Vitesse API

Generated by Doxygen 1.8.14

Contents

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 ib_par_cfg Struct Reference

Generalized data structure for IB parameters.

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- u16 value
- u16 min
- u16 max

3.1.1 Detailed Description

Generalized data structure for IB parameters.

Definition at line 207 of file vtss_phy_10g_api.h.

3.1.2 Field Documentation

3.1.2.1 value

```
u16 ib_par_cfg::value
```

value to be configured

Definition at line 208 of file vtss_phy_10g_api.h.

3.1.2.2 min

```
u16 ib_par_cfg::min
```

Minimum value

Definition at line 209 of file vtss_phy_10g_api.h.

3.1.2.3 max

```
u16 ib_par_cfg::max
```

Maximum value

Definition at line 210 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.2 port_custom_conf_t Struct Reference

Port configuration.

```
#include <port.h>
```

Data Fields

- BOOL enable
- BOOL autoneg
- BOOL fdx
- BOOL flow_control
- vtss_port_speed_t speed
- vtss_fiber_port_speed_t dual_media_fiber_speed
- unsigned int max_length
- BOOL exc_col_cont
- u8 adv_dis
- u8 max_tags
- BOOL oper_up
- BOOL frame_length_chk

3.2.1 Detailed Description

Port configuration.

Definition at line 269 of file port.h.

3.2.2 Field Documentation

3.2.2.1 enable

```
BOOL port_custom_conf_t::enable
```

Admin enable/disable

Definition at line 270 of file port.h.

3.2.2.2 autoneg

```
BOOL port_custom_conf_t::autoneg
```

Auto negotiation

Definition at line 271 of file port.h.

3.2.2.3 fdx

```
BOOL port_custom_conf_t::fdx
```

Forced duplex mode

Definition at line 272 of file port.h.

3.2.2.4 flow_control

```
BOOL port_custom_conf_t::flow_control
```

Flow control (Standard 802.3x)

Definition at line 273 of file port.h.

3.2.2.5 speed

```
vtss_port_speed_t port_custom_conf_t::speed
```

Forced port speed

Definition at line 277 of file port.h.

3.2.2.6 dual_media_fiber_speed

```
vtss_fiber_port_speed_t port_custom_conf_t::dual_media_fiber_speed
```

Speed for dual media fiber ports

Definition at line 278 of file port.h.

3.2.2.7 max_length

```
unsigned int port_custom_conf_t::max_length
```

Max frame length

Definition at line 279 of file port.h.

3.2.2.8 exc_col_cont

```
BOOL port_custom_conf_t::exc_col_cont
```

Excessive collision continuation

Definition at line 283 of file port.h.

3.2.2.9 adv_dis

```
u8 port_custom_conf_t::adv_dis
```

Auto neg advertisement disable

Definition at line 284 of file port.h.

3.2.2.10 max_tags

```
u8 port_custom_conf_t::max_tags
```

Maximum number of tags

Definition at line 285 of file port.h.

3.2.2.11 oper_up

```
BOOL port_custom_conf_t::oper_up
```

Force operational state up

Definition at line 286 of file port.h.

3.2.2.12 frame_length_chk

```
BOOL port_custom_conf_t::frame_length_chk
```

True to do 802.3 frame length check for ethertypes below 0x0600

Definition at line 287 of file port.h.

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

- vtss_api/include/vtss/api/port.h

3.3 vtss_aggr_mode_t Struct Reference

Aggregation traffic distribution mode.

```
#include <l2_types.h>
```

Data Fields

- BOOL smac_enable
- BOOL dmac_enable
- BOOL sip_dip_enable
- BOOL sport_dport_enable

3.3.1 Detailed Description

Aggregation traffic distribution mode.

Definition at line 39 of file l2_types.h.

3.3.2 Field Documentation

3.3.2.1 smac_enable

```
BOOL vtss_aggr_mode_t::smac_enable
```

Source MAC address

Definition at line 41 of file l2_types.h.

3.3.2.2 dmac_enable

```
BOOL vtss_aggr_mode_t::dmac_enable
```

Destination MAC address

Definition at line 42 of file l2_types.h.

3.3.2.3 sip_dip_enable

```
BOOL vtss_aggr_mode_t::sip_dip_enable
```

Source and destination IP address

Definition at line 43 of file l2_types.h.

3.3.2.4 sport_dport_enable

```
BOOL vtss_aggr_mode_t::sport_dport_enable
```

Source and destination UDP/TCP port

Definition at line 44 of file l2_types.h.

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

- vtss_api/include/vtss/api/l2_types.h

3.4 vtss_aneg_t Struct Reference

Auto negotiation struct.

```
#include <types.h>
```

Data Fields

- BOOL obey_pause
- BOOL generate_pause

3.4.1 Detailed Description

Auto negotiation struct.

Definition at line 482 of file types.h.

3.4.2 Field Documentation

3.4.2.1 obey_pause

```
BOOL vtss_aneg_t::obey_pause
```

This port should obey PAUSE frames

Definition at line 484 of file types.h.

3.4.2.2 generate_pause

```
BOOL vtss_aneg_t::generate_pause
```

Link partner obeys PAUSE frames

Definition at line 485 of file types.h.

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

- vtss_api/include/vtss/api/types.h

3.5 vtss_api_lock_t Struct Reference

API lock structure.

```
#include <vtss_misc_api.h>
```

Data Fields

- vtss_inst_t inst
- const char * function
- const char * file
- int line

3.5.1 Detailed Description

API lock structure.

Definition at line 285 of file vtss_misc_api.h.

3.5.2 Field Documentation

3.5.2.1 inst

```
vtss_inst_t vtss_api_lock_t::inst
```

Target instance reference

Definition at line 286 of file vtss_misc_api.h.

3.5.2.2 function

```
const char* vtss_api_lock_t::function
```

Function name

Definition at line 287 of file vtss_misc_api.h.

3.5.2.3 file

```
const char* vtss_api_lock_t::file
```

File name

Definition at line 288 of file vtss_misc_api.h.

3.5.2.4 line

```
int vtss_api_lock_t::line
```

Line number

Definition at line 289 of file vtss_misc_api.h.

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

- vtss_api/include/vtss_misc_api.h

3.6 vtss_debug_info_t Struct Reference

Debug information structure.

```
#include <vtss_misc_api.h>
```

Data Fields

- vtss_debug_layer_t layer
- vtss_debug_group_t group
- vtss_chip_no_t chip_no
- BOOL port_list [VTSS_PORT_ARRAY_SIZE]
- BOOL full
- BOOL clear
- BOOL vml_format

3.6.1 Detailed Description

Debug information structure.

Definition at line 243 of file vtss_misc_api.h.

3.6.2 Field Documentation

3.6.2.1 layer

```
vtss_debug_layer_t vtss_debug_info_t::layer
```

Layer

Definition at line 244 of file vtss_misc_api.h.

3.6.2.2 group

```
vtss_debug_group_t vtss_debug_info_t::group
```

Function group

Definition at line 245 of file vtss_misc_api.h.

3.6.2.3 chip_no

```
vtss_chip_no_t vtss_debug_info_t::chip_no
```

Chip number, multi-chip targets

Definition at line 246 of file vtss_misc_api.h.

3.6.2.4 port_list

```
BOOL vtss_debug_info_t::port_list[VTSS_PORT_ARRAY_SIZE]
```

Port list

Definition at line 247 of file vtss_misc_api.h.

3.6.2.5 full

```
BOOL vtss_debug_info_t::full
```

Full information dump

Definition at line 248 of file vtss_misc_api.h.

3.6.2.6 clear

```
BOOL vtss_debug_info_t::clear
```

Clear counters

Definition at line 249 of file vtss_misc_api.h.

3.6.2.7 vml_format

```
BOOL vtss_debug_info_t::vml_format
```

VML format register dump

Definition at line 250 of file vtss_misc_api.h.

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

- vtss_api/include/vtss_misc_api.h

3.7 vtss_debug_lock_t Struct Reference

API debug lock structure.

```
#include <vtss_misc_api.h>
```

Data Fields

- vtss_chip_no_t chip_no

3.7.1 Detailed Description

API debug lock structure.

Definition at line 307 of file vtss_misc_api.h.

3.7.2 Field Documentation

3.7.2.1 chip_no

```
vtss_chip_no_t vtss_debug_lock_t::chip_no
```

Chip number (if multi-chip instance).

Definition at line 308 of file vtss_misc_api.h.

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

- vtss_api/include/vtss_misc_api.h

3.8 vtss_ewis_aisl_cons_act_s Struct Reference

eWIS AIS-L consequent actions

`#include <vtss_wis_api.h>`

Data Fields

- BOOL ais_on_los
- BOOL ais_on_lof

3.8.1 Detailed Description

eWIS AIS-L consequent actions

Definition at line 77 of file vtss_wis_api.h.

3.8.2 Field Documentation

3.8.2.1 ais_on_los

`BOOL vtss_ewis_aisl_cons_act_s::ais_on_los`

TRUE = enable for AIS-L insertion on LOS

Definition at line 78 of file vtss_wis_api.h.

3.8.2.2 ais_on_lof

`BOOL vtss_ewis_aisl_cons_act_s::ais_on_lof`

TRUE = enable for AIS-L insertion on LOF

Definition at line 79 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.9 vtss_ewis_conf_s Struct Reference

eWIS configuration primitives, used to hold all the configuration parameters in the internal state in the API.

`#include <vtss_wis_api.h>`

Data Fields

- BOOL ewis_init_done
- vtss_ewis_static_conf_t static_conf
- vtss_ewis_mode_t ewis_mode
- vtss_ewis_cons_act_t section_cons_act
- vtss_ewis_tti_t section_txti
- vtss_ewis_force_mode_t force_mode
- vtss_ewis_tti_t path_txti
- vtss_ewis_tx_oh_t tx_oh
- vtss_ewis_tx_oh_passthru_t tx_oh_passthru
- vtss_ewis_sl_conf_t exp_sl
- vtss_ewis_test_conf_t test_conf
- vtss_ewis_counter_threshold_t ewis_cntr_thresh_conf
- vtss_ewis_perf_mode_t perf_mode

3.9.1 Detailed Description

eWIS configuration primitives, used to hold all the configuration parameters in the internal state in the API.

Definition at line 313 of file vtss_wis_api.h.

3.9.2 Field Documentation

3.9.2.1 ewis_init_done

```
BOOL vtss_ewis_conf_s::ewis_init_done
```

Indicate WIS mode is enabled

Definition at line 314 of file vtss_wis_api.h.

3.9.2.2 static_conf

```
vtss_ewis_static_conf_t vtss_ewis_conf_s::static_conf
```

Static configuration

Definition at line 315 of file vtss_wis_api.h.

3.9.2.3 ewis_mode

`vtss_ewis_mode_t vtss_ewis_conf_s::ewis_mode`

EWIS mode configuration

Definition at line 316 of file vtss_wis_api.h.

3.9.2.4 section_cons_act

`vtss_ewis_cons_act_t vtss_ewis_conf_s::section_cons_act`

Section consequent action configuraiton

Definition at line 317 of file vtss_wis_api.h.

3.9.2.5 section_txti

`vtss_ewis_tti_t vtss_ewis_conf_s::section_txti`

Section Trail Trace Identifier configuration

Definition at line 318 of file vtss_wis_api.h.

3.9.2.6 force_mode

`vtss_ewis_force_mode_t vtss_ewis_conf_s::force_mode`

Force mode configuration

Definition at line 319 of file vtss_wis_api.h.

3.9.2.7 path_txti

`vtss_ewis_tti_t vtss_ewis_conf_s::path_txti`

Path Trail Trace Identifier Configuration

Definition at line 320 of file vtss_wis_api.h.

3.9.2.8 tx_oh

vtss_ewis_tx_oh_t vtss_ewis_conf_s::tx_oh

Transmit Overhead

Definition at line 321 of file vtss_wis_api.h.

3.9.2.9 tx_oh_passthru

vtss_ewis_tx_oh_passthru_t vtss_ewis_conf_s::tx_oh_passthru

Transmit Overhead Passthru Configuration

Definition at line 322 of file vtss_wis_api.h.

3.9.2.10 exp_sl

vtss_ewis_sl_conf_t vtss_ewis_conf_s::exp_sl

Expected Signal Label

Definition at line 323 of file vtss_wis_api.h.

3.9.2.11 test_conf

vtss_ewis_test_conf_t vtss_ewis_conf_s::test_conf

EWIS Test Mode configuration

Definition at line 324 of file vtss_wis_api.h.

3.9.2.12 ewis_cntr_thresh_conf

vtss_ewis_counter_threshold_t vtss_ewis_conf_s::ewis_cntr_thresh_conf

EWIS counter threshold configuration

Definition at line 325 of file vtss_wis_api.h.

3.9.2.13 perf_mode

```
vtss_ewis_perf_mode_t vtss_ewis_conf_s::perf_mode
```

EWIS mode configuration

Definition at line 326 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.10 vtss_ewis_cons_act_s Struct Reference

eWIS consequent actions

```
#include <vtss_wis_api.h>
```

Data Fields

- vtss_ewis_aisl_cons_act_t aisl
- vtss_ewis_rdil_cons_act_t rdil
- vtss_ewis_fault_cons_act_t fault

3.10.1 Detailed Description

eWIS consequent actions

Definition at line 91 of file vtss_wis_api.h.

3.10.2 Field Documentation

3.10.2.1 aisl

```
vtss_ewis_aisl_cons_act_t vtss_ewis_cons_act_s::aisl
```

AIS-L consequent action configuration

Definition at line 92 of file vtss_wis_api.h.

3.10.2.2 rdil

`vtss_ewis_rdil_cons_act_t vtss_ewis_cons_act_s::rdil`

RDI-L consequent action configuration

Definition at line 93 of file vtss_wis_api.h.

3.10.2.3 fault

`vtss_ewis_fault_cons_act_t vtss_ewis_cons_act_s::fault`

FAULT condition consequent action configuration

Definition at line 94 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.11 vtss_ewis_counter_s Struct Reference

eWIS performance counters. These counters are free running counters that wraps to zero.

`#include <vtss_wis_api.h>`

Data Fields

- u16 pn_ebc_p
- u16 pf_ebc_p
- u32 pn_ebc_l
- u32 pf_ebc_l
- u16 pn_ebc_s

3.11.1 Detailed Description

eWIS performance counters. These counters are free running counters that wraps to zero.

Definition at line 187 of file vtss_wis_api.h.

3.11.2 Field Documentation

3.11.2.1 pn_ebc_p

```
u16 vtss_ewis_counter_s::pn_ebc_p
```

Path block error count

Definition at line 188 of file vtss_wis_api.h.

3.11.2.2 pf_ebc_p

```
u16 vtss_ewis_counter_s::pf_ebc_p
```

Far end path block error count

Definition at line 189 of file vtss_wis_api.h.

3.11.2.3 pn_ebc_l

```
u32 vtss_ewis_counter_s::pn_ebc_l
```

Near end line block (BIP) error count

Definition at line 190 of file vtss_wis_api.h.

3.11.2.4 pf_ebc_l

```
u32 vtss_ewis_counter_s::pf_ebc_l
```

Far end line block (BIP) error count

Definition at line 191 of file vtss_wis_api.h.

3.11.2.5 pn_ebc_s

```
u16 vtss_ewis_counter_s::pn_ebc_s
```

Section BIP error count

Definition at line 192 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.12 vtss_ewis_counter_threshold_s Struct Reference

eWIS performance counter thresholds.

`#include <vtss_wis_api.h>`

Data Fields

- u32 n_ebc_thr_s
- u32 n_ebc_thr_l
- u32 f_ebc_thr_l
- u32 n_ebc_thr_p
- u32 f_ebc_thr_p

3.12.1 Detailed Description

eWIS performance counter thresholds.

Definition at line 269 of file vtss_wis_api.h.

3.12.2 Field Documentation

3.12.2.1 n_ebc_thr_s

`u32 vtss_ewis_counter_threshold_s::n_ebc_thr_s`

Section error count (B1) threshold

Definition at line 270 of file vtss_wis_api.h.

3.12.2.2 n_ebc_thr_l

`u32 vtss_ewis_counter_threshold_s::n_ebc_thr_l`

Near end line error count (B2) threshold

Definition at line 271 of file vtss_wis_api.h.

3.12.2.3 f_ebc_thr_l

```
u32 vtss_ewis_counter_threshold_s::f_ebc_thr_l
```

Far end line error count threshold

Definition at line 272 of file vtss_wis_api.h.

3.12.2.4 n_ebc_thr_p

```
u32 vtss_ewis_counter_threshold_s::n_ebc_thr_p
```

Path block error count (B3) threshold

Definition at line 273 of file vtss_wis_api.h.

3.12.2.5 f_ebc_thr_p

```
u32 vtss_ewis_counter_threshold_s::f_ebc_thr_p
```

Far end path error count threshold

Definition at line 274 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.13 vtss_ewis_defects_s Struct Reference

eWIS defects

```
#include <vtss_wis_api.h>
```

Data Fields

- BOOL dlos_s
- BOOL doof_s
- BOOL dlof_s
- BOOL dais_l
- BOOL drdi_l
- BOOL dais_p
- BOOL dlop_p
- BOOL duneq_p
- BOOL drdi_p
- BOOL dlcd_p
- BOOL dplm_p
- BOOL dfais_p
- BOOL dfplm_p
- BOOL dfuneq_p

3.13.1 Detailed Description

eWIS defects

Definition at line 153 of file vtss_wis_api.h.

3.13.2 Field Documentation

3.13.2.1 dlos_s

```
BOOL vtss_ewis_defects_s::dlos_s
```

Loss of signal

Definition at line 154 of file vtss_wis_api.h.

3.13.2.2 doof_s

```
BOOL vtss_ewis_defects_s::doof_s
```

Out of frame

Definition at line 155 of file vtss_wis_api.h.

3.13.2.3 dlof_s

```
BOOL vtss_ewis_defects_s::dlof_s
```

Loss of frame

Definition at line 156 of file vtss_wis_api.h.

3.13.2.4 dais_l

```
BOOL vtss_ewis_defects_s::dais_l
```

Line alarm indication signal

Definition at line 157 of file vtss_wis_api.h.

3.13.2.5 drdi_l

```
BOOL vtss_ewis_defects_s::drdi_l
```

Line remote defect indication

Definition at line 158 of file vtss_wis_api.h.

3.13.2.6 dais_p

```
BOOL vtss_ewis_defects_s::dais_p
```

Path alarm indication signal

Definition at line 159 of file vtss_wis_api.h.

3.13.2.7 dlop_p

```
BOOL vtss_ewis_defects_s::dlop_p
```

Loss of pointer

Definition at line 160 of file vtss_wis_api.h.

3.13.2.8 duneq_p

```
BOOL vtss_ewis_defects_s::duneq_p
```

Path Unequipped, not supported in 8487/8488

Definition at line 161 of file vtss_wis_api.h.

3.13.2.9 drdi_p

```
BOOL vtss_ewis_defects_s::drdi_p
```

Path Remote Defect Indication, not supported in 8487/8488

Definition at line 162 of file vtss_wis_api.h.

3.13.2.10 dlcd_p

```
BOOL vtss_ewis_defects_s::dlcd_p
```

Path loss of code-group delineation, not supported in 8492

Definition at line 163 of file vtss_wis_api.h.

3.13.2.11 dplm_p

```
BOOL vtss_ewis_defects_s::dplm_p
```

Path loss of code-group delineation

Definition at line 164 of file vtss_wis_api.h.

3.13.2.12 dfais_p

```
BOOL vtss_ewis_defects_s::dfais_p
```

far-end AIS-P/LOP-P

Definition at line 166 of file vtss_wis_api.h.

3.13.2.13 dfplm_p

```
BOOL vtss_ewis_defects_s::dfplm_p
```

far-end PLM-P/LCD-P defect

Definition at line 167 of file vtss_wis_api.h.

3.13.2.14 dfuneq_p

```
BOOL vtss_ewis_defects_s::dfuneq_p
```

Far End Path Unequipped

Definition at line 168 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.14 vtss_ewis_fault_cons_act_s Struct Reference

eWIS fault mask configuration, i.e set up which defects trigger the Fault condition

`#include <vtss_wis_api.h>`

Data Fields

- BOOL fault_on_feplmp
- BOOL fault_on_feaisp
- BOOL fault_on_rdil
- BOOL fault_on_sef
- BOOL fault_on_lof
- BOOL fault_on_los
- BOOL fault_on_aisl
- BOOL fault_on_lcdp
- BOOL fault_on_plmp
- BOOL fault_on_aisp
- BOOL fault_on_lopp

3.14.1 Detailed Description

eWIS fault mask configuration, i.e set up which defects trigger the Fault condition

Definition at line 62 of file vtss_wis_api.h.

3.14.2 Field Documentation

3.14.2.1 fault_on_feplmp

`BOOL vtss_ewis_fault_cons_act_s::fault_on_feplmp`

TRUE = enable fault condition on far-end PLM-P

Definition at line 63 of file vtss_wis_api.h.

3.14.2.2 fault_on_feaisp

`BOOL vtss_ewis_fault_cons_act_s::fault_on_feaisp`

TRUE = enable fault condition on far-end AIS-P

Definition at line 64 of file vtss_wis_api.h.

3.14.2.3 fault_on_rdil

```
BOOL vtss_ewis_fault_cons_act_s::fault_on_rdil
```

TRUE = enable fault condition on RDI-L

Definition at line 65 of file vtss_wis_api.h.

3.14.2.4 fault_on_sef

```
BOOL vtss_ewis_fault_cons_act_s::fault_on_sef
```

TRUE = enable fault condition on SEF

Definition at line 66 of file vtss_wis_api.h.

3.14.2.5 fault_on_lof

```
BOOL vtss_ewis_fault_cons_act_s::fault_on_lof
```

TRUE = enable fault condition on LOF

Definition at line 67 of file vtss_wis_api.h.

3.14.2.6 fault_on_los

```
BOOL vtss_ewis_fault_cons_act_s::fault_on_los
```

TRUE = enable fault condition on LOS

Definition at line 68 of file vtss_wis_api.h.

3.14.2.7 fault_on_aisl

```
BOOL vtss_ewis_fault_cons_act_s::fault_on_aisl
```

TRUE = enable fault condition on AIS-L

Definition at line 69 of file vtss_wis_api.h.

3.14.2.8 fault_on_lcdp

```
BOOL vtss_ewis_fault_cons_act_s::fault_on_lcdp
```

TRUE = enable fault condition on LCD-P

Definition at line 70 of file vtss_wis_api.h.

3.14.2.9 fault_on_plmp

```
BOOL vtss_ewis_fault_cons_act_s::fault_on_plmp
```

TRUE = enable fault condition on PLM-P

Definition at line 71 of file vtss_wis_api.h.

3.14.2.10 fault_on_aisp

```
BOOL vtss_ewis_fault_cons_act_s::fault_on_aisp
```

TRUE = enable fault condition on AIS-P

Definition at line 72 of file vtss_wis_api.h.

3.14.2.11 fault_on_lopp

```
BOOL vtss_ewis_fault_cons_act_s::fault_on_lopp
```

TRUE = enable fault condition on LOP-P

Definition at line 73 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.15 vtss_ewis_force_mode_s Struct Reference

eWIS force modes

```
#include <vtss_wis_api.h>
```

Data Fields

- vtss_ewis_line_force_mode_t line_rx_force
- vtss_ewis_line_tx_force_mode_t line_tx_force
- vtss_ewis_path_force_mode_t path_force

3.15.1 Detailed Description

eWIS force modes

Definition at line 116 of file vtss_wis_api.h.

3.15.2 Field Documentation

3.15.2.1 line_rx_force

```
vtss_ewis_line_force_mode_t vtss_ewis_force_mode_s::line_rx_force
```

Line force configuration rx direction

Definition at line 117 of file vtss_wis_api.h.

3.15.2.2 line_tx_force

```
vtss_ewis_line_tx_force_mode_t vtss_ewis_force_mode_s::line_tx_force
```

Line force configuration tx direction

Definition at line 118 of file vtss_wis_api.h.

3.15.2.3 path_force

```
vtss_ewis_path_force_mode_t vtss_ewis_force_mode_s::path_force
```

Path force configuration

Definition at line 119 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.16 vtss_ewis_line_force_mode_s Struct Reference

eWIS line force mode

`#include <vtss_wis_api.h>`

Data Fields

- BOOL force_ais
- BOOL force_rdi

3.16.1 Detailed Description

eWIS line force mode

Definition at line 98 of file vtss_wis_api.h.

3.16.2 Field Documentation

3.16.2.1 force_ais

`BOOL vtss_ewis_line_force_mode_s::force_ais`

Force AIS-L configuration

Definition at line 99 of file vtss_wis_api.h.

3.16.2.2 force_rdi

`BOOL vtss_ewis_line_force_mode_s::force_rdi`

Force RDI-L configuration

Definition at line 100 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.17 vtss_ewis_line_tx_force_mode_s Struct Reference

eWIS line TX force mode

`#include <vtss_wis_api.h>`

Data Fields

- BOOL force_ais
- BOOL force_rdi

3.17.1 Detailed Description

eWIS line TX force mode

Definition at line 104 of file vtss_wis_api.h.

3.17.2 Field Documentation

3.17.2.1 force_ais

```
BOOL vtss_ewis_line_tx_force_mode_s::force_ais
```

Force transmission of AIS-L in the K2 byte

Definition at line 105 of file vtss_wis_api.h.

3.17.2.2 force_rdi

```
BOOL vtss_ewis_line_tx_force_mode_s::force_rdi
```

Force transmission of RDI-L in the K2 byte

Definition at line 106 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.18 vtss_ewis_path_force_mode_s Struct Reference

eWIS path force modes

```
#include <vtss_wis_api.h>
```

Data Fields

- BOOL force_uneq
- BOOL force_rdi

3.18.1 Detailed Description

eWIS path force modes

Definition at line 110 of file vtss_wis_api.h.

3.18.2 Field Documentation

3.18.2.1 force_uneq

```
BOOL vtss_ewis_path_force_mode_s::force_uneq
```

Force UNEQ-P configuration

Definition at line 111 of file vtss_wis_api.h.

3.18.2.2 force_rdi

```
BOOL vtss_ewis_path_force_mode_s::force_rdi
```

Force RDI-P configuration

Definition at line 112 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.19 vtss_ewis_perf_mode_s Struct Reference

eWIS Mode(Bit/Block) for the Performence Monitoring Counters

```
#include <vtss_wis_api.h>
```

Data Fields

- vtss_ewis_perf_cntr_mode_t pn_ebc_mode_s
- vtss_ewis_perf_cntr_mode_t pn_ebc_mode_l
- vtss_ewis_perf_cntr_mode_t pf_ebc_mode_l
- vtss_ewis_perf_cntr_mode_t pn_ebc_mode_p
- vtss_ewis_perf_cntr_mode_t pf_ebc_mode_p

3.19.1 Detailed Description

eWIS Mode(Bit/Block) for the Performence Monitoring Counters

Definition at line 129 of file vtss_wis_api.h.

3.19.2 Field Documentation

3.19.2.1 pn_ebc_mode_s

vtss_ewis_perf_cntr_mode_t vtss_ewis_perf_mode_s::pn_ebc_mode_s

Section BIP error count (B1))

Definition at line 130 of file vtss_wis_api.h.

3.19.2.2 pn_ebc_mode_l

vtss_ewis_perf_cntr_mode_t vtss_ewis_perf_mode_s::pn_ebc_mode_l

Near end line block (BIP) error count (B2)

Definition at line 131 of file vtss_wis_api.h.

3.19.2.3 pf_ebc_mode_l

vtss_ewis_perf_cntr_mode_t vtss_ewis_perf_mode_s::pf_ebc_mode_l

Far end line block (BIP) error count (REI-L)

Definition at line 132 of file vtss_wis_api.h.

3.19.2.4 pn_ebc_mode_p

vtss_ewis_perf_cntr_mode_t vtss_ewis_perf_mode_s::pn_ebc_mode_p

Path block error count (B3)

Definition at line 133 of file vtss_wis_api.h.

3.19.2.5 pf_ebc_mode_p

```
vtss_ewis_perf_cntr_mode_t vtss_ewis_perf_mode_s::pf_ebc_mode_p
```

Far end path block error count (REI-P)

Definition at line 134 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.20 vtss_ewis_perf_s Struct Reference

eWIS performance primitives. These data are assumed to be read once every sec. The counters holds increments compared to previous read. The namings and definitions are taken from ITU-T rec G.783.

```
#include <vtss_wis_api.h>
```

Data Fields

- u32 pn_ebc_s
- u32 pn_ebc_l
- u32 pf_ebc_l
- u32 pn_ebc_p
- u32 pf_ebc_p

3.20.1 Detailed Description

eWIS performance primitives. These data are assumed to be read once every sec. The counters holds increments compared to previous read. The namings and definitions are taken from ITU-T rec G.783.

Definition at line 176 of file vtss_wis_api.h.

3.20.2 Field Documentation

3.20.2.1 pn_ebc_s

```
u32 vtss_ewis_perf_s::pn_ebc_s
```

Section BIP error count

Definition at line 177 of file vtss_wis_api.h.

3.20.2.2 pn_ebc_l

```
u32 vtss_ewis_perf_s::pn_ebc_l
```

Near end line block (BIP) error count

Definition at line 178 of file vtss_wis_api.h.

3.20.2.3 pf_ebc_l

```
u32 vtss_ewis_perf_s::pf_ebc_l
```

Far end line block (BIP) error count

Definition at line 179 of file vtss_wis_api.h.

3.20.2.4 pn_ebc_p

```
u32 vtss_ewis_perf_s::pn_ebc_p
```

Path block error count

Definition at line 180 of file vtss_wis_api.h.

3.20.2.5 pf_ebc_p

```
u32 vtss_ewis_perf_s::pf_ebc_p
```

Far end path block error count

Definition at line 181 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.21 vtss_ewis_rdil_cons_act_s Struct Reference

eWIS RDI-L consequent actions

```
#include <vtss_wis_api.h>
```

Data Fields

- BOOL rdil_on_los
- BOOL rdil_on_lof
- BOOL rdil_on_lopc
- BOOL rdil_on_ais_l

3.21.1 Detailed Description

eWIS RDI-L consequent actions

Definition at line 83 of file vtss_wis_api.h.

3.21.2 Field Documentation

3.21.2.1 rdil_on_los

```
BOOL vtss_ewis_rdil_cons_act_s::rdil_on_los
```

TRUE = enable for RDI-L backreporting on LOS

Definition at line 84 of file vtss_wis_api.h.

3.21.2.2 rdil_on_lof

```
BOOL vtss_ewis_rdil_cons_act_s::rdil_on_lof
```

TRUE = enable for RDI-L backreporting on LOF

Definition at line 85 of file vtss_wis_api.h.

3.21.2.3 rdil_on_lopc

```
BOOL vtss_ewis_rdil_cons_act_s::rdil_on_lopc
```

TRUE = enable for RDI-L backreporting on LOPC, not supported in 8492

Definition at line 86 of file vtss_wis_api.h.

3.21.2.4 rdil_on_ais_l

```
BOOL vtss_ewis_rdil_cons_act_s::rdil_on_ais_l
```

TRUE = enable for RDI-L backreporting on AIS_L, not supported in 8492

Definition at line 87 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.22 vtss_ewis_sl_conf_s Struct Reference

signal label configuration

```
#include <vtss_wis_api.h>
```

Data Fields

- u8 exsl

3.22.1 Detailed Description

signal label configuration

Definition at line 306 of file vtss_wis_api.h.

3.22.2 Field Documentation

3.22.2.1 exsl

```
u8 vtss_ewis_sl_conf_s::exsl
```

expected signal label value

Definition at line 307 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.23 vtss_ewis_static_conf_s Struct Reference

eWIS static configuration data,

`#include <vtss_wis_api.h>`

Data Fields

- u16 ewis_txctrl1
- u16 ewis_txctrl2
- u16 ewis_rx_ctrl1
- u16 ewis_mode_ctrl
- u16 ewis_tx_a1_a2
- u16 ewis_tx_c2_h1
- u16 ewis_tx_h2_h3
- u16 ewis_tx_z0_e1
- u16 ewis_rx_frm_ctrl1
- u16 ewis_rx_frm_ctrl2
- u16 ewis_lof_ctrl1
- u16 ewis_lof_ctrl2
- u16 ewis_rx_err_frc1
- u16 ewis_pmtick_ctrl
- u16 ewis_cnt_cfg

3.23.1 Detailed Description

eWIS static configuration data,

Note

This is specific to 8487/8488-15 and should not be used for Daytona.

