi/src/inputFreq.c File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "Input.h"
#include "General.h"
#include "DpllCnfg.h"
#include "Dpll.h"
#include "inputFreq.h"
```

Include dependency graph for inputFreq.c:



Data Structures

- struct T_IDTFreq2bfVal

  Structure to contain translation information for frequency to bitfield value.
- struct T_idtHfDivEntry

  Structure to use as high frequency (HF) divider information.

Functions

- void IDTHlApi_ConfigureInput1PPS (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

Function to provision input port for 1PPS (1 Hz)

- void IDTHlApi_ConfigureInputAmi (T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_idtAmiInputFrequencyBitFieldValue amiSignal)

  Function to provision input AMI port.

- void IDTHlApi_ConfigureInputFrequency (T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_osUint32 nFrequency)

  Function to provision input port.

- void IDTHlApi_DisableAllInputPriorities (T_idtDrvHdlr hdlr)

  Function to disable all inputs priorities.

6.27.1 Function Documentation

6.27.1.1 void IDTHlApi_ConfigureInput1PPS (T_idtDrvHdlr *hdlr*, T_osUint8 *nInputNo*)

Function to provision input port for 1PPS (1 Hz)

Parameters

hdlr	driver handler
nInputNo	input port number

Definition at line 180 of file inputFreq.c.

6.27.1.2 void IDTHlApi_ConfigureInputAmi (T_idtDrvHdlr *hdlr*, T_osUint8 *nInputNo*, T_idtAmiInputFrequencyBitFieldValue *amiSignal*)

Function to provision input AMI port.

Parameters

hdlr	driver handler
nInputNo	input port number
amiSignal	Selected AMI signal /see T_idtAmiInputFrequencyBitFieldValue

Definition at line 222 of file inputFreq.c.

6.27.1.3 void IDTHlApi_ConfigureInputFrequency (T_idtDrvHdlr *hdlr*, T_osUint8 *nInputNo*, T_osUint32 *nFrequency*)

Function to provision input port.

Parameters

hdlr	driver handler
nInputNo	input port number
nFrequency	input frequency (in kHz)

Definition at line 357 of file inputFreq.c.

6.27.1.4 void IDTHlApi_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 priorites tells DPLLs to ignore all inputs.

Use this function when a custom priority is desired. This functions will clear all input priorites first. Input priorities

can then be set by calling IDTInput_SetPriority function.

Parameters

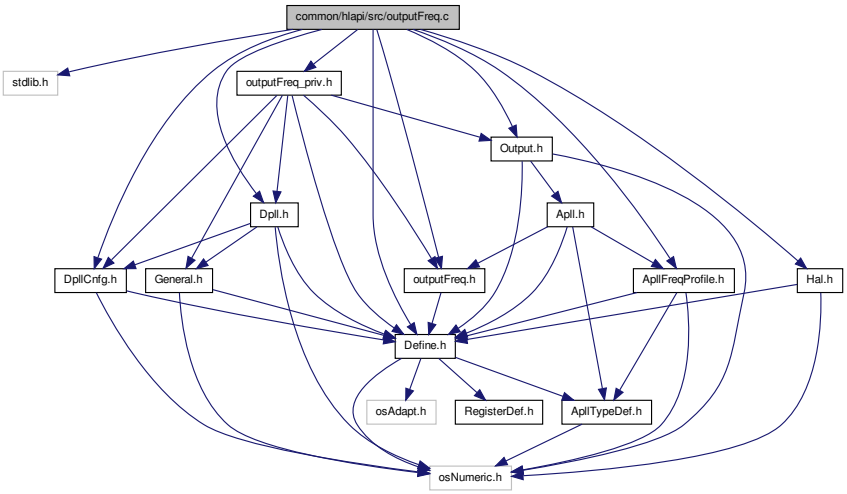
hdlr	Device driver handle

Definition at line 470 of file inputFreq.c.

6.28 common/hlapi/src/outputFreq.c File Reference

```
#include <stdlib.h>
#include "Define.h"
#include "Dpll.h"
#include "Hal.h"
#include "Output.h"
#include "DpllCnfg.h"
#include "outputFreq.h"
#include "outputFreq_priv.h"
#include "ApllFreqProfile.h"
```

Include dependency graph for outputFreq.c:



Macros

- #define MAX_SUPPORTED_FREQUENCIES 9
- #define OUTMUX_ENTRY(sel, selval, div, path, inst) { Selector_##sel, selval, div, OutputPath_##path, inst }

  Macro to reduce size of output path configuration entries. Used to reduce size of table so that it fits inside 80 columns.

Functions

- void IDTHlApi_RetrieveOutputPathMuxFromHardware (T_idtDrvHdlr hdlr, T_IDTPathConfiguration *pathConfig_p)

  Function to retrieve output path configuration from hardware.

- int IDTHlApi_ConfigureOutput (T_idtDrvHdlr hdlr, T_osUint8 nOutN, outputFrequency_t nOutFreq, T_idtDpll-Instance dpllInstance, T_sonetGigeMode sonetGigeMode, T_idtApllConfigSel apllConfSel, T_idtApllOutput-SigType apllOutputSigType, T_idtApllInputClkSrc apllClkSource)

  Function to provision output port.
- void IDTHlApi_DisableAllOutputs (T_idtDrvHdlr hdlr)

  Function to disable all outputs.
- void IDTHlApi_ClearProvisioning (T_idtDrvHdlr hdlr)

  Function to clear saved provisioning used by high level API.

6.28.1 Macro Definition Documentation

6.28.1.1 #define MAX_SUPPORTED_FREQUENCIES 9

Definition at line 53 of file outputFreq.c.

6.28.1.2 #define OUTMUX_ENTRY(*sel, selval, div, path, inst*) { Selector_##sel, selval, div, OutputPath_##path, inst }

Macro to reduce size of output path configuration entries. Used to reduce size of table so that it fits inside 80 columns.

Definition at line 61 of file outputFreq.c.

6.28.2 Function Documentation

6.28.2.1 void IDTHlApi_ClearProvisioning (T_idtDrvHdlr *hdlr*)

Function to clear saved provisioning used by high level API.

Parameters

hdlr	Device driver handle

Definition at line 1464 of file outputFreq.c.

6.28.2.2 int IDTHlApi_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

hdlr	driver handler
nOutN	Output port (1-11)
nOutFreq	Output frequency /see outputFrequency_t for list of supported frequencies
dpllInstance	Dpll Instance (DPLL_INSTANCE_1, DPLL_INSTANCE_2, specify DPLL_INSTANCE_MAX for no specific DPLL)
sonetGigeMode	sonetGige mux configuration.

See Also

T_sonetGigeMode This parameter and dpllInstance determine what ouptut path(s) software may use to generate specified output frequency. The following table illustrates the available paths,

dpllInstance	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

apllConfSel	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
apllOutputSig-Type	APLL output signal type, set to APLL_OUT_SIG_MAX if output port does not go through APLL
apllClkSource	APLL clock source (internal or external), set to APLL_CLK_SEL_MAX if output port does not go through APLL

Returns

0-OK, 1-Error, unable to configure output path mux

Definition at line 1261 of file outputFreq.c.

6.28.2.3 void IDTHlApi_DisableAllOutputs (T_idtDrvHdlr *hdlr*)

Function to disable all outputs.

Parameters

hdlr	driver handler

Definition at line 1443 of file outputFreq.c.

6.28.2.4 void IDTHlApi_RetrieveOutputPathMuxFromHardware (T_idtDrvHdlr *hdlr,* T_IDTPathConfiguration * *pathConfig_p*)

Function to retrieve output path configuration from hardware.

Parameters

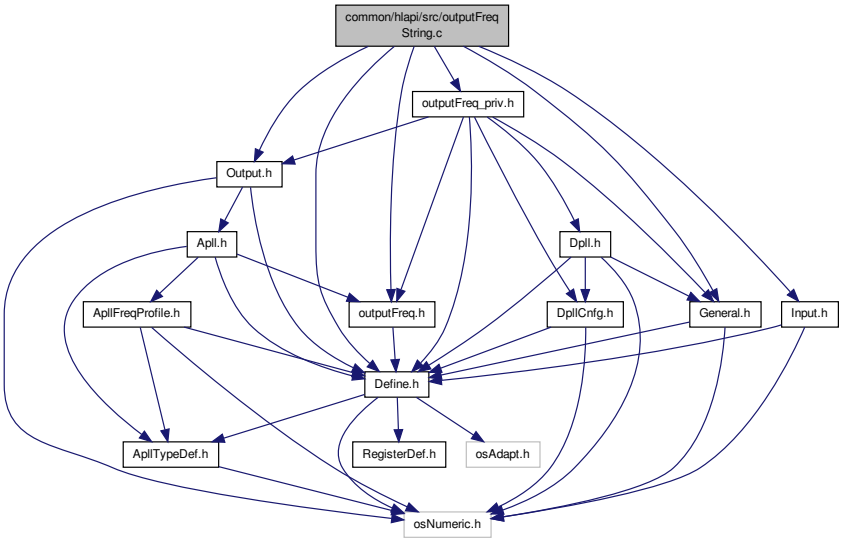
hdlr	Device driver handler
pathConfig_p	Output path configuration

Definition at line 282 of file outputFreq.c.

6.29 common/hlapi/src/outputFreqString.c File Reference