Definition at line 287 of file vtss_wis_api.h.

3.23.2 Field Documentation

3.23.2.1 ewis_txctrl1

`u16 vtss_ewis_static_conf_s::ewis_txctrl1`

WIS Vendor Specific Tx Control 1

Definition at line 288 of file vtss_wis_api.h.

3.23.2.2 ewis_txctrl2

```
u16 vtss_ewis_static_conf_s::ewis_txctrl2
```

WIS Vendor Specific Tx Control 2

Definition at line 289 of file vtss_wis_api.h.

3.23.2.3 ewis_rx_ctrl1

```
u16 vtss_ewis_static_conf_s::ewis_rx_ctrl1
```

E-WIS Rx Control 1

Definition at line 290 of file vtss_wis_api.h.

3.23.2.4 ewis_mode_ctrl

```
u16 vtss_ewis_static_conf_s::ewis_mode_ctrl
```

E-WIS Mode Control (incl expected C2 Path label)

Definition at line 291 of file vtss_wis_api.h.

3.23.2.5 ewis_tx_a1_a2

```
u16 vtss_ewis_static_conf_s::ewis_tx_a1_a2
```

E-WIS Tx A1/A2 Octets (frame alignment)

Definition at line 292 of file vtss_wis_api.h.

3.23.2.6 ewis_tx_c2_h1

```
u16 vtss_ewis_static_conf_s::ewis_tx_c2_h1
```

E-WIS Tx C2/H1 Octets

Definition at line 293 of file vtss_wis_api.h.

3.23.2.7 ewis_tx_h2_h3

```
u16 vtss_ewis_static_conf_s::ewis_tx_h2_h3
```

E-WIS Tx H2/H3 Octets

Definition at line 294 of file vtss_wis_api.h.

3.23.2.8 ewis_tx_z0_e1

```
u16 vtss_ewis_static_conf_s::ewis_tx_z0_e1
```

E-WIS Tx Z0/E1 Octets

Definition at line 295 of file vtss_wis_api.h.

3.23.2.9 ewis_rx_frm_ctrl1

```
u16 vtss_ewis_static_conf_s::ewis_rx_frm_ctrl1
```

E-WIS Rx Framer Control 1

Definition at line 296 of file vtss_wis_api.h.

3.23.2.10 ewis_rx_frm_ctrl2

```
u16 vtss_ewis_static_conf_s::ewis_rx_frm_ctrl2
```

E-WIS Rx Framer Control 2

Definition at line 297 of file vtss_wis_api.h.

3.23.2.11 ewis_lof_ctrl1

```
u16 vtss_ewis_static_conf_s::ewis_lof_ctrl1
```

E-WIS Loss of Frame Control 1

Definition at line 298 of file vtss_wis_api.h.

3.23.2.12 ewis_lof_ctrl2

```
u16 vtss_ewis_static_conf_s::ewis_lof_ctrl2
```

E-WIS Loss of Frame Control 2

Definition at line 299 of file vtss_wis_api.h.

3.23.2.13 ewis_rx_err_frc1

```
u16 vtss_ewis_static_conf_s::ewis_rx_err_frc1
```

E-WIS Rx Error Force Control 1 (incl RXLOF_ON_LOPC and APS_THRES configuration)

Definition at line 300 of file vtss_wis_api.h.

3.23.2.14 ewis_pmtick_ctrl

```
u16 vtss_ewis_static_conf_s::ewis_pmtick_ctrl
```

E-WIS Performance Monitor Control (define how PMTICK works)

Definition at line 301 of file vtss_wis_api.h.

3.23.2.15 ewis_cnt_cfg

```
u16 vtss_ewis_static_conf_s::ewis_cnt_cfg
```

E-WIS Counter Configuration (bit/block mode)

Definition at line 302 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.24 vtss_ewis_status_s Struct Reference

eWIS status

```
#include <vtss_wis_api.h>
```

Data Fields

- BOOL fault
- BOOL link_stat

3.24.1 Detailed Description

eWIS status

Definition at line 147 of file vtss_wis_api.h.

3.24.2 Field Documentation

3.24.2.1 fault

```
BOOL vtss_ewis_status_s::fault
```

Fault condition (Latch on high) i.e. = true if any fault has occurred since previous read

Definition at line 148 of file vtss_wis_api.h.

3.24.2.2 link_stat

```
BOOL vtss_ewis_status_s::link_stat
```

Link status condition (Latch on low) i.e. = false if any link error has occurred since previous read

Definition at line 149 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.25 vtss_ewis_test_conf_s Struct Reference

eWIS test configuration

```
#include <vtss_wis_api.h>
```

Data Fields

- BOOL loopback
- vtss_ewis_test_pattern_t test_pattern_gen
- vtss_ewis_test_pattern_t test_pattern_ana

3.25.1 Detailed Description

eWIS test configuration

Definition at line 206 of file vtss_wis_api.h.

3.25.2 Field Documentation

3.25.2.1 loopback

```
BOOL vtss_ewis_test_conf_s::loopback
```

loop output from Tx to Rx

Definition at line 207 of file vtss_wis_api.h.

3.25.2.2 test_pattern_gen

```
vtss_ewis_test_pattern_t vtss_ewis_test_conf_s::test_pattern_gen
```

test pattern generation configuration

Definition at line 208 of file vtss_wis_api.h.

3.25.2.3 test_pattern_ana

```
vtss_ewis_test_pattern_t vtss_ewis_test_conf_s::test_pattern_ana
```

test pattern analyzer configuration

Definition at line 209 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.26 vtss_ewis_test_status_s Struct Reference

eWIS test status

```
#include <vtss_wis_api.h>
```

Data Fields

- u16 tstpat_cnt
- BOOL ana_sync

3.26.1 Detailed Description

eWIS test status

Definition at line 213 of file vtss_wis_api.h.

3.26.2 Field Documentation

3.26.2.1 tstpat_cnt

```
u16 vtss_ewis_test_status_s::tstpat_cnt
```

PRBS31 test pattern error counter.

Definition at line 214 of file vtss_wis_api.h.

3.26.2.2 ana_sync

```
BOOL vtss_ewis_test_status_s::ana_sync
```

PRBS31 pattern checker is synchronized to the data.

Definition at line 215 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.27 vtss_ewis_tti_s Struct Reference

Trail Trace Identifier type.

```
#include <vtss_wis_api.h>
```

Data Fields

- vtss_ewis_tti_mode_t mode
- u8 tti [64]
- BOOL valid

3.27.1 Detailed Description

Trail Trace Identifier type.

Definition at line 55 of file vtss_wis_api.h.

3.27.2 Field Documentation

3.27.2.1 mode

```
vtss_ewis_tti_mode_t vtss_ewis_tti_s::mode
```

trace identifier mode (1,16,64 bytes)

Definition at line 56 of file vtss_wis_api.h.

3.27.2.2 tti

```
u8 vtss_ewis_tti_s::tti[64]
```

trace identifier value

Definition at line 57 of file vtss_wis_api.h.

3.27.2.3 valid

```
BOOL vtss_ewis_tti_s::valid
```

Identifies the Accepted valid TTI

Definition at line 58 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.28 vtss_ewis_tx_oh_s Struct Reference

WIS transmitted overhead data. only a few oh bytes can be set dynamically. These OH bytes are not configurable from the API: H4 : multiframe indicator M0/M1: STS-1 Line Remote Error Indication (REI) G1: Path status.

```
#include <vtss_wis_api.h>
```

Data Fields

- u8 tx_dcc_s [3]
- u8 tx_e1
- u8 tx_f1
- u8 tx_z0
- u8 tx_dcc_l [9]
- u8 tx_e2
- u16 tx_k1_k2
- u8 tx_s1
- u16 tx_z1_z2
- u8 tx_c2
- u8 tx_f2
- u8 tx_n1
- u16 tx_z3_z4

3.28.1 Detailed Description

WIS transmitted overhead data. only a few oh bytes can be set dynamically. These OH bytes are not configurable from the API: H4 : multiframe indicator M0/M1: STS-1 Line Remote Error Indication (REI) G1: Path status.

Definition at line 224 of file vtss_wis_api.h.

3.28.2 Field Documentation

3.28.2.1 tx_dcc_s

```
u8 vtss_ewis_tx_oh_s::tx_dcc_s[3]
```

< Section Overhead: Section Data Communications Channel(DCC) D1-D3

Definition at line 226 of file vtss_wis_api.h.

3.28.2.2 tx_e1

```
u8 vtss_ewis_tx_oh_s::tx_e1
```

Orderwire

Definition at line 227 of file vtss_wis_api.h.

3.28.2.3 tx_f1

```
u8 vtss_ewis_tx_oh_s::tx_f1
```

Section User Channel

Definition at line 228 of file vtss_wis_api.h.

3.28.2.4 tx_z0

```
u8 vtss_ewis_tx_oh_s::tx_z0
```

Reserved for Section growth line overhead:

Definition at line 229 of file vtss_wis_api.h.

3.28.2.5 tx_dcc_l

```
u8 vtss_ewis_tx_oh_s::tx_dcc_l[9]
```

Line Data Communications Channel (DCC) D4-D12

Definition at line 231 of file vtss_wis_api.h.

3.28.2.6 tx_e2

```
u8 vtss_ewis_tx_oh_s::tx_e2
```

Orderwire

Definition at line 232 of file vtss_wis_api.h.

3.28.2.7 tx_k1_k2

```
u16 vtss_ewis_tx_oh_s::tx_k1_k2
```

Automatic protection switch (APS) channel and Line Remote Defect Identifier (RDI-L)

Definition at line 233 of file vtss_wis_api.h.

3.28.2.8 tx_s1

```
u8 vtss_ewis_tx_oh_s::tx_s1
```

Synchronization messaging

Definition at line 234 of file vtss_wis_api.h.

3.28.2.9 tx_z1_z2

```
u16 vtss_ewis_tx_oh_s::tx_z1_z2
```

Reserved for Line growth path overhead:

Definition at line 235 of file vtss_wis_api.h.

3.28.2.10 tx_c2

```
u8 vtss_ewis_tx_oh_s::tx_c2
```

Transmitted C2 path label

Definition at line 237 of file vtss_wis_api.h.

3.28.2.11 tx_f2

```
u8 vtss_ewis_tx_oh_s::tx_f2
```

Path User Channel

Definition at line 238 of file vtss_wis_api.h.

3.28.2.12 tx_n1

```
u8 vtss_ewis_tx_oh_s::tx_n1
```

Tandem connection maintenance/Path data channel

Definition at line 239 of file vtss_wis_api.h.

3.28.2.13 tx_z3_z4

```
u16 vtss_ewis_tx_oh_s::tx_z3_z4
```

Reserved for Path growth

Definition at line 240 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.29 vtss_ewis_tx_passthru_s Struct Reference

eWIS overhead passthru configuration.

```
#include <vtss_wis_api.h>
```

Data Fields

- BOOL tx_j0
- BOOL tx_z0
- BOOL tx_b1
- BOOL tx_e1
- BOOL tx_f1
- BOOL tx_dcc_s
- BOOL tx_soh
- BOOL tx_b2
- BOOL tx_k1
- BOOL tx_k2
- BOOL tx_reil
- BOOL tx_dcc_l
- BOOL tx_s1
- BOOL tx_e2
- BOOL tx_z1_z2
- BOOL tx_loh

3.29.1 Detailed Description

eWIS overhead passthru configuration.

Definition at line 245 of file vtss_wis_api.h.

3.29.2 Field Documentation

3.29.2.1 tx_j0

```
BOOL vtss_ewis_tx_passthru_s::tx_j0
```

< Section Overhead: j0 Section TTI passthrough

Definition at line 247 of file vtss_wis_api.h.

3.29.2.2 tx_z0

```
BOOL vtss_ewis_tx_passthru_s::tx_z0
```

z0 Section growth passthrough

Definition at line 248 of file vtss_wis_api.h.

3.29.2.3 tx_b1

```
BOOL vtss_ewis_tx_passthru_s::tx_b1
```

b1 BIP passthrough

Definition at line 249 of file vtss_wis_api.h.

3.29.2.4 tx_e1

```
BOOL vtss_ewis_tx_passthru_s::tx_e1
```

e1 order wire passthrough

Definition at line 250 of file vtss_wis_api.h.

3.29.2.5 tx_f1

```
BOOL vtss_ewis_tx_passthru_s::tx_f1
```

f1 Section user channel

Definition at line 251 of file vtss_wis_api.h.

3.29.2.6 tx_dcc_s

```
BOOL vtss_ewis_tx_passthru_s::tx_dcc_s
```

Section Data communication channel passthrough D1-D3

Definition at line 252 of file vtss_wis_api.h.

3.29.2.7 tx_soh

```
BOOL vtss_ewis_tx_passthru_s::tx_soh
```

Section Reserved National and unused bytes passthrough line overhead:

Definition at line 253 of file vtss_wis_api.h.

3.29.2.8 tx_b2

```
BOOL vtss_ewis_tx_passthru_s::tx_b2
```

b2 BIP passthrough

Definition at line 256 of file vtss_wis_api.h.

3.29.2.9 tx_k1

```
BOOL vtss_ewis_tx_passthru_s::tx_k1
```

k1 passthrough

Definition at line 257 of file vtss_wis_api.h.

3.29.2.10 tx_k2

```
BOOL vtss_ewis_tx_passthru_s::tx_k2
```

k2 passthrough

Definition at line 258 of file vtss_wis_api.h.

3.29.2.11 tx_reil

```
BOOL vtss_ewis_tx_passthru_s::tx_reil
```

reil passthrough

Definition at line 259 of file vtss_wis_api.h.

3.29.2.12 tx_dcc_l

```
BOOL vtss_ewis_tx_passthru_s::tx_dcc_l
```

Section Data communication channel passthrough D4-D12

Definition at line 260 of file vtss_wis_api.h.

3.29.2.13 tx_s1

```
BOOL vtss_ewis_tx_passthru_s::tx_s1
```

Synchronization messaging passthrough

Definition at line 261 of file vtss_wis_api.h.

3.29.2.14 tx_e2

```
BOOL vtss_ewis_tx_passthru_s::tx_e2
```

order wire passthrough

Definition at line 262 of file vtss_wis_api.h.

3.29.2.15 tx_z1_z2

```
BOOL vtss_ewis_tx_passthru_s::tx_z1_z2
```

Reserved for path growth passthrough

Definition at line 263 of file vtss_wis_api.h.

3.29.2.16 tx_loh

```
BOOL vtss_ewis_tx_passthru_s::tx_loh
```

Line Reserved National and unused bytes passthrough

Definition at line 264 of file vtss_wis_api.h.

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

- vtss_api/include/vtss_wis_api.h

3.30 vtss_gpio_10g_gpio_mode_t Struct Reference

GPIO configured mode.

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- vtss_10g_phy_gpio_t mode
- vtss_port_no_t port
- vtss_gpio_10g_input_t input
- vtss_gpio_10g_gpio_intr_sgnl_t in_sig
- vtss_gpio_10g_no_t p_gpio
- vtss_gpio_10g_chan_intrpt_t c_intrpt
- u16 source
- u32 aggr_intrpt
- BOOL use_as_intrpt
- BOOL p_gpio_intrpt

3.30.1 Detailed Description

GPIO configured mode.

GPIO input modes

Definition at line 2480 of file vtss_phy_10g_api.h.

3.30.2 Field Documentation

3.30.2.1 mode

vtss_10g_phy_gpio_t vtss_gpio_10g_gpio_mode_t::mode

Mode of this GPIO pin

Definition at line 2482 of file vtss_phy_10g_api.h.

3.30.2.2 port

vtss_port_no_t vtss_gpio_10g_gpio_mode_t::port

In case of VTSS_10G_PHY_GPIO_WIS_INT mode, this is the interrupt port number that is related to this GPIO In case of VTSS_10G_PHY_GPIO_PCS_RX_FAULT mode, this is the PCS status port number that is related to this GPIO

Definition at line 2483 of file vtss_phy_10g_api.h.

3.30.2.3 input

vtss_gpio_10g_input_t vtss_gpio_10g_gpio_mode_t::input

GPIO input modes

Definition at line 2485 of file vtss_phy_10g_api.h.

3.30.2.4 in_sig

vtss_gpio_10g_gpio_intr_sgnl_t vtss_gpio_10g_gpio_mode_t::in_sig

Internal signal that to be routed through GPIO

Definition at line 2486 of file vtss_phy_10g_api.h.

3.30.2.5 p_gpio

vtss_gpio_10g_no_t vtss_gpio_10g_gpio_mode_t::p_gpio

Per channel GPIO number

Definition at line 2487 of file vtss_phy_10g_api.h.

3.30.2.6 c_intrpt

```
vtss_gpio_10g_chan_intrpt_t vtss_gpio_10g_gpio_mode_t::c_intrpt
```

Per Channel interrupt,

Definition at line 2488 of file vtss_phy_10g_api.h.

3.30.2.7 source

```
u16 vtss_gpio_10g_gpio_mode_t::source
```

source of GPIO,approriate value from GPIO_OUT_CFG_X register field GPIO_X_SEL

Definition at line 2489 of file vtss_phy_10g_api.h.

3.30.2.8 aggr_intrpt

```
u32 vtss_gpio_10g_gpio_mode_t::aggr_intrpt
```

Bitmask corresponds to aggregated interrupt

Definition at line 2490 of file vtss_phy_10g_api.h.

3.30.2.9 use_as_intrpt

```
BOOL vtss_gpio_10g_gpio_mode_t::use_as_intrpt
```

Change in GPIO generates GPIO interrupt ,supported on MALIBU

Definition at line 2491 of file vtss_phy_10g_api.h.

3.30.2.10 p_gpio_intrpt

```
BOOL vtss_gpio_10g_gpio_mode_t::p_gpio_intrpt
```

Port GPIO Change interrupt

Definition at line 2492 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.31 vtss_init_conf_t Struct Reference

Initialization configuration.

`#include <vtss_init_api.h>`

Data Fields

- vtss_reg_read_t reg_read
- vtss_reg_write_t reg_write
- vtss_miim_read_t miim_read
- vtss_miim_write_t miim_write
- vtss_mmd_read_t mmd_read
- vtss_mmd_read_inc_t mmd_read_inc
- vtss_mmd_write_t mmd_write
- vtss_spi_read_write_t spi_read_write
- vtss_spi_32bit_read_write_t spi_32bit_read_write
- vtss_spi_64bit_read_write_t spi_64bit_read_write
- BOOL warm_start_enable
- vtss_restart_info_src_t restart_info_src
- vtss_port_no_t restart_info_port
- vtss_pi_conf_t pi

3.31.1 Detailed Description

Initialization configuration.

Definition at line 515 of file vtss_init_api.h.

3.31.2 Field Documentation

3.31.2.1 reg_read

`vtss_reg_read_t vtss_init_conf_t::reg_read`

Register read function

Definition at line 517 of file vtss_init_api.h.

3.31.2.2 reg_write

`vtss_reg_write_t vtss_init_conf_t::reg_write`

Register write function

Definition at line 518 of file vtss_init_api.h.

3.31.2.3 miim_read

`vtss_miim_read_t vtss_init_conf_t::miim_read`

MII management read function

Definition at line 521 of file vtss_init_api.h.

3.31.2.4 miim_write

`vtss_miim_write_t vtss_init_conf_t::miim_write`

MII management write function

Definition at line 522 of file vtss_init_api.h.

3.31.2.5 mmd_read

`vtss_mmd_read_t vtss_init_conf_t::mmd_read`

MMD management read function

Definition at line 525 of file vtss_init_api.h.

3.31.2.6 mmd_read_inc

`vtss_mmd_read_inc_t vtss_init_conf_t::mmd_read_inc`

MMD management read increment function

Definition at line 526 of file vtss_init_api.h.

3.31.2.7 mmd_write

`vtss_mmd_write_t vtss_init_conf_t::mmd_write`

MMD management write function

Definition at line 527 of file vtss_init_api.h.

3.31.2.8 spi_read_write

`vtss_spi_read_write_t vtss_init_conf_t::spi_read_write`

Board specific SPI read/write callout function

Definition at line 529 of file vtss_init_api.h.

3.31.2.9 spi_32bit_read_write

`vtss_spi_32bit_read_write_t vtss_init_conf_t::spi_32bit_read_write`

Board specific SPI read/write callout function for 32 bit data

Definition at line 531 of file vtss_init_api.h.

3.31.2.10 spi_64bit_read_write

`vtss_spi_64bit_read_write_t vtss_init_conf_t::spi_64bit_read_write`

Board specific SPI read/write callout function for 64 bit data

Definition at line 532 of file vtss_init_api.h.

3.31.2.11 warm_start_enable

`BOOL vtss_init_conf_t::warm_start_enable`

Allow warm start

Definition at line 535 of file vtss_init_api.h.

3.31.2.12 restart_info_src

`vtss_restart_info_src_t vtss_init_conf_t::restart_info_src`

Source of restart information

Definition at line 536 of file vtss_init_api.h.

3.31.2.13 restart_info_port

```
vtss_port_no_t vtss_init_conf_t::restart_info_port
```

Port used to store PHY restart information

Definition at line 537 of file vtss_init_api.h.

3.31.2.14 pi

```
vtss_pi_conf_t vtss_init_conf_t::pi
```

Parallel Interface configuration

Definition at line 547 of file vtss_init_api.h.

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

- vtss_api/include/vtss_init_api.h

3.32 vtss_inst_create_t Struct Reference

Create structure.

```
#include <vtss_init_api.h>
```

Data Fields

- vtss_target_type_t target

3.32.1 Detailed Description

Create structure.

Definition at line 78 of file vtss_init_api.h.

3.32.2 Field Documentation

3.32.2.1 target

```
vtss_target_type_t vtss_inst_create_t::target
```

Target type

Definition at line 79 of file vtss_init_api.h.

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

- vtss_api/include/vtss_init_api.h

3.33 vtss_ip_addr_t Struct Reference

Either an IPv4 or IPv6 address.

```
#include <types.h>
```

Data Fields

- vtss_ip_type_t type
- union {
  vtss_ipv4_t ipv4
  vtss_ipv6_t ipv6
  } addr

3.33.1 Detailed Description

Either an IPv4 or IPv6 address.

Definition at line 813 of file types.h.

3.33.2 Field Documentation

3.33.2.1 type

```
vtss_ip_type_t vtss_ip_addr_t::type
```

Union type

Definition at line 814 of file types.h.

3.33.2.2 ipv4

```
vtss_ipv4_t vtss_ip_addr_t::ipv4
```

IPv4 address

Definition at line 816 of file types.h.

3.33.2.3 ipv6

```
vtss_ipv6_t vtss_ip_addr_t::ipv6
```

IPv6 address

Definition at line 817 of file types.h.

3.33.2.4 addr

```
union { ... } vtss_ip_addr_t::addr
```

IP address

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

- vtss_api/include/vtss/api/types.h

3.34 vtss_ip_network_t Struct Reference

IPv6 network.

```
#include <types.h>
```

Data Fields

- vtss_ip_addr_t address
- vtss_prefix_size_t prefix_size

3.34.1 Detailed Description

IPv6 network.

Definition at line 836 of file types.h.

3.34.2 Field Documentation

3.34.2.1 address

`vtss_ip_addr_t vtss_ip_network_t::address`

Network address

Definition at line 838 of file types.h.

3.34.2.2 prefix_size

`vtss_prefix_size_t vtss_ip_network_t::prefix_size`

Prefix size

Definition at line 839 of file types.h.

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

- vtss_api/include/vtss/api/types.h

3.35 vtss_ipv4_network_t Struct Reference

IPv4 network.

`#include <types.h>`

Data Fields

- vtss_ipv4_t address
- vtss_prefix_size_t prefix_size

3.35.1 Detailed Description

IPv4 network.

Definition at line 822 of file types.h.

3.35.2 Field Documentation

3.35.2.1 address

```
vtss_ipv4_t vtss_ipv4_network_t::address
```

Network address

Definition at line 824 of file types.h.

3.35.2.2 prefix_size

```
vtss_prefix_size_t vtss_ipv4_network_t::prefix_size
```

Prefix size

Definition at line 825 of file types.h.

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

- vtss_api/include/vtss/api/types.h

3.36 vtss_ipv4_uc_t Struct Reference

IPv4 unicast routing entry.

```
#include <types.h>
```

Data Fields

- vtss_ipv4_network_t network
- vtss_ipv4_t destination

3.36.1 Detailed Description

IPv4 unicast routing entry.

Definition at line 852 of file types.h.

3.36.2 Field Documentation

3.36.2.1 network

```
vtss_ipv4_network_t vtss_ipv4_uc_t::network
```

Network to route

Definition at line 854 of file types.h.

3.36.2.2 destination

```
vtss_ipv4_t vtss_ipv4_uc_t::destination
```

IP address of next-hop router. Zero if local route

Definition at line 855 of file types.h.

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

- vtss_api/include/vtss/api/types.h

3.37 vtss_ipv6_network_t Struct Reference

IPv6 network.

```
#include <types.h>
```

Data Fields

- vtss_ipv6_t address
- vtss_prefix_size_t prefix_size

3.37.1 Detailed Description

IPv6 network.

Definition at line 829 of file types.h.

3.37.2 Field Documentation

3.37.2.1 address

```
vtss_ipv6_t vtss_ipv6_network_t::address
```

Network address

Definition at line 831 of file types.h.

3.37.2.2 prefix_size

```
vtss_prefix_size_t vtss_ipv6_network_t::prefix_size
```

Prefix size

Definition at line 832 of file types.h.

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

- vtss_api/include/vtss/api/types.h

3.38 vtss_ipv6_t Struct Reference

IPv6 address/mask.

```
#include <types.h>
```

Data Fields

- u8 addr [16]

3.38.1 Detailed Description

IPv6 address/mask.

Definition at line 797 of file types.h.

3.38.2 Field Documentation

3.38.2.1 addr

```
u8 vtss_ipv6_t::addr[16]
```

Address

Definition at line 799 of file types.h.

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

- vtss_api/include/vtss/api/types.h

3.39 vtss_ipv6_uc_t Struct Reference

IPv6 routing entry.

```
#include <types.h>
```

Data Fields

- vtss_ipv6_network_t network
- vtss_ipv6_t destination

3.39.1 Detailed Description

IPv6 routing entry.

Definition at line 860 of file types.h.

3.39.2 Field Documentation

3.39.2.1 network

```
vtss_ipv6_network_t vtss_ipv6_uc_t::network
```

Network to route

Definition at line 862 of file types.h.

3.39.2.2 destination

```
vtss_ipv6_t vtss_ipv6_uc_t::destination
```

IP address of next-hop router. Zero if local route

Definition at line 863 of file types.h.

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

- vtss_api/include/vtss/api/types.h

3.40 vtss_l3_counters_t Struct Reference

Routing interface statics counter.

```
#include <types.h>
```

Data Fields

- u64 ipv4uc_received_octets
- u64 ipv4uc_received_frames
- u64 ipv6uc_received_octets
- u64 ipv6uc_received_frames
- u64 ipv4uc_transmitted_octets
- u64 ipv4uc_transmitted_frames
- u64 ipv6uc_transmitted_octets
- u64 ipv6uc_transmitted_frames

3.40.1 Detailed Description

Routing interface statics counter.

Definition at line 886 of file types.h.

3.40.2 Field Documentation

3.40.2.1 ipv4uc_received_octets

```
u64 vtss_l3_counters_t::ipv4uc_received_octets
```

IPv4UC octets received and hardware forwarded

Definition at line 887 of file types.h.

3.40.2.2 ipv4uc_received_frames

```
u64 vtss_l3_counters_t::ipv4uc_received_frames
```

IPv4UC frames received and hardware forwarded

Definition at line 888 of file types.h.

3.40.2.3 ipv6uc_received_octets

```
u64 vtss_l3_counters_t::ipv6uc_received_octets
```

IPv6UC octets received and hardware forwarded

Definition at line 889 of file types.h.

3.40.2.4 ipv6uc_received_frames

```
u64 vtss_l3_counters_t::ipv6uc_received_frames
```

IPv6UC frames received and hardware forwarded

Definition at line 890 of file types.h.

3.40.2.5 ipv4uc_transmitted_octets

```
u64 vtss_l3_counters_t::ipv4uc_transmitted_octets
```

IPv4UC octets transmitted

Definition at line 892 of file types.h.

3.40.2.6 ipv4uc_transmitted_frames

```
u64 vtss_l3_counters_t::ipv4uc_transmitted_frames
```

IPv4UC frames transmitted

Definition at line 893 of file types.h.

3.40.2.7 ipv6uc_transmitted_octets

```
u64 vtss_l3_counters_t::ipv6uc_transmitted_octets
```

IPv6UC octets transmitted

Definition at line 894 of file types.h.

3.40.2.8 ipv6uc_transmitted_frames

```
u64 vtss_l3_counters_t::ipv6uc_transmitted_frames
```

IPv6UC frames transmitted

Definition at line 895 of file types.h.

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

- vtss_api/include/vtss/api/types.h

3.41 vtss_mac_t Struct Reference

MAC Address.

```
#include <types.h>
```

Data Fields

- u8 addr [6]

3.41.1 Detailed Description

MAC Address.

Definition at line 643 of file types.h.

3.41.2 Field Documentation

3.41.2.1 addr

```
u8 vtss_mac_t::addr[6]
```

Network byte order

Definition at line 645 of file types.h.

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

- vtss_api/include/vtss/api/types.h

3.42 vtss_mtimer_t Struct Reference

Timer structure.

```
#include <vtss_os_linux.h>
```

Data Fields

- struct timeval timeout
- struct timeval now

3.42.1 Detailed Description

Timer structure.

Definition at line 88 of file vtss_os_linux.h.

3.42.2 Field Documentation

3.42.2.1 timeout

```
struct timeval vtss_mtimer_t::timeout
```

Timeout

Definition at line 89 of file vtss_os_linux.h.

3.42.2.2 now

```
struct timeval vtss_mtimer_t::now
```

Time right now

Definition at line 90 of file vtss_os_linux.h.

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

- vtss_api/include/vtss_os_linux.h

3.43 vtss_os_timestamp_t Struct Reference

```
#include <vtss_misc_api.h>
```

Data Fields

- unsigned int hw_cnt

3.43.1 Detailed Description

VTSS_OS_TIMESTAMP_TYPE VTSS_OS_TIMESTAMP() These two provides a mean to have the API timestamp events for use by the application. It is up to the platform specific code to implement the actual functions to obtain the timestamp. The implementation *must* be callable from interrupt context, so no implicit waits or sleeps are allowed.

Definition at line 1072 of file vtss_misc_api.h.

3.43.2 Field Documentation

3.43.2.1 hw_cnt

```
unsigned int vtss_os_timestamp_t::hw_cnt
```

hardware counter

Definition at line 1073 of file vtss_misc_api.h.

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

- vtss_api/include/vtss_misc_api.h

3.44 vtss_phy_10g_apc_conf_t Struct Reference

10G Phy APC configuration

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- vtss_phy_10g_ib_apc_op_mode_t op_mode
- BOOL op_mode_flag

3.44.1 Detailed Description

10G Phy APC configuration

Definition at line 241 of file vtss_phy_10g_api.h.

3.44.2 Field Documentation

3.44.2.1 op_mode

```
vtss_phy_10g_ib_apc_op_mode_t vtss_phy_10g_apc_conf_t::op_mode
```

APC operation

Definition at line 242 of file vtss_phy_10g_api.h.

3.44.2.2 op_mode_flag

```
BOOL vtss_phy_10g_apc_conf_t::op_mode_flag
```

APC operation flag,eg: TRUE= APC_RESET,FALSE = APC_RESET clear

Definition at line 243 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.45 vtss_phy_10g_apc_status_t Struct Reference

10G Phy APC status

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL reset
- BOOL freeze

3.45.1 Detailed Description

10G Phy APC status

Definition at line 247 of file vtss_phy_10g_api.h.