```
#include "Define.h"
#include "Output.h"
#include "Input.h"
#include "General.h"
#include "outputFreq.h"
#include "outputFreq_priv.h"
```

Include dependency graph for outputFreqString.c:



Data Structures

- struct T_apllOutputFrequencyEntry

  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 IDTHlApi_PrintAvailFrequencies (validFreq_t frequencyArray)

  Function to print frequency array in string format.
- void IDTHlApi_PrintFrequencyList (const outputFrequency_t frequencies[])

  Function to print frequency list in string format.
- void IDTHlApi_PrintConfiguration (T_IDTPathConfiguration pathConfig)

  Function to print output path configuration to a string format.
- void IDTHlApi_PrintOption (pathSelectorConfig_t option, T_osUint8 printPort)

  Function to print output path configuration option to string format.
- void IDTHlApi_PrintSelectedOutputPath (T_outputPathType nPath)

  Function to display selected output port path.
- void IDTHlApi_PrintOutputHwConfig (networkType_t network, T_osUint8 nOutN, T_osUint8 internalPort, T_osUint8 isEnabled, T_osUint8 hasApll, T_idtApllDividerValue apllDivVal, T_idtApllXtalType xtalType, T_osUint8 outDiv, T_outputPathType outPath, T_idtDpllInstance dpllInstOrSonetGigEthInst, T_idtDpllInstance sonetGigEthInput, T_idtSonetGigEthOutput sonetGigeMode, T_idtDpllOutputSelectorA dpllSelA, T_idtDpllOutputSelectorB dpllSelB)
- void IDTHlApi_PrintInputHwConfig (T_osUint8 nInputN, T_osUint8 internalPort, networkType_t networkType, T_osUint8 isEnable, T_idtHighFreqencyDivisor nUsage, T_osUint16 nFactor, T_idtInputDividerMux nDivider, T_idtInputFrequencyBitfieldValue nFrequency, T_osUint8 numberOfDpll, T_osUint8 priorities[])
- void IDTHlApi_PrintInput (T_idtDrvHdlr hdlr)
- void IDTHlApi_PrintOutput (T_idtDrvHdlr hdlr)

Variables

- const char * IDTHlApi_Frequency2String_c [OutFreq_MAX]

6.29.1 Enumeration Type Documentation

6.29.1.1 enum T_outFreqLookupIdx

Enumerator

outFreqLookupIdx_Dpll77p76 DPLL 77.76MHz

outFreqLookupIdx_Dpll12E1 DPLL 12E1

outFreqLookupIdx_Dpll16E1 DPLL 16E1

outFreqLookupIdx_Dpll16T1 DPLL 16T1

outFreqLookupIdx_DpllE3 DPLL E3

outFreqLookupIdx_DpllT3 DPLL T3

outFreqLookupIdx_DpllGsm DPLL GSM

outFreqLookupIdx_DpllObsai DPLL OBSAI

outFreqLookupIdx_DpllEth DPLL 24T1

outFreqLookupIdx_Dpll24T1 DPLL 24T1

outFreqLookupIdx_DpllGps DPLL GPS

outFreqLookupIdx_SonetGigEthSonet SONET/GIGE SONET

outFreqLookupIdx_SonetGigEthEth SONET/GIGE ETH

outFreqLookupIdx_SonetGigEthFec SONET/GIGE FEC

outFreqLookupIdx_MAX

Definition at line 45 of file outputFreqString.c.

6.29.2 Function Documentation

6.29.2.1 void IDTHlApi_PrintAvailFrequencies (validFreq_t *frequencyArray*)

Function to print frequency array in string format.

Parameters

frequencyArray	Frequency array

Definition at line 317 of file outputFreqString.c.

6.29.2.2 void IDTHlApi_PrintConfiguration (T_IDTPathConfiguration *pathConfig*)

Function to print output path configuration to a string format.

Parameters

pathConfig	Output path config

Definition at line 358 of file outputFreqString.c.

6.29.2.3 void IDTHlApi_PrintFrequencyList (const outputFrequency_t *frequencies[]*)

Function to print frequency list in string format.

Parameters

frequencies	Frequency list

Definition at line 339 of file outputFreqString.c.

6.29.2.4 void IDTHlApi_PrintInput (T_idtDrvHdlr *hdlr*)

Definition at line 948 of file outputFreqString.c.

6.29.2.5 void IDTHlApi_PrintInputHwConfig (T_osUint8 *nInputN,* T_osUint8 *internalPort,* networkType_t *networkType,* T_osUint8 *isEnable,* T_idtHighFreqencyDivisor *nUsage,* T_osUint16 *nFactor,* T_idtInputDividerMux *nDivider,* T_idtInputFrequencyBitfieldValue *nFrequency,* T_osUint8 *numberOfDpll,* T_osUint8 *priorities[]*)

Definition at line 773 of file outputFreqString.c.

6.29.2.6 void IDTHlApi_PrintOption (pathSelectorConfig_t *option,* T_osUint8 *printPort*)

Function to print output path configuration option to string format.

Parameters

option	Output path configuration option.
printPort	Flag to indicate whether port information is also displayed

Definition at line 387 of file outputFreqString.c.

6.29.2.7 void IDTHlApi_PrintOutput (T_idtDrvHdlr *hdlr*)

Definition at line 1011 of file outputFreqString.c.

6.29.2.8 void IDTHlApi_PrintOutputHwConfig (networkType_t *network,* T_osUint8 *nOutN,* T_osUint8 *internalPort,* T_osUint8 *isEnabled,* T_osUint8 *hasApll,* T_idtApllDividerValue *apllDivVal,* T_idtApllXtalType *xtalType,* T_osUint8 *outDiv,* T_outputPathType *outPath,* T_idtDpllInstance *dpllInstOrSonetGigEthInst,* T_idtDpllInstance *sonetGigEthInput,* T_idtSonetGigEthOutput *sonetGigeMode,* T_idtDpllOutputSelectorA *dpllSelA,* T_idtDpllOutputSelectorB *dpllSelB*)

Definition at line 581 of file outputFreqString.c.

6.29.2.9 void IDTHlApi_PrintSelectedOutputPath (T_outputPathType *nPath*)

Function to display selected output port path.

Parameters

nPath	output port path

Definition at line 443 of file outputFreqString.c.

6.29.3 Variable Documentation

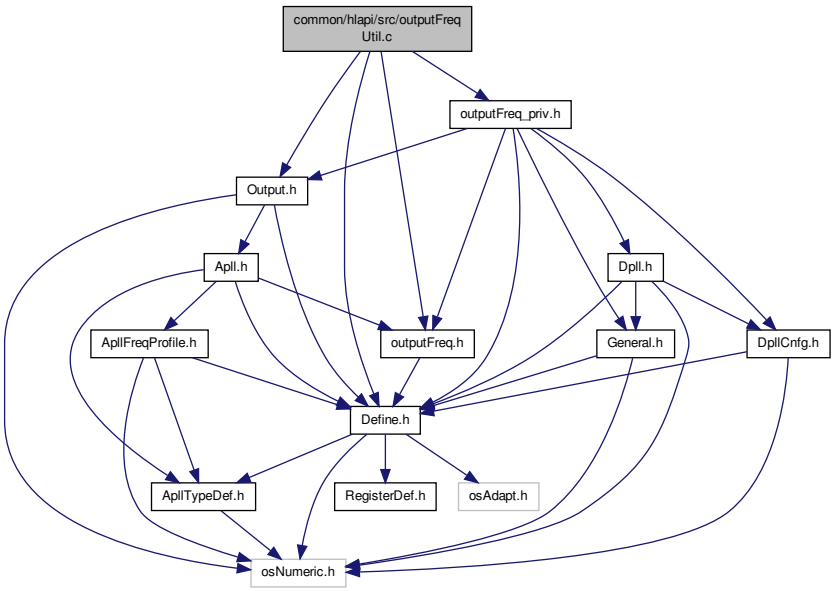
6.29.3.1 const char* IDTHlApi_Frequency2String_c[OutFreq_MAX]

Definition at line 77 of file outputFreqString.c.

6.30 common/hlapi/src/outputFreqUtil.c File Reference

```
#include "Define.h"
#include "Output.h"
#include "outputFreq.h"
#include "outputFreq_priv.h"
```

Include dependency graph for outputFreqUtil.c:



Functions

- void IDTHlApi_ConvertFromFreqListToFreqArray (outputFrequency_t frequencies[], validFreq_t validFrequencies)

  Convert from a frequency list to frequency array.
- void IDTHlApi_ApplyFrequencies (const outputFrequency_t *frequencies, validFreq_t validFrequencies, int value)

  Function to set status of frequency in frequency array.
- int IDTHlApi_IsFreqArrayInFreqList (const outputFrequency_t frequencies[], validFreq_t validFrequencies)

  Function to compare group of frequencies with used frequency list.
- int IDTHlApi_IsFrequencyInFreqList (outputFrequency_t frequency, const outputFrequency_t validFrequencies[])

  Function to verify that frequency is within frequency list.

6.30.1 Function Documentation

6.30.1.1 void IDTHlApi_ApplyFrequencies (const outputFrequency_t * *frequencies*, validFreq_t *validFrequencies*, int *value*)

Function to set status of frequency in frequency array.

Parameters

frequencies	Constant frequency list
validFrequencies	
value	Status to set for frequency array

Definition at line 79 of file outputFreqUtil.c.

6.30.1.2 void IDTHIApi_ConvertFromFreqListToFreqArray (outputFrequency_t *frequencies[],* validFreq_t *validFrequencies*)

Convert from a frequency list to frequency array.

Parameters

frequencies	Frequency list to convert
validFrequencies	Output frequency array

Definition at line 57 of file outputFreqUtil.c.

6.30.1.3 int IDTHIApi_IsFreqArrayInFreqList (const outputFrequency_t *frequencies[],* validFreq_t *validFrequencies*)

Function to compare group of frequencies with used frequency list.

Parameters

frequencies	List to frequencies to check for
validFrequencies	List of frequencies to check against

Returns

0-Not in frequency list, 1-All in frequency list

Definition at line 100 of file outputFreqUtil.c.

6.30.1.4 int IDTHIApi_IsFrequencyInFreqList (outputFrequency_t *frequency,* const outputFrequency_t *validFrequencies[]*)

Function to verify that frequency is within frequency list.

Parameters

frequency	Frequency to check
validFrequencies	Frequency list to check against

Returns

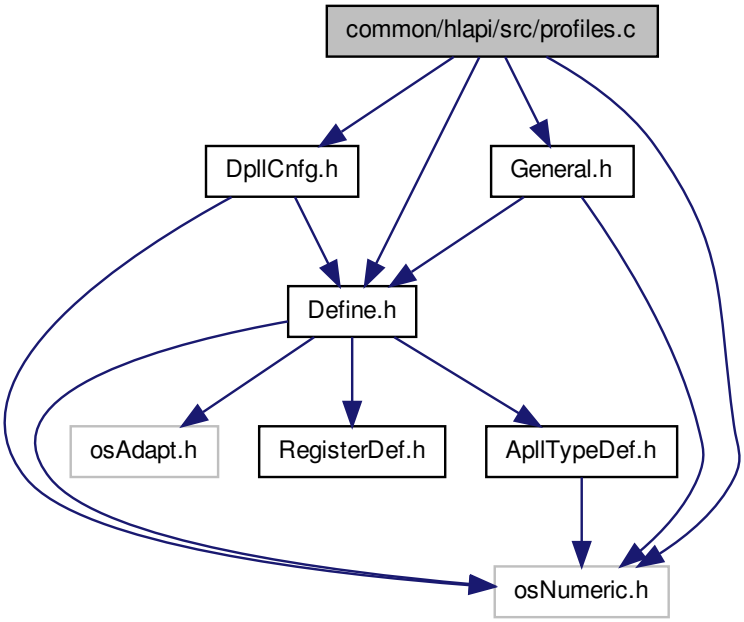
1-Frequency is in list, 0-Frequency is not in list

Definition at line 124 of file outputFreqUtil.c.

6.31 common/hlapi/src/profiles.c File Reference