3.45.2 Field Documentation

3.45.2.1 reset

```
BOOL vtss_phy_10g_apc_status_t::reset
```

APC reset status

Definition at line 248 of file vtss_phy_10g_api.h.

3.45.2.2 freeze

```
BOOL vtss_phy_10g_apc_status_t::freeze
```

APC freeze status

Definition at line 249 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.46 vtss_phy_10g_auto_failover_conf_t Struct Reference

10G PHY Automatic Failover configuration

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- vtss_port_no_t port_no
- vtss_phy_10g_auto_failover_event_t evnt
- u16 trig_ch_id
- BOOL is_host_side
- u16 channel_id
- vtss_gpio_10g_no_t v_gpio
- vtss_gpio_10g_no_t a_gpio
- BOOL enable
- vtss_phy_10g_auto_failover_filter_t filter
- u16 fltr_val

3.46.1 Detailed Description

10G PHY Automatic Failover configuration

Definition at line 1807 of file vtss_phy_10g_api.h.

3.46.2 Field Documentation

3.46.2.1 port_no

```
vtss_port_no_t vtss_phy_10g_auto_failover_conf_t::port_no
```

port number

Definition at line 1808 of file vtss_phy_10g_api.h.

3.46.2.2 evnt

```
vtss_phy_10g_auto_failover_event_t vtss_phy_10g_auto_failover_conf_t::evnt
```

Auto failover event selection

Definition at line 1809 of file vtss_phy_10g_api.h.

3.46.2.3 trig_ch_id

```
u16 vtss_phy_10g_auto_failover_conf_t::trig_ch_id
```

Channel ID that triggers event,source

Definition at line 1810 of file vtss_phy_10g_api.h.

3.46.2.4 is_host_side

```
BOOL vtss_phy_10g_auto_failover_conf_t::is_host_side
```

Protection switch configuration is on line(RX) or host side(Rx)

Definition at line 1811 of file vtss_phy_10g_api.h.

3.46.2.5 channel_id

```
u16 vtss_phy_10g_auto_failover_conf_t::channel_id
```

channel to be switched,destination

Definition at line 1812 of file vtss_phy_10g_api.h.

3.46.2.6 v_gpio

```
vtss_gpio_10g_no_t vtss_phy_10g_auto_failover_conf_t::v_gpio
```

virtual GPIO pin to be triggering the event

Definition at line 1813 of file vtss_phy_10g_api.h.

3.46.2.7 a_gpio

```
vtss_gpio_10g_no_t vtss_phy_10g_auto_failover_conf_t::a_gpio
```

actual GPIO pin to be triggering the event

Definition at line 1814 of file vtss_phy_10g_api.h.

3.46.2.8 enable

```
BOOL vtss_phy_10g_auto_failover_conf_t::enable
```

Enable or disable auto failover

Definition at line 1815 of file vtss_phy_10g_api.h.

3.46.2.9 filter

```
vtss_phy_10g_auto_failover_filter_t vtss_phy_10g_auto_failover_conf_t::filter
```

Number of CSR clock cycles

Definition at line 1816 of file vtss_phy_10g_api.h.

3.46.2.10 fltr_val

```
u16 vtss_phy_10g_auto_failover_conf_t::fltr_val
```

value of filter if chosen

Definition at line 1817 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.47 vtss_phy_10g_ckout_conf_t Struct Reference

10G Phy CKOUT config data

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- vtss_ckout_data_sel_t mode
- vtss_phy_10g_squelch_src_t src
- vtss_phy_10g_ckout_freq_t freq
- BOOL squelch_inv
- BOOL enable
- ckout_sel_t ckout_sel

3.47.1 Detailed Description

10G Phy CKOUT config data

Malibu Only

Definition at line 962 of file vtss_phy_10g_api.h.

3.47.2 Field Documentation

3.47.2.1 mode

vtss_ckout_data_sel_t vtss_phy_10g_ckout_conf_t::mode

CKOUT output clock mode

Definition at line 963 of file vtss_phy_10g_api.h.

3.47.2.2 src

vtss_phy_10g_squelch_src_t vtss_phy_10g_ckout_conf_t::src

CKOUT squelch source

Definition at line 964 of file vtss_phy_10g_api.h.

3.47.2.3 freq

vtss_phy_10g_ckout_freq_t vtss_phy_10g_ckout_conf_t::freq

CKOUT clock frequency

Definition at line 965 of file vtss_phy_10g_api.h.

3.47.2.4 squelch_inv

BOOL vtss_phy_10g_ckout_conf_t::squelch_inv

'0'- Use squelch source src as is, '1'-Invert

Definition at line 966 of file vtss_phy_10g_api.h.

3.47.2.5 enable

BOOL vtss_phy_10g_ckout_conf_t::enable

'1'- Enable CKOUT, '0'-Disable

Definition at line 967 of file vtss_phy_10g_api.h.

3.47.2.6 ckout_sel

```
ckout_sel_t vtss_phy_10g_ckout_conf_t::ckout_sel
```

CKOUT sel eg-'0' for CKOUT0, '1' for CKOUT1

Definition at line 968 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.48 vtss_phy_10g_clause_37_adv_t Struct Reference

Advertisement control data for Clause 37 aneg.

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL fdx
- BOOL hdx
- BOOL symmetric_pause
- BOOL asymmetric_pause
- vtss_phy_10g_clause_37_remote_fault_t remote_fault
- BOOL acknowledge
- BOOL next_page

3.48.1 Detailed Description

Advertisement control data for Clause 37 aneg.

Definition at line 1507 of file vtss_phy_10g_api.h.

3.48.2 Field Documentation

3.48.2.1 fdx

```
BOOL vtss_phy_10g_clause_37_adv_t::fdx
```

(FD)

Definition at line 1509 of file vtss_phy_10g_api.h.

3.48.2.2 hdx

```
BOOL vtss_phy_10g_clause_37_adv_t::hdx
```

(HD) ,Not supported

Definition at line 1510 of file vtss_phy_10g_api.h.

3.48.2.3 symmetric_pause

```
BOOL vtss_phy_10g_clause_37_adv_t::symmetric_pause
```

(PS1)

Definition at line 1511 of file vtss_phy_10g_api.h.

3.48.2.4 asymmetric_pause

```
BOOL vtss_phy_10g_clause_37_adv_t::asymmetric_pause
```

(PS2)

Definition at line 1512 of file vtss_phy_10g_api.h.

3.48.2.5 remote_fault

```
vtss_phy_10g_clause_37_remote_fault_t vtss_phy_10g_clause_37_adv_t::remote_fault
```

(RF1) + (RF2) , would be generated according to condition

Definition at line 1513 of file vtss_phy_10g_api.h.

3.48.2.6 acknowledge

```
BOOL vtss_phy_10g_clause_37_adv_t::acknowledge
```

(Ack) , would be generated according to condition

Definition at line 1514 of file vtss_phy_10g_api.h.

3.48.2.7 next_page

```
BOOL vtss_phy_10g_clause_37_adv_t::next_page
```

(NP) ,Not supported

Definition at line 1515 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.49 vtss_phy_10g_clause_37_cmn_status_t Struct Reference

Clause 37 Auto-negotiation status for line and host.

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- vtss_phy_10g_clause_37_status_t line
- vtss_phy_10g_clause_37_status_t host

3.49.1 Detailed Description

Clause 37 Auto-negotiation status for line and host.

Definition at line 1529 of file vtss_phy_10g_api.h.

3.49.2 Field Documentation

3.49.2.1 line

```
vtss_phy_10g_clause_37_status_t vtss_phy_10g_clause_37_cmn_status_t::line
```

Line clause 37 status

Definition at line 1531 of file vtss_phy_10g_api.h.

3.49.2.2 host

```
vtss_phy_10g_clause_37_status_t vtss_phy_10g_clause_37_cmn_status_t::host
```

Host clause 37 status

Definition at line 1532 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.50 vtss_phy_10g_clause_37_control_t Struct Reference

Clause 37 control struct.

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL enable
- vtss_phy_10g_clause_37_adv_t advertisement
- BOOL enable_pass_thru
- BOOL line
- BOOL host
- BOOL l_h

3.50.1 Detailed Description

Clause 37 control struct.

Definition at line 1537 of file vtss_phy_10g_api.h.

3.50.2 Field Documentation

3.50.2.1 enable

```
BOOL vtss_phy_10g_clause_37_control_t::enable
```

Enable of Autoneg

Definition at line 1539 of file vtss_phy_10g_api.h.

3.50.2.2 advertisement

```
vtss_phy_10g_clause_37_adv_t vtss_phy_10g_clause_37_control_t::advertisement
```

Clause 37 Advertisement data

Definition at line 1540 of file vtss_phy_10g_api.h.

3.50.2.3 enable_pass_thru

```
BOOL vtss_phy_10g_clause_37_control_t::enable_pass_thru
```

Enables pass through mode in VENICE/MALIBU

Definition at line 1541 of file vtss_phy_10g_api.h.

3.50.2.4 line

```
BOOL vtss_phy_10g_clause_37_control_t::line
```

Line:TRUE for line side

Definition at line 1542 of file vtss_phy_10g_api.h.

3.50.2.5 host

```
BOOL vtss_phy_10g_clause_37_control_t::host
```

Host:True for host side

Definition at line 1543 of file vtss_phy_10g_api.h.

3.50.2.6 l_h

```
BOOL vtss_phy_10g_clause_37_control_t::l_h
```

Both Host and line side

Definition at line 1544 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.51 vtss_phy_10g_clause_37_status_t Struct Reference

Clause 37 Auto-negotiation status.

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL link
- struct {
  BOOL complete
  vtss_phy_10g_clause_37_adv_t partner_advertisement
  } autoneg

3.51.1 Detailed Description

Clause 37 Auto-negotiation status.

Definition at line 1519 of file vtss_phy_10g_api.h.

3.51.2 Field Documentation

3.51.2.1 link

```
BOOL vtss_phy_10g_clause_37_status_t::link
```

FALSE if link has been down since last status read

Definition at line 1521 of file vtss_phy_10g_api.h.

3.51.2.2 complete

```
BOOL vtss_phy_10g_clause_37_status_t::complete
```

Aneg completion status

Definition at line 1523 of file vtss_phy_10g_api.h.

3.51.2.3 partner_advertisement

```
vtss_phy_10g_clause_37_adv_t vtss_phy_10g_clause_37_status_t::partner_advertisement
```

Clause 37 Advertisement control data

Definition at line 1524 of file vtss_phy_10g_api.h.

3.51.2.4 autoneg

```
struct { ... } vtss_phy_10g_clause_37_status_t::autoneg
```

Autoneg status

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

- vtss_api/include/vtss_phy_10g_api.h

3.52 vtss_phy_10g_clk_src_t Struct Reference

10G Phy CLOCK Source Selection

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL is_high_amp

3.52.1 Detailed Description

10G Phy CLOCK Source Selection

Definition at line 192 of file vtss_phy_10g_api.h.

3.52.2 Field Documentation

3.52.2.1 is_high_amp

```
BOOL vtss_phy_10g_clk_src_t::is_high_amp
```

Amplitude selection HIGH or LOW

Definition at line 193 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.53 vtss_phy_10g_cnt_t Struct Reference

10G Phy Sublayer counters

`#include <vtss_phy_10g_api.h>`

Data Fields

- vtss_phy_pcs_cnt_t pcs

3.53.1 Detailed Description

10G Phy Sublayer counters

Definition at line 1676 of file vtss_phy_10g_api.h.

3.53.2 Field Documentation

3.53.2.1 pcs

`vtss_phy_pcs_cnt_t vtss_phy_10g_cnt_t::pcs`

PCS counters

Definition at line 1679 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.54 vtss_phy_10g_fw_status_t Struct Reference

Firmware status.

`#include <vtss_phy_10g_api.h>`

Data Fields

- u16 edc_fw_rev
- BOOL edc_fw_chksum
- BOOL icpu_activity
- BOOL edc_fw_api_load

3.54.1 Detailed Description

Firmware status.

Definition at line 2741 of file vtss_phy_10g_api.h.

3.54.2 Field Documentation

3.54.2.1 edc_fw_rev

```
u16 vtss_phy_10g_fw_status_t::edc_fw_rev
```

FW revision

Definition at line 2742 of file vtss_phy_10g_api.h.

3.54.2.2 edc_fw_chksum

```
BOOL vtss_phy_10g_fw_status_t::edc_fw_chksum
```

FW chksum. Fail=0, Pass=1

Definition at line 2743 of file vtss_phy_10g_api.h.

3.54.2.3 icpu_activity

```
BOOL vtss_phy_10g_fw_status_t::icpu_activity
```

iCPU activity. Not Running=0, Running=1

Definition at line 2744 of file vtss_phy_10g_api.h.

3.54.2.4 edc_fw_api_load

```
BOOL vtss_phy_10g_fw_status_t::edc_fw_api_load
```

EDC FW is loaded through API No=0, Yes=1

Definition at line 2745 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.55 vtss_phy_10g_host_clk_conf_t Struct Reference

10G Phy Host clock config data

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- vtss_phy_10g_clk_sel_t mode
- vtss_phy_10g_recvrd_clk_sel_t recvrd_clk_sel
- u8 clk_sel_no

3.55.1 Detailed Description

10G Phy Host clock config data

Malibu Only

Definition at line 1052 of file vtss_phy_10g_api.h.

3.55.2 Field Documentation

3.55.2.1 mode

```
vtss_phy_10g_clk_sel_t vtss_phy_10g_host_clk_conf_t::mode
```

Host side output clock mode

Definition at line 1053 of file vtss_phy_10g_api.h.

3.55.2.2 recvrd_clk_sel

```
vtss_phy_10g_recvrd_clk_sel_t vtss_phy_10g_host_clk_conf_t::recvrd_clk_sel
```

Recovered clock selection

Definition at line 1054 of file vtss_phy_10g_api.h.

3.55.2.3 clk_sel_no

```
u8 vtss_phy_10g_host_clk_conf_t::clk_sel_no
```

Host clock select No(0-3)

Definition at line 1055 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.56 vtss_phy_10g_ib_conf_t Struct Reference

10G Phy IB configuration

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- ib_par_cfg offs
- ib_par_cfg gain
- ib_par_cfg gainadj
- ib_par_cfg l
- ib_par_cfg c
- ib_par_cfg agc
- ib_par_cfg dfe1
- ib_par_cfg dfe2
- ib_par_cfg dfe3
- ib_par_cfg dfe4
- u8 ld
- u8 prbs
- BOOL prbs_inv
- u32 apc_bit_mask
- u32 freeze_bit_mask
- u32 config_bit_mask
- BOOL is_host

3.56.1 Detailed Description

10G Phy IB configuration

Definition at line 213 of file vtss_phy_10g_api.h.

3.56.2 Field Documentation

3.56.2.1 offs

ib_par_cfg vtss_phy_10g_ib_conf_t::offs

Equalizer offset value

Definition at line 214 of file vtss_phy_10g_api.h.

3.56.2.2 gain

ib_par_cfg vtss_phy_10g_ib_conf_t::gain

Equalizer gain value

Definition at line 215 of file vtss_phy_10g_api.h.

3.56.2.3 gainadj

ib_par_cfg vtss_phy_10g_ib_conf_t::gainadj

IB gain adjustment

Definition at line 216 of file vtss_phy_10g_api.h.

3.56.2.4 l

ib_par_cfg vtss_phy_10g_ib_conf_t::l

Equalizer L value

Definition at line 217 of file vtss_phy_10g_api.h.

3.56.2.5 c

ib_par_cfg vtss_phy_10g_ib_conf_t::c

Equalizer C value

Definition at line 218 of file vtss_phy_10g_api.h.

3.56.2.6 agc

ib_par_cfg vtss_phy_10g_ib_conf_t::agc

AGC value

Definition at line 219 of file vtss_phy_10g_api.h.

3.56.2.7 dfe1

ib_par_cfg vtss_phy_10g_ib_conf_t::dfe1

DFE1 active value

Definition at line 220 of file vtss_phy_10g_api.h.

3.56.2.8 dfe2

ib_par_cfg vtss_phy_10g_ib_conf_t::dfe2

DFE2 active value

Definition at line 221 of file vtss_phy_10g_api.h.

3.56.2.9 dfe3

ib_par_cfg vtss_phy_10g_ib_conf_t::dfe3

DFE3 active value

Definition at line 222 of file vtss_phy_10g_api.h.

3.56.2.10 dfe4

ib_par_cfg vtss_phy_10g_ib_conf_t::dfe4

DFE4 active value

Definition at line 223 of file vtss_phy_10g_api.h.

3.56.2.11 ld

```
u8 vtss_phy_10g_ib_conf_t::ld
```

level detect

Definition at line 224 of file vtss_phy_10g_api.h.

3.56.2.12 prbs

```
u8 vtss_phy_10g_ib_conf_t::prbs
```

PRBS RX pattern selected

Definition at line 225 of file vtss_phy_10g_api.h.

3.56.2.13 prbs_inv

```
BOOL vtss_phy_10g_ib_conf_t::prbs_inv
```

PRBS inversions selected

Definition at line 226 of file vtss_phy_10g_api.h.

3.56.2.14 apc_bit_mask

```
u32 vtss_phy_10g_ib_conf_t::apc_bit_mask
```

Bit mask that has the information of the all the parameters whether they are being controlled by APC

Definition at line 227 of file vtss_phy_10g_api.h.

3.56.2.15 freeze_bit_mask

```
u32 vtss_phy_10g_ib_conf_t::freeze_bit_mask
```

Bit mask that has the information of the all parameters that are frozen to the value

Definition at line 228 of file vtss_phy_10g_api.h.

3.56.2.16 config_bit_mask

```
u32 vtss_phy_10g_ib_conf_t::config_bit_mask
```

Bit mask that has the information of the all parameters that are to be configured

Definition at line 229 of file vtss_phy_10g_api.h.

3.56.2.17 is_host

```
BOOL vtss_phy_10g_ib_conf_t::is_host
```

Configuration is on Host or line

Definition at line 230 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.57 vtss_phy_10g_ib_status_t Struct Reference

10G Phy IB configuration

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- vtss_phy_10g_ib_conf_t ib_conf
- BOOL sig_det
- u16 bit_errors

3.57.1 Detailed Description

10G Phy IB configuration

Definition at line 234 of file vtss_phy_10g_api.h.

3.57.2 Field Documentation

3.57.2.1 ib_conf

```
vtss_phy_10g_ib_conf_t vtss_phy_10g_ib_status_t::ib_conf
```

Current status of IB configuraion

Definition at line 235 of file vtss_phy_10g_api.h.

3.57.2.2 sig_det

```
BOOL vtss_phy_10g_ib_status_t::sig_det
```

Signal detect

Definition at line 236 of file vtss_phy_10g_api.h.

3.57.2.3 bit_errors

```
u16 vtss_phy_10g_ib_status_t::bit_errors
```

Bit errors if PRBS is enabled

Definition at line 237 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.58 vtss_phy_10g_ib_storage_t Struct Reference

VSCOPE fast scan storage.

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL ib_storage_bool [BOOLEAN_STORAGE_COUNT]
- u32 ib_storage [UNSIGNED_STORAGE_COUNT]

3.58.1 Detailed Description

VSCOPE fast scan storage.

Definition at line 1898 of file vtss_phy_10g_api.h.

3.58.2 Field Documentation

3.58.2.1 ib_storage_bool

```
BOOL vtss_phy_10g_ib_storage_t::ib_storage_bool[BOOLEAN_STORAGE_COUNT]
```

boolean values to be stored in vtss_state during vscope fast scan configuration

Definition at line 1899 of file vtss_phy_10g_api.h.

3.58.2.2 ib_storage

```
u32 vtss_phy_10g_ib_storage_t::ib_storage[UNSIGNED_STORAGE_COUNT]
```

u8 values to be stored in vtss_state during vscope fast scan configuration

Definition at line 1900 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.59 vtss_phy_10g_id_t Struct Reference

10G Phy part number and revision

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- u16 part_number
- u16 revision
- u16 channel_id
- vtss_phy_10g_family_t family
- vtss_phy_10g_type_t type
- vtss_port_no_t phy_api_base_no
- u16 device_feature_status

3.59.1 Detailed Description

10G Phy part number and revision

Definition at line 2269 of file vtss_phy_10g_api.h.

3.59.2 Field Documentation

3.59.2.1 part_number

```
u16 vtss_phy_10g_id_t::part_number
```

Part number (Hex)

Definition at line 2271 of file vtss_phy_10g_api.h.

3.59.2.2 revision

```
u16 vtss_phy_10g_id_t::revision
```

Chip revision

Definition at line 2272 of file vtss_phy_10g_api.h.

3.59.2.3 channel_id

```
u16 vtss_phy_10g_id_t::channel_id
```

Channel id

Definition at line 2273 of file vtss_phy_10g_api.h.

3.59.2.4 family

```
vtss_phy_10g_family_t vtss_phy_10g_id_t::family
```

Phy Family

Definition at line 2274 of file vtss_phy_10g_api.h.

3.59.2.5 type

```
vtss_phy_10g_type_t vtss_phy_10g_id_t::type
```

Phy id (Decimal)

Definition at line 2275 of file vtss_phy_10g_api.h.

3.59.2.6 phy_api_base_no

```
vtss_port_no_t vtss_phy_10g_id_t::phy_api_base_no
```

First API no within this phy (in case of multiple channels)

Definition at line 2276 of file vtss_phy_10g_api.h.

3.59.2.7 device_feature_status

```
u16 vtss_phy_10g_id_t::device_feature_status
```

Device features depending on EFUSE

Definition at line 2277 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.60 vtss_phy_10g_init_parm_t Struct Reference

10G Phy Initialization configuration

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- vtss_channel_t channel_conf

3.60.1 Detailed Description

10G Phy Initialization configuration

Definition at line 432 of file vtss_phy_10g_api.h.

3.60.2 Field Documentation

3.60.2.1 channel_conf

```
vtss_channel_t vtss_phy_10g_init_parm_t::channel_conf
```

Channel configuration selection,manual or auto

Definition at line 433 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.61 vtss_phy_10g_jitter_conf_t Struct Reference

10G Phy Optimisation of jitter performance

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL incr_levn
- u8 levn
- u8 vtail

3.61.1 Detailed Description

10G Phy Optimisation of jitter performance

Definition at line 302 of file vtss_phy_10g_api.h.

3.61.2 Field Documentation

3.61.2.1 incr_levn

```
BOOL vtss_phy_10g_jitter_conf_t::incr_levn
```

Increase LevN

Definition at line 303 of file vtss_phy_10g_api.h.

3.61.2.2 levn

u8 vtss_phy_10g_jitter_conf_t::levn

Selects levn value depending on incr_levn value incr_levn=1: levn: it is from 31: ∼300mVpp to 0: ∼1075mVpp. incr_levn=0: levn: it is from 31: ∼500mVpp to 0: ∼1275mVpp. Maximum achievable amplitude depends on supply Voltage Recommended settings for SR at 10.3125Gbps For Insertion loss < 4db Levn: 7 Incr_levn: 1 (Also the API Default) Recommended settings for DAC Irrespective of Insertion loss Levn: 7 Incr_levn: 1 (Also the API Default)

Definition at line 304 of file vtss_phy_10g_api.h.

3.61.2.3 vtail

u8 vtss_phy_10g_jitter_conf_t::vtail

Vtail configuration 0: reserved, 1: 75mV, 2:100mV. Recommended settings(default in API) SR mode(@10.3125↩ Gbps), insertion loss < 4dB, value for vtail: 2 DAC mode, insertion loss independent,value for vtail: 2

Definition at line 314 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.62 vtss_phy_10g_lane_sync_conf_t Struct Reference

10G Phy Lane SYNC Configuration

#include <vtss_phy_10g_api.h>

Data Fields

- BOOL enable
- vtss_phy_10g_tx_macro_t tx_macro
- vtss_phy_10g_rx_macro_t rx_macro
- u8 rx_ch
- u8 tx_ch

3.62.1 Detailed Description

10G Phy Lane SYNC Configuration

Malibu Only

Definition at line 1130 of file vtss_phy_10g_api.h.

3.62.2 Field Documentation

3.62.2.1 enable

```
BOOL vtss_phy_10g_lane_sync_conf_t::enable
```

Enable/Disable LANE SYNC

Definition at line 1131 of file vtss_phy_10g_api.h.

3.62.2.2 tx_macro

```
vtss_phy_10g_tx_macro_t vtss_phy_10g_lane_sync_conf_t::tx_macro
```

Tx Macro to lane sync to (destination)

Definition at line 1132 of file vtss_phy_10g_api.h.

3.62.2.3 rx_macro

```
vtss_phy_10g_rx_macro_t vtss_phy_10g_lane_sync_conf_t::rx_macro
```

Rx Macro to lane sync from (Source)

Definition at line 1133 of file vtss_phy_10g_api.h.

3.62.2.4 rx_ch

```
u8 vtss_phy_10g_lane_sync_conf_t::rx_ch
```

0[Default] to 3- NA If rx_macro is SREFCLK

Definition at line 1134 of file vtss_phy_10g_api.h.

3.62.2.5 tx_ch

```
u8 vtss_phy_10g_lane_sync_conf_t::tx_ch
```

0[Default] to 3- NA If tx_macro is SCKOUT

Definition at line 1135 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.63 vtss_phy_10g_line_clk_conf_t Struct Reference

10G Phy Line clock config data

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- vtss_phy_10g_clk_sel_t mode
- vtss_phy_10g_recvrd_clk_sel_t recvrd_clk_sel
- u8 clk_sel_no

3.63.1 Detailed Description

10G Phy Line clock config data

Malibu Only

Definition at line 1026 of file vtss_phy_10g_api.h.

3.63.2 Field Documentation

3.63.2.1 mode

```
vtss_phy_10g_clk_sel_t vtss_phy_10g_line_clk_conf_t::mode
```

Line side output clock mode

Definition at line 1027 of file vtss_phy_10g_api.h.

3.63.2.2 recvrd_clk_sel

```
vtss_phy_10g_recvrd_clk_sel_t vtss_phy_10g_line_clk_conf_t::recvrd_clk_sel
```

Recovered clock selection

Definition at line 1028 of file vtss_phy_10g_api.h.

3.63.2.3 clk_sel_no

```
u8 vtss_phy_10g_line_clk_conf_t::clk_sel_no
```

Line clock select No(0-3)

Definition at line 1029 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.64 vtss_phy_10g_loopback_t Struct Reference

10G Phy system and network loopbacks

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- vtss_lb_type_t lb_type
- BOOL enable

3.64.1 Detailed Description

10G Phy system and network loopbacks

Definition at line 1625 of file vtss_phy_10g_api.h.

3.64.2 Field Documentation

3.64.2.1 lb_type

```
vtss_lb_type_t vtss_phy_10g_loopback_t::lb_type
```

Looback types

Definition at line 1626 of file vtss_phy_10g_api.h.

3.64.2.2 enable

```
BOOL vtss_phy_10g_loopback_t::enable
```

Enable/Disable loopback given in <lb_type>

Definition at line 1627 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.65 vtss_phy_10g_mode_t Struct Reference

10G Phy operating mode

```
#include <vtss_phy_10g_api.h>
```

Public Types

- enum { VTSS_EDC_FW_LOAD_MDIO, VTSS_EDC_FW_LOAD_NOTHING }
  EDC modes.

Data Fields

- oper_mode_t oper_mode
- vtss_phy_interface_mode interface
- vtss_wrefclk_t wrefclk
- BOOL high_input_gain
- BOOL xfi_pol_invert
- BOOL xaui_lane_flip
- vtss_channel_t channel_id
- BOOL hl_clk_synth
- vtss_recvrd_t rcvrd_clk
- vtss_recvrdclk_cdr_div_t rcvrd_clk_div
- vtss_srefclk_div_t sref_clk_div
- vtss_wref_clk_div_t wref_clk_div
- enum vtss_phy_10g_mode_t:: { ... } edc_fw_load
  EDC modes.

- struct {
  BOOL use_conf
  BOOL ob_conf
  BOOL ib_conf
  BOOL dig_offset_reg
  BOOL apc_offs_ctrl
  BOOL apc_line_ld_ctrl
  BOOL apc_host_ld_ctrl
  u32 d_filter
  u32 cfg0
  u32 ib_ini_lp
  u32 ib_min_lp
  u32 ib_max_lp
  u32 apc_eqz_offs_par_cfg
  u32 apc_line_eqz_ld_ctrl
  u32 apc_host_eqz_ld_ctrl
  BOOL l_offset_guard
  BOOL h_offset_guard
  } serdes_conf

  Serdes parameters.
- apc_ib_regulator_t apc_ib_regulator
- u16 pma_txratecontrol
- BOOL venice_rev_a_los_detection_workaround
- ddr_mode_t ddr_mode
- clk_mstr_t master
- vtss_rptr_rate_t rate
- vtss_phy_10g_polarity_inv_t polarity
- BOOL is_host_wan
- vtss_phy_10g_clk_src_t h_clk_src
- vtss_phy_10g_clk_src_t l_clk_src
- BOOL lref_for_host
- vtss_phy_6g_link_partner_distance_t link_6g_distance
- vtss_phy_10g_media_t h_media
- vtss_phy_10g_media_t l_media
- vtss_phy_10g_ib_conf_t h_ib_conf
- vtss_phy_10g_ib_conf_t l_ib_conf
- vtss_phy_10g_apc_conf_t h_apc_conf
- vtss_phy_10g_apc_conf_t l_apc_conf
- BOOL enable_pass_thru
- BOOL is_init
- BOOL sd6g_calib_done

3.65.1 Detailed Description

10G Phy operating mode

Definition at line 333 of file vtss_phy_10g_api.h.

3.65.2 Member Enumeration Documentation

3.65.2.1 anonymous enum

```
anonymous enum
```

EDC modes.

Enumerator

VTSS_EDC_FW_LOAD_MDIO	Load EDC FW through MDIO to iCPU
VTSS_EDC_FW_LOAD_NOTHING	Do not load FW to iCPU

Definition at line 357 of file vtss_phy_10g_api.h.

3.65.3 Field Documentation

3.65.3.1 oper_mode

```
oper_mode_t vtss_phy_10g_mode_t::oper_mode
```

Phy operational mode

Definition at line 334 of file vtss_phy_10g_api.h.

3.65.3.2 interface

```
vtss_phy_interface_mode vtss_phy_10g_mode_t::interface
```

Interface mode.

Definition at line 336 of file vtss_phy_10g_api.h.

3.65.3.3 wrefclk

```
vtss_wrefclk_t vtss_phy_10g_mode_t::wrefclk
```

848X only: WAN ref clock

Definition at line 338 of file vtss_phy_10g_api.h.

3.65.3.4 high_input_gain

```
BOOL vtss_phy_10g_mode_t::high_input_gain
```

Disable=0 (default), Enable=1. Should not be enabled unless needed

Definition at line 340 of file vtss_phy_10g_api.h.

3.65.3.5 xfi_pol_invert

```
BOOL vtss_phy_10g_mode_t::xfi_pol_invert
```

Selects polarity ot the TX XFI data. 1:Invert 0:Normal

Definition at line 341 of file vtss_phy_10g_api.h.

3.65.3.6 xaui_lane_flip

```
BOOL vtss_phy_10g_mode_t::xaui_lane_flip
```

Swaps lane 0 <–> 3 and 1 <–> 2 for both RX and TX

Definition at line 342 of file vtss_phy_10g_api.h.

3.65.3.7 channel_id

```
vtss_channel_t vtss_phy_10g_mode_t::channel_id
```

Channel id of this instance of the Phy

Definition at line 343 of file vtss_phy_10g_api.h.

3.65.3.8 hl_clk_synth

```
BOOL vtss_phy_10g_mode_t::hl_clk_synth
```

0: Free running clock 1: Hitless clock

Definition at line 346 of file vtss_phy_10g_api.h.

3.65.3.9 rcvrd_clk

```
vtss_recvrd_t vtss_phy_10g_mode_t::rcvrd_clk
```

RXCLKOUT/TXCLKOUT used as recovered clock (not used any more, instead use the api functions: vtss_phy_↩ 10g_rxckout_set and vtss_phy_10g_txckout_set

Definition at line 347 of file vtss_phy_10g_api.h.

3.65.3.10 rcvrd_clk_div

`vtss_recvrdclk_cdr_div_t vtss_phy_10g_mode_t::rcvrd_clk_div`

8488 only: recovered clock's divisor

Definition at line 350 of file vtss_phy_10g_api.h.

3.65.3.11 sref_clk_div

`vtss_srefclk_div_t vtss_phy_10g_mode_t::sref_clk_div`

8488 only: SRERCLK divisor

Definition at line 351 of file vtss_phy_10g_api.h.

3.65.3.12 wref_clk_div

`vtss_wref_clk_div_t vtss_phy_10g_mode_t::wref_clk_div`

8488 only: WREFCLK divisor

Definition at line 352 of file vtss_phy_10g_api.h.

3.65.3.13 edc_fw_load

`enum { ... } vtss_phy_10g_mode_t::edc_fw_load`

EDC modes.

EDC Firmware load

3.65.3.14 use_conf

`BOOL vtss_phy_10g_mode_t::use_conf`

Use this configuration instead of default(only for setting 'd_filter'in Venice)

Definition at line 365 of file vtss_phy_10g_api.h.

3.65.3.15 ob_conf

```
BOOL vtss_phy_10g_mode_t::ob_conf
```

Configuration for SD10F OB instead of default (only for Venice family)

Definition at line 366 of file vtss_phy_10g_api.h.

3.65.3.16 ib_conf

```
BOOL vtss_phy_10g_mode_t::ib_conf
```

Configuration for SD6G ib_ini_lp, ib_min_lp & ib_max_lp (only for Venice family)

Definition at line 367 of file vtss_phy_10g_api.h.

3.65.3.17 dig_offset_reg

```
BOOL vtss_phy_10g_mode_t::dig_offset_reg
```

Digital offset regulation for SD6G IB. Default is Analog(only for Venice family)

Definition at line 368 of file vtss_phy_10g_api.h.

3.65.3.18 apc_offs_ctrl

```
BOOL vtss_phy_10g_mode_t::apc_offs_ctrl
```

Parameter used to control APC offset(overwrite APC_EQZ_OFFS_PAR_CFG default value with apc_eqz_offs_↩ par_cfg

Definition at line 369 of file vtss_phy_10g_api.h.

3.65.3.19 apc_line_ld_ctrl

```
BOOL vtss_phy_10g_mode_t::apc_line_ld_ctrl
```

Parameter used to control APC Line LD Ctrl (overwrite LD_LEV_INI, line apc value with apc_line_eqz_ld_ctrl

Definition at line 370 of file vtss_phy_10g_api.h.

3.65.3.20 apc_host_ld_ctrl

```
BOOL vtss_phy_10g_mode_t::apc_host_ld_ctrl
```

Parameter used to control APC Host LD Ctrl (overwrite LD_LEV_INI, host apc value with apc_line_eqz_ld_ctrl

Definition at line 371 of file vtss_phy_10g_api.h.

3.65.3.21 d_filter

```
u32 vtss_phy_10g_mode_t::d_filter
```

SD10G Transmit filter coefficients for FIR taps (default 0x7DF820)

Definition at line 372 of file vtss_phy_10g_api.h.

3.65.3.22 cfg0

```
u32 vtss_phy_10g_mode_t::cfg0
```

SD10G OB CFG0 value, configurable by USER (only for Venice family)

Definition at line 373 of file vtss_phy_10g_api.h.

3.65.3.23 ib_ini_lp

```
u32 vtss_phy_10g_mode_t::ib_ini_lp
```

SD6G Init force value for low-pass gain regulation (default 1)

Definition at line 374 of file vtss_phy_10g_api.h.

3.65.3.24 ib_min_lp

```
u32 vtss_phy_10g_mode_t::ib_min_lp
```

SD6G Min value for low-pass gain regulation (default 0)

Definition at line 375 of file vtss_phy_10g_api.h.

3.65.3.25 ib_max_lp

u32 vtss_phy_10g_mode_t::ib_max_lp

SD6G Max value for low-pass gain regulation (default 63)

Definition at line 376 of file vtss_phy_10g_api.h.

3.65.3.26 apc_eqz_offs_par_cfg

u32 vtss_phy_10g_mode_t::apc_eqz_offs_par_cfg

APC EQZ_OFFS Parameter control(value of register APC_EQZ_OFFS_PAR_CFG,updated when apc_offs_ctrl is set

Definition at line 377 of file vtss_phy_10g_api.h.

3.65.3.27 apc_line_eqz_ld_ctrl

u32 vtss_phy_10g_mode_t::apc_line_eqz_ld_ctrl

APC EQZ Line LD control(value of LD_LEV_INI, line apc value. Updated when apc_line_ld_ctrl is set) range = 0x18-0x2C default values (SR mode): Venice-A/B 0x18; Malibu-A, Venice-C 0x2C; Malibu-B 0x18

Definition at line 378 of file vtss_phy_10g_api.h.

3.65.3.28 apc_host_eqz_ld_ctrl

u32 vtss_phy_10g_mode_t::apc_host_eqz_ld_ctrl

APC EQZ Host LD control(value of LD_LEV_INI, host apc value. Updated when apc_line_ld_ctrl is set) range = 0x18-0x2C default values (SR mode): Venice-A/B 0x18; Malibu-A, Venice-C 0x2C; Malibu-B 0x18

Definition at line 380 of file vtss_phy_10g_api.h.

3.65.3.29 l_offset_guard

BOOL vtss_phy_10g_mode_t::l_offset_guard

This variable is deprecated, not to be used

Definition at line 382 of file vtss_phy_10g_api.h.

3.65.3.30 h_offset_guard

```
BOOL vtss_phy_10g_mode_t::h_offset_guard
```

This variable is deprecated, not to be used

Definition at line 383 of file vtss_phy_10g_api.h.

3.65.3.31 serdes_conf

```
struct { ... } vtss_phy_10g_mode_t::serdes_conf
```

Serdes parameters.

Serdes configuration

3.65.3.32 apc_ib_regulator

```
apc_ib_regulator_t vtss_phy_10g_mode_t::apc_ib_regulator
```

Analog Parameter Control / IB equalizer (only for Venice family)

Definition at line 386 of file vtss_phy_10g_api.h.

3.65.3.33 pma_txratecontrol

```
u16 vtss_phy_10g_mode_t::pma_txratecontrol
```

Normal pma_txratecontrol value to be restored when loopback is disabled

Definition at line 387 of file vtss_phy_10g_api.h.

3.65.3.34 venice_rev_a_los_detection_workaround

```
BOOL vtss_phy_10g_mode_t::venice_rev_a_los_detection_workaround
```

TRUE => LOS detection woak around enabled. Requires interrupt handling

Definition at line 388 of file vtss_phy_10g_api.h.

3.65.3.35 ddr_mode

ddr_mode_t vtss_phy_10g_mode_t::ddr_mode

DDR Interleave mode

Definition at line 389 of file vtss_phy_10g_api.h.

3.65.3.36 master

clk_mstr_t vtss_phy_10g_mode_t::master

Clock Master

Definition at line 390 of file vtss_phy_10g_api.h.

3.65.3.37 rate

vtss_rptr_rate_t vtss_phy_10g_mode_t::rate

Data rate in repeater mode

Definition at line 391 of file vtss_phy_10g_api.h.

3.65.3.38 polarity

vtss_phy_10g_polarity_inv_t vtss_phy_10g_mode_t::polarity

polarity inversion configuration

Definition at line 392 of file vtss_phy_10g_api.h.

3.65.3.39 is_host_wan

BOOL vtss_phy_10g_mode_t::is_host_wan

Flag that gives information of WAN rate is supported at host interface

Definition at line 393 of file vtss_phy_10g_api.h.

3.65.3.40 h_clk_src

```
vtss_phy_10g_clk_src_t vtss_phy_10g_mode_t::h_clk_src
```

Host side clock configuration

Definition at line 394 of file vtss_phy_10g_api.h.

3.65.3.41 l_clk_src

```
vtss_phy_10g_clk_src_t vtss_phy_10g_mode_t::l_clk_src
```

Line side clock configuration

Definition at line 395 of file vtss_phy_10g_api.h.

3.65.3.42 lref_for_host

```
BOOL vtss_phy_10g_mode_t::lref_for_host
```

Clock source selection HREF or LREF on HOST side

Definition at line 396 of file vtss_phy_10g_api.h.

3.65.3.43 link_6g_distance

```
vtss_phy_6g_link_partner_distance_t vtss_phy_10g_mode_t::link_6g_distance
```

Gives information of link partner distance from 6G macro

Definition at line 397 of file vtss_phy_10g_api.h.

3.65.3.44 h_media

```
vtss_phy_10g_media_t vtss_phy_10g_mode_t::h_media
```

Gives information of media type connected on HOST direction

Definition at line 398 of file vtss_phy_10g_api.h.

3.65.3.45 l_media

vtss_phy_10g_media_t vtss_phy_10g_mode_t::l_media

Gives information of media type connected on LINE direction For Venice rev C and Malibu rev B, it is recommended to use smart control for the media type settings regardless what the actual media type application is used.

Definition at line 399 of file vtss_phy_10g_api.h.

3.65.3.46 h_ib_conf

vtss_phy_10g_ib_conf_t vtss_phy_10g_mode_t::h_ib_conf

Host Input buffer configuration

Definition at line 403 of file vtss_phy_10g_api.h.

3.65.3.47 l_ib_conf

vtss_phy_10g_ib_conf_t vtss_phy_10g_mode_t::l_ib_conf

Line Input buffer configuration

Definition at line 404 of file vtss_phy_10g_api.h.

3.65.3.48 h_apc_conf

vtss_phy_10g_apc_conf_t vtss_phy_10g_mode_t::h_apc_conf

HOST APC configuration

Definition at line 405 of file vtss_phy_10g_api.h.

3.65.3.49 l_apc_conf

vtss_phy_10g_apc_conf_t vtss_phy_10g_mode_t::l_apc_conf

LINE APC configuration

Definition at line 406 of file vtss_phy_10g_api.h.

3.65.3.50 enable_pass_thru

```
BOOL vtss_phy_10g_mode_t::enable_pass_thru
```

Enables Pass through mode in VENICE

Definition at line 407 of file vtss_phy_10g_api.h.

3.65.3.51 is_init

```
BOOL vtss_phy_10g_mode_t::is_init
```

To identify intialization Phase

Definition at line 408 of file vtss_phy_10g_api.h.

3.65.3.52 sd6g_calib_done

```
BOOL vtss_phy_10g_mode_t::sd6g_calib_done
```

to identify initialization Phase for ib calibration

Definition at line 409 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.66 vtss_phy_10g_ob_status_t Struct Reference

10G Phy OB status

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- u8 r_ctrl
- u8 c_ctrl
- u8 slew
- u8 levn
- u32 d_fltr
- int v3
- int vp
- int v4
- int v5
- BOOL is_host

3.66.1 Detailed Description

10G Phy OB status

Definition at line 1171 of file vtss_phy_10g_api.h.

3.66.2 Field Documentation

3.66.2.1 r_ctrl

```
u8 vtss_phy_10g_ob_status_t::r_ctrl
```

slew rate r active value

Definition at line 1172 of file vtss_phy_10g_api.h.

3.66.2.2 c_ctrl

```
u8 vtss_phy_10g_ob_status_t::c_ctrl
```

slew rate c active value

Definition at line 1173 of file vtss_phy_10g_api.h.

3.66.2.3 slew

```
u8 vtss_phy_10g_ob_status_t::slew
```

slew rate

Definition at line 1174 of file vtss_phy_10g_api.h.

3.66.2.4 levn

```
u8 vtss_phy_10g_ob_status_t::levn
```

amplitude

Definition at line 1175 of file vtss_phy_10g_api.h.

3.66.2.5 d_fltr

```
u32 vtss_phy_10g_ob_status_t::d_fltr
```

d-filter value

Definition at line 1176 of file vtss_phy_10g_api.h.

3.66.2.6 v3

```
int vtss_phy_10g_ob_status_t::v3
```

d_filter tap v3

Definition at line 1177 of file vtss_phy_10g_api.h.

3.66.2.7 vp

```
int vtss_phy_10g_ob_status_t::vp
```

d_filter tap vp

Definition at line 1178 of file vtss_phy_10g_api.h.

3.66.2.8 v4

```
int vtss_phy_10g_ob_status_t::v4
```

d_filter tap v4

Definition at line 1179 of file vtss_phy_10g_api.h.

3.66.2.9 v5

```
int vtss_phy_10g_ob_status_t::v5
```

d_filter tap v5

Definition at line 1180 of file vtss_phy_10g_api.h.

3.66.2.10 is_host

```
BOOL vtss_phy_10g_ob_status_t::is_host
```

flag that says host/line

Definition at line 1181 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.67 vtss_phy_10g_pcs_prbs_gen_conf_t Struct Reference

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL prbs_gen

3.67.1 Detailed Description

\ brief 10G PHY pcs prbs generator configuration

Definition at line 1941 of file vtss_phy_10g_api.h.

3.67.2 Field Documentation

3.67.2.1 prbs_gen

```
BOOL vtss_phy_10g_pcs_prbs_gen_conf_t::prbs_gen
```

enable or disable prbs test pattern mode on transmit path

Definition at line 1942 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.68 vtss_phy_10g_pcs_prbs_mon_conf_t Struct Reference

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL prbs_mon
- u32 error_counter

3.68.1 Detailed Description

\ brief 10G PHY pcs prbs analyzer configuration

Definition at line 1976 of file vtss_phy_10g_api.h.

3.68.2 Field Documentation

3.68.2.1 prbs_mon

```
BOOL vtss_phy_10g_pcs_prbs_mon_conf_t::prbs_mon
```

enable or disable prbs test pattern mode on receive path

Definition at line 1977 of file vtss_phy_10g_api.h.

3.68.2.2 error_counter

```
u32 vtss_phy_10g_pcs_prbs_mon_conf_t::error_counter
```

Error counters for pcs prbs

Definition at line 1978 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.69 vtss_phy_10g_pkt_gen_conf_t Struct Reference

10G PHY Packet generator configuration

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL enable
- BOOL ptp
- BOOL ingress
- BOOL frames
- BOOL frame_single
- u16 etype
- u8 pkt_len
- u32 ipg_len
- vtss_mac_addr_t smac
- vtss_mac_addr_t dmac
- u8 ptp_ts_sec
- u8 ptp_ts_ns
- u8 srate

3.69.1 Detailed Description

10G PHY Packet generator configuration

Definition at line 2133 of file vtss_phy_10g_api.h.

3.69.2 Field Documentation

3.69.2.1 enable

```
BOOL vtss_phy_10g_pkt_gen_conf_t::enable
```

Enable or disable packet generator

Definition at line 2134 of file vtss_phy_10g_api.h.

3.69.2.2 ptp

```
BOOL vtss_phy_10g_pkt_gen_conf_t::ptp
```

PTP or standard frame

Definition at line 2135 of file vtss_phy_10g_api.h.

3.69.2.3 ingress

```
BOOL vtss_phy_10g_pkt_gen_conf_t::ingress
```

Ingress or egress

Definition at line 2136 of file vtss_phy_10g_api.h.

3.69.2.4 frames

```
BOOL vtss_phy_10g_pkt_gen_conf_t::frames
```

frames or idles

Definition at line 2137 of file vtss_phy_10g_api.h.

3.69.2.5 frame_single

```
BOOL vtss_phy_10g_pkt_gen_conf_t::frame_single
```

Generate single packet

Definition at line 2138 of file vtss_phy_10g_api.h.

3.69.2.6 etype

```
u16 vtss_phy_10g_pkt_gen_conf_t::etype
```

Ethertype

Definition at line 2139 of file vtss_phy_10g_api.h.

3.69.2.7 pkt_len

```
u8 vtss_phy_10g_pkt_gen_conf_t::pkt_len
```

Packet length,min=64,max=16KB

Definition at line 2140 of file vtss_phy_10g_api.h.

3.69.2.8 ipg_len

```
u32 vtss_phy_10g_pkt_gen_conf_t::ipg_len
```

Inter Packet Gap

Definition at line 2141 of file vtss_phy_10g_api.h.

3.69.2.9 smac

```
vtss_mac_addr_t vtss_phy_10g_pkt_gen_conf_t::smac
```

Source MAC address

Definition at line 2142 of file vtss_phy_10g_api.h.

3.69.2.10 dmac

```
vtss_mac_addr_t vtss_phy_10g_pkt_gen_conf_t::dmac
```

Destination MAC address

Definition at line 2143 of file vtss_phy_10g_api.h.

3.69.2.11 ptp_ts_sec

```
u8 vtss_phy_10g_pkt_gen_conf_t::ptp_ts_sec
```

Seconds part of timestamp value

Definition at line 2144 of file vtss_phy_10g_api.h.

3.69.2.12 ptp_ts_ns

```
u8 vtss_phy_10g_pkt_gen_conf_t::ptp_ts_ns
```

NanoSeconds part of ts value

Definition at line 2145 of file vtss_phy_10g_api.h.

3.69.2.13 srate

```
u8 vtss_phy_10g_pkt_gen_conf_t::srate
```

Srate for ptp frames

Definition at line 2146 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.70 vtss_phy_10g_pkt_mon_conf_t Struct Reference

10G PHY Packet Monitor configuration

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL enable
- BOOL update
- vtss_phy_10g_pkt_mon_rst_t reset
- vtss_32_cntr_t good_crc
- vtss_32_cntr_t bad_crc
- vtss_32_cntr_t frag
- vtss_32_cntr_t lfault
- vtss_32_cntr_t ber

3.70.1 Detailed Description

10G PHY Packet Monitor configuration

Definition at line 2178 of file vtss_phy_10g_api.h.

3.70.2 Field Documentation

3.70.2.1 enable

```
BOOL vtss_phy_10g_pkt_mon_conf_t::enable
```

Enable or disable packet monitor

Definition at line 2179 of file vtss_phy_10g_api.h.

3.70.2.2 update

```
BOOL vtss_phy_10g_pkt_mon_conf_t::update
```

update and reads monitor counters

Definition at line 2180 of file vtss_phy_10g_api.h.

3.70.2.3 reset

```
vtss_phy_10g_pkt_mon_rst_t vtss_phy_10g_pkt_mon_conf_t::reset
```

resets all monitor counters

Definition at line 2181 of file vtss_phy_10g_api.h.

3.70.2.4 good_crc

```
vtss_32_cntr_t vtss_phy_10g_pkt_mon_conf_t::good_crc
```

Good CRC packet count

Definition at line 2182 of file vtss_phy_10g_api.h.

3.70.2.5 bad_crc

```
vtss_32_cntr_t vtss_phy_10g_pkt_mon_conf_t::bad_crc
```

Bad CRC packet count

Definition at line 2183 of file vtss_phy_10g_api.h.

3.70.2.6 frag

```
vtss_32_cntr_t vtss_phy_10g_pkt_mon_conf_t::frag
```

Fragmented packet count

Definition at line 2184 of file vtss_phy_10g_api.h.

3.70.2.7 lfault

```
vtss_32_cntr_t vtss_phy_10g_pkt_mon_conf_t::lfault
```

Local fault packet count

Definition at line 2185 of file vtss_phy_10g_api.h.

3.70.2.8 ber

```
vtss_32_cntr_t vtss_phy_10g_pkt_mon_conf_t::ber
```

B-errored packet count

Definition at line 2186 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.71 vtss_phy_10g_polarity_inv_t Struct Reference

10G Phy Polarity inversion

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL line_rx
- BOOL line_tx
- BOOL host_rx
- BOOL host_tx

3.71.1 Detailed Description

10G Phy Polarity inversion

Definition at line 162 of file vtss_phy_10g_api.h.

3.71.2 Field Documentation

3.71.2.1 line_rx

```
BOOL vtss_phy_10g_polarity_inv_t::line_rx
```

Line side Receive path

Definition at line 163 of file vtss_phy_10g_api.h.

3.71.2.2 line_tx

```
BOOL vtss_phy_10g_polarity_inv_t::line_tx
```

Line side Transmit path

Definition at line 164 of file vtss_phy_10g_api.h.

3.71.2.3 host_rx

```
BOOL vtss_phy_10g_polarity_inv_t::host_rx
```

Host side Receive path

Definition at line 165 of file vtss_phy_10g_api.h.

3.71.2.4 host_tx

```
BOOL vtss_phy_10g_polarity_inv_t::host_tx
```

Host side Transmit path

Definition at line 166 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.72 vtss_phy_10g_prbs_gen_conf_t Struct Reference

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL enable
- u8 prbsn_tx_sel
- BOOL line
- BOOL prbsn_tx_io
- u8 prbsn_tx_iw

3.72.1 Detailed Description

\ brief 10G PHY prbs generator configuration

Definition at line 2026 of file vtss_phy_10g_api.h.

3.72.2 Field Documentation

3.72.2.1 enable

```
BOOL vtss_phy_10g_prbs_gen_conf_t::enable
```

enable or disable prbs generator

Definition at line 2027 of file vtss_phy_10g_api.h.

3.72.2.2 prbsn_tx_sel

```
u8 vtss_phy_10g_prbs_gen_conf_t::prbsn_tx_sel
```

select the prbs to be implemented, min=0, max=5

Definition at line 2028 of file vtss_phy_10g_api.h.

3.72.2.3 line

```
BOOL vtss_phy_10g_prbs_gen_conf_t::line
```

select the line side or host side, 1 for line side

Definition at line 2029 of file vtss_phy_10g_api.h.

3.72.2.4 prbsn_tx_io

```
BOOL vtss_phy_10g_prbs_gen_conf_t::prbsn_tx_io
```

Invert PRBS TX pattern

Definition at line 2030 of file vtss_phy_10g_api.h.

3.72.2.5 prbsn_tx_iw

```
u8 vtss_phy_10g_prbs_gen_conf_t::prbsn_tx_iw
```

select the prbs interface widtdh ,range 0-5

Definition at line 2031 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.73 vtss_phy_10g_prbs_mon_conf_t Struct Reference

10G PHY prbs monitor Configuration

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL enable
- BOOL line
- u16 max_bist_frames
- u16 error_states
- u16 des_interface_width
- u16 prbsn_sel
- BOOL prbs_check_input_invert
- u16 no_of_errors
- u16 bist_mode
- u32 error_status
- u32 PRBS_status
- u32 main_status
- BOOL stuck_at_par
- BOOL stuck_at_01
- BOOL no_sync
- BOOL instable
- BOOL incomplete
- BOOL active

3.73.1 Detailed Description

10G PHY prbs monitor Configuration

Definition at line 2065 of file vtss_phy_10g_api.h.

3.73.2 Field Documentation

3.73.2.1 enable

`BOOL vtss_phy_10g_prbs_mon_conf_t::enable`

enable or disable the prbs monitor

Definition at line 2066 of file vtss_phy_10g_api.h.

3.73.2.2 line

`BOOL vtss_phy_10g_prbs_mon_conf_t::line`

select line side or host side, 1 for line side

Definition at line 2067 of file vtss_phy_10g_api.h.

3.73.2.3 max_bist_frames

`u16 vtss_phy_10g_prbs_mon_conf_t::max_bist_frames`

threshold to iterate counter for max_bist_frames [15:0]

Definition at line 2068 of file vtss_phy_10g_api.h.

3.73.2.4 error_states

`u16 vtss_phy_10g_prbs_mon_conf_t::error_states`

States in which error counting is enabled3:all but IDLE; 2:check 1:stable+check,0:wait_stable+stable+check

Definition at line 2069 of file vtss_phy_10g_api.h.

3.73.2.5 des_interface_width

```
u16 vtss_phy_10g_prbs_mon_conf_t::des_interface_width
```

DES interface width 0:8,1:10,2:16,3:20,4:32,5:40 (default)

Definition at line 2070 of file vtss_phy_10g_api.h.

3.73.2.6 prbsn_sel

```
u16 vtss_phy_10g_prbs_mon_conf_t::prbsn_sel
```

select the prbs to be implemented, min=0, max=5>

Definition at line 2071 of file vtss_phy_10g_api.h.

3.73.2.7 prbs_check_input_invert

```
BOOL vtss_phy_10g_prbs_mon_conf_t::prbs_check_input_invert
```

Enables PRBS checker input inversion

Definition at line 2072 of file vtss_phy_10g_api.h.

3.73.2.8 no_of_errors

```
u16 vtss_phy_10g_prbs_mon_conf_t::no_of_errors
```

Number of consecutive errors/non-errors before transitioning to respective state , value = num-40-bits-words + 1

Definition at line 2073 of file vtss_phy_10g_api.h.

3.73.2.9 bist_mode

```
u16 vtss_phy_10g_prbs_mon_conf_t::bist_mode
```

0: off, 1: BIST, 2: BER, 3:CONT(infinite mode)

Definition at line 2074 of file vtss_phy_10g_api.h.

3.73.2.10 error_status

```
u32 vtss_phy_10g_prbs_mon_conf_t::error_status
```

Error stautus of PRBS

Definition at line 2075 of file vtss_phy_10g_api.h.

3.73.2.11 PRBS_status

```
u32 vtss_phy_10g_prbs_mon_conf_t::PRBS_status
```

PRBS status

Definition at line 2076 of file vtss_phy_10g_api.h.

3.73.2.12 main_status

```
u32 vtss_phy_10g_prbs_mon_conf_t::main_status
```

Main stauts

Definition at line 2077 of file vtss_phy_10g_api.h.

3.73.2.13 stuck_at_par

```
BOOL vtss_phy_10g_prbs_mon_conf_t::stuck_at_par
```

Data input is unchanged for all 40 parallel bits for at least 7 clock cycles

Definition at line 2078 of file vtss_phy_10g_api.h.

3.73.2.14 stuck_at_01

```
BOOL vtss_phy_10g_prbs_mon_conf_t::stuck_at_01
```

Data input is unchanged for all 40 parallel bits for at least 7 clock cycles

Definition at line 2079 of file vtss_phy_10g_api.h.

3.73.2.15 no_sync

```
BOOL vtss_phy_10g_prbs_mon_conf_t::no_sync
```

no sync found since BIST enabled

Definition at line 2080 of file vtss_phy_10g_api.h.

3.73.2.16 instable

```
BOOL vtss_phy_10g_prbs_mon_conf_t::instable
```

BIST input data not stable

Definition at line 2081 of file vtss_phy_10g_api.h.

3.73.2.17 incomplete

```
BOOL vtss_phy_10g_prbs_mon_conf_t::incomplete
```

BIST not complete i.e. it has not reached a stable state

Definition at line 2082 of file vtss_phy_10g_api.h.

3.73.2.18 active

```
BOOL vtss_phy_10g_prbs_mon_conf_t::active
```

BIST is active but has not entered a final state

Definition at line 2083 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.74 vtss_phy_10g_rxckout_conf_t Struct Reference

10G Phy RXCKOUT config data

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- vtss_recvrd_clkout_t mode
- BOOL squelch_on_pcs_fault
- BOOL squelch_on_lopc

3.74.1 Detailed Description

10G Phy RXCKOUT config data

Definition at line 706 of file vtss_phy_10g_api.h.

3.74.2 Field Documentation

3.74.2.1 mode

```
vtss_recvrd_clkout_t vtss_phy_10g_rxckout_conf_t::mode
```

RXCKOUT output mode (DISABLE/RX_CLK/TX_CLK)

Definition at line 707 of file vtss_phy_10g_api.h.

3.74.2.2 squelch_on_pcs_fault

```
BOOL vtss_phy_10g_rxckout_conf_t::squelch_on_pcs_fault
```

Enable squelching on PCS_FAULT (supported on revision no = 1 (Rev C) and above)

Definition at line 708 of file vtss_phy_10g_api.h.

3.74.2.3 squelch_on_lopc

```
BOOL vtss_phy_10g_rxckout_conf_t::squelch_on_lopc
```

Enable squelching on LOPC (supported on revision no = 1 (Rev C) and above)

Definition at line 709 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.75 vtss_phy_10g_sckout_conf_t Struct Reference

10G Phy SCKOUT config data

`#include <vtss_phy_10g_api.h>`

Data Fields

- vtss_phy_10g_clk_sel_t mode
- vtss_phy_10g_squelch_src_t src
- vtss_phy_10g_sckout_freq_t freq
- BOOL squelch_inv
- BOOL enable

3.75.1 Detailed Description

10G Phy SCKOUT config data

Malibu Only

Definition at line 982 of file vtss_phy_10g_api.h.

3.75.2 Field Documentation

3.75.2.1 mode

`vtss_phy_10g_clk_sel_t vtss_phy_10g_sckout_conf_t::mode`

SCKOUT output clock mode

Definition at line 983 of file vtss_phy_10g_api.h.

3.75.2.2 src

`vtss_phy_10g_squelch_src_t vtss_phy_10g_sckout_conf_t::src`

SCKOUT squelch source

Definition at line 984 of file vtss_phy_10g_api.h.

3.75.2.3 freq

`vtss_phy_10g_sckout_freq_t vtss_phy_10g_sckout_conf_t::freq`

SCKOUT freq(156.25MHz, 125MHz only)

Definition at line 985 of file vtss_phy_10g_api.h.

3.75.2.4 squelch_inv

`BOOL vtss_phy_10g_sckout_conf_t::squelch_inv`

'0'- Use squelch source src as is, '1'-Invert

Definition at line 986 of file vtss_phy_10g_api.h.

3.75.2.5 enable

`BOOL vtss_phy_10g_sckout_conf_t::enable`

Enable/Disable SCKOUT

Definition at line 987 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.76 vtss_phy_10g_serdes_status_t Struct Reference

10G Phy SERDES status

`#include <vtss_phy_10g_api.h>`

Data Fields

- u8 rcomp
- BOOL h_pll5g_lock_status
- BOOL h_pll5g_fsm_lock
- u8 h_pll5g_fsm_stat
- u8 h_pll5g_gain
- BOOL l_pll5g_lock_status
- BOOL l_pll5g_fsm_lock
- u8 l_pll5g_fsm_stat
- u8 l_pll5g_gain
- BOOL h_rx_rcpll_lock_status
- u8 h_rx_rcpll_range
- u8 h_rx_rcpll_vco_load
- u8 h_rx_rcpll_fsm_status
- BOOL l_rx_rcpll_lock_status
- u8 l_rx_rcpll_range
- u8 l_rx_rcpll_vco_load
- u8 l_rx_rcpll_fsm_status
- BOOL h_tx_rcpll_lock_status
- u8 h_tx_rcpll_range
- u8 h_tx_rcpll_vco_load
- u8 h_tx_rcpll_fsm_status
- BOOL l_tx_rcpll_lock_status
- u8 l_tx_rcpll_range
- u8 l_tx_rcpll_vco_load
- u8 l_tx_rcpll_fsm_status
- vtss_sublayer_status_t h_pma
- vtss_sublayer_status_t h_pcs
- vtss_sublayer_status_t l_pma
- vtss_sublayer_status_t l_pcs
- vtss_sublayer_status_t wis

3.76.1 Detailed Description

10G Phy SERDES status

Definition at line 253 of file vtss_phy_10g_api.h.

3.76.2 Field Documentation

3.76.2.1 rcomp

```
u8 vtss_phy_10g_serdes_status_t::rcomp
```

Measured Resistor value

Definition at line 254 of file vtss_phy_10g_api.h.

3.76.2.2 h_pll5g_lock_status

```
BOOL vtss_phy_10g_serdes_status_t::h_pll5g_lock_status
```

TRUE value says its locked

Definition at line 257 of file vtss_phy_10g_api.h.

3.76.2.3 h_pll5g_fsm_lock

```
BOOL vtss_phy_10g_serdes_status_t::h_pll5g_fsm_lock
```

TRUE value says fsm is locked

Definition at line 258 of file vtss_phy_10g_api.h.

3.76.2.4 h_pll5g_fsm_stat

```
u8 vtss_phy_10g_serdes_status_t::h_pll5g_fsm_stat
```

FSM status

Definition at line 259 of file vtss_phy_10g_api.h.