```
#include "Define.h"
#include "osNumeric.h"
#include "DpllCnfg.h"
#include "General.h"
```

Include dependency graph for profiles.c:



Functions

- void IDTHlApi_gr253SonetStratum3 (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)
  DPLL configuration for GR-253 (SONET) Stratum 3.
- void IDTHlApi_gr1244Stratum3 (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)
  DPLL configuration for GR-1244 Stratum 3.
- void IDTHlApi_smc (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)
  DPLL configuration for SMC.
- void IDTHlApi_g8262EecOption1 (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)
  DPLL configuration for G-8262 Option 1.
- void IDTHlApi_g8262EecOption2 (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)
  DPLL configuration for G-8262 Option 2.
- void IDTHlApi_g813Option1 (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)
  DPLL configuration for G-813 Option 1.
- void IDTHlApi_g813Option2 (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)
  DPLL configuration for G-813 Option 2.
- void IDTHlApi_pps (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)
  DPLL configuration for 1 pulse per second (PPS)
- void IDTHlApi_wideband (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)
  DPLL configuration for wideband.

6.31.1 Detailed Description

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

Definition in file profiles.c.

6.31.2 Function Documentation

6.31.2.1 void IDTHlApi_g813Option1 (T_idtDrvHdlr *hdlr,* T_idtDpllInstance *nDpll*)

DPLL configuration for G-813 Option 1.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 396 of file profiles.c.

6.31.2.2 void IDTHlApi_g813Option2 (T_idtDrvHdlr *hdlr,* T_idtDpllInstance *nDpll*)

DPLL configuration for G-813 Option 2.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 408 of file profiles.c.

6.31.2.3 void IDTHlApi_g8262EecOption1 (T_idtDrvHdlr *hdlr,* T_idtDpllInstance *nDpll*)

DPLL configuration for G-8262 Option 1.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 258 of file profiles.c.

6.31.2.4 void IDTHlApi_g8262EecOption2 (T_idtDrvHdlr *hdlr,* T_idtDpllInstance *nDpll*)

DPLL configuration for G-8262 Option 2.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 327 of file profiles.c.

6.31.2.5 void IDTHlApi_gr1244Stratum3 (T_idtDrvHdlr *hdlr,* T_idtDpllInstance *nDpll*)

DPLL configuration for GR-1244 Stratum 3.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 120 of file profiles.c.

6.31.2.6 void IDTHlApi_gr253SonetStratum3 (T_idtDrvHdlr *hdlr,* T_idtDpllInstance *nDpll*)

DPLL configuration for GR-253 (SONET) Stratum 3.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 51 of file profiles.c.

6.31.2.7 void IDTHlApi_pps (T_idtDrvHdlr *hdlr,* T_idtDpllInstance *nDpll*)

DPLL configuration for 1 pulse per second (PPS)

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 421 of file profiles.c.

6.31.2.8 void IDTHlApi_smc (T_idtDrvHdlr *hdlr,* T_idtDpllInstance *nDpll*)

DPLL configuration for SMC.

Parameters

hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 189 of file profiles.c.

6.31.2.9 void IDTHlApi_wideband (T_idtDrvHdlr *hdlr,* T_idtDpllInstance *nDpll*)

DPLL configuration for wideband.

Parameters

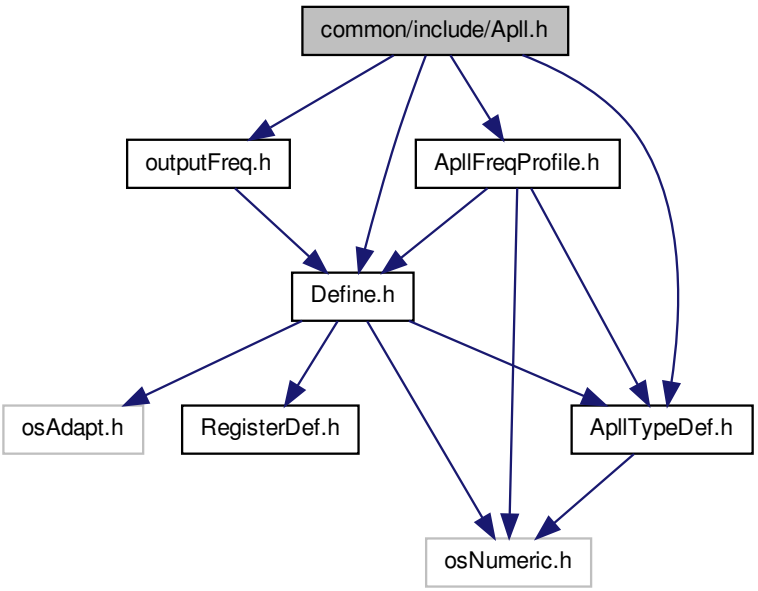
hdlr	Device driver handler
nDpll	DPLL instance

Definition at line 523 of file profiles.c.

6.32 common/include/ApII.h File Reference

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

Include dependency graph for Apll.h:



Data Structures

- struct T_idtApllConfigPerOutput

  Structure containing APLL per output configuration.
- struct T_idtApllConfig

  Structure containing APLL full configuration.
- struct T_idtApllConfig::preDiv_s
- struct T_idtApllConfig::fbDiv_s
- struct T_idtApllConfig::outputDiv_s
- struct T_idtApllConfig::outputDiv_s::qaDiv_s
- struct T_idtApllConfig::outputDiv_s::qbDiv_s
- struct T_idtApllConfig::outputConfig_s
- struct T_idtApllConfig::outputConfig_s::qaConfig_s
- struct T_idtApllConfig::outputConfig_s::qbConfig_s

Macros

- #define IDT_GET_OUTPUT_APLL_CONFIG(hdlr, output, outputApll)

  Utility macro to translate output port to APLL instance.

Functions

- void IDTApll_SetApllConfigPerOutput (T_idtDrvHdlr hdlr, T_idtApllId apllId, T_idtApllOutput output, T_idtApll-ConfigPerOutput *apllConfigPerOutput)

  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 confiugred for the APLL.

- T_osBool IDTApll_ValidXtalInput (const T_idtDrvDeviceConfig *deviceConfig, T_idtApllXtal *xtalList, T_osUint8 numberOfXtal)

  Checks if there is input XTAL information is valid for the device.

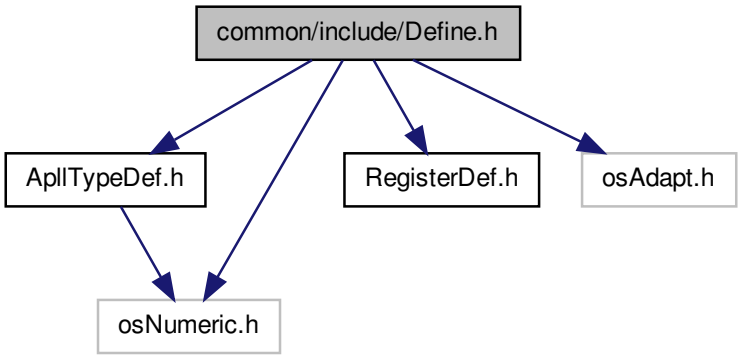
- T_idtApllXtal * IDTApll_GetSupportedXtal (const T_idtDrvDeviceConfig *deviceConfig, T_osUint8 *numberOfXtal)

  Return supported xtal configuration by the device.

6.33 common/include/Define.h File Reference

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

Include dependency graph for Define.h:



Data Structures

- struct T_idtDrvAccess

  Initialization structure detailing device access information for device driver.
- struct T_idtOutputApllConfig

  Structure containing APLL inforatmion.
- 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(apllConfig)

  Utility macro to check if APLL is configured.
- #define IDT_DRV_MAX_INPUT 14

  Define maximum number of supported inputs.
- #define IDT_DRV_MAX_OUTPUT 11

  Define maximum number of supported outputs.
- #define IDT_DRV_MAX_APLL_XTAL 4

  Define maximum number of APLL XTALs per device.
- #define IDT_DRV_MAX_XTAL_PER_APLL 2

  Define maximum number of XTALs per APLL.

Typedefs

- typedef T_osUint8(∗ T_idtReadFunc)(void ∗usrData, T_osUint8 deviceAddr, T_osUint8 addrOff)

  Function type for reading from i2c device.
- typedef void(∗ T_idtWriteFunc)(void ∗usrData, T_osUint8 deviceAddr, T_osUint8 addrOff, T_osUint8 data)

  Function type to writing to i2c device.
- typedef void(∗ T_idtAccessInitFunc)(void ∗userData)

  Function type to initialize device access.
- typedef void(∗ T_idtAccessDeInitFunc)(void ∗userData)

  Function type to deintialize device access.
- typedef T_osUint8(∗ T_idtI2CaddrTransFunc)(T_osUint8 i2cAddr, void ∗transData)

  Function type to translate i2c address used internally by the driver.
- typedef T_osBool(∗ T_idtInputFreqValid)(T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_osUint32 nFrequency, void ∗auxData)

  Function type to sanity check the input frequency.
- typedef T_osBool(∗ T_idtOuputFreqValid)(T_idtDrvHdlr hdlr, T_osUint8 nOutN, T_osUint32 nOutFreq, void ∗auxData)

  Function type to sanity check the output frequency.

Enumerations

- enum T_idtDpllType { Dpll_T0 = 0, Dpll_T4 = 1, Dpll_MAX }

  Enumerated type for selecting DPLLs (T0 or T4)
- enum T_idtDeviceType {
  IDT_82V3910, IDT_82V3911, IDT_82V33930, IDT_8V89316,
  IDT_8V89317, IDT_DEVICE_MAX }

  Enumerated type listing supported device(s)
- enum T_idtDpllInstance { DPLL_INSTANCE_1, DPLL_INSTANCE_2, DPLL_INSTANCE_MAX }

  Define maximum number of supported DPLLs.

6.33.1 Macro Definition Documentation

6.33.1.1 #define APLL_CONFIGURED(*apllConfig*)

Value:

```
((apllConfig.apllInst != APLL_INST_MAX) &&      \
    (apllConfig.apllInput) &&                     \
    (apllConfig.apllOutput != APLL_OUTPUT_MAX) &&  \
    (apllConfig.extOutput))                          \
    ? 1 : 0
```

Utility macro to check if APLL is configured.

Parameters

apllConfig	APLL configuration

Definition at line 81 of file Define.h.

6.33.1.2 #define IDT_DRV_MAX_APLL_XTAL 4

Define maximum number of APLL XTALs per device.

Definition at line 110 of file Define.h.

6.33.1.3 #define IDT_DRV_MAX_INPUT 14

Define maximum number of supported inputs.

Definition at line 104 of file Define.h.

6.33.1.4 #define IDT_DRV_MAX_OUTPUT 11

Define maximum number of supported outputs.

Definition at line 107 of file Define.h.

6.33.1.5 #define IDT_DRV_MAX_XTAL_PER_APLL 2

Define maximum number of XTALs per APLL.

Definition at line 113 of file Define.h.

6.33.1.6 #define MPU_I2C_MODE 0x00

Definition at line 62 of file Define.h.

6.33.1.7 #define MPU_SERIAL_MODE 0x01

Definition at line 63 of file Define.h.

6.33.1.8 #define NULL_APLL_CONFIG

Value:

```
{                           \
    APLL_INST_MAX,          \
    0,                      \
    APLL_OUTPUT_MAX,        \
    0,                      \
}
```

Utility macro to set APLL configuration to NULL for the output path that does not have APLL.

Definition at line 69 of file Define.h.

6.33.2 Typedef Documentation

6.33.2.1 typedef void(* T_idtAccessDeInitFunc)(void *userData)

Function type to deintialize device access.

Optional binding for device access

Parameters

userData	optional user data passed to deinitialization function

Definition at line 156 of file Define.h.

6.33.2.2 typedef void(∗ T_idtAccessInitFunc)(void ∗userData)

Function type to initialize device access.

Optional binding for device access

Parameters

userData	optional user data passed to initialization function

Definition at line 147 of file Define.h.

6.33.2.3 typedef T_osUint8(∗ T_idtI2CaddrTransFunc)(T_osUint8 i2cAddr, void ∗transData)

Function type to translate i2c address used internally by the driver.

Parameters

i2cAddr	7-bit unshifted i2c slave address for the WAN PLL
transData	Helper data for translating

Returns

Translated i2c address used internally by the device driver

Definition at line 164 of file Define.h.

6.33.2.4 typedef T_osBool(∗ T_idtInputFreqValid)(T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_osUint32 nFrequency, void ∗auxData)

Function type to sanity check the input frequency.

Parameters

hdlr	driver handler
nInputNo	input port number
nFrequency	input frequency (in Hz)
auxData	auxiliary data (optional)

Returns

E_osTrue - nFrequency is supported, E_osFalse - nFrequency is not supported

Definition at line 175 of file Define.h.

6.33.2.5 typedef T_osBool(∗ T_idtOuputFreqValid)(T_idtDrvHdlr hdlr, T_osUint8 nOutN, T_osUint32 nOutFreq, void ∗auxData)

Function type to sanity check the output frequency.

Parameters

hdlr	driver handler
nOutN	output port number
nOutFreq	Output frequency /see outputFrequency_t for list of supported frequencies
auxData	auxiliary data (optional)

Returns

E_osTrue - nOutFreq is supported, E_osFalse - nOutFreq is not supported

Definition at line 187 of file Define.h.

6.33.2.6 typedef T_osUint8(* T_idtReadFunc)(void *usrData, T_osUint8 deviceAddr, T_osUint8 addrOff)

Function type for reading from i2c device.

Parameters

usrData	optional data passed to write function
deviceAddr	device address for bus that requires slave address (e.g. i2c)
addrOff	address offset

Returns

read data

Definition at line 129 of file Define.h.

6.33.2.7 typedef void(* T_idtWriteFunc)(void *usrData, T_osUint8 deviceAddr, T_osUint8 addrOff, T_osUint8 data)

Function type to writing to i2c device.

Parameters

usrData	optional user data passed to read function
deviceAddr	for bus that requires slave address (e.g. i2c)
addrOff	address offset
data	data to write

Definition at line 138 of file Define.h.

6.33.3 Enumeration Type Documentation

6.33.3.1 enum T_idtDeviceType

Enumerated type listing supported device(s)

Enumerator

IDT_82V3910 82V3910
IDT_82V3911 82V3911
IDT_82V33930 82V33930
IDT_8V89316 8V89316
IDT_8V89317 8V89317
IDT_DEVICE_MAX

Definition at line 93 of file Define.h.

6.33.3.2 enum T_idtDpllInstance

Define maximum number of supported DPLLs.

Enumerator

DPLL_INSTANCE_1 DPLL instance #1

DPLL_INSTANCE_2 DPLL instance #2

DPLL_INSTANCE_MAX

Definition at line 116 of file Define.h.

6.33.3.3 enum T_idtDpllType

Enumerated type for selecting DPLLs (T0 or T4)

Enumerator

Dpll_T0 Select T0 DPLL

Dpll_T4 Select T4 DPLL

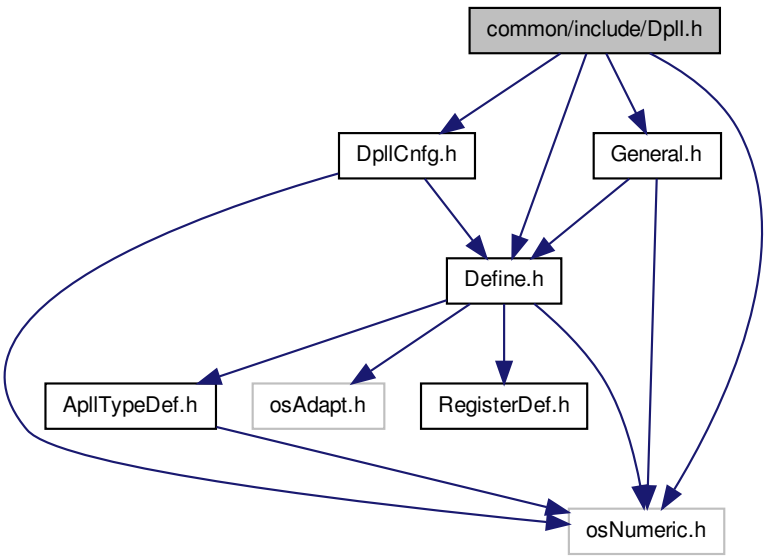
Dpll_MAX

Definition at line 53 of file Define.h.

6.34 common/include/Dpll.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "DpllCnfg.h"
#include "General.h"
```

Include dependency graph for Dpll.h:



Data Structures

- struct T_IDTPathConfiguration

Structure containing output path configuration.