3.76.2.5 h_pll5g_gain

```
u8 vtss_phy_10g_serdes_status_t::h_pll5g_gain
```

Gain

Definition at line 260 of file vtss_phy_10g_api.h.

3.76.2.6 l_pll5g_lock_status

```
BOOL vtss_phy_10g_serdes_status_t::l_pll5g_lock_status
```

TRUE value says its locked

Definition at line 263 of file vtss_phy_10g_api.h.

3.76.2.7 l_pll5g_fsm_lock

```
BOOL vtss_phy_10g_serdes_status_t::l_pll5g_fsm_lock
```

TRUE value says fsm is locked

Definition at line 264 of file vtss_phy_10g_api.h.

3.76.2.8 l_pll5g_fsm_stat

```
u8 vtss_phy_10g_serdes_status_t::l_pll5g_fsm_stat
```

FSM status

Definition at line 265 of file vtss_phy_10g_api.h.

3.76.2.9 l_pll5g_gain

```
u8 vtss_phy_10g_serdes_status_t::l_pll5g_gain
```

Gain

Definition at line 266 of file vtss_phy_10g_api.h.

3.76.2.10 h_rx_rcpll_lock_status

```
BOOL vtss_phy_10g_serdes_status_t::h_rx_rcpll_lock_status
```

TRUE value says its locked

Definition at line 269 of file vtss_phy_10g_api.h.

3.76.2.11 h_rx_rcpll_range

```
u8 vtss_phy_10g_serdes_status_t::h_rx_rcpll_range
```

TRUE value says with in range

Definition at line 270 of file vtss_phy_10g_api.h.

3.76.2.12 h_rx_rcpll_vco_load

```
u8 vtss_phy_10g_serdes_status_t::h_rx_rcpll_vco_load
```

Actual value of VCV load

Definition at line 271 of file vtss_phy_10g_api.h.

3.76.2.13 h_rx_rcpll_fsm_status

```
u8 vtss_phy_10g_serdes_status_t::h_rx_rcpll_fsm_status
```

Actual value of FSM stage

Definition at line 272 of file vtss_phy_10g_api.h.

3.76.2.14 l_rx_rcpll_lock_status

```
BOOL vtss_phy_10g_serdes_status_t::l_rx_rcpll_lock_status
```

TRUE value says its locked

Definition at line 273 of file vtss_phy_10g_api.h.

3.76.2.15 l_rx_rcpll_range

```
u8 vtss_phy_10g_serdes_status_t::l_rx_rcpll_range
```

TRUE value says with in range

Definition at line 274 of file vtss_phy_10g_api.h.

3.76.2.16 l_rx_rcpll_vco_load

```
u8 vtss_phy_10g_serdes_status_t::l_rx_rcpll_vco_load
```

Actual value of VCV load

Definition at line 275 of file vtss_phy_10g_api.h.

3.76.2.17 l_rx_rcpll_fsm_status

```
u8 vtss_phy_10g_serdes_status_t::l_rx_rcpll_fsm_status
```

Actual value of FSM stage

Definition at line 276 of file vtss_phy_10g_api.h.

3.76.2.18 h_tx_rcpll_lock_status

```
BOOL vtss_phy_10g_serdes_status_t::h_tx_rcpll_lock_status
```

TRUE value says its locked

Definition at line 279 of file vtss_phy_10g_api.h.

3.76.2.19 h_tx_rcpll_range

```
u8 vtss_phy_10g_serdes_status_t::h_tx_rcpll_range
```

TRUE value says with in range

Definition at line 280 of file vtss_phy_10g_api.h.

3.76.2.20 h_tx_rcpll_vco_load

```
u8 vtss_phy_10g_serdes_status_t::h_tx_rcpll_vco_load
```

Actual value of VCV load

Definition at line 281 of file vtss_phy_10g_api.h.

3.76.2.21 h_tx_rcpll_fsm_status

```
u8 vtss_phy_10g_serdes_status_t::h_tx_rcpll_fsm_status
```

Actual value of FSM stage

Definition at line 282 of file vtss_phy_10g_api.h.

3.76.2.22 l_tx_rcpll_lock_status

```
BOOL vtss_phy_10g_serdes_status_t::l_tx_rcpll_lock_status
```

TRUE value says its locked

Definition at line 283 of file vtss_phy_10g_api.h.

3.76.2.23 l_tx_rcpll_range

```
u8 vtss_phy_10g_serdes_status_t::l_tx_rcpll_range
```

TRUE value says with in range

Definition at line 284 of file vtss_phy_10g_api.h.

3.76.2.24 l_tx_rcpll_vco_load

```
u8 vtss_phy_10g_serdes_status_t::l_tx_rcpll_vco_load
```

Actual value of VCV load

Definition at line 285 of file vtss_phy_10g_api.h.

3.76.2.25 l_tx_rcpll_fsm_status

```
u8 vtss_phy_10g_serdes_status_t::l_tx_rcpll_fsm_status
```

Actual value of FSM stage

Definition at line 286 of file vtss_phy_10g_api.h.

3.76.2.26 h_pma

```
vtss_sublayer_status_t vtss_phy_10g_serdes_status_t::h_pma
```

Host pma status

Definition at line 289 of file vtss_phy_10g_api.h.

3.76.2.27 h_pcs

```
vtss_sublayer_status_t vtss_phy_10g_serdes_status_t::h_pcs
```

Host pcs status

Definition at line 290 of file vtss_phy_10g_api.h.

3.76.2.28 l_pma

```
vtss_sublayer_status_t vtss_phy_10g_serdes_status_t::l_pma
```

Line pma status

Definition at line 293 of file vtss_phy_10g_api.h.

3.76.2.29 l_pcs

```
vtss_sublayer_status_t vtss_phy_10g_serdes_status_t::l_pcs
```

Line pcs status

Definition at line 294 of file vtss_phy_10g_api.h.

3.76.2.30 wis

```
vtss_sublayer_status_t vtss_phy_10g_serdes_status_t::wis
```

WIS status

Definition at line 297 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.77 vtss_phy_10g_srefclk_mode_t Struct Reference

10G Phy srefclk config data

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL enable
- vtss_phy_10g_srefclk_freq_t freq

3.77.1 Detailed Description

10G Phy srefclk config data

Definition at line 787 of file vtss_phy_10g_api.h.

3.77.2 Field Documentation

3.77.2.1 enable

```
BOOL vtss_phy_10g_srefclk_mode_t::enable
```

Enable locking line tx clock to srefclk input

Definition at line 788 of file vtss_phy_10g_api.h.

3.77.2.2 freq

```
vtss_phy_10g_srefclk_freq_t vtss_phy_10g_srefclk_mode_t::freq
```

The srefclk input frequency

Definition at line 789 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.78 vtss_phy_10g_status_t Struct Reference

10G Phy link and fault status for all sublayers

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- vtss_sublayer_status_t pma
- vtss_sublayer_status_t hpma
- vtss_sublayer_status_t wis
- vtss_sublayer_status_t pcs
- vtss_sublayer_status_t hpcs
- vtss_sublayer_status_t xs
- BOOL lpcs_1g
- BOOL hpcs_1g
- BOOL status
- BOOL block_lock
- BOOL lopc_stat

3.78.1 Detailed Description

10G Phy link and fault status for all sublayers

Definition at line 1428 of file vtss_phy_10g_api.h.

3.78.2 Field Documentation

3.78.2.1 pma

```
vtss_sublayer_status_t vtss_phy_10g_status_t::pma
```

Status for Line PMA sublayer

Definition at line 1429 of file vtss_phy_10g_api.h.

3.78.2.2 hpma

```
vtss_sublayer_status_t vtss_phy_10g_status_t::hpma
```

Status for Host PMA sublayer

Definition at line 1430 of file vtss_phy_10g_api.h.

3.78.2.3 wis

```
vtss_sublayer_status_t vtss_phy_10g_status_t::wis
```

Status for WIS sublayer

Definition at line 1431 of file vtss_phy_10g_api.h.

3.78.2.4 pcs

```
vtss_sublayer_status_t vtss_phy_10g_status_t::pcs
```

Status for Line PCS sublayer

Definition at line 1432 of file vtss_phy_10g_api.h.

3.78.2.5 hpcs

```
vtss_sublayer_status_t vtss_phy_10g_status_t::hpcs
```

Status for HOST PCS sublayer,pcs xaui in case of venice

Definition at line 1433 of file vtss_phy_10g_api.h.

3.78.2.6 xs

```
vtss_sublayer_status_t vtss_phy_10g_status_t::xs
```

Status for XAUI sublayer

Definition at line 1434 of file vtss_phy_10g_api.h.

3.78.2.7 lpcs_1g

```
BOOL vtss_phy_10g_status_t::lpcs_1g
```

Status for Line 1G_PCS sublayer

Definition at line 1435 of file vtss_phy_10g_api.h.

3.78.2.8 hpcs_1g

```
BOOL vtss_phy_10g_status_t::hpcs_1g
```

Status for Host 1G_PCS sublayer

Definition at line 1436 of file vtss_phy_10g_api.h.

3.78.2.9 status

```
BOOL vtss_phy_10g_status_t::status
```

Status of whole PHY , based on operation mode and PHY type

Definition at line 1437 of file vtss_phy_10g_api.h.

3.78.2.10 block_lock

```
BOOL vtss_phy_10g_status_t::block_lock
```

Gives block lock information

Definition at line 1438 of file vtss_phy_10g_api.h.

3.78.2.11 lopc_stat

```
BOOL vtss_phy_10g_status_t::lopc_stat
```

LOPC status

Definition at line 1439 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.79 vtss_phy_10g_timestamp_val_t Struct Reference

10G PHY timestamp value array(holder)

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- u16 timestamp [10][5]

3.79.1 Detailed Description

10G PHY timestamp value array(holder)

Definition at line 2190 of file vtss_phy_10g_api.h.

3.79.2 Field Documentation

3.79.2.1 timestamp

```
u16 vtss_phy_10g_timestamp_val_t::timestamp[10][5]
```

5 bytes each of 10 timestamp values

Definition at line 2191 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.80 vtss_phy_10g_txckout_conf_t Struct Reference

10G Phy TXCKOUT config data

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- vtss_recvrd_clkout_t mode

3.80.1 Detailed Description

10G Phy TXCKOUT config data

Definition at line 743 of file vtss_phy_10g_api.h.

3.80.2 Field Documentation

3.80.2.1 mode

```
vtss_recvrd_clkout_t vtss_phy_10g_txckout_conf_t::mode
```

TXCKOUT output mode (DISABLE/RX_CLK/TX_CLK)

Definition at line 744 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.81 vtss_phy_10g_vscope_conf_t Struct Reference

`#include <vtss_phy_10g_api.h>`

Data Fields

- vtss_phy_10g_vscope_scan_t scan_type
- BOOL line
- BOOL enable
- u32 error_thres

3.81.1 Detailed Description

\ brief VSCOPE scan configuration

Definition at line 1856 of file vtss_phy_10g_api.h.

3.81.2 Field Documentation

3.81.2.1 scan_type

`vtss_phy_10g_vscope_scan_t vtss_phy_10g_vscope_conf_t::scan_type`

selects the type of scan to be implemented

Definition at line 1857 of file vtss_phy_10g_api.h.

3.81.2.2 line

`BOOL vtss_phy_10g_vscope_conf_t::line`

select line side or host side, TRUE for line side

Definition at line 1858 of file vtss_phy_10g_api.h.

3.81.2.3 enable

`BOOL vtss_phy_10g_vscope_conf_t::enable`

enable or disable vscope fast scan

Definition at line 1859 of file vtss_phy_10g_api.h.

3.81.2.4 error_thres

```
u32 vtss_phy_10g_vscope_conf_t::error_thres
```

error_threshold for vscope calculations

Definition at line 1860 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.82 vtss_phy_10g_vscope_scan_conf_t Struct Reference

VSCOPE scan configuration.

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL line
- u32 x_start
- u32 y_start
- u32 x_incr
- u32 y_incr
- u32 x_count
- u32 y_count
- u32 ber

3.82.1 Detailed Description

VSCOPE scan configuration.

Definition at line 1904 of file vtss_phy_10g_api.h.

3.82.2 Field Documentation

3.82.2.1 line

```
BOOL vtss_phy_10g_vscope_scan_conf_t::line
```

selects line or host side, 1 for line

Definition at line 1905 of file vtss_phy_10g_api.h.

3.82.2.2 x_start

```
u32 vtss_phy_10g_vscope_scan_conf_t::x_start
```

start value for x (0-127)

Definition at line 1906 of file vtss_phy_10g_api.h.

3.82.2.3 y_start

```
u32 vtss_phy_10g_vscope_scan_conf_t::y_start
```

start value for y (0-63)

Definition at line 1907 of file vtss_phy_10g_api.h.

3.82.2.4 x_incr

```
u32 vtss_phy_10g_vscope_scan_conf_t::x_incr
```

increment value for x during the scan

Definition at line 1908 of file vtss_phy_10g_api.h.

3.82.2.5 y_incr

```
u32 vtss_phy_10g_vscope_scan_conf_t::y_incr
```

increment value for y during the scan

Definition at line 1909 of file vtss_phy_10g_api.h.

3.82.2.6 x_count

```
u32 vtss_phy_10g_vscope_scan_conf_t::x_count
```

max value for x (upto which scan is to be performed)

Definition at line 1910 of file vtss_phy_10g_api.h.

3.82.2.7 y_count

```
u32 vtss_phy_10g_vscope_scan_conf_t::y_count
```

max value for y (upto which scan is to be performed)

Definition at line 1911 of file vtss_phy_10g_api.h.

3.82.2.8 ber

```
u32 vtss_phy_10g_vscope_scan_conf_t::ber
```

bit error rate

Definition at line 1912 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.83 vtss_phy_10g_vscope_scan_status_t Struct Reference

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- vtss_phy_10g_vscope_scan_conf_t scan_conf
- i32 error_free_x
- i32 error_free_y
- i32 amp_range
- u32 errors [PHASE_POINTS][AMPLITUDE_POINTS]

3.83.1 Detailed Description

\ brief Vscope eye scan status

Definition at line 1919 of file vtss_phy_10g_api.h.

3.83.2 Field Documentation

3.83.2.1 scan_conf

```
vtss_phy_10g_vscope_scan_conf_t vtss_phy_10g_vscope_scan_status_t::scan_conf
```

scan configuration data

Definition at line 1920 of file vtss_phy_10g_api.h.

3.83.2.2 error_free_x

```
i32 vtss_phy_10g_vscope_scan_status_t::error_free_x
```

error free x values in case of fast eye scan

Definition at line 1921 of file vtss_phy_10g_api.h.

3.83.2.3 error_free_y

```
i32 vtss_phy_10g_vscope_scan_status_t::error_free_y
```

error free y values in case of fast eye scan

Definition at line 1922 of file vtss_phy_10g_api.h.

3.83.2.4 amp_range

```
i32 vtss_phy_10g_vscope_scan_status_t::amp_range
```

amp range in case of fast eye scan

Definition at line 1923 of file vtss_phy_10g_api.h.

3.83.2.5 errors

```
u32 vtss_phy_10g_vscope_scan_status_t::errors[PHASE_POINTS][AMPLITUDE_POINTS]
```

error matrix in full scan mode

Definition at line 1924 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.84 vtss_phy_pcs_cnt_t Struct Reference

10G Phy PCS counters

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL block_lock_latched
- BOOL high_ber_latched
- u8 ber_cnt
- u8 err_blk_cnt

3.84.1 Detailed Description

10G Phy PCS counters

Definition at line 1668 of file vtss_phy_10g_api.h.

3.84.2 Field Documentation

3.84.2.1 block_lock_latched

```
BOOL vtss_phy_pcs_cnt_t::block_lock_latched
```

Latched block status

Definition at line 1669 of file vtss_phy_10g_api.h.

3.84.2.2 high_ber_latched

```
BOOL vtss_phy_pcs_cnt_t::high_ber_latched
```

Lathced high ber status

Definition at line 1670 of file vtss_phy_10g_api.h.

3.84.2.3 ber_cnt

```
u8 vtss_phy_pcs_cnt_t::ber_cnt
```

BER counter. Saturating, clear on read

Definition at line 1671 of file vtss_phy_10g_api.h.

3.84.2.4 err_blk_cnt

```
u8 vtss_phy_pcs_cnt_t::err_blk_cnt
```

ERROR block counter. Saturating, clear on read

Definition at line 1672 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.85 vtss_pi_conf_t Struct Reference

PI configuration.

```
#include <vtss_init_api.h>
```

Data Fields

- u32 cs_wait_ns

3.85.1 Detailed Description

PI configuration.

Definition at line 322 of file vtss_init_api.h.

3.85.2 Field Documentation

3.85.2.1 cs_wait_ns

```
u32 vtss_pi_conf_t::cs_wait_ns
```

Minimum CS wait time in nanoseconds

Definition at line 327 of file vtss_init_api.h.

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

- vtss_api/include/vtss_init_api.h

3.86 vtss_port_bridge_counters_t Struct Reference

Port bridge counter structure (RFC 4188)

```
#include <port.h>
```

Data Fields

- vtss_port_counter_t dot1dTpPortInDiscards

3.86.1 Detailed Description

Port bridge counter structure (RFC 4188)

Definition at line 201 of file port.h.

3.86.2 Field Documentation

3.86.2.1 dot1dTpPortInDiscards

```
vtss_port_counter_t vtss_port_bridge_counters_t::dot1dTpPortInDiscards
```

Rx bridge discards

Definition at line 203 of file port.h.

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

- vtss_api/include/vtss/api/port.h

3.87 vtss_port_counters_t Struct Reference

Port counter structure.

`#include <port.h>`

Data Fields

- vtss_port_rmon_counters_t rmon
- vtss_port_if_group_counters_t if_group
- vtss_port_ethernet_like_counters_t ethernet_like
- vtss_port_proprietary_counters_t prop

3.87.1 Detailed Description

Port counter structure.

Definition at line 220 of file port.h.

3.87.2 Field Documentation

3.87.2.1 rmon

`vtss_port_rmon_counters_t vtss_port_counters_t::rmon`

RMON counters

Definition at line 222 of file port.h.

3.87.2.2 if_group

`vtss_port_if_group_counters_t vtss_port_counters_t::if_group`

Interfaces Group counters

Definition at line 223 of file port.h.

3.87.2.3 ethernet_like

`vtss_port_ethernet_like_counters_t vtss_port_counters_t::ethernet_like`

Ethernet-like Interface counters

Definition at line 224 of file port.h.

3.87.2.4 prop

`vtss_port_proprietary_counters_t vtss_port_counters_t::prop`

Proprietary counters

Definition at line 230 of file port.h.

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

- vtss_api/include/vtss/api/port.h

3.88 vtss_port_ethernet_like_counters_t Struct Reference

Ethernet-like Interface counter structure (RFC 3635)

`#include <port.h>`

Data Fields

- vtss_port_counter_t dot3InPauseFrames
- vtss_port_counter_t dot3OutPauseFrames

3.88.1 Detailed Description

Ethernet-like Interface counter structure (RFC 3635)

Definition at line 163 of file port.h.

3.88.2 Field Documentation

3.88.2.1 dot3InPauseFrames

```
vtss_port_counter_t vtss_port_ethernet_like_counters_t::dot3InPauseFrames
```

Rx pause

Definition at line 171 of file port.h.

3.88.2.2 dot3OutPauseFrames

```
vtss_port_counter_t vtss_port_ethernet_like_counters_t::dot3OutPauseFrames
```

Tx pause

Definition at line 181 of file port.h.

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

- vtss_api/include/vtss/api/port.h

3.89 vtss_port_if_group_counters_t Struct Reference

Interfaces Group counter structure (RFC 2863)

```
#include <port.h>
```

Data Fields

- vtss_port_counter_t ifInOctets
- vtss_port_counter_t ifInUcastPkts
- vtss_port_counter_t ifInMulticastPkts
- vtss_port_counter_t ifInBroadcastPkts
- vtss_port_counter_t ifInNUcastPkts
- vtss_port_counter_t ifInDiscards
- vtss_port_counter_t ifInErrors
- vtss_port_counter_t ifOutOctets
- vtss_port_counter_t ifOutUcastPkts
- vtss_port_counter_t ifOutMulticastPkts
- vtss_port_counter_t ifOutBroadcastPkts
- vtss_port_counter_t ifOutNUcastPkts
- vtss_port_counter_t ifOutDiscards
- vtss_port_counter_t ifOutErrors

3.89.1 Detailed Description

Interfaces Group counter structure (RFC 2863)

Definition at line 144 of file port.h.

3.89.2 Field Documentation

3.89.2.1 ifInOctets

```
vtss_port_counter_t vtss_port_if_group_counters_t::ifInOctets
```

Rx octets

Definition at line 145 of file port.h.

3.89.2.2 ifInUcastPkts

```
vtss_port_counter_t vtss_port_if_group_counters_t::ifInUcastPkts
```

Rx unicasts

Definition at line 146 of file port.h.

3.89.2.3 ifInMulticastPkts

```
vtss_port_counter_t vtss_port_if_group_counters_t::ifInMulticastPkts
```

Rx multicasts

Definition at line 147 of file port.h.

3.89.2.4 ifInBroadcastPkts

```
vtss_port_counter_t vtss_port_if_group_counters_t::ifInBroadcastPkts
```

Rx broadcasts

Definition at line 148 of file port.h.

3.89.2.5 ifInNUcastPkts

```
vtss_port_counter_t vtss_port_if_group_counters_t::ifInNUcastPkts
```

Rx non-unicasts

Definition at line 149 of file port.h.

3.89.2.6 ifInDiscards

```
vtss_port_counter_t vtss_port_if_group_counters_t::ifInDiscards
```

Rx discards

Definition at line 150 of file port.h.

3.89.2.7 ifInErrors

```
vtss_port_counter_t vtss_port_if_group_counters_t::ifInErrors
```

Rx errors

Definition at line 151 of file port.h.

3.89.2.8 ifOutOctets

```
vtss_port_counter_t vtss_port_if_group_counters_t::ifOutOctets
```

Tx octets

Definition at line 153 of file port.h.

3.89.2.9 ifOutUcastPkts

```
vtss_port_counter_t vtss_port_if_group_counters_t::ifOutUcastPkts
```

Tx unicasts

Definition at line 154 of file port.h.

3.89.2.10 ifOutMulticastPkts

```
vtss_port_counter_t vtss_port_if_group_counters_t::ifOutMulticastPkts
```

Tx multicasts

Definition at line 155 of file port.h.

3.89.2.11 ifOutBroadcastPkts

```
vtss_port_counter_t vtss_port_if_group_counters_t::ifOutBroadcastPkts
```

Tx broadcasts

Definition at line 156 of file port.h.

3.89.2.12 ifOutNUcastPkts

```
vtss_port_counter_t vtss_port_if_group_counters_t::ifOutNUcastPkts
```

Tx non-unicasts

Definition at line 157 of file port.h.

3.89.2.13 ifOutDiscards

```
vtss_port_counter_t vtss_port_if_group_counters_t::ifOutDiscards
```

Tx discards

Definition at line 158 of file port.h.

3.89.2.14 ifOutErrors

```
vtss_port_counter_t vtss_port_if_group_counters_t::ifOutErrors
```

Tx errors

Definition at line 159 of file port.h.

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

- vtss_api/include/vtss/api/port.h

3.90 vtss_port_proprietary_counters_t Struct Reference

Port proprietary counter structure.

```
#include <port.h>
```

3.90.1 Detailed Description

Port proprietary counter structure.

Definition at line 207 of file port.h.

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

- vtss_api/include/vtss/api/port.h

3.91 vtss_port_rmon_counters_t Struct Reference

RMON counter structure (RFC 2819)

```
#include <port.h>
```

Data Fields

- vtss_port_counter_t rx_etherStatsDropEvents
- vtss_port_counter_t rx_etherStatsOctets
- vtss_port_counter_t rx_etherStatsPkts
- vtss_port_counter_t rx_etherStatsBroadcastPkts
- vtss_port_counter_t rx_etherStatsMulticastPkts
- vtss_port_counter_t rx_etherStatsCRCAlignErrors
- vtss_port_counter_t rx_etherStatsUndersizePkts
- vtss_port_counter_t rx_etherStatsOversizePkts
- vtss_port_counter_t rx_etherStatsFragments
- vtss_port_counter_t rx_etherStatsJabbers
- vtss_port_counter_t rx_etherStatsPkts64Octets
- vtss_port_counter_t rx_etherStatsPkts65to127Octets
- vtss_port_counter_t rx_etherStatsPkts128to255Octets
- vtss_port_counter_t rx_etherStatsPkts256to511Octets
- vtss_port_counter_t rx_etherStatsPkts512to1023Octets
- vtss_port_counter_t rx_etherStatsPkts1024to1518Octets
- vtss_port_counter_t rx_etherStatsPkts1519toMaxOctets
- vtss_port_counter_t tx_etherStatsDropEvents
- vtss_port_counter_t tx_etherStatsOctets
- vtss_port_counter_t tx_etherStatsPkts
- vtss_port_counter_t tx_etherStatsBroadcastPkts
- vtss_port_counter_t tx_etherStatsMulticastPkts
- vtss_port_counter_t tx_etherStatsCollisions
- vtss_port_counter_t tx_etherStatsPkts64Octets
- vtss_port_counter_t tx_etherStatsPkts65to127Octets
- vtss_port_counter_t tx_etherStatsPkts128to255Octets
- vtss_port_counter_t tx_etherStatsPkts256to511Octets
- vtss_port_counter_t tx_etherStatsPkts512to1023Octets
- vtss_port_counter_t tx_etherStatsPkts1024to1518Octets
- vtss_port_counter_t tx_etherStatsPkts1519toMaxOctets

3.91.1 Detailed Description

RMON counter structure (RFC 2819)

Definition at line 109 of file port.h.

3.91.2 Field Documentation

3.91.2.1 rx_etherStatsDropEvents

```
vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsDropEvents
```

Rx drop events

Definition at line 110 of file port.h.

3.91.2.2 rx_etherStatsOctets

```
vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsOctets
```

Rx octets

Definition at line 111 of file port.h.

3.91.2.3 rx_etherStatsPkts

```
vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsPkts
```

Rx packets

Definition at line 112 of file port.h.

3.91.2.4 rx_etherStatsBroadcastPkts

```
vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsBroadcastPkts
```

Rx broadcasts

Definition at line 113 of file port.h.

3.91.2.5 rx_etherStatsMulticastPkts

```
vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsMulticastPkts
```

Rx multicasts

Definition at line 114 of file port.h.

3.91.2.6 rx_etherStatsCRCAlignErrors

```
vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsCRCAlignErrors
```

Rx CRC/alignment errors

Definition at line 115 of file port.h.

3.91.2.7 rx_etherStatsUndersizePkts

```
vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsUndersizePkts
```

Rx undersize packets

Definition at line 116 of file port.h.

3.91.2.8 rx_etherStatsOversizePkts

```
vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsOversizePkts
```

Rx oversize packets

Definition at line 117 of file port.h.

3.91.2.9 rx_etherStatsFragments

```
vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsFragments
```

Rx fragments

Definition at line 118 of file port.h.

3.91.2.10 rx_etherStatsJabbers

```
vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsJabbers
```

Rx jabbers

Definition at line 119 of file port.h.

3.91.2.11 rx_etherStatsPkts64Octets

```
vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsPkts64Octets
```

Rx 64 byte packets

Definition at line 120 of file port.h.

3.91.2.12 rx_etherStatsPkts65to127Octets

```
vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsPkts65to127Octets
```

Rx 65-127 byte packets

Definition at line 121 of file port.h.

3.91.2.13 rx_etherStatsPkts128to255Octets

```
vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsPkts128to255Octets
```

Rx 128-255 byte packets

Definition at line 122 of file port.h.

3.91.2.14 rx_etherStatsPkts256to511Octets

```
vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsPkts256to511Octets
```

Rx 256-511 byte packets

Definition at line 123 of file port.h.

3.91.2.15 rx_etherStatsPkts512to1023Octets

vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsPkts512to1023Octets

Rx 512-1023 byte packet

Definition at line 124 of file port.h.

3.91.2.16 rx_etherStatsPkts1024to1518Octets

vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsPkts1024to1518Octets

Rx 1024-1518 byte packets

Definition at line 125 of file port.h.

3.91.2.17 rx_etherStatsPkts1519toMaxOctets

vtss_port_counter_t vtss_port_rmon_counters_t::rx_etherStatsPkts1519toMaxOctets

Rx 1519- byte packets

Definition at line 126 of file port.h.

3.91.2.18 tx_etherStatsDropEvents

vtss_port_counter_t vtss_port_rmon_counters_t::tx_etherStatsDropEvents

Tx drop events

Definition at line 128 of file port.h.

3.91.2.19 tx_etherStatsOctets

vtss_port_counter_t vtss_port_rmon_counters_t::tx_etherStatsOctets

Tx octets

Definition at line 129 of file port.h.

3.91.2.20 tx_etherStatsPkts

```
vtss_port_counter_t vtss_port_rmon_counters_t::tx_etherStatsPkts
```

Tx packets

Definition at line 130 of file port.h.

3.91.2.21 tx_etherStatsBroadcastPkts

```
vtss_port_counter_t vtss_port_rmon_counters_t::tx_etherStatsBroadcastPkts
```

Tx broadcasts

Definition at line 131 of file port.h.

3.91.2.22 tx_etherStatsMulticastPkts

```
vtss_port_counter_t vtss_port_rmon_counters_t::tx_etherStatsMulticastPkts
```

Tx multicasts

Definition at line 132 of file port.h.

3.91.2.23 tx_etherStatsCollisions

```
vtss_port_counter_t vtss_port_rmon_counters_t::tx_etherStatsCollisions
```

Tx collisions

Definition at line 133 of file port.h.

3.91.2.24 tx_etherStatsPkts64Octets

```
vtss_port_counter_t vtss_port_rmon_counters_t::tx_etherStatsPkts64Octets
```

Tx 64 byte packets

Definition at line 134 of file port.h.

3.91.2.25 tx_etherStatsPkts65to127Octets

vtss_port_counter_t vtss_port_rmon_counters_t::tx_etherStatsPkts65to127Octets

Tx 65-127 byte packets

Definition at line 135 of file port.h.

3.91.2.26 tx_etherStatsPkts128to255Octets

vtss_port_counter_t vtss_port_rmon_counters_t::tx_etherStatsPkts128to255Octets

Tx 128-255 byte packets

Definition at line 136 of file port.h.

3.91.2.27 tx_etherStatsPkts256to511Octets

vtss_port_counter_t vtss_port_rmon_counters_t::tx_etherStatsPkts256to511Octets

Tx 256-511 byte packets

Definition at line 137 of file port.h.

3.91.2.28 tx_etherStatsPkts512to1023Octets

vtss_port_counter_t vtss_port_rmon_counters_t::tx_etherStatsPkts512to1023Octets

Tx 512-1023 byte packet

Definition at line 138 of file port.h.

3.91.2.29 tx_etherStatsPkts1024to1518Octets

vtss_port_counter_t vtss_port_rmon_counters_t::tx_etherStatsPkts1024to1518Octets

Tx 1024-1518 byte packets

Definition at line 139 of file port.h.

3.91.2.30 tx_etherStatsPkts1519toMaxOctets

```
vtss_port_counter_t vtss_port_rmon_counters_t::tx_etherStatsPkts1519toMaxOctets
```

Tx 1519- byte packets

Definition at line 140 of file port.h.

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

- vtss_api/include/vtss/api/port.h

3.92 vtss_port_status_t Struct Reference

Port status parameter struct.

```
#include <port.h>
```

Data Fields

- vtss_event_t link_down
- BOOL link
- vtss_port_speed_t speed
- BOOL fdx
- BOOL remote_fault
- BOOL aneg_complete
- BOOL unidirectional_ability
- vtss_aneg_t aneg
- BOOL mdi_cross
- BOOL fiber
- BOOL copper

3.92.1 Detailed Description

Port status parameter struct.

Definition at line 295 of file port.h.

3.92.2 Field Documentation

3.92.2.1 link_down

```
vtss_event_t vtss_port_status_t::link_down
```

Link down event occurred since last call

Definition at line 297 of file port.h.