Macros

- #define SonetGigEthSel_NUMMAX 2

  Constant defining number of instances of SonetGigEth selectors.

Enumerations

- enum T_idtT0DpllSelectorA { T0DpllSelA_16E1, T0DpllSelA_16T1, T0DpllSelA_GSM, T0DpllSelA_OBSAI, T0DpllSelA_MAX }

  Enumerated type listing T0 DPLL selector A options.

- enum T_idtT0DpllSelectorB { T0DpllSelB_12E1, T0DpllSelB_GPS, T0DpllSelB_E3, T0DpllSelB_T3, T0DpllSelB_MAX }

  Enumerated type listing T0 DPLL selector B options.

- enum T_idtT4DpllSelectorA { T4DpllSelA_16E1, T4DpllSelA_16T1, T4DpllSelA_GSM, T4DpllSelA_GPS, T4DpllSelA_MAX }

  Enumerated type listing T4 DPLL selector A options.

- enum T_idtT4DpllSelectorB { T4DpllSelB_12E1, T4DpllSelB_24T1, T4DpllSelB_E3, T4DpllSelB_T3, T4DpllSelB_MAX }

  Enumerated type listing T4 DPLL selector B options.

- enum T_idtSonetGigEthSelector { SonetGigEthSel_T0DpllSonet = 0, SonetGigEthSel_T4DpllSonet = 4, SonetGigEthSel_T0Dpll10GbE = 8, SonetGigEthSel_T0Dpll10GbE_6664 = 9, SonetGigEthSel_T4Dpll10GbE = 12, SonetGigEthSel_T4Dpll10GbE_6664 = 13, SonetGigEthSel_MAX }

  Enumerated type listing Sonet/GigE path selection options.

Variables

- const T_IDTPathConfiguration IDTDpll_pathConfigurationNULL_c

  Output path configuration no path constant.

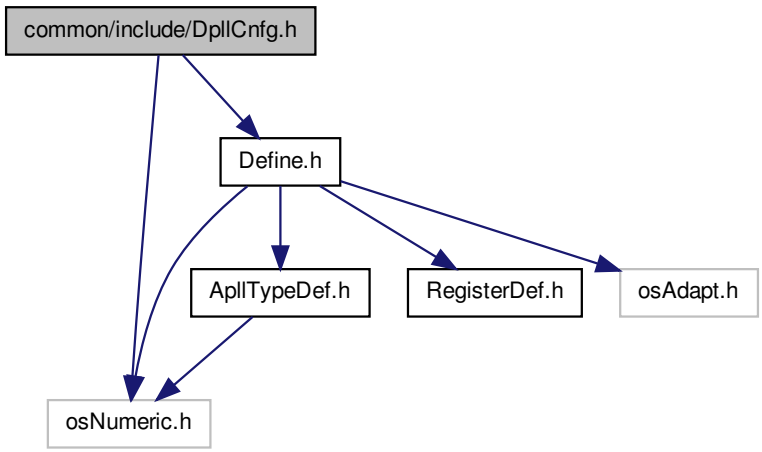
- const T_idtSonetGigEthSelector IDTDpll_sonetGigEthPathConfig2Bitfield [Dpll_MAX][SonetGigEthOutput_-MAX]

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

6.35 common/include/DpllCnfg.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
```

Include dependency graph for DpllCnfg.h:



Macros

- #define DCO_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 { dpllBandwidthLimitStart, dpllBandwidthLimitAcquire, dpllBandwidthLimit-Locked, 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_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.

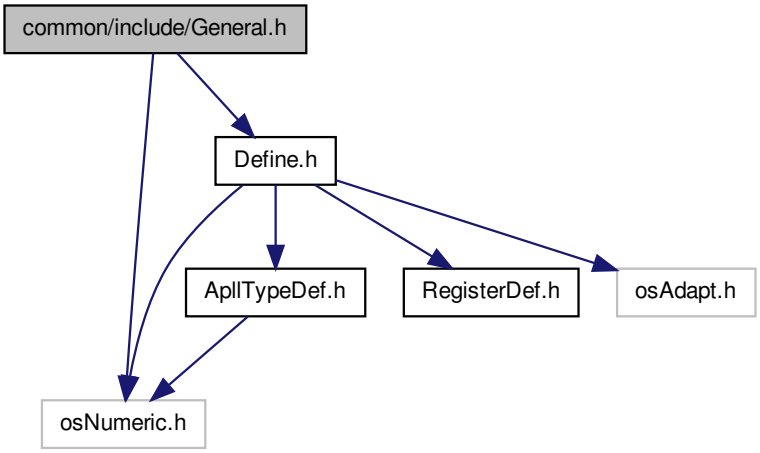
- void IDTDpll_GetPpsPhase (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_idtDpllPpsPhasePostion *nPhase_p, T_idtDpllPpsPulseWidth *nPulse_p)

  Get PPS phase and pulse for DPLL.

6.36 common/include/General.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
```

Include dependency graph for General.h:



Data Structures

- struct T_idtI2CtransData

  I2C address translation information used internally by the driver.

Macros

- #define Activate_Edge_Rising 0
- #define Activate_Edge_Falling 1
- #define INT_Active_Low 0
- #define INT_Active_High 1
- #define NOMINAL_PPT_TO_2COMPL(ppt) (T_osUint32)(((ppt)∗10)/884)

  Conversion from parts per trillion to register unit value for nominal (oscillator) offset.
- #define NOMINAL_2COMPL_TO_PPT(twoCompl) (long)(((twoCompl)∗884)/10)

  Conversion from parts per trillion to register unit value for nominal (oscillator) offset.
- #define NOMINAL_MAX_PPT 94893

  Maximum oscillator offset value.
- #define NOMINAL_MIN_PPT -94893

  Minimum oscillator offset value.

Enumerations

- enum networkType_t { Network_SDH = 0, Network_SONET = 1, Network_MAX }

  Enumerated type listing types of selectable network type.
- enum phaseTimeoutMultiplier_t {
  phaseTimeoutMultiplier_2 = 0, phaseTimeoutMultiplier_4 = 1, phaseTimeoutMultiplier_8 = 2, phaseTimeoutMultiplier_16 = 3,
  phaseTimeoutMultiplier_MAX }

  Enumerated type listing phase alarm timeout multipliers.

Functions

- T_osUint16 IDTGeneral_GetDeviceID (T_idtDrvHdlr hdlr)

  Get ID number of the device.
- void IDTGeneral_SetOscFreqOffset (T_idtDrvHdlr hdlr, long pptOffset)

  Set compensation to the oscillator offset.
- long IDTGeneral_GetOscFreqOffset (T_idtDrvHdlr hdlr)

  Get compensated oscillator offset.
- void IDTGeneral_SetNetworkType (T_idtDrvHdlr hdlr, networkType_t nType)

  Set the network type, SDH or SONET.
- networkType_t IDTGeneral_GetNetworkType (T_idtDrvHdlr hdlr)

  Get the network type, SDH or SONET.
- void IDTGeneral_SetRevertiveMode (T_idtDrvHdlr hdlr, T_osUint8 nEnable)

  Set the device running in Revertive mode or not.
- T_osUint8 IDTGeneral_GetRevertiveMode (T_idtDrvHdlr hdlr)

  Get the device running in Revertive mode or not.
- void IDTGeneral_SetTimeoutValue (T_idtDrvHdlr hdlr, phaseTimeoutMultiplier_t nMultiFactor, T_osUint8 nTimeout)

  Set the value of time out of phase alarm Timeout(second)= nMultiFactor ∗ nTimeout.
- void IDTGeneral_GetTimeoutValue (T_idtDrvHdlr hdlr, phaseTimeoutMultiplier_t ∗nMultiFactor_p, T_osUint8 ∗nTimeout_p)

  Get the value of time out of phase alarm Timeout(second)= nMultiFactor ∗ nTimeout.
- void IDTGeneral_SetOscActiveEdge (T_idtDrvHdlr hdlr, T_osUint8 nEdge)

  Set the activate edge of main oscillator.
- T_osUint8 IDTGeneral_GetOscActiveEdge (T_idtDrvHdlr hdlr)

  Get the activate edge of main oscillator.
- void IDTGeneral_SetProtectMode (T_idtDrvHdlr hdlr)

  Enable protected register access mode.
- void IDTGeneral_SetSingleUnprotectMode (T_idtDrvHdlr hdlr)

  Set single access register access mode.
- void IDTGeneral_SetFullyUnprotectMode (T_idtDrvHdlr hdlr)

  Set fully un-protected register access mode.
- void IDTGeneral_SetFlagOnTDO (T_idtDrvHdlr hdlr, T_osUint8 nEnable)

  Set TDO pin to indicate the fail of selected clock.
- T_osUint8 IDTGeneral_GetFlagOnTDO (T_idtDrvHdlr hdlr)

  Get TDO pin to indicate the fail of selected clock.
- void IDTGeneral_SetUltraFastSwitch (T_idtDrvHdlr hdlr, T_osUint8 nEnable)

  Set Ultra-fast-switch enable or not.
- T_osUint8 IDTGeneral_GetUltraFastSwitch (T_idtDrvHdlr hdlr)

  Get Ultra-fast-switch enable or not.
- void IDTGeneral_SetHardAlarm (T_idtDrvHdlr hdlr, T_osUint8 nEnable)

  Set Hard-alarm enable when the input exceeds the threshold.
- T_osUint8 IDTGeneral_GetHardAlarm (T_idtDrvHdlr hdlr)

  Get Hard-alarm enable when the input exceeds the threshold.
- T_osUint8 IDTGeneral_i2cTranslate (T_osUint8 i2cAddr, void ∗transData)

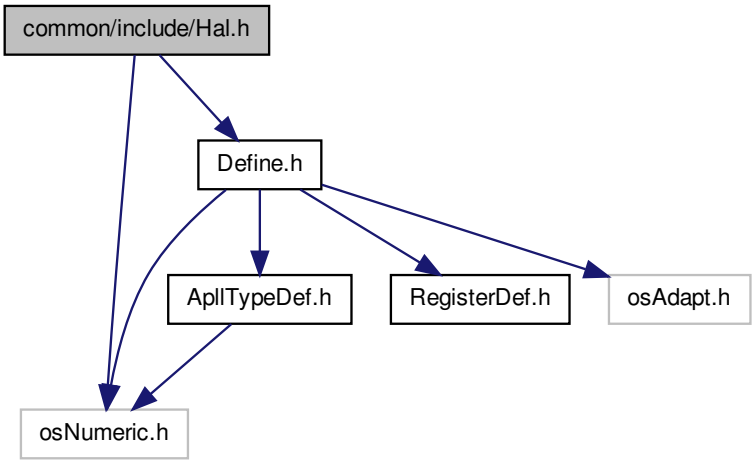
  I2C address translation function used internally by the driver.
- void IDTGeneral_InitDeviceCommon (T_idtDrvHdlr hdlr)
- T_idtDrvHdlr IDTGeneral_InitDriverCommon (const char ∗name, T_idtDrvAccess drvAccess, T_idtDeviceType deviceType, const T_idtDrvDeviceConfig ∗deviceConfig, int useHighLevelApi)
- void IDTGeneral_DeInitDriver (T_idtDrvHdlr hdlr)

  De-initialize driver handler.

6.37 common/include/Hal.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
```

Include dependency graph for Hal.h:



Macros

- #define IDT_HAL_USE_MACRO 0
- #define IDTHAL_ModifyBitfield(regVal, mask, lsb, data)

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

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

Functions

- void IDTHAL_WriteBitfield (T_idtDrvHdlr hdlr, T_osUint8 offset, T_osUint8 mask, T_osUint8 lsb, T_osUint8 data)

  Define function to write a bitfield to a device.
- void IDTHAL_WriteBitfield_Ext (T_idtDrvHdlr hdlr, T_osUint8 offset, T_osUint8 mask, T_osUint8 lsb, T_osUint8 data)

  Define function to write a bitfield to a device.
- T_osUint8 IDTHAL_ReadBitfield (T_idtDrvHdlr hdlr, T_osUint8 offset, T_osUint8 mask, T_osUint8 lsb)

  Define function to read a bitfield from device.
- T_osUint8 IDTHAL_ReadBitfield_Ext (T_idtDrvHdlr hdlr, T_osUint8 offset, T_osUint8 mask, T_osUint8 lsb)

  Define function to read a bitfield from device.
- T_osUint8 IDTHal_Read (T_idtDrvHdlr hdlr, T_osUint8 offset)

  Function to read from device.
- void IDTHal_Write (T_idtDrvHdlr hdlr, T_osUint8 offset, T_osUint8 data)

  Define function to write to device.
- T_osUint8 IDTHal_Read_Ext (T_idtDrvHdlr hdlr, T_osUint8 offset)

  Function to read from device apll address space.

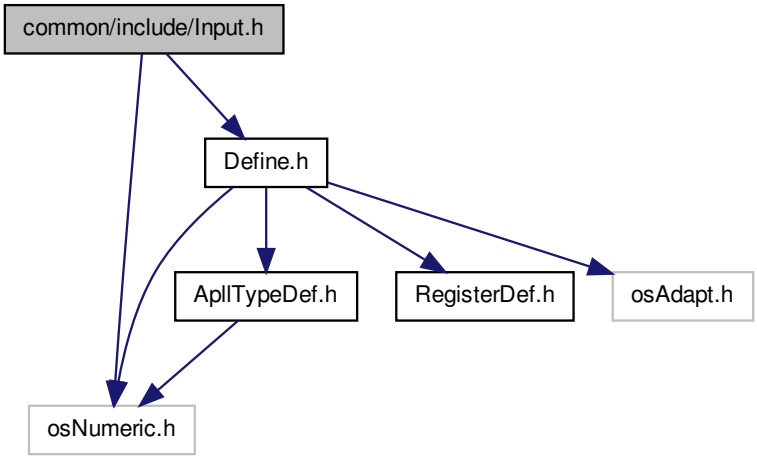
- void IDTHal_Write_Ext (T_idtDrvHdlr hdlr, T_osUint8 offset, T_osUint8 data)

  Function to write to device apll address space.

6.38 common/include/Input.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
```

Include dependency graph for Input.h:



Macros

- #define Input_Phase_Lock_Alarm 1
- #define Input_No_Activity_Alarm 2
- #define Input_Freq_Hard_Alarm 4
- #define IDT_TRANSLATE_EXT_TO_INT_INPORT(hdlr, ext, internal)

  Utility macro to check if input port is supported and to translated from external port to internal port.
- #define IDT_EXT_INPORT_POPULATED(hdlr, ext) (hdlr->deviceConfig_m->inputExt2IntXlate_ma[(ext)])

  Utility macro to check if input port is populated.

Enumerations

- enum T_idtInputDividerMux { Divider_Bypass = 0, Divider_Lock8k = 1, Divider_DivN = 2, Divider_MAX }

  Enumerated type listing input divider modes.
- enum T_idtHighFreqencyDivisor { 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_idtHighFreqencyDivisor nUsage)

  Set the usage of high frequency (> 155.52MHz) divider.
- T_idtHighFreqencyDivisor IDTInput_GetInputHFdivider (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Get high frequency configuration divider.
- T_osUint8 IDTInput_GetInputFreqOffset (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Get the frequency offset of the specified input clock. The returned value is updated every 16 seconds.
- T_osUint8 IDTInput_GetInputClockStatus (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Get the status of a specified input clock.
- T_osUint8 IDTInput_GetInputClockValidation (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Get the clock validation status of an input port.
- void IDTInput_EnableAllInputs (T_idtDrvHdlr hdlr)

  Enable to select anyone of input clocks.
- void IDTInput_DisableAllInputs (T_idtDrvHdlr hdlr)

  Disable to select anyone of input clocks.
- void IDTInput_SetInputEnable (T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_osUint8 enable)

  Enable/Disable selected input.
- T_osUint8 IDTInput_GetInputEnable (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Retrieve enable/disable status of selected input.
- void IDTInput_SetSyncFrequency (T_idtDrvHdlr hdlr, T_idtInputSyncFreq syncFreq)

  Function to select input external sync frequency.
- T_idtInputSyncFreq IDTInput_GetSyncFrequency (T_idtDrvHdlr hdlr)

  Function to retrieve select input external sync frequency.
- void IDTInput_SetSyncSampling (T_idtDrvHdlr hdlr, T_osUint8 exSyncN, T_externalSyncSampling sampling)

  Function to provision sampling configuration for external sync.
- T_externalSyncSampling IDTInput_GetSyncSampling (T_idtDrvHdlr hdlr, T_osUint8 exSyncN)

  Function to retrieve sampling configuration for external sync.
- void IDTInput_SetSyncEdge (T_idtDrvHdlr hdlr, T_externalSyncEdge edge)

  Function to provision synchronization alignment.
- T_externalSyncEdge IDTInput_GetSyncEdge (T_idtDrvHdlr hdlr)

  Function to retrieve synchronization alignment.
- void IDTInput_SetSyncAlarmRange (T_idtDrvHdlr hdlr, T_externalSyncAlarmRange range)

Function to provision the sync allowable range before alarming.

- T_externalSyncAlarmRange IDTInput_GetSyncAlarmRange (T_idtDrvHdlr hdlr)

  Function to retrieve the sync allowable range before alarming.
- void IDTInput_SetSyncBypass (T_idtDrvHdlr hdlr, T_externalSyncBypass bypass)

  Function to provision sync bypass mode.
- T_externalSyncBypass IDTInput_GetSyncBypass (T_idtDrvHdlr hdlr)

  Function to retrieve sync bypass mode.
- void IDTInput_SetSyncEnable (T_idtDrvHdlr hdlr, T_externalSyncEnable enable)

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

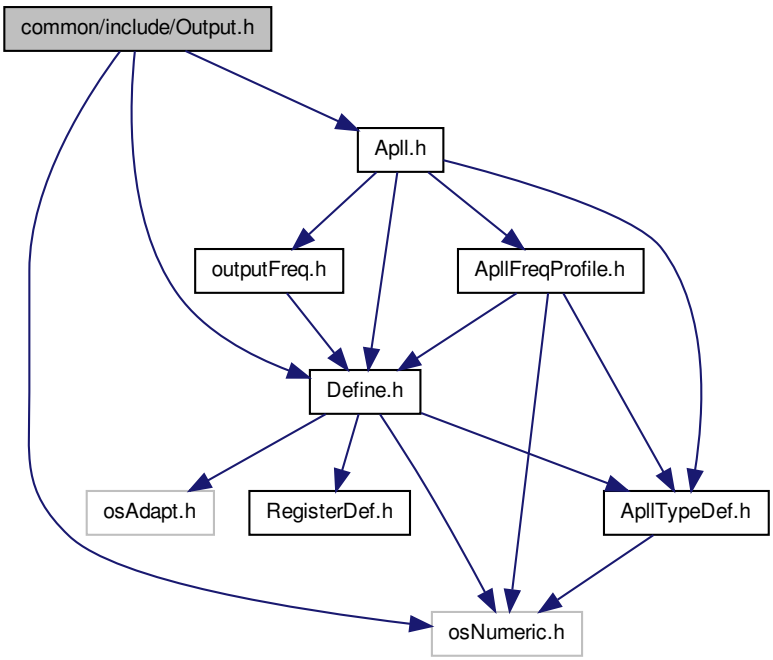
  Function to retrieve external sync enable mode.
- T_idtInputFrequencyFamily IDTInput_InFreqBitValue2Family (T_idtDrvHdlr hdlr, T_idtInputFrequencyBitfield-Value inFreqBitValue)

  Translate input frequency bit value to input frequency family.

6.39 common/include/Output.h File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "Apll.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_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_SetOuputpuInterface (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_GetOuputpuInterface (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.