3.92.2.2 link

```
BOOL vtss_port_status_t::link
```

Link is up. Remaining fields only valid if TRUE

Definition at line 298 of file port.h.

3.92.2.3 speed

```
vtss_port_speed_t vtss_port_status_t::speed
```

Speed

Definition at line 299 of file port.h.

3.92.2.4 fdx

```
BOOL vtss_port_status_t::fdx
```

Full duplex

Definition at line 300 of file port.h.

3.92.2.5 remote_fault

```
BOOL vtss_port_status_t::remote_fault
```

Remote fault signalled

Definition at line 301 of file port.h.

3.92.2.6 aneg_complete

```
BOOL vtss_port_status_t::aneg_complete
```

Autoneg completed (for clause_37 and Cisco aneg)

Definition at line 302 of file port.h.

3.92.2.7 unidirectional_ability

```
BOOL vtss_port_status_t::unidirectional_ability
```

TRUE: PHY able to transmit from media independent interface regardless of whether the PHY has determined that a valid link has been established.FALSE: PHY able to transmit from media independent interface only when the PHY has determined that a valid link has been established. Note This bit is only applicable to 100BASE-FX and 1000BASE-X fiber media modes.

Definition at line 303 of file port.h.

3.92.2.8 aneg

```
vtss_aneg_t vtss_port_status_t::aneg
```

Auto negotiation result

Definition at line 307 of file port.h.

3.92.2.9 mdi_cross

```
BOOL vtss_port_status_t::mdi_cross
```

Indication of if Auto-MDIX crossover is performed

Definition at line 308 of file port.h.

3.92.2.10 fiber

```
BOOL vtss_port_status_t::fiber
```

Indication of if the link is a fiber link, TRUE if link is a fiber link. FALSE if link is cu link or No Media

Definition at line 309 of file port.h.

3.92.2.11 copper

```
BOOL vtss_port_status_t::copper
```

Indication of if the link is a copper link, TRUE if link is a copper link. FALSE if link is fiber link or No Media

Definition at line 310 of file port.h.

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

- vtss_api/include/vtss/api/port.h

3.93 vtss_restart_status_t Struct Reference

Restart status.

`#include <vtss_init_api.h>`

Data Fields

- vtss_restart_t restart
- vtss_version_t prev_version
- vtss_version_t cur_version

3.93.1 Detailed Description

Restart status.

Definition at line 608 of file vtss_init_api.h.

3.93.2 Field Documentation

3.93.2.1 restart

`vtss_restart_t vtss_restart_status_t::restart`

Previous restart mode

Definition at line 609 of file vtss_init_api.h.

3.93.2.2 prev_version

`vtss_version_t vtss_restart_status_t::prev_version`

Previous API version

Definition at line 610 of file vtss_init_api.h.

3.93.2.3 cur_version

```
vtss_version_t vtss_restart_status_t::cur_version
```

Current API version

Definition at line 611 of file vtss_init_api.h.

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

- vtss_api/include/vtss_init_api.h

3.94 vtss_routing_entry_t Struct Reference

Routing entry.

```
#include <types.h>
```

Data Fields

- vtss_routing_entry_type_t type
- union {
  vtss_ipv4_uc_t ipv4_uc
  vtss_ipv6_uc_t ipv6_uc
  } route

- vtss_vid_t vlan

3.94.1 Detailed Description

Routing entry.

Definition at line 868 of file types.h.

3.94.2 Field Documentation

3.94.2.1 type

```
vtss_routing_entry_type_t vtss_routing_entry_t::type
```

Type of route

Definition at line 871 of file types.h.

3.94.2.2 ipv4_uc

```
vtss_ipv4_uc_t vtss_routing_entry_t::ipv4_uc
```

IPv6 unicast route

Definition at line 875 of file types.h.

3.94.2.3 ipv6_uc

```
vtss_ipv6_uc_t vtss_routing_entry_t::ipv6_uc
```

IPv6 unicast route

Definition at line 878 of file types.h.

3.94.2.4 route

```
union { ... } vtss_routing_entry_t::route
```

Route

3.94.2.5 vlan

```
vtss_vid_t vtss_routing_entry_t::vlan
```

Link-local addresses needs to specify a egress vlan.

Definition at line 882 of file types.h.

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

- vtss_api/include/vtss/api/types.h

3.95 vtss_sublayer_status_t Struct Reference

10G Phy link and fault status

```
#include <vtss_phy_10g_api.h>
```

Data Fields

- BOOL rx_link
- vtss_event_t link_down
- BOOL rx_fault
- BOOL tx_fault

3.95.1 Detailed Description

10G Phy link and fault status

Definition at line 86 of file vtss_phy_10g_api.h.

3.95.2 Field Documentation

3.95.2.1 rx_link

```
BOOL vtss_sublayer_status_t::rx_link
```

The rx link status

Definition at line 87 of file vtss_phy_10g_api.h.

3.95.2.2 link_down

```
vtss_event_t vtss_sublayer_status_t::link_down
```

Link down event status. Clear on read

Definition at line 88 of file vtss_phy_10g_api.h.

3.95.2.3 rx_fault

```
BOOL vtss_sublayer_status_t::rx_fault
```

Rx fault event status. Clear on read

Definition at line 89 of file vtss_phy_10g_api.h.

3.95.2.4 tx_fault

```
BOOL vtss_sublayer_status_t::tx_fault
```

Tx fault event status. Clear on read

Definition at line 90 of file vtss_phy_10g_api.h.

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

- vtss_api/include/vtss_phy_10g_api.h

3.96 vtss_timeofday_t Struct Reference

Time of day structure.

`#include <vtss_os_ecos.h>`

Data Fields

- u32 sec
- time_t sec

3.96.1 Detailed Description

Time of day structure.

Definition at line 59 of file vtss_os_ecos.h.

3.96.2 Field Documentation

3.96.2.1 sec [1/2]

`u32 vtss_timeofday_t::sec`

Time of day in seconds

Definition at line 60 of file vtss_os_ecos.h.

3.96.2.2 sec [2/2]

`time_t vtss_timeofday_t::sec`

Time of day in seconds

Definition at line 109 of file vtss_os_linux.h.

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

- vtss_api/include/vtss_os_ecos.h
- vtss_api/include/vtss_os_linux.h

3.97 vtss_timestamp_t Struct Reference

Time stamp in seconds and nanoseconds.

```
#include <types.h>
```

Data Fields

- u16 sec_msb
- u32 seconds
- u32 nanoseconds

3.97.1 Detailed Description

Time stamp in seconds and nanoseconds.

Definition at line 1212 of file types.h.

3.97.2 Field Documentation

3.97.2.1 sec_msb

```
u16 vtss_timestamp_t::sec_msb
```

Seconds msb

Definition at line 1213 of file types.h.

3.97.2.2 seconds

```
u32 vtss_timestamp_t::seconds
```

Seconds

Definition at line 1214 of file types.h.

3.97.2.3 nanoseconds

```
u32 vtss_timestamp_t::nanoseconds
```

nanoseconds

Definition at line 1215 of file types.h.

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

- vtss_api/include/vtss/api/types.h

3.98 vtss_trace_conf_t Struct Reference

Trace group configuration.

```
#include <vtss_misc_api.h>
```

Data Fields

- vtss_trace_level_t level [VTSS_TRACE_LAYER_COUNT]

3.98.1 Detailed Description

Trace group configuration.

Definition at line 97 of file vtss_misc_api.h.

3.98.2 Field Documentation

3.98.2.1 level

```
vtss_trace_level_t vtss_trace_conf_t::level[VTSS_TRACE_LAYER_COUNT]
```

Trace level per layer

Definition at line 99 of file vtss_misc_api.h.

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

- vtss_api/include/vtss_misc_api.h

3.99 vtss_vid_mac_t Struct Reference

MAC Address in specific VLAN.

`#include <types.h>`

Data Fields

- vtss_vid_t vid
- vtss_mac_t mac

3.99.1 Detailed Description

MAC Address in specific VLAN.

Definition at line 652 of file types.h.

3.99.2 Field Documentation

3.99.2.1 vid

`vtss_vid_t vtss_vid_mac_t::vid`

VLAN ID

Definition at line 654 of file types.h.

3.99.2.2 mac

`vtss_mac_t vtss_vid_mac_t::mac`

MAC address

Definition at line 655 of file types.h.

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

- vtss_api/include/vtss/api/types.h

Chapter 4

File Documentation

4.1 vtss_api/include/vtss/api/l2_types.h File Reference

Layer 2 Public API Header for l2.

```
#include <vtss/api/types.h>
```

Data Structures

- struct vtss_aggr_mode_t

  Aggregation traffic distribution mode.

Enumerations

- enum vtss_sflow_type_t { VTSS_SFLOW_TYPE_NONE = 0, VTSS_SFLOW_TYPE_RX, VTSS_SFLOW_TYPE_TX, VTSS_SFLOW_TYPE_ALL }

  sFlow sampler type.

4.1.1 Detailed Description

Layer 2 Public API Header for l2.

This header file describes public Layer 2 datatypes

4.1.2 Enumeration Type Documentation

4.1.2.1 vtss_sflow_type_t

```
enum vtss_sflow_type_t
```

sFlow sampler type.

The API supports sampling ingress and egress separately, as well as simultaneously.

Enumerator

VTSS_SFLOW_TYPE_NONE	Sampler is not enabled on the port.
VTSS_SFLOW_TYPE_RX	Sampler is enabled for ingress on the port.
VTSS_SFLOW_TYPE_TX	Sampler is enabled for egress on the port.
VTSS_SFLOW_TYPE_ALL	Sampler is enabled for both ingress and egress on the port.

Definition at line 53 of file l2_types.h.

4.2 vtss_api/include/vtss/api/options.h File Reference

Features and options.

Macros

- #define VTSS_OPT_TRACE 1
- #define VTSS_OPT_VAUI_EQ_CTRL 6
- #define VTSS_PHY_OPT_VERIPHY 1
- #define VTSS_FEATURE_SYNCE_10G
- #define VTSS_FEATURE_EDC_FW_LOAD
- #define VTSS_FEATURE_WIS
- #define VTSS_FEATURE_WARM_START
- #define VTSS_ARCH_MALIBU
- #define VTSS_ARCH_MALIBU_B
- #define VTSS_ARCH_VENICE_C

4.2.1 Detailed Description

Features and options.

This header file describes target features and compile-time options

4.2.2 Macro Definition Documentation

4.2.2.1 VTSS_OPT_TRACE

```
#define VTSS_OPT_TRACE 1
```

Trace enabled by default

Definition at line 548 of file options.h.

4.2.2.2 VTSS_OPT_VAUI_EQ_CTRL

```
#define VTSS_OPT_VAUI_EQ_CTRL 6
```

Default equalization control

Definition at line 605 of file options.h.

4.2.2.3 VTSS_PHY_OPT_VERIPHY

```
#define VTSS_PHY_OPT_VERIPHY 1
```

VeriPHY enabled by default

Definition at line 613 of file options.h.

4.2.2.4 VTSS_FEATURE_SYNCE_10G

```
#define VTSS_FEATURE_SYNCE_10G
```

SYNCE - L1 syncronization feature for 10G PHYs

Definition at line 621 of file options.h.

4.2.2.5 VTSS_FEATURE_EDC_FW_LOAD

```
#define VTSS_FEATURE_EDC_FW_LOAD
```

848x EDC firmware will get loaded at initilization

Definition at line 622 of file options.h.

4.2.2.6 VTSS_FEATURE_WIS

```
#define VTSS_FEATURE_WIS
```

WAN interface sublayer functionality

Definition at line 623 of file options.h.

4.2.2.7 VTSS_FEATURE_WARM_START

```
#define VTSS_FEATURE_WARM_START
```

Warm start

Definition at line 624 of file options.h.

4.2.2.8 VTSS_ARCH_MALIBU

```
#define VTSS_ARCH_MALIBU
```

Used for Malibu-A PHY

Definition at line 625 of file options.h.

4.2.2.9 VTSS_ARCH_MALIBU_B

```
#define VTSS_ARCH_MALIBU_B
```

Used for Malibu-B PHY

Definition at line 626 of file options.h.

4.2.2.10 VTSS_ARCH_VENICE_C

```
#define VTSS_ARCH_VENICE_C
```

Used for Venice-C PHY

Definition at line 627 of file options.h.

4.3 vtss_api/include/vtss/api/phy.h File Reference

PHY Public API Header.

```
#include <vtss/api/types.h>
```

Macros

- #define VTSS_PHY_POWER_ACTIPHY_BIT 0
- #define VTSS_PHY_POWER_DYNAMIC_BIT 1

Enumerations

- enum vtss_phy_power_mode_t { VTSS_PHY_POWER_NOMINAL = 0, VTSS_PHY_POWER_ACTIPHY = 1 << VTSS_PHY_POWER_ACTIPHY_BIT, VTSS_PHY_POWER_DYNAMIC = 1 << VTSS_PHY_POWE↩ R_DYNAMIC_BIT, VTSS_PHY_POWER_ENABLED = VTSS_PHY_POWER_ACTIPHY + VTSS_PHY_P↩ OWER_DYNAMIC }

  PHY power reduction modes.
- enum vtss_phy_veriphy_status_t { VTSS_VERIPHY_STATUS_OK = 0, VTSS_VERIPHY_STATUS_OPEN = 1, VTSS_VERIPHY_STATUS_SHORT = 2, VTSS_VERIPHY_STATUS_ABNORM = 4, VTSS_VERIPHY_STATUS_SHORT_A = 8, VTSS_VERIPHY_STATUS_SHORT_B = 9, VTSS_VERIPHY_STATUS_SHORT_C = 10, VTSS_VERIPHY_STATUS_SHORT_D = 11, VTSS_VERIPHY_STATUS_COUPL_A = 12, VTSS_VERIPHY_STATUS_COUPL_B = 13, VTSS_VERIPHY_STATUS_COUPL = 14, VTSS_VERIPHY_STATUS_COUPL_D = 15, VTSS_VERIPHY_STATUS_UNKNOWN = 16, VTSS_VERIPHY_STATUS_RUNNING = 17 }

  VeriPHY status.

4.3.1 Detailed Description

PHY Public API Header.

This header file describes public PHY data-types

4.3.2 Macro Definition Documentation

4.3.2.1 VTSS_PHY_POWER_ACTIPHY_BIT

```
#define VTSS_PHY_POWER_ACTIPHY_BIT 0
```

Defines the bit used to signaling that ActiPhy is enabled

Definition at line 42 of file phy.h.

4.3.2.2 VTSS_PHY_POWER_DYNAMIC_BIT

```
#define VTSS_PHY_POWER_DYNAMIC_BIT 1
```

Defines the bit used to signaling that PerfectReach is enabled

Definition at line 43 of file phy.h.

4.3.3 Enumeration Type Documentation

4.3.3.1 vtss_phy_power_mode_t

```
enum vtss_phy_power_mode_t
```

PHY power reduction modes.

Enumerator

VTSS_PHY_POWER_NOMINAL	Default power settings
VTSS_PHY_POWER_ACTIPHY	ActiPHY - Link down power savings enabled (Bit 0)
VTSS_PHY_POWER_DYNAMIC	PerfectReach - Link up power savings enabled (Bit 1)
VTSS_PHY_POWER_ENABLED	ActiPHY + PerfectReach enabled

Definition at line 46 of file phy.h.

4.3.3.2 vtss_phy_veriphy_status_t

```
enum vtss_phy_veriphy_status_t
```

VeriPHY status.

Enumerator

VTSS_VERIPHY_STATUS_OK	Correctly terminated pair
VTSS_VERIPHY_STATUS_OPEN	Open pair
VTSS_VERIPHY_STATUS_SHORT	Short pair
VTSS_VERIPHY_STATUS_ABNORM	Abnormal termination
VTSS_VERIPHY_STATUS_SHORT_A	Cross-pair short to pair A
VTSS_VERIPHY_STATUS_SHORT_B	Cross-pair short to pair B
VTSS_VERIPHY_STATUS_SHORT_C	Cross-pair short to pair C
VTSS_VERIPHY_STATUS_SHORT_D	Cross-pair short to pair D
VTSS_VERIPHY_STATUS_COUPL_A	Abnormal cross-pair coupling, pair A
VTSS_VERIPHY_STATUS_COUPL_B	Abnormal cross-pair coupling, pair B
VTSS_VERIPHY_STATUS_COUPL_C	Abnormal cross-pair coupling, pair C
VTSS_VERIPHY_STATUS_COUPL_D	Abnormal cross-pair coupling, pair D
VTSS_VERIPHY_STATUS_UNKNOWN	Unknown - VeriPhy never started ?
VTSS_VERIPHY_STATUS_RUNNING	VeriPhy is still running - No result yet

Definition at line 54 of file phy.h.

4.4 vtss_api/include/vtss/api/port.h File Reference

Port Public API Header.

```
#include <vtss/api/types.h>
#include <vtss/api/phy.h>
```

Data Structures

- struct vtss_port_rmon_counters_t

  RMON counter structure (RFC 2819)
- struct vtss_port_if_group_counters_t

  Interfaces Group counter structure (RFC 2863)
- struct vtss_port_ethernet_like_counters_t

  Ethernet-like Interface counter structure (RFC 3635)
- struct vtss_port_bridge_counters_t

  Port bridge counter structure (RFC 4188)
- struct vtss_port_proprietary_counters_t

  Port proprietary counter structure.
- struct vtss_port_counters_t

  Port counter structure.
- struct port_custom_conf_t

  Port configuration.
- struct vtss_port_status_t

  Port status parameter struct.

Macros

- #define PORT_CAP_NONE 0x00000000
- #define PORT_CAP_AUTONEG 0x00000001
- #define PORT_CAP_10M_HDX 0x00000002
- #define PORT_CAP_10M_FDX 0x00000004
- #define PORT_CAP_100M_HDX 0x00000008
- #define PORT_CAP_100M_FDX 0x00000010
- #define PORT_CAP_1G_FDX 0x00000020
- #define PORT_CAP_2_5G_FDX 0x00000040
- #define PORT_CAP_5G_FDX 0x00000080
- #define PORT_CAP_10G_FDX 0x00000100
- #define PORT_CAP_FLOW_CTRL 0x00001000
- #define PORT_CAP_COPPER 0x00002000
- #define PORT_CAP_FIBER 0x00004000
- #define PORT_CAP_DUAL_COPPER 0x00008000
- #define PORT_CAP_DUAL_FIBER 0x00010000
- #define PORT_CAP_SD_ENABLE 0x00020000
- #define PORT_CAP_SD_HIGH 0x00040000
- #define PORT_CAP_SD_INTERNAL 0x00080000
- #define PORT_CAP_DUAL_FIBER_100FX 0x00100000
- #define PORT_CAP_XAUI_LANE_FLIP 0x00200000
- #define PORT_CAP_VTSS_10G_PHY 0x00400000
- #define PORT_CAP_SFP_DETECT 0x00800000
- #define PORT_CAP_STACKING 0x01000000
- #define PORT_CAP_DUAL_SFP_DETECT 0x02000000
- #define PORT_CAP_SFP_ONLY 0x04000000
- #define PORT_CAP_DUAL_COPPER_100FX 0x08000000
- #define PORT_CAP_HDX (PORT_CAP_10M_HDX | PORT_CAP_100M_HDX)
- #define PORT_CAP_TRI_SPEED_FDX (PORT_CAP_AUTONEG | PORT_CAP_1G_FDX | PORT_CAP_100M_FDX | PORT_CAP_10M_FDX | PORT_CAP_FLOW_CTRL)
- #define PORT_CAP_TRI_SPEED (PORT_CAP_TRI_SPEED_FDX | PORT_CAP_HDX)
- #define PORT_CAP_1G_PHY (PORT_CAP_COPPER | PORT_CAP_FIBER | PORT_CAP_DUAL_COPPER | PORT_CAP_DUAL_FIBER | PORT_CAP_DUAL_FIBER_100FX)

- #define PORT_CAP_TRI_SPEED_COPPER (PORT_CAP_TRI_SPEED | PORT_CAP_COPPER)
- #define PORT_CAP_TRI_SPEED_FIBER (PORT_CAP_TRI_SPEED | PORT_CAP_FIBER)
- #define PORT_CAP_TRI_SPEED_DUAL_COPPER (PORT_CAP_TRI_SPEED | PORT_CAP_DUAL_COPPER)
- #define PORT_CAP_TRI_SPEED_DUAL_FIBER (PORT_CAP_TRI_SPEED | PORT_CAP_DUAL_FIBER)
- #define PORT_CAP_TRI_SPEED_DUAL_FIBER_100FX (PORT_CAP_TRI_SPEED | PORT_CAP_DUAL_FIBER_100FX)
- #define PORT_CAP_ANY_FIBER (PORT_CAP_FIBER | PORT_CAP_DUAL_FIBER_100FX | PORT_CAP_DUAL_FIBER | PORT_CAP_DUAL_COPPER | PORT_CAP_SFP_DETECT)
- #define PORT_CAP_SPEED_DUAL_ANY_FIBER_FIXED_SPEED (PORT_CAP_DUAL_FIBER_100FX | PORT_CAP_DUAL_FIBER | PORT_CAP_DUAL_COPPER)
- #define PORT_CAP_SPEED_DUAL_ANY_FIBER (PORT_CAP_DUAL_COPPER | PORT_CAP_DUAL_FIBER| PORT_CAP_DUAL_FIBER_100FX | PORT_CAP_DUAL_SFP_DETECT)
- #define PORT_CAP_TRI_SPEED_DUAL_ANY_FIBER (PORT_CAP_TRI_SPEED | PORT_CAP_SPEED_DUAL_ANY_FIBER
- #define PORT_CAP_TRI_SPEED_DUAL_ANY_FIBER_FIXED_SFP_SPEED (PORT_CAP_TRI_SPEED | PORT_CAP_SPEED_DUAL_ANY_FIBER_FIXED_SPEED)
- #define PORT_CAP_DUAL_FIBER_1000X (PORT_CAP_DUAL_FIBER | PORT_CAP_DUAL_COPPER)
- #define PORT_CAP_SFP_1G (PORT_CAP_AUTONEG | PORT_CAP_100M_FDX | PORT_CAP_1G_FDX | PORT_CAP_FLOW_CTRL | PORT_CAP_SFP_ONLY)
- #define PORT_CAP_SFP_2_5G (PORT_CAP_SFP_1G | PORT_CAP_2_5G_FDX)
- #define PORT_CAP_SFP_SD_HIGH (PORT_CAP_SD_ENABLE | PORT_CAP_SD_HIGH | PORT_CAP_SD_INTERNAL | PORT_CAP_SFP_DETECT | PORT_CAP_SFP_ONLY)
- #define PORT_CAP_2_5G_TRI_SPEED_FDX (PORT_CAP_AUTONEG | PORT_CAP_2_5G_FDX | PORT_CAP_1G_FDX | PORT_CAP_100M_FDX | PORT_CAP_FLOW_CTRL)
- #define PORT_CAP_2_5G_TRI_SPEED (PORT_CAP_2_5G_TRI_SPEED_FDX | PORT_CAP_100M_HDX)
- #define PORT_CAP_2_5G_TRI_SPEED_COPPER (PORT_CAP_2_5G_TRI_SPEED | PORT_CAP_COPPER)

Typedefs

- typedef u32 port_cap_t
- typedef u64 vtss_port_counter_t

  Counter type.

Enumerations

- enum vtss_port_speed_t {
  VTSS_SPEED_UNDEFINED, VTSS_SPEED_10M, VTSS_SPEED_100M, VTSS_SPEED_1G,
  VTSS_SPEED_2500M, VTSS_SPEED_5G, VTSS_SPEED_10G, VTSS_SPEED_12G }

  Port speed.
- enum vtss_fiber_port_speed_t {
  VTSS_SPEED_FIBER_NOT_SUPPORTED_OR_DISABLED = 0, VTSS_SPEED_FIBER_100FX = 2,
  VTSS_SPEED_FIBER_1000X = 3, VTSS_SPEED_FIBER_AUTO = 4,
  VTSS_SPEED_FIBER_DISABLED = 5 }

  Fiber Port speed.

4.4.1 Detailed Description

Port Public API Header.

This header file describes public port data-types

4.4.2 Macro Definition Documentation

4.4.2.1 PORT_CAP_NONE

```
#define PORT_CAP_NONE 0x00000000
```

No capabilities

Definition at line 46 of file port.h.

4.4.2.2 PORT_CAP_AUTONEG

```
#define PORT_CAP_AUTONEG 0x00000001
```

Auto negotiation

Definition at line 47 of file port.h.

4.4.2.3 PORT_CAP_10M_HDX

```
#define PORT_CAP_10M_HDX 0x00000002
```

10 Mbps, half duplex

Definition at line 48 of file port.h.

4.4.2.4 PORT_CAP_10M_FDX

```
#define PORT_CAP_10M_FDX 0x00000004
```

10 Mbps, full duplex

Definition at line 49 of file port.h.

4.4.2.5 PORT_CAP_100M_HDX

```
#define PORT_CAP_100M_HDX 0x00000008
```

100 Mbps, half duplex

Definition at line 50 of file port.h.

4.4.2.6 PORT_CAP_100M_FDX

```
#define PORT_CAP_100M_FDX 0x00000010
```

100 Mbps, full duplex

Definition at line 51 of file port.h.

4.4.2.7 PORT_CAP_1G_FDX

```
#define PORT_CAP_1G_FDX 0x00000020
```

1 Gbps, full duplex

Definition at line 52 of file port.h.

4.4.2.8 PORT_CAP_2_5G_FDX

```
#define PORT_CAP_2_5G_FDX 0x00000040
```

2.5 Gbps, full duplex

Definition at line 53 of file port.h.

4.4.2.9 PORT_CAP_5G_FDX

```
#define PORT_CAP_5G_FDX 0x00000080
```

5Gbps, full duplex

Definition at line 54 of file port.h.

4.4.2.10 PORT_CAP_10G_FDX

```
#define PORT_CAP_10G_FDX 0x00000100
```

10Gbps, full duplex

Definition at line 55 of file port.h.

4.4.2.11 PORT_CAP_FLOW_CTRL

```
#define PORT_CAP_FLOW_CTRL 0x00001000
```

Flow control

Definition at line 56 of file port.h.

4.4.2.12 PORT_CAP_COPPER

```
#define PORT_CAP_COPPER 0x00002000
```

Copper media

Definition at line 57 of file port.h.

4.4.2.13 PORT_CAP_FIBER

```
#define PORT_CAP_FIBER 0x00004000
```

Fiber media

Definition at line 58 of file port.h.

4.4.2.14 PORT_CAP_DUAL_COPPER

```
#define PORT_CAP_DUAL_COPPER 0x00008000
```

Dual media, copper preferred

Definition at line 59 of file port.h.

4.4.2.15 PORT_CAP_DUAL_FIBER

```
#define PORT_CAP_DUAL_FIBER 0x00010000
```

Dual media, fiber preferred

Definition at line 60 of file port.h.

4.4.2.16 PORT_CAP_SD_ENABLE

```
#define PORT_CAP_SD_ENABLE 0x00020000
```

Signal Detect enabled

Definition at line 61 of file port.h.

4.4.2.17 PORT_CAP_SD_HIGH

```
#define PORT_CAP_SD_HIGH 0x00040000
```

Signal Detect active high

Definition at line 62 of file port.h.

4.4.2.18 PORT_CAP_SD_INTERNAL

```
#define PORT_CAP_SD_INTERNAL 0x00080000
```

Signal Detect select internal

Definition at line 63 of file port.h.

4.4.2.19 PORT_CAP_DUAL_FIBER_100FX

```
#define PORT_CAP_DUAL_FIBER_100FX 0x00100000
```

Dual media (Fiber = 100FX), fiber preferred

Definition at line 64 of file port.h.

4.4.2.20 PORT_CAP_XAUI_LANE_FLIP

```
#define PORT_CAP_XAUI_LANE_FLIP 0x00200000
```

Flip the XAUI lanes

Definition at line 65 of file port.h.

4.4.2.21 PORT_CAP_VTSS_10G_PHY

```
#define PORT_CAP_VTSS_10G_PHY 0x00400000
```

Connected to VTSS 10G PHY

Definition at line 66 of file port.h.

4.4.2.22 PORT_CAP_SFP_DETECT

```
#define PORT_CAP_SFP_DETECT 0x00800000
```

Auto detect the SFP module

Definition at line 67 of file port.h.

4.4.2.23 PORT_CAP_STACKING

```
#define PORT_CAP_STACKING 0x01000000
```

Stack port candidate

Definition at line 68 of file port.h.

4.4.2.24 PORT_CAP_DUAL_SFP_DETECT

```
#define PORT_CAP_DUAL_SFP_DETECT 0x02000000
```

Auto detect the SFP module for dual media

Definition at line 69 of file port.h.

4.4.2.25 PORT_CAP_SFP_ONLY

```
#define PORT_CAP_SFP_ONLY 0x04000000
```

SFP only port (not dual media)

Definition at line 70 of file port.h.

4.4.2.26 PORT_CAP_DUAL_COPPER_100FX

```
#define PORT_CAP_DUAL_COPPER_100FX 0x08000000
```

Dual media (Fiber = 100FX), copper preferred

Definition at line 71 of file port.h.

4.4.2.27 PORT_CAP_HDX

```
#define PORT_CAP_HDX (PORT_CAP_10M_HDX | PORT_CAP_100M_HDX)
```

Half duplex

Definition at line 74 of file port.h.

4.4.2.28 PORT_CAP_TRI_SPEED_FDX

```
#define PORT_CAP_TRI_SPEED_FDX (PORT_CAP_AUTONEG | PORT_CAP_1G_FDX | PORT_CAP_100M_FDX | PORT_CAP_10M_FDX
| PORT_CAP_FLOW_CTRL)
```

Tri-speed port full duplex only

Definition at line 75 of file port.h.

4.4.2.29 PORT_CAP_TRI_SPEED

```
#define PORT_CAP_TRI_SPEED (PORT_CAP_TRI_SPEED_FDX | PORT_CAP_HDX)
```

Tri-speed port, both full and half duplex

Definition at line 76 of file port.h.

4.4.2.30 PORT_CAP_1G_PHY

```
#define PORT_CAP_1G_PHY (PORT_CAP_COPPER | PORT_CAP_FIBER | PORT_CAP_DUAL_COPPER | PORT_CAP_DUAL_FIBER
| PORT_CAP_DUAL_FIBER_100FX)
```

1G PHY present

Definition at line 77 of file port.h.

4.4.2.31 PORT_CAP_TRI_SPEED_COPPER

```
#define PORT_CAP_TRI_SPEED_COPPER (PORT_CAP_TRI_SPEED | PORT_CAP_COPPER)
```

Tri-speed port copper only

Definition at line 79 of file port.h.

4.4.2.32 PORT_CAP_TRI_SPEED_FIBER

```
#define PORT_CAP_TRI_SPEED_FIBER (PORT_CAP_TRI_SPEED | PORT_CAP_FIBER)
```

Tri-speed port fiber only

Definition at line 80 of file port.h.

4.4.2.33 PORT_CAP_TRI_SPEED_DUAL_COPPER

```
#define PORT_CAP_TRI_SPEED_DUAL_COPPER (PORT_CAP_TRI_SPEED | PORT_CAP_DUAL_COPPER)
```

Tri-speed port both fiber and copper. Copper prefered

Definition at line 81 of file port.h.

4.4.2.34 PORT_CAP_TRI_SPEED_DUAL_FIBER

```
#define PORT_CAP_TRI_SPEED_DUAL_FIBER (PORT_CAP_TRI_SPEED | PORT_CAP_DUAL_FIBER)
```

Tri-speed port both fiber and copper. Fiber prefered

Definition at line 82 of file port.h.

4.4.2.35 PORT_CAP_TRI_SPEED_DUAL_FIBER_100FX

```
#define PORT_CAP_TRI_SPEED_DUAL_FIBER_100FX (PORT_CAP_TRI_SPEED | PORT_CAP_DUAL_FIBER_100FX)
```

Tri-speed port both fiber (100FX) and copper. Copper prefered

Definition at line 83 of file port.h.

4.4.2.36 PORT_CAP_ANY_FIBER

```
#define PORT_CAP_ANY_FIBER (PORT_CAP_FIBER | PORT_CAP_DUAL_FIBER_100FX | PORT_CAP_DUAL_FIBER |
PORT_CAP_DUAL_COPPER | PORT_CAP_SFP_DETECT)
```

Any fiber mode

Definition at line 84 of file port.h.

4.4.2.37 PORT_CAP_SPEED_DUAL_ANY_FIBER_FIXED_SPEED

```
#define PORT_CAP_SPEED_DUAL_ANY_FIBER_FIXED_SPEED (PORT_CAP_DUAL_FIBER_100FX | PORT_CAP_DUAL_FIBER
| PORT_CAP_DUAL_COPPER)
```

Any fiber mode, but auto detection not supported

Definition at line 85 of file port.h.

4.4.2.38 PORT_CAP_SPEED_DUAL_ANY_FIBER

```
#define PORT_CAP_SPEED_DUAL_ANY_FIBER (PORT_CAP_DUAL_COPPER | PORT_CAP_DUAL_FIBER| PORT_CAP_DUAL_FIBER_100FX
| PORT_CAP_DUAL_SFP_DETECT)
```

Any fiber mode, auto detection supported

Definition at line 86 of file port.h.

4.4.2.39 PORT_CAP_TRI_SPEED_DUAL_ANY_FIBER

```
#define PORT_CAP_TRI_SPEED_DUAL_ANY_FIBER (PORT_CAP_TRI_SPEED | PORT_CAP_SPEED_DUAL_ANY_FIBER)
```

Copper 5 Fiber mode, auto detection supported

Definition at line 87 of file port.h.

4.4.2.40 PORT_CAP_TRI_SPEED_DUAL_ANY_FIBER_FIXED_SFP_SPEED

```
#define PORT_CAP_TRI_SPEED_DUAL_ANY_FIBER_FIXED_SFP_SPEED (PORT_CAP_TRI_SPEED | PORT_CAP_SPEED_DUAL_ANY_FIBER_
```

Copper & Fiber mode, but SFP auto detection not supported

Definition at line 88 of file port.h.

4.4.2.41 PORT_CAP_DUAL_FIBER_1000X

```
#define PORT_CAP_DUAL_FIBER_1000X (PORT_CAP_DUAL_FIBER | PORT_CAP_DUAL_COPPER)
```

1000Base-X fiber mode

Definition at line 89 of file port.h.

4.4.2.42 PORT_CAP_SFP_1G

```
#define PORT_CAP_SFP_1G (PORT_CAP_AUTONEG | PORT_CAP_100M_FDX | PORT_CAP_1G_FDX | PORT_CAP_FLOW_CTRL
| PORT_CAP_SFP_ONLY)
```

SFP fiber port 100FX/1G with auto negotiation and flow control

Definition at line 91 of file port.h.

4.4.2.43 PORT_CAP_SFP_2_5G

```
#define PORT_CAP_SFP_2_5G (PORT_CAP_SFP_1G | PORT_CAP_2_5G_FDX)
```

SFP fiber port 100FX/1G/2.5G with auto negotiation and flow control

Definition at line 92 of file port.h.

4.4.2.44 PORT_CAP_SFP_SD_HIGH

```
#define PORT_CAP_SFP_SD_HIGH (PORT_CAP_SD_ENABLE | PORT_CAP_SD_HIGH | PORT_CAP_SD_INTERNAL |
PORT_CAP_SFP_DETECT | PORT_CAP_SFP_ONLY)
```

SFP fiber port 100FX/1G/2.5G with auto negotiation and flow control, signal detect high

Definition at line 93 of file port.h.

4.4.2.45 PORT_CAP_2_5G_TRI_SPEED_FDX

```
#define PORT_CAP_2_5G_TRI_SPEED_FDX (PORT_CAP_AUTONEG | PORT_CAP_2_5G_FDX | PORT_CAP_1G_FDX |
PORT_CAP_100M_FDX | PORT_CAP_FLOW_CTRL)
```

100M/1G/2.5G Tri-speed port full duplex only

Definition at line 95 of file port.h.

4.4.2.46 PORT_CAP_2_5G_TRI_SPEED

```
#define PORT_CAP_2_5G_TRI_SPEED (PORT_CAP_2_5G_TRI_SPEED_FDX | PORT_CAP_100M_HDX)
```

100M/1G/2.5G Tri-speed port, all full duplex and 100M half duplex

Definition at line 96 of file port.h.

4.4.2.47 PORT_CAP_2_5G_TRI_SPEED_COPPER

```
#define PORT_CAP_2_5G_TRI_SPEED_COPPER (PORT_CAP_2_5G_TRI_SPEED | PORT_CAP_COPPER)
```

100M/1G/2.5G Tri-speed port copper only

Definition at line 97 of file port.h.

4.4.3 Typedef Documentation

4.4.3.1 port_cap_t

```
typedef u32 port_cap_t
```

Bit-mask containing the port capabilities

Definition at line 99 of file port.h.

4.4.4 Enumeration Type Documentation

4.4.4.1 vtss_port_speed_t

```
enum vtss_port_speed_t
```

Port speed.

Enumerator

VTSS_SPEED_UNDEFINED	Undefined
VTSS_SPEED_10M	10 M
VTSS_SPEED_100M	100 M
VTSS_SPEED_1G	1 G
VTSS_SPEED_2500M	2.5G
VTSS_SPEED_5G	5G or 2x2.5G
VTSS_SPEED_10G	10 G
VTSS_SPEED_12G	12G

Definition at line 242 of file port.h.

4.4.4.2 vtss_fiber_port_speed_t

```
enum vtss_fiber_port_speed_t
```

Fiber Port speed.

Enumerator

VTSS_SPEED_FIBER_NOT_SUPPORTED_OR_DISABLED	Fiber not supported/ Fiber port disabled
VTSS_SPEED_FIBER_100FX	100BASE-FX
VTSS_SPEED_FIBER_1000X	1000BASE-X
VTSS_SPEED_FIBER_AUTO	Auto detection
VTSS_SPEED_FIBER_DISABLED	Obsolete - use VTSS_SPEED_FIBER_NOT_SUPPORTED_OR_DISABLED instead

Definition at line 255 of file port.h.

4.5 vtss_api/include/vtss/api/types.h File Reference

Generic types API.

```
#include <vtss/api/options.h>
#include <sys/bsdtypes.h>
```

Data Structures

- struct vtss_aneg_t
  Auto negotiation struct.
- struct vtss_mac_t
  MAC Address.
- struct vtss_vid_mac_t
  MAC Address in specific VLAN.
- struct vtss_ipv6_t
  IPv6 address/mask.
- struct vtss_ip_addr_t
  Either an IPv4 or IPv6 address.
- struct vtss_ipv4_network_t
  IPv4 network.
- struct vtss_ipv6_network_t
  IPv6 network.
- struct vtss_ip_network_t

IPv6 network.

- struct vtss_ipv4_uc_t

  IPv4 unicast routing entry.
- struct vtss_ipv6_uc_t

  IPv6 routing entry.
- struct vtss_routing_entry_t

  Routing entry.
- struct vtss_l3_counters_t

  Routing interface statics counter.
- struct vtss_timestamp_t

  Time stamp in seconds and nanoseconds.

Macros

- #define PRIu64 "llu"
- #define PRIi64 "lli"
- #define PRIx64 "llx"
- #define VTSS_BIT64(x) (1ULL << (x))
- #define VTSS_BITMASK64(x) ((1ULL << (x)) - 1)
- #define VTSS_EXTRACT_BITFIELD64(x, o, w) (((x) >> (o)) & VTSS_BITMASK64(w))
- #define VTSS_ENCODE_BITFIELD64(x, o, w) (((u64)(x) & VTSS_BITMASK64(w)) << (o))
- #define VTSS_ENCODE_BITMASK64(o, w) (VTSS_BITMASK64(w) << (o))
- #define TRUE 1
- #define FALSE 0
- #define VTSS_PACKET_RATE_DISABLED 0xffffffff
- #define VTSS_PORT_COUNT 1
- #define VTSS_PORTS VTSS_OPT_PORT_COUNT
- #define VTSS_PORT_NO_NONE (0xffffffff)
- #define VTSS_PORT_NO_CPU (0xfffffffe)
- #define VTSS_PORT_NO_START (0)
- #define VTSS_PORT_NO_END (VTSS_PORT_NO_START+VTSS_PORTS)
- #define VTSS_PORT_ARRAY_SIZE VTSS_PORT_NO_END
- #define VTSS_PORT_IS_PORT(x) ((x)<VTSS_PORT_NO_END)
- #define VTSS_VID_NULL ((const vtss_vid_t)0)
- #define VTSS_VID_DEFAULT ((const vtss_vid_t)1)
- #define VTSS_VID_RESERVED ((const vtss_vid_t)0xFFF)
- #define VTSS_VIDS ((const vtss_vid_t)4096)
- #define VTSS_VID_ALL ((const vtss_vid_t)0x1000)
- #define VTSS_ETYPE_VTSS 0x8880
- #define VTSS_MAC_ADDR_SZ_BYTES 6
- #define MAC_ADDR_BROADCAST {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF}
- #define VTSS_EVCS 256
- #define VTSS_ISDX_NONE (0)
- #define VTSS_AGGRS (VTSS_PORTS/2)
- #define VTSS_AGGR_NO_NONE 0xffffffff
- #define VTSS_AGGR_NO_START 0
- #define VTSS_AGGR_NO_END (VTSS_AGGR_NO_START+VTSS_AGGRS)
- #define VTSS_GLAGS 2
- #define VTSS_GLAG_NO_NONE 0xffffffff
- #define VTSS_GLAG_NO_START 0
- #define VTSS_GLAG_NO_END (VTSS_GLAG_NO_START+VTSS_GLAGS)
- #define VTSS_GLAG_PORTS 8
- #define VTSS_GLAG_PORT_START 0

- #define VTSS_GLAG_PORT_END (VTSS_GLAG_PORT_START+VTSS_GLAG_PORTS)
- #define VTSS_GLAG_PORT_ARRAY_SIZE VTSS_GLAG_PORT_END
- #define VTSS_HQOS_COUNT 256
- #define VTSS_HQOS_ID_NONE 0xffff
- #define VTSS_ONE_MIA 1000000000
- #define VTSS_ONE_MILL 1000000
- #define VTSS_MAX_TIMEINTERVAL 0x7fffffffffffffffLL
- #define VTSS_INTERVAL_SEC(t) ((i32)VTSS_DIV64((t)>>16, VTSS_ONE_MIA))
- #define VTSS_INTERVAL_MS(t) ((i32)VTSS_DIV64((t)>>16, VTSS_ONE_MILL))
- #define VTSS_INTERVAL_US(t) ((i32)VTSS_DIV64((t)>>16, 1000))
- #define VTSS_INTERVAL_NS(t) ((i32)VTSS_MOD64((t)>>16, VTSS_ONE_MIA))
- #define VTSS_INTERVAL_PS(t) (((((i32)(t & 0xffff))*1000)+0x8000)/0x10000)
- #define VTSS_SEC_NS_INTERVAL(s, n) (((vtss_timeinterval_t)(n)+(vtss_timeinterval_t)(s)*VTSS_ONE_MIA)<<16)
- #define VTSS_CLOCK_IDENTITY_LENGTH 8
- #define VTSS_SYNCE_CLK_PORT_ARRAY_SIZE 2

Typedefs

- typedef char i8

  Fallback Integer types.
- typedef signed short i16
- typedef signed int i32
- typedef signed long long i64
- typedef unsigned char u8
- typedef unsigned short u16
- typedef unsigned int u32
- typedef unsigned long long u64
- typedef unsigned char BOOL
- typedef unsigned int uintptr_t
- typedef int vtss_rc

  Error code type.
- typedef u32 vtss_chip_no_t

  Chip number used for targets with multiple chips.
- typedef struct vtss_state_s * vtss_inst_t

  Instance identifier.
- typedef BOOL vtss_event_t

  Description: Event type. When a variable of this type is used as an input parameter, the API will set the variable if the event has occured. The API will never clear the variable. If is up to the application to clear the variable, when the event has been handled.
- typedef u32 vtss_packet_rate_t

  Policer packet rate in PPS.
- typedef u32 vtss_port_no_t

  Port Number.
- typedef u32 vtss_phys_port_no_t

  Physical port number.
- typedef u16 vtss_vid_t

  VLAN Identifier.
- typedef u16 vtss_etype_t

  Ethernet Type.
- typedef u8 vtss_mac_addr_t[VTSS_MAC_ADDR_SZ_BYTES]
- typedef u16 vtss_evc_id_t

  EVC ID.

- typedef u32 vtss_isdx_t
- typedef u32 vtss_aggr_no_t

  Aggregation Number.
- typedef u32 vtss_glag_no_t

  Description: GLAG number.
- typedef u16 vtss_udp_tcp_t

  Description: UDP/TCP port number.
- typedef u32 vtss_ip_t

  IPv4 address/mask.
- typedef vtss_ip_t vtss_ipv4_t

  IPv4 address/mask.
- typedef u32 vtss_prefix_size_t

  Prefix size.
- typedef u16 vtss_hqos_id_t

  HQoS entry identifier (HQoS ID)
- typedef i64 vtss_clk_adj_rate_t

  *Clock adjustment rate in parts per billion (ppb) * 1<<16. Range is +-2**47 ppb For example, 8.25 ppb is expressed as 0x0000.0000.0008.4000.*
- typedef i64 vtss_timeinterval_t

  *Time interval in ns * 1<<16 range +-2**47 ns = 140737 sec = 39 hours For example, 2.5 ns is expressed as 0x0000.0000.0002.8000.*
- typedef u8 vtss_clock_identity[VTSS_CLOCK_IDENTITY_LENGTH]

  PTP clock unique identifier.

Enumerations

- enum {
  VTSS_RC_OK = 0, VTSS_RC_ERROR = -1, VTSS_RC_INV_STATE = -2, VTSS_RC_INCOMPLETE = -3,
  VTSS_RC_ERR_CLK_CONF_NOT_SUPPORTED = -6, VTSS_RC_ERR_KR_CONF_NOT_SUPPORTED
  = -7, VTSS_RC_ERR_KR_CONF_INVALID_PARAMETER = -8, VTSS_RC_ERR_PHY_BASE_NO_NOT_FOUND
  = -50,
  VTSS_RC_ERR_PHY_6G_MACRO_SETUP = -51, VTSS_RC_ERR_PHY_MEDIA_IF_NOT_SUPPORTED
  = -52, VTSS_RC_ERR_PHY_CLK_CONF_NOT_SUPPORTED = -53, VTSS_RC_ERR_PHY_GPIO_ALT_MODE_NOT_SUPP
  = -54,
  VTSS_RC_ERR_PHY_GPIO_PIN_NOT_SUPPORTED = -55, VTSS_RC_ERR_PHY_PORT_OUT_RANGE
  = -56, VTSS_RC_ERR_PHY_PATCH_SETTING_NOT_SUPPORTED = -57, VTSS_RC_ERR_PHY_LCPLL_NOT_SUPPORTE
  = -58,
  VTSS_RC_ERR_PHY_RCPLL_NOT_SUPPORTED = -59, VTSS_RC_ERR_MACSEC_INVALID_SCI_MACADDR
  = -60, VTSS_RC_ERR_MACSEC_NOT_ENABLED = -61, VTSS_RC_ERR_MACSEC_SECY_ALREADY_IN_USE
  = -63,
  VTSS_RC_ERR_MACSEC_NO_SECY_FOUND = -64, VTSS_RC_ERR_MACSEC_NO_SECY_VACANCY
  = -65, VTSS_RC_ERR_MACSEC_INVALID_VALIDATE_FRM = -66, VTSS_RC_ERR_MACSEC_COULD_NOT_PRG_SA_MA
  = -67,
  VTSS_RC_ERR_MACSEC_COULD_NOT_PRG_SA_FLOW = -68, VTSS_RC_ERR_MACSEC_COULD_NOT_ENA_SA
  = -69, VTSS_RC_ERR_MACSEC_COULD_NOT_SET_SA = -70, VTSS_RC_ERR_MACSEC_INVALID_BYPASS_HDR_LEN
  = -71,
  VTSS_RC_ERR_MACSEC_SC_NOT_FOUND = -72, VTSS_RC_ERR_MACSEC_NO_CTRL_FRM_MATCH
  = -73, VTSS_RC_ERR_MACSEC_COULD_NOT_SET_PATTERN = -74, VTSS_RC_ERR_MACSEC_TIMEOUT_ISSUE
  = -75,
  VTSS_RC_ERR_MACSEC_COULD_NOT_EMPTY_EGRESS = -76, VTSS_RC_ERR_MACSEC_AN_NOT_CREATED
  = -77, VTSS_RC_ERR_MACSEC_COULD_NOT_EMPTY_INGRESS = -78, VTSS_RC_ERR_MACSEC_TX_SC_NOT_EXIST
  = -80,
  VTSS_RC_ERR_MACSEC_COULD_NOT_DISABLE_SA = -81, VTSS_RC_ERR_MACSEC_COULD_NOT_DEL_RX_SA
  = -82, VTSS_RC_ERR_MACSEC_COULD_NOT_DEL_TX_SA = -83, VTSS_RC_ERR_MACSEC_PATTERN_NOT_SET

= -84,
VTSS_RC_ERR_MACSEC_HW_RESOURCE_EXHUSTED = -85, VTSS_RC_ERR_MACSEC_SCI_ALREADY_EXISTS
= -86, VTSS_RC_ERR_MACSEC_SC_RESOURCE_NOT_FOUND = -87, VTSS_RC_ERR_MACSEC_RX_AN_ALREADY_IN_
= -88,
VTSS_RC_ERR_MACSEC_EMPTY_RECORD = -89, VTSS_RC_ERR_MACSEC_COULD_NOT_PRG_XFORM
= -90, VTSS_RC_ERR_MACSEC_COULD_NOT_TOGGLE_SA = -91, VTSS_RC_ERR_MACSEC_TX_AN_ALREADY_IN_US
= -92,
VTSS_RC_ERR_MACSEC_ALL_AVAILABLE_SA_IN_USE = -93, VTSS_RC_ERR_MACSEC_MATCH_DISABLE
= -94, VTSS_RC_ERR_MACSEC_ALL_CP_RULES_IN_USE = -95, VTSS_RC_ERR_MACSEC_PATTERN_PRIO_NOT_VAL
= -96,
VTSS_RC_ERR_MACSEC_BUFFER_TOO_SMALL = -97, VTSS_RC_ERR_MACSEC_FRAME_TOO_LONG
= -98, VTSS_RC_ERR_MACSEC_FRAME_TRUNCATED = -99, VTSS_RC_ERR_MACSEC_PHY_POWERED_DOWN
= -100,
VTSS_RC_ERR_MACSEC_PHY_NOT_MACSEC_CAPABLE = -101, VTSS_RC_ERR_MACSEC_AN_NOT_EXIST
= -102, VTSS_RC_ERR_MACSEC_NO_PATTERN_CFG = -103, VTSS_RC_ERR_MACSEC_MAX_MTU =
-105,
VTSS_RC_ERR_MACSEC_UNEXPECT_CP_MODE = -106, VTSS_RC_ERR_MACSEC_COULD_NOT_DISABLE_AN
= -107, VTSS_RC_ERR_MACSEC_RULE_OUT_OF_RANGE = -108, VTSS_RC_ERR_MACSEC_RULE_NOT_EXIST
= -109,
VTSS_RC_ERR_MACSEC_CSR_READ = -110, VTSS_RC_ERR_MACSEC_CSR_WRITE = -111,
VTSS_RC_ERR_PHY_6G_RCPLL_ON_BASE_PORT_ONLY = -112, VTSS_RC_ERR_INVALID_NULL_PTR
= -200 }

Error codes.

- enum vtss_mem_flags_t { VTSS_MEM_FLAGS_NONE = 0x0, VTSS_MEM_FLAGS_DMA = 0x1,
VTSS_MEM_FLAGS_PERSIST = 0x2 }

  Memory allocation flags.

- enum vtss_port_interface_t {
VTSS_PORT_INTERFACE_NO_CONNECTION, VTSS_PORT_INTERFACE_LOOPBACK, VTSS_PORT_INTERFACE_INTE
VTSS_PORT_INTERFACE_MII,
VTSS_PORT_INTERFACE_GMII, VTSS_PORT_INTERFACE_RGMII, VTSS_PORT_INTERFACE_TBI,
VTSS_PORT_INTERFACE_RTBI,
VTSS_PORT_INTERFACE_SGMII, VTSS_PORT_INTERFACE_SGMII_CISCO, VTSS_PORT_INTERFACE_SERDES,
VTSS_PORT_INTERFACE_VAUI,
VTSS_PORT_INTERFACE_100FX, VTSS_PORT_INTERFACE_XAUI, VTSS_PORT_INTERFACE_RXAUI,
VTSS_PORT_INTERFACE_XGMII,
VTSS_PORT_INTERFACE_SPI4, VTSS_PORT_INTERFACE_QSGMII, VTSS_PORT_INTERFACE_SFI }

  The different interfaces for connecting MAC and PHY.

- enum vtss_serdes_mode_t {
VTSS_SERDES_MODE_DISABLE, VTSS_SERDES_MODE_XAUI_12G, VTSS_SERDES_MODE_XAUI,
VTSS_SERDES_MODE_RXAUI,
VTSS_SERDES_MODE_RXAUI_12G, VTSS_SERDES_MODE_2G5, VTSS_SERDES_MODE_QSGMII,
VTSS_SERDES_MODE_SGMII,
VTSS_SERDES_MODE_100FX, VTSS_SERDES_MODE_1000BaseX, VTSS_SERDES_MODE_SFI,
VTSS_SERDES_MODE_SFI_DAC,
VTSS_SERDES_MODE_IDLE }

  Serdes macro mode.

- enum vtss_vlan_frame_t { VTSS_VLAN_FRAME_ALL, VTSS_VLAN_FRAME_TAGGED, VTSS_VLAN_FRAME_UNTAGGED
}

  VLAN acceptable frame type.

- enum vtss_vdd_t { VTSS_VDD_1V0, VTSS_VDD_1V2 }

  VDD power supply.

- enum vtss_ip_type_t { VTSS_IP_TYPE_NONE = 0, VTSS_IP_TYPE_IPV4 = 1, VTSS_IP_TYPE_IPV6 = 2 }

  IP address type.

- enum vtss_routing_entry_type_t { **VTSS_ROUTING_ENTRY_TYPE_INVALID** = 0, **VTSS_ROUTING_EN**↩
TRY_TYPE_IPV6_UC = 1, **VTSS_ROUTING_ENTRY_TYPE_IPV4_MC** = 2, **VTSS_ROUTING_ENTRY_**↩
TYPE_IPV4_UC = 3 }

Routing entry type.

- enum vtss_hqos_sch_mode_t { VTSS_HQOS_SCH_MODE_NORMAL, VTSS_HQOS_SCH_MODE_BASIC, VTSS_HQOS_SCH_MODE_HIERARCHICAL }

  HQoS port scheduling mode.

4.5.1 Detailed Description

Generic types API.

This header file describes generic types used in the API

4.5.2 Macro Definition Documentation

4.5.2.1 PRIu64

```
#define PRIu64 "llu"
```

Fallback un-signed 64-bit formatting string

Definition at line 111 of file types.h.

4.5.2.2 PRIi64

```
#define PRIi64 "lli"
```

Fallback signed 64-bit formatting string

Definition at line 115 of file types.h.

4.5.2.3 PRIx64

```
#define PRIx64 "llx"
```

Fallback hex 64-bit formatting string

Definition at line 119 of file types.h.

4.5.2.4 VTSS_BIT64

```
#define VTSS_BIT64(
              x ) (1ULL << (x))
```

Set one bit in a 64-bit mask

Definition at line 122 of file types.h.

4.5.2.5 VTSS_BITMASK64

```
#define VTSS_BITMASK64(
              x ) ((1ULL << (x)) - 1)
```

Get a bitmask consisting of x ones

Definition at line 123 of file types.h.

4.5.2.6 VTSS_EXTRACT_BITFIELD64

```
#define VTSS_EXTRACT_BITFIELD64(
              x,
              o,
              w ) (((x) >> (o)) & VTSS_BITMASK64(w))
```

Extract w bits from bit position o in x

Definition at line 124 of file types.h.

4.5.2.7 VTSS_ENCODE_BITFIELD64

```
#define VTSS_ENCODE_BITFIELD64(
              x,
              o,
              w ) (((u64)(x) & VTSS_BITMASK64(w)) << (o))
```

Place w bits of x at bit position o

Definition at line 125 of file types.h.

4.5.2.8 VTSS_ENCODE_BITMASK64

```
#define VTSS_ENCODE_BITMASK64(
              o,
              w ) (VTSS_BITMASK64(w) << (o))
```

Create a bitmask of w bits positioned at o

Definition at line 126 of file types.h.

4.5.2.9 TRUE

```
#define TRUE 1
```

True boolean value

Definition at line 129 of file types.h.

4.5.2.10 FALSE

```
#define FALSE 0
```

False boolean value

Definition at line 132 of file types.h.

4.5.2.11 VTSS_PACKET_RATE_DISABLED

```
#define VTSS_PACKET_RATE_DISABLED 0xffffffff
```

Special value for disabling packet policer

Definition at line 238 of file types.h.

4.5.2.12 VTSS_PORT_COUNT

```
#define VTSS_PORT_COUNT 1
```

Default number of ports

Definition at line 281 of file types.h.

4.5.2.13 VTSS_PORTS

```
#define VTSS_PORTS VTSS_OPT_PORT_COUNT
```

Number of ports

Definition at line 442 of file types.h.

4.5.2.14 VTSS_PORT_NO_NONE

```
#define VTSS_PORT_NO_NONE (0xffffffff)
```

Port number none

Definition at line 448 of file types.h.

4.5.2.15 VTSS_PORT_NO_CPU

```
#define VTSS_PORT_NO_CPU (0xfffffffe)
```

Port number for CPU for special purposes

Definition at line 449 of file types.h.

4.5.2.16 VTSS_PORT_NO_START

```
#define VTSS_PORT_NO_START (0)
```

Port start number

Definition at line 450 of file types.h.

4.5.2.17 VTSS_PORT_NO_END

```
#define VTSS_PORT_NO_END (VTSS_PORT_NO_START+VTSS_PORTS)
```

Port end number

Definition at line 451 of file types.h.

4.5.2.18 VTSS_PORT_ARRAY_SIZE

```
#define VTSS_PORT_ARRAY_SIZE VTSS_PORT_NO_END
```

Port number array size

Definition at line 452 of file types.h.

4.5.2.19 VTSS_PORT_IS_PORT

```
#define VTSS_PORT_IS_PORT(
              x ) ((x)<VTSS_PORT_NO_END)
```

Valid port number

Definition at line 454 of file types.h.

4.5.2.20 VTSS_VID_NULL

```
#define VTSS_VID_NULL ((const vtss_vid_t)0)
```

NULL VLAN ID

Definition at line 609 of file types.h.

4.5.2.21 VTSS_VID_DEFAULT

```
#define VTSS_VID_DEFAULT ((const vtss_vid_t)1)
```

Default VLAN ID

Definition at line 610 of file types.h.

4.5.2.22 VTSS_VID_RESERVED

```
#define VTSS_VID_RESERVED ((const vtss_vid_t)0xFFF)
```

Reserved VLAN ID

Definition at line 611 of file types.h.

4.5.2.23 VTSS_VIDS

```
#define VTSS_VIDS ((const vtss_vid_t)4096)
```

Number of VLAN IDs

Definition at line 612 of file types.h.

4.5.2.24 VTSS_VID_ALL

```
#define VTSS_VID_ALL ((const vtss_vid_t)0x1000)
```

Untagged VID: All VLAN IDs

Definition at line 613 of file types.h.

4.5.2.25 VTSS_ETYPE_VTSS

```
#define VTSS_ETYPE_VTSS 0x8880
```

VLAN tag with "arbitrary" TPID.Vitesse Ethernet Type

Definition at line 640 of file types.h.

4.5.2.26 VTSS_MAC_ADDR_SZ_BYTES

```
#define VTSS_MAC_ADDR_SZ_BYTES 6
```

Number of bytes for representing MAC address (SMAC/DMAC) type

Definition at line 648 of file types.h.

4.5.2.27 MAC_ADDR_BROADCAST

```
#define MAC_ADDR_BROADCAST {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF}
```

Broadcast address used for addr in the vtss_mac_t struct

Definition at line 658 of file types.h.

4.5.2.28 VTSS_EVCS

```
#define VTSS_EVCS 256
```

Maximum number of Ethernet Virtual Connections

Definition at line 670 of file types.h.

4.5.2.29 VTSS_ISDX_NONE

```
#define VTSS_ISDX_NONE (0)
```

Ingress Service Index number none

Definition at line 674 of file types.h.

4.5.2.30 VTSS_AGGRS

```
#define VTSS_AGGRS (VTSS_PORTS/2)
```

Number of LLAGs

Definition at line 678 of file types.h.

4.5.2.31 VTSS_AGGR_NO_NONE

```
#define VTSS_AGGR_NO_NONE 0xffffffff
```

Aggregation number none

Definition at line 679 of file types.h.

4.5.2.32 VTSS_AGGR_NO_START

```
#define VTSS_AGGR_NO_START 0
```

Aggregation start number

Definition at line 680 of file types.h.

4.5.2.33 VTSS_AGGR_NO_END

```
#define VTSS_AGGR_NO_END (VTSS_AGGR_NO_START+VTSS_AGGRS)
```

Aggregation number end

Definition at line 681 of file types.h.

4.5.2.34 VTSS_GLAGS

```
#define VTSS_GLAGS 2
```

Number of GLAGs

Definition at line 689 of file types.h.

4.5.2.35 VTSS_GLAG_NO_NONE

```
#define VTSS_GLAG_NO_NONE 0xffffffff
```

GLAG number none

Definition at line 691 of file types.h.

4.5.2.36 VTSS_GLAG_NO_START

```
#define VTSS_GLAG_NO_START 0
```

GLAG start number

Definition at line 692 of file types.h.

4.5.2.37 VTSS_GLAG_NO_END

```
#define VTSS_GLAG_NO_END (VTSS_GLAG_NO_START+VTSS_GLAGS)
```

GLAG end number

Definition at line 693 of file types.h.

4.5.2.38 VTSS_GLAG_PORTS

```
#define VTSS_GLAG_PORTS 8
```

Number of GLAG ports

Definition at line 696 of file types.h.

4.5.2.39 VTSS_GLAG_PORT_START

```
#define VTSS_GLAG_PORT_START 0
```

GLAG port start number

Definition at line 697 of file types.h.

4.5.2.40 VTSS_GLAG_PORT_END

```
#define VTSS_GLAG_PORT_END (VTSS_GLAG_PORT_START+VTSS_GLAG_PORTS)
```

GLAG port end number

Definition at line 698 of file types.h.

4.5.2.41 VTSS_GLAG_PORT_ARRAY_SIZE

```
#define VTSS_GLAG_PORT_ARRAY_SIZE VTSS_GLAG_PORT_END
```

GLAG port array size

Definition at line 699 of file types.h.

4.5.2.42 VTSS_HQOS_COUNT

```
#define VTSS_HQOS_COUNT 256
```

Maximum number of HQoS entries

Definition at line 1165 of file types.h.

4.5.2.43 VTSS_HQOS_ID_NONE

```
#define VTSS_HQOS_ID_NONE 0xffff
```

Special HQoS ID value

Definition at line 1167 of file types.h.

4.5.2.44 VTSS_ONE_MIA

```
#define VTSS_ONE_MIA 1000000000
```

One billion

Definition at line 1198 of file types.h.

4.5.2.45 VTSS_ONE_MILL

```
#define VTSS_ONE_MILL 1000000
```

One million

Definition at line 1199 of file types.h.

4.5.2.46 VTSS_MAX_TIMEINTERVAL

```
#define VTSS_MAX_TIMEINTERVAL 0x7fffffffffffffLL
```

Maximum time interval

Definition at line 1200 of file types.h.