6.40 common/include/RegisterDef.h File Reference

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

Enumerations

- enum {
  REG_ID_LOW = 0x00, REG_ID_HIGH = 0x01, REG_MPU_PIN_STS = 0x02, REG_NOMINAL_FREQ_LOW_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_CNFG = 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_DIVN_LOW_CNFG = 0x24,
  REG_PRE_DIVN_HIGH_CNFG = 0x25, REG_IN1_IN2_SEL_PRIORITY_CNFG = 0x26, REG_IN3_IN4_SEL_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_IN11_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_BUCKET_SIZE_0_CNFG = 0x33, REG_DECAY_RATE_0_CNFG = 0x34,
  REG_UPPER_THRESHOLD_1_CNFG = 0x35, REG_LOWER_THRESHOLD_1_CNFG = 0x36, REG_BUCKET_SIZE_1_CNFG = 0x37, REG_DECAY_RATE_1_CNFG = 0x38,
  REG_UPPER_THRESHOLD_2_CNFG = 0x39, REG_LOWER_THRESHOLD_2_CNFG = 0x3A, REG_BUCKET_SIZE_2_CNFG = 0x3B, REG_DECAY_RATE_2_CNFG = 0x3C,
  REG_UPPER_THRESHOLD_3_CNFG = 0x3D, REG_LOWER_THRESHOLD_3_CNFG = 0x3E, REG_BUCKET_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_STS = 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_CNFG = 0x53, REG_T4_OPERATION_MODE_CNFG = 0x54,
  REG_T0_DPLL_APLL_PATH_CNFG = 0x55, REG_T0_DPLL_START_BW_DAMPING_CNFG = 0x56, REG_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_HOLDOVER_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_FREQ_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, REG_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_NOMINAL_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_SEL_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_MULTI_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_SIZE = 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_SONET_SDH_SIZE = 1,
BF_IN_SONET_SDH_MASK = 0x04, BF_IN_SONET_SDH_LSB = 2, BF_MASTER_SLAVE_SIZE = 1, BF_MASTER_SLAVE_MASK = 0x02,
BF_MASTER_SLAVE_LSB = 1, BF_REVERTIVE_MODE_SIZE = 1, BF_REVERTIVE_MODE_MASK = 0x01, BF_REVERTIVE_MODE_LSB = 0,
BF_OSC_EDGE_SIZE = 1, BF_OSC_EDGE_MASK = 0x04, BF_OSC_EDGE_LSB = 2, BF_OUT7_PECL_LVDS_SIZE = 1,
BF_OUT7_PECL_LVDS_MASK = 0x02, BF_OUT7_PECL_LVDS_LSB = 1, BF_OUT6_PECL_LVDS_SIZE = 1, BF_OUT6_PECL_LVDS_MASK = 0x01,
BF_OUT6_PECL_LVDS_LSB = 0, BF_FREQ_MON_CLK_SIZE = 1, BF_FREQ_MON_CLK_MASK = 0x80, BF_FREQ_MON_CLK_LSB = 7,
BF_LOS_FLAG_TO_TDO_SIZE = 1, BF_LOS_FLAG_TO_TDO_MASK = 0x40, BF_LOS_FLAG_TO_TDO_LSB = 6, BF_ULTR_FAST_SW_SIZE = 1,
BF_ULTR_FAST_SW_MASK = 0x20, BF_ULTR_FAST_SW_LSB = 5, BF_EXT_SW_SIZE = 1, BF_EXT_SW_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, 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_VALIDATED_MASK = 0xF0, BF_THIRD_HIGHEST_PRIORITY_VALIDATED_LSB = 4, BF_SECOND_HIGHEST_PRIORITY_VALIDATED_SIZE = 4,
BF_SECOND_HIGHEST_PRIORITY_VALIDATED_MASK = 0x0F, BF_SECOND_HIGHEST_PRIORITY_VALIDATED_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_LOCK_T0_LSB = 6,
BF_T0_FOR_T4_SIZE = 1, BF_T0_FOR_T4_MASK = 0x20, BF_T0_FOR_T4_LSB = 5, BF_T4_TEST_T0_PH_SIZE = 1,
BF_T4_TEST_T0_PH_MASK = 0x10, BF_T4_TEST_T0_PH_LSB = 4, BF_T4_INPUT_SEL_SIZE = 4, BF_T4_INPUT_SEL_MASK = 0x0F,
BF_T4_INPUT_SEL_LSB = 0, BF_EX_SYNC_ALARM_MON_SIZE = 1, BF_EX_SYNC_ALARM_MON_MASK = 0x80, BF_EX_SYNC_ALARM_MON_LSB = 7,
BF_T4_DPLL_LOCK_SIZE = 1, BF_T4_DPLL_LOCK_MASK = 0x40, BF_T4_DPLL_LOCK_LSB = 6, BF_T0_DPLL_SOFT_FREQ_ALARM_SIZE = 1,
BF_T0_DPLL_SOFT_FREQ_ALARM_MASK = 0x20, BF_T0_DPLL_SOFT_FREQ_ALARM_LSB = 5, BF_T4_DPLL_SOFT_FREQ_ALARM_SIZE = 1, BF_T4_DPLL_SOFT_FREQ_ALARM_MASK = 0x10,
BF_T4_DPLL_SOFT_FREQ_ALARM_LSB = 4, BF_T0_DPLL_LOCK_SIZE = 1, BF_T0_DPLL_LOCK_MASK = 0x08, BF_T0_DPLL_LOCK_LSB = 3,
BF_T0_DPLL_OPERATING_MODE_SIZE = 3, BF_T0_DPLL_OPERATING_MODE_MASK = 0x07, BF_T0_DPLL_OPERATING_MODE_LSB = 0, BF_T0_WDCO_SIZE = 1,
BF_T0_WDCO_MASK = 0x08, BF_T0_WDCO_LSB = 3, BF_T0_OPERATING_MODE_SIZE = 3, BF_T0_OPERATING_MODE_MASK = 0x07,
BF_T0_OPERATING_MODE_LSB = 0, BF_T4_WDCO_SIZE = 1, BF_T4_WDCO_MASK = 0x08, BF_T4_WDCO_LSB = 3,
BF_T4_OPERATING_MODE_SIZE = 3, BF_T4_OPERATING_MODE_MASK = 0x07, BF_T4_OPERATING_MODE_LSB = 0, BF_T0_APLL_PATH_SIZE = 4,
BF_T0_APLL_PATH_MASK = 0xF0, BF_T0_APLL_PATH_LSB = 4, BF_T0_GSM_OBSAI_16E1_16T1_SEL_SIZE = 2, BF_T0_GSM_OBSAI_16E1_16T1_SEL_MASK = 0x0C,
BF_T0_GSM_OBSAI_16E1_16T1_SEL_LSB = 2, BF_T0_12E1_GPS_E3_T3_SEL_SIZE = 2, BF_T0_12E1_GPS_E3_T3_SEL_MASK = 0x03, BF_T0_12E1_GPS_E3_T3_SEL_LSB = 0,
BF_T0_DPLL_START_DAMPING_SIZE = 3, BF_T0_DPLL_START_DAMPING_MASK = 0xE0, BF_T0_DPLL_START_DAMPING_LSB = 5, BF_T0_DPLL_START_BW_SIZE = 5,
BF_T0_DPLL_START_BW_MASK = 0x1F, BF_T0_DPLL_START_BW_LSB = 0, BF_T0_DPLL_ACQ_DAMPING_SIZE = 3, BF_T0_DPLL_ACQ_DAMPING_MASK = 0xE0,
BF_T0_DPLL_ACQ_DAMPING_LSB = 5, BF_T0_DPLL_ACQ_BW_SIZE = 5, BF_T0_DPLL_ACQ_BW_MASK = 0x1F, BF_T0_DPLL_ACQ_BW_LSB = 0,
BF_T0_DPLL_LOCKED_DAMPING_SIZE = 3, BF_T0_DPLL_LOCKED_DAMPING_MASK = 0xE0, BF_T0_DPLL_LOCKED_DAMPING_LSB = 5, BF_T0_DPLL_LOCKED_BW_SIZE = 5,
BF_T0_DPLL_LOCKED_BW_MASK = 0x1F, BF_T0_DPLL_LOCKED_BW_LSB = 0, BF_AUTO_BW_SEL_SIZE = 1, BF_AUTO_BW_SEL_MASK = 0x80,
BF_AUTO_BW_SEL_LSB = 7, BF_T0_LIMIT_SIZE = 1, BF_T0_LIMIT_MASK = 0x08, BF_T0_LIMIT_LSB = 3,
BF_COARSE_PH_LOS_LIMT_EN_SIZE = 1, BF_COARSE_PH_LOS_LIMT_EN_MASK = 0x80, BF_COARSE_PH_LOS_LIMT_EN_LSB = 7, BF_WIDE_EN_SIZE = 1,
BF_WIDE_EN_MASK = 0x40, BF_WIDE_EN_LSB = 6, BF_MULTI_PH_APP_SIZE = 1, BF_MULTI_PH_APP_MASK = 0x20,
BF_MULTI_PH_APP_LSB = 5, BF_MULTI_PH_8K_4K_2K_EN_SIZE = 1, BF_MULTI_PH_8K_4K_2K_EN_MASK = 0x10, BF_MULTI_PH_8K_4K_2K_EN_LSB = 4,
BF_PH_LOS_COARSE_LIMT_SIZE = 4, BF_PH_LOS_COARSE_LIMT_MASK = 0x0F, BF_PH_LOS_COARSE_LIMT_LSB = 0, BF_FINE_PH_LOS_LIMT_EN_SIZE = 1,
BF_FINE_PH_LOS_LIMT_EN_MASK = 0x80, BF_FINE_PH_LOS_LIMT_EN_LSB = 7, BF_FAST_LOS_SW_SIZE = 1, BF_FAST_LOS_SW_MASK = 0x40,
BF_FAST_LOS_SW_LSB = 6, BF_PH_LOS_FINE_LIMT_SIZE = 3, BF_PH_LOS_FINE_LIMT_MASK = 0x07, BF_PH_LOS_FINE_LIMT_LSB = 0,
BF_MAN_HOLDOVER_SIZE = 1, BF_MAN_HOLDOVER_MASK = 0x80, BF_MAN_HOLDOVER_LSB = 7,

BF_AUTO_AVG_SIZE = 1,
BF_AUTO_AVG_MASK = 0x40, BF_AUTO_AVG_LSB = 6, BF_FAST_AVG_SIZE = 1, BF_FAST_AVG_MASK = 0x20,
BF_FAST_AVG_LSB = 5, BF_READ_AVG_SIZE = 1, BF_READ_AVG_MASK = 0x10, BF_READ_AVG_LSB = 4,
BF_TEMP_HOLDOVER_MODE_SIZE = 2, BF_TEMP_HOLDOVER_MODE_MASK = 0x0C, BF_TEMP_HOLDOVER_MODE_LSB = 2, BF_HOLDOVER_FREQ_A_SIZE = 8,
BF_HOLDOVER_FREQ_A_MASK = 0xFF, BF_HOLDOVER_FREQ_A_LSB = 0, BF_HOLDOVER_FREQ_B_SIZE = 8, BF_HOLDOVER_FREQ_B_MASK = 0xFF,
BF_HOLDOVER_FREQ_B_LSB = 0, BF_HOLDOVER_FREQ_C_SIZE = 8, BF_HOLDOVER_FREQ_C_MASK = 0xFF, BF_HOLDOVER_FREQ_C_LSB = 0,
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_LOCKED_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_LOCKED_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_DPLL_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_LOS_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_FREQ_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_SEL_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_FAIL_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_IN_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_SYNC_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 offsers and bit field constants.

6.40.1 Detailed Description

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

See Also

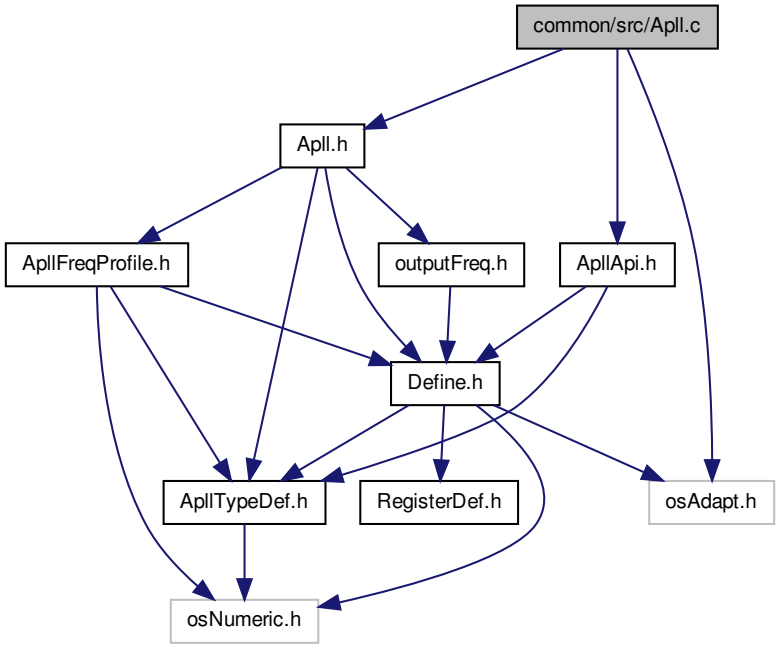
Datasheet for detailed register descriptions.

Definition in file RegisterDef.h.

6.41 common/src/Apll.c File Reference

```
#include "Apll.h"
#include "ApllApi.h"
#include "osAdapt.h"
```

Include dependency graph for Apll.c:



Functions

- void IDTApll_SetApllConfigPerOutput (T_idtDrvHdlr hdlr, T_idtApllId apllInstId, T_idtApllOutput output, T_idt-ApllConfigPerOutput *apllConfigPerOutput)

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 confiugred 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.

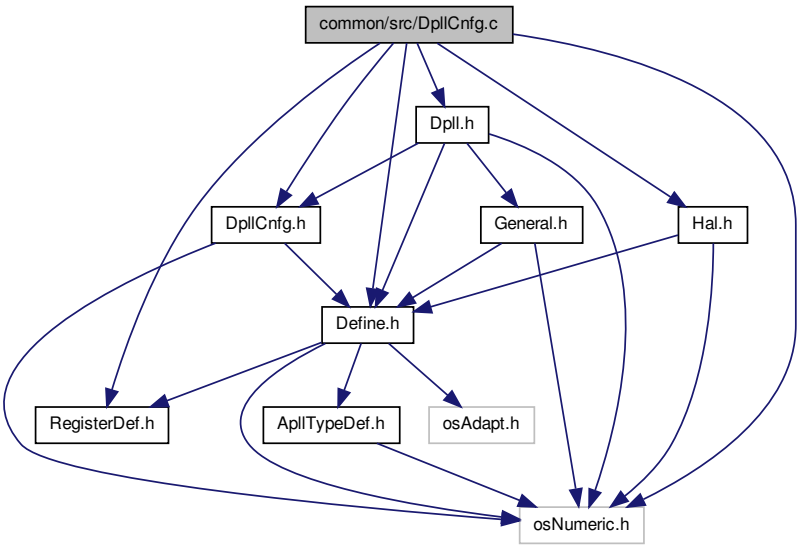
- T_idtApllXtal * IDTApll_GetSupportedXtal (const T_idtDrvDeviceConfig *deviceConfig, T_osUint8 *numberOfXtal)

  Return supported xtal configuration by the device.

6.42 common/src/DpllCnfg.c File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "Hal.h"
#include "DpllCnfg.h"
#include "RegisterDef.h"
#include "Dpll.h"
```

Include dependency graph for DpllCnfg.c:



Data Structures

- struct T_idtSonetGigEthBitfield2PathConfig

  Structure Translate from Sonet/GigE bitfield to path configuration.

Macros

- #define INVALID_REGISTER_OFFSET ((T_osUint8)(-1))

  Tag used to indicating invalid register offset.

Functions

- T_idtDpllType IDTDpll_GetTypeOfDpll (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  Function to determine type of DPLL give a DPLL instance.
- void IDTDpll_SetAutoSelInput (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_osUint8 mode)

  Set the DPLL as automatic reference clock selection.
- T_osUint8 IDTDpll_GetAutoSelInput (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  Get automatic input selection status.
- void IDTDpll_SetForceSelInput (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_osUint8 nInputNo)

  Set the DPLL to force to lock to a specified reference clock.
- T_osUint8 IDTDpll_GetForceSelInput (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  Set the DPLL to force to lock to a specified reference clock.
- T_osUint8 IDTDpll_Get1stPriorityInput (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  Get input number of a valid clock with the highest priority.
- T_osUint8 IDTDpll_Get2ndPriorityInput (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  Get input number of a valid clock with the second highest priority.
- T_osUint8 IDTDpll_Get3rdPriorityInput (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

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

- T_osUint8 IDTDpll_GetCurrentSelectInput (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  Get the current selected input clock.
- void IDTDpll_SetDpllOperMod (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_idtDpllOperMode nMode)

  Set DPLL working at the specified operating mode.
- T_idtDpllOperMode IDTDpll_GetDpllOperMod (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  Set DPLL working at the specified operating mode.
- T_osUint8 IDTDpll_IsDpllLocked (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  Inquire if the DPLL is in lock state.
- void IDTDpll_SetBandWidthDamping (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_idtBandwidthLimitStage stage, T_idtDpllBandwidth nBandWidth, T_idtDampingFactor nDamping)

  Set the bandwidth and damping factor used for bandwidth limiting.
- void IDTDpll_GetBandWidthDamping (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_idtBandwidthLimitStage stage, T_idtDpllBandwidth *nBandWidth_p, T_idtDampingFactor *nDamping_p)

  Get configured bandwidth and damping factor for bandwidth limiting.
- void IDTDpll_SetAutoSelBandWidth (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_osUint8 nEnable)

  Set device to select a suitable bandwidth and damping factor automatically.
- T_osUint8 IDTDpll_GetAutoSelBandWidth (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  Get auto bandwidth/damping factor status automatically.
- void IDTDpll_SetDpllFrozen (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_osUint8 nEnable)

  Set DPLL integral path value frozen.
- T_osUint8 IDTDpll_GetDpllFrozen (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  Get DPLL integral path value frozen.
- void IDTDpll_SetPhaseCoarseDetector (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_osUint8 nEnable, T_osUint8 nWide, T_osUint8 nMultiPhase, T_osUint8 nMultiPh8kEn)

  Set the coarse phase lose detector enable.
- void IDTDpll_GetPhaseCoarseDetector (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_osUint8 *nEnable_p, T_osUint8 *nWide_p, T_osUint8 *nMultiPhase_p, T_osUint8 *nMultiPh8kEn_p)

  Set the coarse phase lose detector enable.
- void IDTDpll_SetPhaseCoarseLimit (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_idtCoarsePhaseLimit nLimit)

  Set the limitation of the coarse phase lose detector.
- T_idtCoarsePhaseLimit IDTDpll_GetPhaseCoarseLimit (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  Get the limitation of the coarse phase lose detector.
- void IDTDpll_SetPhaseFineDetectorEnable (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_osUint8 nEnable)

  Set the fine phase lose detector enable.
- T_osUint8 IDTDpll_GetPhaseFineDetectorEnable (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  Get the fine phase lose detector enable.
- void IDTDpll_SetPhaseFineLimit (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_idtFinePhaseLimit nLimit)

  Set the limitation of the fine phase lose detector.
- T_idtFinePhaseLimit IDTDpll_GetPhaseFineLimit (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll)

  Set the limitation of the fine phase lose detector.
- void IDTDpll_SetFastLossSwitch (T_idtDrvHdlr hdlr, T_osUint8 nEnable)

  Set reference clock switch if a fast loss occurs.
- T_osUint8 IDTDpll_GetFastLossSwitch (T_idtDrvHdlr hdlr)

  Get reference clock switch if a fast loss occurs.
- void IDTDpll_SetHoldoverMode (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_idtDpllHoldover nMode, T_osUint8 nReadMode)

  Set calculation mode of holdover frequency and read back mode.
- void IDTDpll_GetHoldoverMode (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_idtDpllHoldover *nMode_p, T_osUint8 *nReadMode_p)

  Get calculation mode of holdover frequency and read back mode.

- void IDTDpll_SetTempHoldoverMode (T_idtDrvHdlr hdlr, T_idtDpllInstance nDpll, T_idtTemp_holdover 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.

Variables

- const T_IDTPathConfiguration IDTDpll_pathConfigurationNULL_c

  Invalid output path configuration.

- const T_osUint8 IDTDpll_bandwidthLimitRegisters_a [Dpll_MAX][dpllBandwidthLimit_MAX]

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

- const T_idtSonetGigEthSelector IDTDpll_sonetGigEthPathConfig2Bitfield [Dpll_MAX][SonetGigEthOutput_-MAX]

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

6.42.1 Macro Definition Documentation

6.42.1.1 #define INVALID_REGISTER_OFFSET ((T_osUint8)(-1))

Tag used to indicating invalid register offset.

Definition at line 46 of file DpllCnfg.c.

6.42.2 Variable Documentation

6.42.2.1 const T_osUint8 IDTDpll_bandwidthLimitRegisters_a[Dpll_MAX][dpllBandwidthLimit_MAX]

Initial value:

```
=
{
    {
        REG_T0_DPLL_START_BW_DAMPING_CNFG,
        REG_T0_DPLL_ACQ_BW_DAMPING_CNFG,
        REG_T0_DPLL_LOCKED_BW_DAMPING_CNFG
    },
    {
        INVALID_REGISTER_OFFSET,
        INVALID_REGISTER_OFFSET,
        REG_T4_DPLL_LOCKED_BW_DAMPING_CNFG
    } }
```

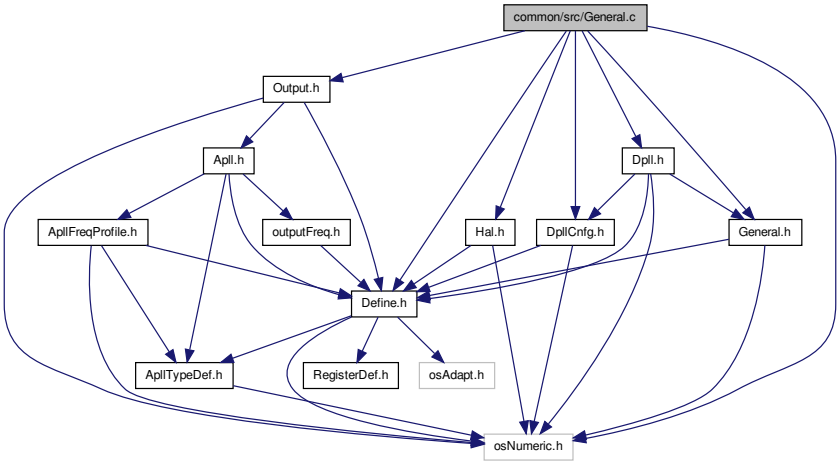
Structure used to determine which register to access for bandwidth limiting.

Definition at line 75 of file DpllCnfg.c.

6.43 common/src/General.c File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "Hal.h"
#include "DpllCnfg.h"
#include "Dpll.h"
#include "General.h"
#include "Output.h"
```

Include dependency graph for General.c:



Functions

- T_osUint16 IDTGeneral_GetDeviceID (T_idtDrvHdlr hdlr)

  Get ID number of the device.
- void IDTGeneral_SetOscFreqOffset (T_idtDrvHdlr hdlr, long pptOffset)

  Set compensation to the oscillator offset.
- long IDTGeneral_GetOscFreqOffset (T_idtDrvHdlr hdlr)

  Get compensated oscillator offset.
- void IDTGeneral_SetNetworkType (T_idtDrvHdlr hdlr, networkType_t nType)

  Set the network type, SDH or SONET.
- networkType_t IDTGeneral_GetNetworkType (T_idtDrvHdlr hdlr)

  Get the network type, SDH or SONET.
- void IDTGeneral_SetRevertiveMode (T_idtDrvHdlr hdlr, T_osUint8 nEnable)

  Set the device running in Revertive mode or not.
- T_osUint8 IDTGeneral_GetRevertiveMode (T_idtDrvHdlr hdlr)

  Get the device running in Revertive mode or not.
- void IDTGeneral_SetTimeoutValue (T_idtDrvHdlr hdlr, phaseTimeoutMultiplier_t nMultiFactor, T_osUint8 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)

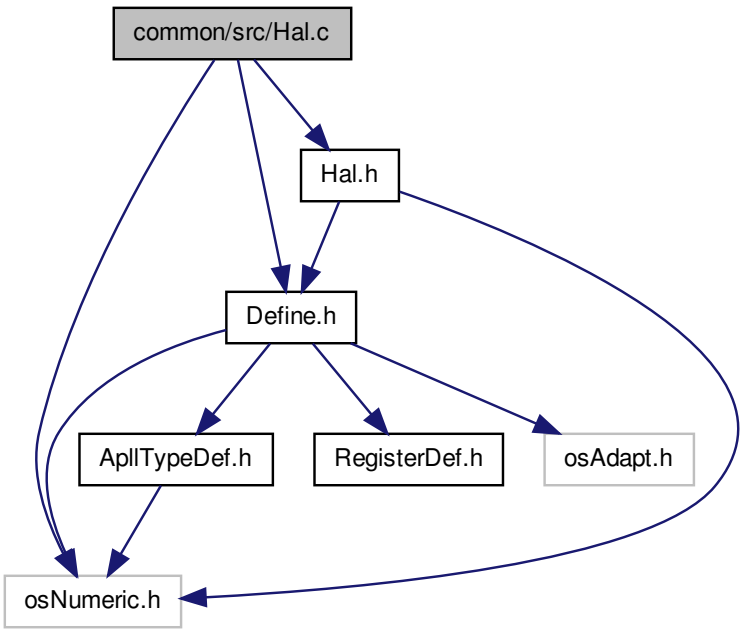
  I2C address translation function used internally by the driver.
- void IDTGeneral_InitDeviceCommon (T_idtDrvHdlr hdlr)
- T_idtDrvHdlr IDTGeneral_InitDriverCommon (const char ∗name, T_idtDrvAccess drvAccess, T_idtDeviceType deviceType, const T_idtDrvDeviceConfig ∗deviceConfig, int useHighLevelApi)
- void IDTGeneral_DeInitDriver (T_idtDrvHdlr hdlr)

  De-initialize driver handler.

6.44 common/src/Hal.c File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "Hal.h"
```

Include dependency graph for Hal.c:



Macros

- #define DEBUG_IDT_HAL

Functions

- void IDTHAL_WriteBitfield (T_idtDrvHdlr hdlr, T_osUint8 offset, T_osUint8 mask, T_osUint8 lsb, T_osUint8 data)

  Define function to write a bitfield to a device.

- void IDTHAL_WriteBitfield_Ext (T_idtDrvHdlr hdlr, T_osUint8 offset, T_osUint8 mask, T_osUint8 lsb, T_osUint8 data)

  Define function to write a bitfield to a device.

- T_osUint8 IDTHAL_ReadBitfield (T_idtDrvHdlr hdlr, T_osUint8 offset, T_osUint8 mask, T_osUint8 lsb)

  Define function to read a bitfield from device.

- T_osUint8 IDTHAL_ReadBitfield_Ext (T_idtDrvHdlr hdlr, T_osUint8 offset, T_osUint8 mask, T_osUint8 lsb)

  Define function to read a bitfield from device.

- T_osUint8 IDTHal_Read (T_idtDrvHdlr hdlr, T_osUint8 offset)

  Function to read from device address space.

- void IDTHal_Write (T_idtDrvHdlr hdlr, T_osUint8 offset, T_osUint8 data)

  Function to write to device address space.

- T_osUint8 IDTHal_Read_Ext (T_idtDrvHdlr hdlr, T_osUint8 offset)

  Function to read from device apll address space.

- void IDTHal_Write_Ext (T_idtDrvHdlr hdlr, T_osUint8 offset, T_osUint8 data)

  Function to write to device apll address space.

6.44.1 Macro Definition Documentation

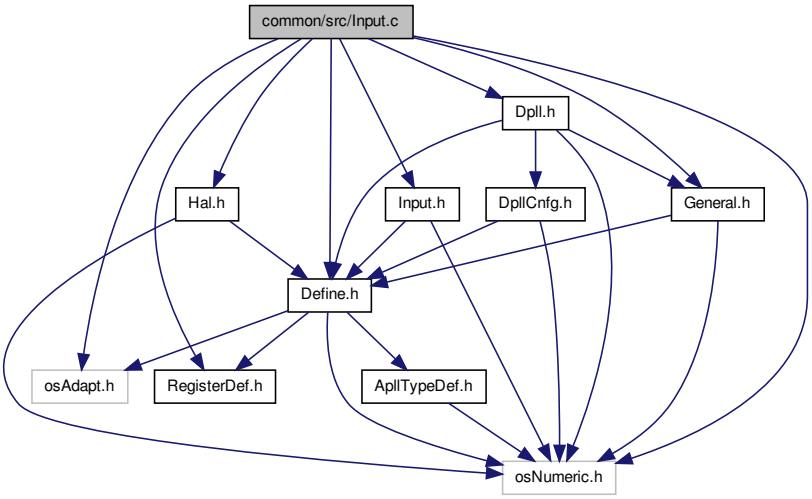
6.44.1.1 #define DEBUG_IDT_HAL

Definition at line 41 of file Hal.c.

6.45 common/src/Input.c File Reference

```
#include "osNumeric.h"
#include "osAdapt.h"
#include "Define.h"
#include "Hal.h"
#include "General.h"
#include "Input.h"
#include "Dpll.h"
#include "RegisterDef.h"
```

Include dependency graph for Input.c:



Data Structures

- struct bucketReg_t

  Structure used to store leaky bucket register offsets.

Functions

- T_idtInputFrequencyFamily IDTInput_InFreqBitValue2Family (T_idtDrvHdlr hdlr, T_idtInputFrequencyBitfieldValue inFreqBitValue)

  Translate input frequency bit value to input frequency family.
- void IDTInput_SetPriority (T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_idtDpllInstance nDpll, T_osUint8 nPriority)

  Set specified port to the priority.
- T_osUint8 IDTInput_GetPriority (T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_idtDpllInstance nDpll)

  Get the current priority of the specified port.
- void IDTInput_SetInputFrequency (T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_idtInputFrequencyBitfieldValue nFrequency)

  Set the frequency of the input clock at the specified port.
- T_idtInputFrequencyBitfieldValue IDTInput_GetInputFrequency (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Get frequency of the input clock at the specified port.
- void IDTInput_SetActivityMonitor (T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_osUint8 nBucket)

  Select one of the four activity monitor configurations for the specified input port.
- T_osUint8 IDTInput_GetActivityMonitor (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Retrieve selected activity monitoring configuration.
- void IDTInput_SetBucketConfig (T_idtDrvHdlr hdlr, T_osUint8 nBucket, T_osUint8 nUpper, T_osUint8 nLower, T_osUint8 nSize, T_osUint8 nDecay)

  Set the configuration to Leaky Bucket activity monitoring.
- void IDTInput_GetBucketConfig (T_idtDrvHdlr hdlr, T_osUint8 nBucket, T_osUint8 *nUpper_p, T_osUint8 *nLower_p, T_osUint8 *nSize_p, T_osUint8 *nDecay_p)

  Get activity monitoring configuration (Leaky Bucket)

- void IDTInput_SetPreDivider (T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_idtInputDividerMux nDivider)

  Assign a pre-divider for the specified input.
- T_idtInputDividerMux IDTInput_GetPreDivider (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Get configured input port pre-divider.
- void IDTInput_SetDivisionFactor (T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_osUint16 nFactor)

  Set the dividen factor to pre-divider assigned to a certain input.
- T_osUint16 IDTInput_GetDivisionFactor (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Get the dividen factor to pre-divider assigned to a certain input.
- void IDTInput_SetInputHFdivider (T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_idtHighFreqencyDivisor nUsage)

  Set the usage of high frequency (> 155.52MHz) divider.
- T_idtHighFreqencyDivisor IDTInput_GetInputHFdivider (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Get high frequency configuration divider.
- T_osUint8 IDTInput_GetInputFreqOffset (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Get the frequency offset of the specified input clock. The returned value is updated every 16 seconds.
- T_osUint8 IDTInput_GetInputClockStatus (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Get the status of a specified input clock.
- T_osUint8 IDTInput_GetInputClockValidation (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Get the clock validation status of an input port.
- void IDTInput_EnableAllInputs (T_idtDrvHdlr hdlr)

  Enable to select anyone of input clocks.
- void IDTInput_DisableAllInputs (T_idtDrvHdlr hdlr)

  Disable to select anyone of input clocks.
- void IDTInput_SetInputEnable (T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_osUint8 enable)

  Enable/Disable selected input.
- T_osUint8 IDTInput_GetInputEnable (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Retrieve enable/disable status of selected input.
- void IDTInput_SetSyncFrequency (T_idtDrvHdlr hdlr, T_idtInputSyncFreq syncFreq)

  Function to select input external sync frequency.
- T_idtInputSyncFreq IDTInput_GetSyncFrequency (T_idtDrvHdlr hdlr)

  Function to retrieve select input external sync frequency.
- void IDTInput_SetSyncSampling (T_idtDrvHdlr hdlr, T_osUint8 exSyncN, T_externalSyncSampling sampling)

  Function to provision sampling configuration for external sync.
- T_externalSyncSampling IDTInput_GetSyncSampling (T_idtDrvHdlr hdlr, T_osUint8 exSyncN)

  Function to retrieve sampling configuration for external sync.
- void IDTInput_SetSyncEdge (T_idtDrvHdlr hdlr, T_externalSyncEdge edge)

  Function to provision synchronization alignment.
- T_externalSyncEdge IDTInput_GetSyncEdge (T_idtDrvHdlr hdlr)

  Function to retrieve synchronization alignment.
- void IDTInput_SetSyncAlarmRange (T_idtDrvHdlr hdlr, T_externalSyncAlarmRange range)

  Function to provision the sync allowable range before alarming.
- T_externalSyncAlarmRange IDTInput_GetSyncAlarmRange (T_idtDrvHdlr hdlr)

  Function to retrieve the sync allowable range before alarming.
- void IDTInput_SetSyncBypass (T_idtDrvHdlr hdlr, T_externalSyncBypass bypass)

  Function to provision sync bypass mode.
- T_externalSyncBypass IDTInput_GetSyncBypass (T_idtDrvHdlr hdlr)

  Function to retrieve sync bypass mode.
- void IDTInput_SetSyncEnable (T_idtDrvHdlr hdlr, T_externalSyncEnable enable)

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

Function to retrieve external sync enable mode.

- void IDTInput_SetAmiInputFrequency (T_idtDrvHdlr hdlr, T_osUint8 nInputNo, T_idtAmiInputFrequencyBitFieldValue amiSignal)

  Provision AMI signal for input port.

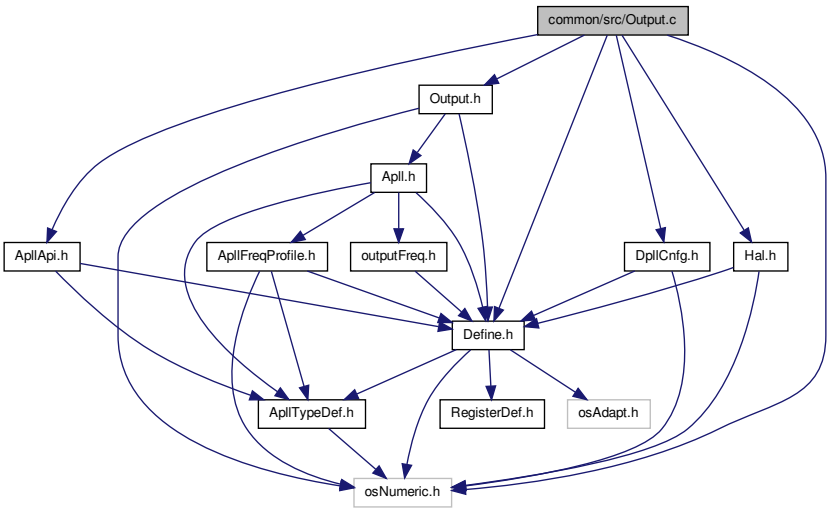
- T_idtAmiInputFrequencyBitFieldValue IDTInput_GetAmiInputFrequency (T_idtDrvHdlr hdlr, T_osUint8 nInputNo)

  Function to retrieve AMI input frequency for input port.

6.46 common/src/Output.c File Reference

```
#include "osNumeric.h"
#include "Define.h"
#include "Hal.h"
#include "Output.h"
#include "DpllCnfg.h"
#include "ApllApi.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_SetOuputpuInterface (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_GetOuputpuInterface (T_idtDrvHdlr hdlr, T_osUint8 nOutN)

  Get output port interface type. Consult data sheet for supported interfaces per port.
- void IDTOutput_SetFrameSync (T_idtDrvHdlr hdlr, T_frameSync frameSync, T_osUint8 enable, T_osUint8 invert, T_osUint8 pulse)

  Function to control output frame syncs.
- void IDTOutput_GetFrameSync (T_idtDrvHdlr hdlr, T_frameSync frameSync, T_osUint8 *enable_p, T_osUint8 *invert_p, T_osUint8 *pulse_p)

  Function to retrieve output frame sync configuration.
- void IDTOutput_SetFrameSyncPulseEdge (T_idtDrvHdlr hdlr, T_osUint8 isRisingEdge)

  Function provision trigger for frame sync pulses.
- T_osUint8 IDTOutput_GetFrameSyncPulseEdge (T_idtDrvHdlr hdlr)

  Function retrieve trigger for frame sync pulses.
- void IDTOutput_DisableApllInput (T_idtDrvHdlr hdlr, T_idtOutputApllConfig outputApllConfig)

  Function disables APLL input.
- void IDTOutput_DisableAllIntOutput (T_idtDrvHdlr hdlr)

  Function disables all internal outputs.

6.46.1 Enumeration Type Documentation

6.46.1.1 anonymous enum

Enumerated type supported output port interfaces in bit mask format.

Enumerator

OUTPUTIF_AMI_MASK AMI Interface mask

OUTPUTIF_CMOS_MASK CMOS Interface mask

OUTPUTIF_LVDS_MASK LVDS Interface mask

OUTPUTIF_PECL_MASK PECL Interface mask

Definition at line 78 of file Output.c.

6.46.1.2 enum T_idtOutputPathBfVal

Enumerated type listing possible output path selections.

Enumerator

OutputPathBfVal_T0SonetGigEth From T0 Sonet/GigE path

OutputPathBfVal_T0Dpll_Eth From T0 DPLL ETH path

OutputPathBfVal_T0Dpll_7776kHz From T0 DPLL 77.76 MHz path

OutputPathBfVal_T0Dpll_12E1_GPS_E3_T3 From T0 DPLL 12E1/GPS/E3/T3 path

OutputPathBfVal_T0Dpll_16E1_16T1 From T0 DPLL 16E1/16T1 path

OutputPathBfVal_T0Dpll_GSM_OBSAI_16E1_16T1 From T0 DPLL GSM/OBSAI/16E1/16T1 path

OutputPathBfVal_T4SonetGigEth From T4 Sonet/GigE path

OutputPathBfVal_T4Dpll_Eth From T4 DPLL ETH path

OutputPathBfVal_T4Dpll_7776kHz From T4 DPLL 77.76 MHz path

OutputPathBfVal_T4Dpll_12E1_24T1_E3_T3 From T4 DPLL 12E1/E3/T3/24T1 path

OutputPathBfVal_T4Dpll_16E1_16T1 From T4 DPLL 16E1/16T1 path

OutputPathBfVal_T4Dpll_GSM_GPS_16E1_16T1 From T4 DPLL GSM/GPS/16E1/16T1 path

OutputPathBfVal_NULL Not supported

OutputPathBfVal_MAX

Definition at line 48 of file Output.c.

Index