4.5.2.47 VTSS_INTERVAL_SEC

```
#define VTSS_INTERVAL_SEC(
             t ) ((i32)VTSS_DIV64((t)>>16, VTSS_ONE_MIA))
```

One Second time interval

Definition at line 1202 of file types.h.

4.5.2.48 VTSS_INTERVAL_MS

```
#define VTSS_INTERVAL_MS(
              t ) ((i32)VTSS_DIV64((t)>>16, VTSS_ONE_MILL))
```

One millisecond time interval

Definition at line 1203 of file types.h.

4.5.2.49 VTSS_INTERVAL_US

```
#define VTSS_INTERVAL_US(
              t ) ((i32)VTSS_DIV64((t)>>16, 1000))
```

One microsecond time interval

Definition at line 1204 of file types.h.

4.5.2.50 VTSS_INTERVAL_NS

```
#define VTSS_INTERVAL_NS(
              t ) ((i32)VTSS_MOD64((t)>>16, VTSS_ONE_MIA))
```

This returns the ns part of the interval, not the total number of ns

Definition at line 1205 of file types.h.

4.5.2.51 VTSS_INTERVAL_PS

```
#define VTSS_INTERVAL_PS(
              t ) (((((i32)(t & 0xffff))*1000)+0x8000)/0x10000)
```

This returns the ps part of the interval, not the total number of ps

Definition at line 1206 of file types.h.

4.5.2.52 VTSS_SEC_NS_INTERVAL

```
#define VTSS_SEC_NS_INTERVAL(
              s,
              n ) (((vtss_timeinterval_t)(n)+(vtss_timeinterval_t)(s)*VTSS_ONE_MIA)<<16)
```

TBD

Definition at line 1207 of file types.h.

4.5.2.53 VTSS_CLOCK_IDENTITY_LENGTH

```
#define VTSS_CLOCK_IDENTITY_LENGTH 8
```

Length of unique PTP identifier

Definition at line 1218 of file types.h.

4.5.2.54 VTSS_SYNCE_CLK_PORT_ARRAY_SIZE

```
#define VTSS_SYNCE_CLK_PORT_ARRAY_SIZE 2
```

SYNCE clock out port numberarray size

Definition at line 1232 of file types.h.

4.5.3 Typedef Documentation

4.5.3.1 i8

```
typedef char i8
```

Fallback Integer types.

8-bit signed

Definition at line 68 of file types.h.

4.5.3.2 i16

```
typedef signed short i16
```

16-bit signed

Definition at line 69 of file types.h.

4.5.3.3 i32

```
typedef signed int i32
```

32-bit signed

Definition at line 70 of file types.h.

4.5.3.4 i64

```
typedef signed long long i64
```

64-bit signed

Definition at line 71 of file types.h.

4.5.3.5 u8

```
typedef unsigned char u8
```

8-bit unsigned

Definition at line 73 of file types.h.

4.5.3.6 u16

```
typedef unsigned short u16
```

16-bit unsigned

Definition at line 74 of file types.h.

4.5.3.7 u32

```
typedef unsigned int u32
```

32-bit unsigned

Definition at line 75 of file types.h.

4.5.3.8 u64

```
typedef unsigned long long u64
```

64-bit unsigned

Definition at line 76 of file types.h.

4.5.3.9 BOOL

```
typedef unsigned char BOOL
```

Boolean implemented as 8-bit unsigned

Definition at line 78 of file types.h.

4.5.3.10 uintptr_t

```
typedef unsigned int uintptr_t
```

Unsigned integer big enough to hold pointers

Definition at line 79 of file types.h.

4.5.3.11 vtss_mac_addr_t

```
typedef u8 vtss_mac_addr_t[VTSS_MAC_ADDR_SZ_BYTES]
```

MAC address (SMAC/DMAC)

Definition at line 649 of file types.h.

4.5.3.12 vtss_isdx_t

```
typedef u32 vtss_isdx_t
```

Ingress Service Index type

Definition at line 673 of file types.h.

4.5.4 Enumeration Type Documentation

4.5.4.1 anonymous enum

```
anonymous enum
```

Error codes.

Enumerator

VTSS_RC_OK	Success
VTSS_RC_ERROR	Unspecified error
VTSS_RC_INV_STATE	Invalid state for operation
VTSS_RC_INCOMPLETE	Incomplete result
VTSS_RC_ERR_CLK_CONF_NOT_SUPPORTED	The PHY doesn't support clock configuration (for SynceE)
VTSS_RC_ERR_KR_CONF_NOT_SUPPORTED	The PHY doesn't support 10GBASE_KR equalization
VTSS_RC_ERR_KR_CONF_INVALID_PARAMETER	One of the parameters are out of range
VTSS_RC_ERR_PHY_BASE_NO_NOT_FOUND	Port base number (first port within a chip) is not found
VTSS_RC_ERR_PHY_6G_MACRO_SETUP	Setup of 6G macro failed
VTSS_RC_ERR_PHY_MEDIA_IF_NOT_SUPPORTED	PHY does not support the selected media mode
VTSS_RC_ERR_PHY_CLK_CONF_NOT_SUPPORTED	The PHY doesn't support clock configuration (for SynceE)
VTSS_RC_ERR_PHY_GPIO_ALT_MODE_NOT_SUPPORTED	The PHY doesn't support the alternative mode for the selected GPIO pin
VTSS_RC_ERR_PHY_GPIO_PIN_NOT_SUPPORTED	The PHY doesn't support the selected GPIO pin
VTSS_RC_ERR_PHY_PORT_OUT_RANGE	PHY API called with port number larger than VTSS_PORTS
VTSS_RC_ERR_PHY_PATCH_SETTING_NOT_SUPPORTED	PHY API micro patch setting not supported for the port in question
VTSS_RC_ERR_PHY_LCPLL_NOT_SUPPORTED	PHY API LC-PLL status not supported for the port
VTSS_RC_ERR_PHY_RCPLL_NOT_SUPPORTED	PHY API RC-PLL status not supported for the port
VTSS_RC_ERR_MACSEC_INVALID_SCI_MACADDR	From IEEE 802.1AE-2006, section 9.9 - The 64-bit value FF-FF-FF-FF-FF-FF is never used as an SCI and is reserved for use by implementations to indicate the absence of an SC or an SCI in contexts where an SC can be present
VTSS_RC_ERR_MACSEC_NOT_ENABLED	Trying to access port where MACSEC is not enabled
VTSS_RC_ERR_MACSEC_SECY_ALREADY_IN_USE	Trying to use a secy which is already in use
VTSS_RC_ERR_MACSEC_NO_SECY_FOUND	No SecY found for the specific port
VTSS_RC_ERR_MACSEC_NO_SECY_VACANCY	No secy vacancy
VTSS_RC_ERR_MACSEC_INVALID_VALIDATE_FRM	Validate_frames value invalid
VTSS_RC_ERR_MACSEC_COULD_NOT_PRG_SA_MATCH	Could not program the SA match
VTSS_RC_ERR_MACSEC_COULD_NOT_PRG_SA_FLOW	Could not program the SA flow
VTSS_RC_ERR_MACSEC_COULD_NOT_ENA_SA	Could not enable the SA
VTSS_RC_ERR_MACSEC_COULD_NOT_SET_SA	Could not set SA to in use
VTSS_RC_ERR_MACSEC_INVALID_BYPASS_HDR_LEN	Invalid header bypass length
VTSS_RC_ERR_MACSEC_SC_NOT_FOUND	Could not find SC (from sci)
VTSS_RC_ERR_MACSEC_NO_CTRL_FRM_MATCH	No control frame match
VTSS_RC_ERR_MACSEC_COULD_NOT_SET_PATTERN	Could no set bypass pattern for CP rule
VTSS_RC_ERR_MACSEC_TIMEOUT_ISSUE	Internal timeout issue, bailing out

Enumerator

VTSS_RC_ERR_MACSEC_COULD_NOT_EMPT↩ Y_EGRESS	Could not empty the egress pipeline
VTSS_RC_ERR_MACSEC_AN_NOT_CREATED	AN not created.
VTSS_RC_ERR_MACSEC_COULD_NOT_EMPT↩ Y_INGRESS	Could not empty the ingress pipeline
VTSS_RC_ERR_MACSEC_TX_SC_NOT_EXIST	No tx SC found
VTSS_RC_ERR_MACSEC_COULD_NOT_DISAB↩ LE_SA	Could not disable sa
VTSS_RC_ERR_MACSEC_COULD_NOT_DEL_R↩ X_SA	Could not delete rx sa
VTSS_RC_ERR_MACSEC_COULD_NOT_DEL_T↩ X_SA	Could not delete tx sa
VTSS_RC_ERR_MACSEC_PATTERN_NOT_SET	Pattern not set
VTSS_RC_ERR_MACSEC_HW_RESOURCE_EX↩ HUSTED	HW resources exhausted
VTSS_RC_ERR_MACSEC_SCI_ALREADY_EXISTS	SCI already exists
VTSS_RC_ERR_MACSEC_SC_RESOURCE_NO↩ T_FOUND	Could not find SC resources
VTSS_RC_ERR_MACSEC_RX_AN_ALREADY_I↩ N_USE	Rx AN is in use
VTSS_RC_ERR_MACSEC_EMPTY_RECORD	Could not get an empty record
VTSS_RC_ERR_MACSEC_COULD_NOT_PRG_↩ XFORM	Could not program the xform record
VTSS_RC_ERR_MACSEC_COULD_NOT_TOGG↩ LE_SA	Could not toggle SA
VTSS_RC_ERR_MACSEC_TX_AN_ALREADY_I↩ N_USE	Tx AN is in use
VTSS_RC_ERR_MACSEC_ALL_AVAILABLE_SA↩ _IN_USE	All available SA's are in use
VTSS_RC_ERR_MACSEC_MATCH_DISABLE	MACSEC match disabled
VTSS_RC_ERR_MACSEC_ALL_CP_RULES_IN_↩ USE	All CP rules of the specific type are in use
VTSS_RC_ERR_MACSEC_PATTERN_PRIO_NO↩ T_VALID	The pattern priority is not valid
VTSS_RC_ERR_MACSEC_BUFFER_TOO_SMALL	Buffer to small, must be greater than VTSS_MACSEC_FRAME_CAPTURE_SIZE_MAX
VTSS_RC_ERR_MACSEC_FRAME_TOO_LONG	Frame length is supposed to be less than the amount of data in the fifo
VTSS_RC_ERR_MACSEC_FRAME_TRUNCATED	Frame is Truncated
VTSS_RC_ERR_MACSEC_PHY_POWERED_DO↩ WN	Phy is powered down, i.e. the MacSec block is not accessible
VTSS_RC_ERR_MACSEC_PHY_NOT_MACSEC↩ _CAPABLE	Port/Phy is not MacSec capable
VTSS_RC_ERR_MACSEC_AN_NOT_EXIST	AN does not exist
VTSS_RC_ERR_MACSEC_NO_PATTERN_CFG	No pattern is configured
VTSS_RC_ERR_MACSEC_MAX_MTU	Maximum MTU allowed is 32761 (+ 4 bytes for VLAN)
VTSS_RC_ERR_MACSEC_UNEXPECT_CP_MODE	Unexpected CP mode
VTSS_RC_ERR_MACSEC_COULD_NOT_DISAB↩ LE_AN	Could not disable AN
VTSS_RC_ERR_MACSEC_RULE_OUT_OF_RAN↩ GE	Rule id is out of range. Must not be larger than VTSS_MACSEC_CP_RULES

Enumerator

VTSS_RC_ERR_MACSEC_RULE_NOT_EXIST	Rule does not exist
VTSS_RC_ERR_MACSEC_CSR_READ	Could not do CSR read
VTSS_RC_ERR_MACSEC_CSR_WRITE	Could not do CSR write
VTSS_RC_ERR_PHY_6G_RCPLL_ON_BASE_P↩ ORT_ONLY	PHY API 6G RC-PLL status support only on Base port
VTSS_RC_ERR_INVALID_NULL_PTR	A pointer was unexpected NULL

Definition at line 139 of file types.h.

4.5.4.2 vtss_mem_flags_t

`enum vtss_mem_flags_t`

Memory allocation flags.

The VTSS API asks the application to allocate dynamic memory for its internal structures through calls to VTSS_OS_MALLOC().

The application should normally just associate this with a call to malloc() or kmalloc() depending on the OS and the runtime model (API running in Kernel or User space).

However, on some OSs, it's required to allocate specially if the memory is going to be associated with DMA, hence the VTSS_MEM_FLAGS_DMA enumeration.

Also, to be able to support warm restart, another enumeration, VTSS_MEM_FLAGS_PERSIST, tells the application to allocate the memory in a part of RAM that won't be affected by a subsequent boot.

VTSS_OS_MALLOC() must not block or make waiting points if called with flags != VTSS_MEM_FLAGS_NONE.

Each of the enumerations are ORed together to form the final flags that are used in a call to VTSS_OS_MALLOC().

The same set of flags are used in calls to VTSS_OS_FREE().

Enumerator

VTSS_MEM_FLAGS_NONE	Allocate normally according to runtime model (User or Kernel space).
VTSS_MEM_FLAGS_DMA	Allocate memory that can be used with a DMA.
VTSS_MEM_FLAGS_PERSIST	Allocate memory that will survive a warm restart.

Definition at line 275 of file types.h.

4.5.4.3 vtss_port_interface_t

`enum vtss_port_interface_t`

The different interfaces for connecting MAC and PHY.

Enumerator

VTSS_PORT_INTERFACE_NO_CONNECTION	No connection
VTSS_PORT_INTERFACE_LOOPBACK	Internal loopback in MAC
VTSS_PORT_INTERFACE_INTERNAL	Internal interface
VTSS_PORT_INTERFACE_MII	MII (RMII does not exist)
VTSS_PORT_INTERFACE_GMII	GMII
VTSS_PORT_INTERFACE_RGMII	RGMII
VTSS_PORT_INTERFACE_TBI	TBI
VTSS_PORT_INTERFACE_RTBI	RTBI
VTSS_PORT_INTERFACE_SGMII	SGMII
VTSS_PORT_INTERFACE_SGMII_CISCO	SGMII using Cisco aneg
VTSS_PORT_INTERFACE_SERDES	SERDES
VTSS_PORT_INTERFACE_VAUI	VAUI
VTSS_PORT_INTERFACE_100FX	100FX
VTSS_PORT_INTERFACE_XAUI	XAUI
VTSS_PORT_INTERFACE_RXAUI	RXAUI
VTSS_PORT_INTERFACE_XGMII	XGMII
VTSS_PORT_INTERFACE_SPI4	SPI4
VTSS_PORT_INTERFACE_QSGMII	QSGMII
VTSS_PORT_INTERFACE_SFI	SFI/LAN

Definition at line 457 of file types.h.

4.5.4.4 vtss_serdes_mode_t

```
enum vtss_serdes_mode_t
```

Serdes macro mode.

Enumerator

VTSS_SERDES_MODE_DISABLE	Disable serdes
VTSS_SERDES_MODE_XAUI_12G	XAUI 12G mode
VTSS_SERDES_MODE_XAUI	XAUI 10G mode
VTSS_SERDES_MODE_RXAUI	RXAUI 10G mode
VTSS_SERDES_MODE_RXAUI_12G	RXAUI 12G mode
VTSS_SERDES_MODE_2G5	2.5G mode
VTSS_SERDES_MODE_QSGMII	QSGMII mode
VTSS_SERDES_MODE_SGMII	SGMII mode
VTSS_SERDES_MODE_100FX	100FX mode
VTSS_SERDES_MODE_1000BaseX	1000BaseX mode
VTSS_SERDES_MODE_SFI	LAN/10G mode
VTSS_SERDES_MODE_SFI_DAC	LAN/10G DAC(CU)
VTSS_SERDES_MODE_IDLE	Send idles

Definition at line 490 of file types.h.

4.5.4.5 vtss_vlan_frame_t

enum vtss_vlan_frame_t

VLAN acceptable frame type.

Enumerator

VTSS_VLAN_FRAME_ALL	Accept all frames
VTSS_VLAN_FRAME_TAGGED	Accept tagged frames only
VTSS_VLAN_FRAME_UNTAGGED	Accept untagged frames only

Definition at line 618 of file types.h.

4.5.4.6 vtss_vdd_t

enum vtss_vdd_t

VDD power supply.

Enumerator

VTSS_VDD_1V0	1.0V (default)
VTSS_VDD_1V2	1.2V

Definition at line 776 of file types.h.

4.5.4.7 vtss_ip_type_t

enum vtss_ip_type_t

IP address type.

Enumerator

VTSS_IP_TYPE_NONE	Matches "InetAddressType_unknown"
VTSS_IP_TYPE_IPV4	Matches "InetAddressType_ipv4"
VTSS_IP_TYPE_IPV6	Matches "InetAddressType_ipv6"

Definition at line 806 of file types.h.

4.5.4.8 vtss_hqos_sch_mode_t

```
enum vtss_hqos_sch_mode_t
```

HQoS port scheduling mode.

The scheduling mode for the port affects which egress QoS options are available.

Enumerator

VTSS_HQOS_SCH_MODE_NORMAL	Normal QoS configuration available for non-service traffic only (default)
VTSS_HQOS_SCH_MODE_BASIC	Basic QoS configuration available for non-service traffic only
VTSS_HQOS_SCH_MODE_HIERARCHICAL	Basic QoS configuration available per HQoS entry (HQoS)

Definition at line 1173 of file types.h.

4.6 vtss_api/include/vtss_ae_api.h File Reference

ae API

```
#include <vtss/api/types.h>
```

4.6.1 Detailed Description

ae API

4.7 vtss_api/include/vtss_afi_api.h File Reference

AFI API.

```
#include <vtss/api/options.h>
```

4.7.1 Detailed Description

AFI API.

This header file describes Automatic Frame Injector functions.

4.8 vtss_api/include/vtss_aneg_api.h File Reference

ANEG API.

```
#include <vtss/api/types.h>
```

4.8.1 Detailed Description

ANEG API.

4.9 vtss_api/include/vtss_api.h File Reference

Vitesse API main header file.

```
#include <vtss/api/options.h>
#include <vtss_os.h>
#include <vtss/api/types.h>
#include <vtss_init_api.h>
#include <vtss_phy_10g_api.h>
#include <vtss_wis_api.h>
```

4.9.1 Detailed Description

Vitesse API main header file.

This is the only header file which must be included by the application

4.10 vtss_api/include/vtss_evc_api.h File Reference

EVC API.

```
#include <vtss/api/types.h>
```

4.10.1 Detailed Description

EVC API.

This header file describes EVC functions

4.11 vtss_api/include/vtss_fdma_api.h File Reference

Frame DMA API.

```
#include <vtss/api/types.h>
```

4.11.1 Detailed Description

Frame DMA API.

4.12 vtss_api/include/vtss_gfp_api.h File Reference

GFP API.

```
#include <vtss/api/types.h>
```

4.12.1 Detailed Description

GFP API.

4.13 vtss_api/include/vtss_hqos_api.h File Reference

HQoS API.

```
#include <vtss/api/types.h>
```

4.13.1 Detailed Description

HQoS API.

This header file describes Hierarchical Quality of Service (HQoS) functions

HQoS is enabled on a port when the port scheduling mode is set to hierarchical.

HQoS parameters are configured using HQoS IDs.

Traffic can be mapped to HQoS IDs by using the corresponding modules.

4.14 vtss_api/include/vtss_i2c_api.h File Reference

I2C API.

```
#include <vtss/api/types.h>
```

4.14.1 Detailed Description

I2C API.

4.15 vtss_api/include/vtss_init_api.h File Reference

Initialization API.

```
#include <vtss/api/types.h>
```

Data Structures

- struct vtss_inst_create_t

  Create structure.
- struct vtss_pi_conf_t

  PI configuration.
- struct vtss_init_conf_t

  Initialization configuration.
- struct vtss_restart_status_t

  Restart status.

Macros

- #define VTSS_I2C_NO_MULTIPLEXER -1

Typedefs

- typedef vtss_rc(∗ vtss_reg_read_t) (const vtss_chip_no_t chip_no, const u32 addr, u32 ∗const value)

  Register read function.
- typedef vtss_rc(∗ vtss_reg_write_t) (const vtss_chip_no_t chip_no, const u32 addr, const u32 value)

  Register write function.
- typedef vtss_rc(∗ vtss_i2c_read_t) (const vtss_port_no_t port_no, const u8 i2c_addr, const u8 addr, u8 ∗const data, const u8 cnt, const i8 i2c_clk_sel)

  I2C read function.
- typedef vtss_rc(∗ vtss_i2c_write_t) (const vtss_port_no_t port_no, const u8 i2c_addr, u8 ∗const data, const u8 cnt, const i8 i2c_clk_sel)

  I2C write function.
- typedef vtss_rc(∗ vtss_spi_read_write_t) (const vtss_inst_t inst, const vtss_port_no_t port_no, const u8 bitsize, u8 ∗const bitstream)

  SPI read/write function.
- typedef vtss_rc(∗ vtss_spi_32bit_read_write_t) (const vtss_inst_t inst, vtss_port_no_t port_no, BOOL read, u8 dev, u16 reg_num, u32 ∗const data)

  SPI 32 bit read/write function.
- typedef vtss_rc(∗ vtss_spi_64bit_read_write_t) (const vtss_inst_t inst, vtss_port_no_t port_no, BOOL read, u8 dev, u16 reg_num, u64 ∗const data)

  SPI 64 bit read/write function.
- typedef vtss_rc(∗ vtss_miim_read_t) (const vtss_inst_t inst, const vtss_port_no_t port_no, const u8 addr, u16 ∗const value)

  MII management read function (IEEE 802.3 clause 22)
- typedef vtss_rc(∗ vtss_miim_write_t) (const vtss_inst_t inst, const vtss_port_no_t port_no, const u8 addr, const u16 value)

  MII management write function (IEEE 802.3 clause 22)

- typedef vtss_rc(∗ vtss_mmd_read_t) (const vtss_inst_t inst, const vtss_port_no_t port_no, const u8 mmd, const u16 addr, u16 ∗const value)

  MMD management read function (IEEE 802.3 clause 45)
- typedef vtss_rc(∗ vtss_mmd_read_inc_t) (const vtss_inst_t inst, const vtss_port_no_t port_no, const u8 mmd, const u16 addr, u16 ∗const buf, u8 count)

  MMD management read increment function (IEEE 802.3 clause 45)
- typedef vtss_rc(∗ vtss_mmd_write_t) (const vtss_inst_t inst, const vtss_port_no_t port_no, const u8 mmd, const u16 addr, const u16 value)

  MMD management write function (IEEE 802.3 clause 45)
- typedef u16 vtss_version_t

  API version.

Enumerations

- enum vtss_target_type_t { VTSS_TARGET_CU_PHY, VTSS_TARGET_10G_PHY, VTSS_TARGET_SPARX_III_11 = 0x7414, VTSS_TARGET_SERVAL_LITE = 0x7416, VTSS_TARGET_SERVAL = 0x7418, VTSS_TARGET_SEVILLE = 0x9953, VTSS_TARGET_SPARX_III_10_UM = 0x7420, VTSS_TARGET_SPARX_III_17_UM = 0x7421, VTSS_TARGET_SPARX_III_25_UM = 0x7422, VTSS_TARGET_CARACAL_LITE = 0x7423, VTSS_TARGET_SPARX_III_10 = 0x7424, VTSS_TARGET_SPARX_III_18 = 0x7425, VTSS_TARGET_SPARX_III_24 = 0x7426, VTSS_TARGET_SPARX_III_26 = 0x7427, VTSS_TARGET_SPARX_III_10_01 = 0x17424, VTSS_TARGET_CARACAL_1 = 0x7428, VTSS_TARGET_CARACAL_2 = 0x7429, VTSS_TARGET_JAGUAR_1 = 0x7460, VTSS_TARGET_LYNX_1 = 0x7462, VTSS_TARGET_E_STAX_III_48 = 0x7432, VTSS_TARGET_E_STAX_III_68 = 0x7434, VTSS_TARGET_E_STAX_III_24_DUAL = 0xD7431, VTSS_TARGET_E_STAX_III_68_DUAL = 0xD7434, VTSS_TARGET_DAYTONA = 0x8492, VTSS_TARGET_TALLADEGA = 0x8494, VTSS_TARGET_SERVAL_2 = 0x7438, VTSS_TARGET_LYNX_2 = 0x7464, VTSS_TARGET_JAGUAR_2 = 0x7468, VTSS_TARGET_SPARX_IV_52 = 0x7442, VTSS_TARGET_SPARX_IV_44 = 0x7444, VTSS_TARGET_SPARX_IV_80 = 0x7448, VTSS_TARGET_SPARX_IV_90 = 0x7449 }

  Target chip type.
- enum vtss_pi_width_t { **VTSS_PI_WIDTH_16** = 0, **VTSS_PI_WIDTH_8** }

  PI data width.
- enum vtss_restart_info_src_t { **VTSS_RESTART_INFO_SRC_NONE**, **VTSS_RESTART_INFO_SRC_CU_PHY**, **VTSS_RESTART_INFO_SRC_10G_PHY** }

  Restart information source.
- enum vtss_restart_t { VTSS_RESTART_COLD, VTSS_RESTART_COOL, VTSS_RESTART_WARM }

  Restart type.

Functions

- vtss_rc vtss_inst_get (const vtss_target_type_t target, vtss_inst_create_t ∗const create)

  Initialize create structure for target.
- vtss_rc vtss_inst_create (const vtss_inst_create_t ∗const create, vtss_inst_t ∗const inst)

  Create target instance.
- vtss_rc vtss_inst_destroy (const vtss_inst_t inst)

  Destroy target instance.
- vtss_rc vtss_init_conf_get (const vtss_inst_t inst, vtss_init_conf_t ∗const conf)

  Get default initialization configuration.
- vtss_rc vtss_init_conf_set (const vtss_inst_t inst, const vtss_init_conf_t ∗const conf)

  Set initialization configuration.

- vtss_rc vtss_restart_conf_end (const vtss_inst_t inst)

  Indicate configuration end. If a warm start has been done, the stored configuration will be applied.
- vtss_rc vtss_restart_status_get (const vtss_inst_t inst, vtss_restart_status_t *const status)

  Get restart status.
- vtss_rc vtss_restart_conf_get (const vtss_inst_t inst, vtss_restart_t *const restart)

  Get restart configuration (next restart mode)
- vtss_rc vtss_restart_conf_set (const vtss_inst_t inst, const vtss_restart_t restart)

  Set restart configuration (next restart mode)

4.15.1 Detailed Description

Initialization API.

This header file describes functions used to create and initialize targets

4.15.2 Macro Definition Documentation

4.15.2.1 VTSS_I2C_NO_MULTIPLEXER

```
#define VTSS_I2C_NO_MULTIPLEXER -1
```

Used to signal not to use internal i2c clock multiplexing

Definition at line 139 of file vtss_init_api.h.

4.15.3 Typedef Documentation

4.15.3.1 vtss_reg_read_t

```
typedef vtss_rc(* vtss_reg_read_t) (const vtss_chip_no_t chip_no, const u32 addr, u32 *const
value)
```

Register read function.

Parameters

chip_no	[IN] Chip number, for targets with multiple chips
addr	[IN] Register address
value	[OUT] Register value

Returns

Return code.

Definition at line 122 of file vtss_init_api.h.

4.15.3.2 vtss_reg_write_t

```
typedef vtss_rc(* vtss_reg_write_t) (const vtss_chip_no_t chip_no, const u32 addr, const u32 value)
```

Register write function.

Parameters

chip_no	[IN] Chip number, for targets with multiple chips
addr	[IN] Register address
value	[IN] Register value

Returns

Return code.

Definition at line 135 of file vtss_init_api.h.

4.15.3.3 vtss_i2c_read_t

```
typedef vtss_rc(* vtss_i2c_read_t) (const vtss_port_no_t port_no, const u8 i2c_addr, const u8 addr, u8 *const data, const u8 cnt, const i8 i2c_clk_sel)
```

I2C read function.

Parameters

port_no	[IN] Port number
i2c_addr	[IN] I2C device address
addr	[IN] Register address
data	[OUT] Pointer the register(s) data value.
cnt	[IN] Number of registers to read
i2c_clk_sel	[IN] If i2c clock multiplexing is supported then this is the i2c mux, else use VTSS_I2C_NO_MULTIPLEXER

Returns

Return code.

Definition at line 152 of file vtss_init_api.h.

4.15.3.4 vtss_i2c_write_t

```
typedef vtss_rc(* vtss_i2c_write_t) (const vtss_port_no_t port_no, const u8 i2c_addr, u8 *const
data, const u8 cnt, const i8 i2c_clk_sel)
```

I2C write function.

Parameters

port_no	[IN] Port number
i2c_addr	[IN] I2C device address
data	[OUT] Pointer the data to be written.
cnt	[IN] Number of data bytes to write
i2c_clk_sel	[IN] If i2c clock multiplexing is supported then this is the i2c mux, else use VTSS_I2C_NO_MULTIPLEXER

Returns

Return code.

Definition at line 170 of file vtss_init_api.h.

4.15.3.5 vtss_spi_read_write_t

```
typedef vtss_rc(* vtss_spi_read_write_t) (const vtss_inst_t inst, const vtss_port_no_t port_↩
no, const u8 bitsize, u8 *const bitstream)
```

SPI read/write function.

Parameters

inst	[IN] Vitesse API instance.
port_no	[IN] Port number.
bitsize	[IN] Size (in bytes) of bitstream following this parameter.
data	[IN|OUT] Pointer to the data to be written to SPI Slave, if doing write operation. Pointer to the data read from SPI Slave, if doing read operation.

Returns

Return code.

Definition at line 187 of file vtss_init_api.h.

4.15.3.6 vtss_spi_32bit_read_write_t

```
typedef vtss_rc(* vtss_spi_32bit_read_write_t) (const vtss_inst_t inst, vtss_port_no_t port_↩
no, BOOL read, u8 dev, u16 reg_num, u32 *const data)
```

SPI 32 bit read/write function.

Parameters

inst	[IN] Vitesse API instance.
port_no	[IN] Port number.
read	[IN] Read/Write.
dev	[IN] MMD device number.
reg_num	[IN] Register offset.
data	[IN|OUT] Pointer to the data to be written to SPI Slave, if doing write operation. Pointer to the data read from SPI Slave, if doing read operation.

Returns

Return code.

Definition at line 205 of file vtss_init_api.h.

4.15.3.7 vtss_spi_64bit_read_write_t

```
typedef vtss_rc(* vtss_spi_64bit_read_write_t) (const vtss_inst_t inst, vtss_port_no_t port_↩
no, BOOL read, u8 dev, u16 reg_num, u64 *const data)
```

SPI 64 bit read/write function.

Parameters

inst	[IN] Vitesse API instance.
port_no	[IN] Port number.
read	[IN] Read/Write.
dev	[IN] MMD device number.
reg_num	[IN] Register offset.
data	[IN|OUT] Pointer to the data to be written to SPI Slave, if doing write operation. Pointer to the data read from SPI Slave, if doing read operation.

Returns

Return code.

Definition at line 225 of file vtss_init_api.h.

4.15.3.8 vtss_miim_read_t

```
typedef vtss_rc(* vtss_miim_read_t) (const vtss_inst_t inst, const vtss_port_no_t port_no,
const u8 addr, u16 *const value)
```

MII management read function (IEEE 802.3 clause 22)

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
addr	[IN] Register address (0-31)
value	[OUT] Register value

Returns

Return code.

Definition at line 242 of file vtss_init_api.h.

4.15.3.9 vtss_miim_write_t

```
typedef vtss_rc(* vtss_miim_write_t) (const vtss_inst_t inst, const vtss_port_no_t port_no,
const u8 addr, const u16 value)
```

MII management write function (IEEE 802.3 clause 22)

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
addr	[IN] Register address (0-31)
value	[IN] Register value

Returns

Return code.

Definition at line 257 of file vtss_init_api.h.

4.15.3.10 vtss_mmd_read_t

```
typedef vtss_rc(* vtss_mmd_read_t) (const vtss_inst_t inst, const vtss_port_no_t port_no,
const u8 mmd, const u16 addr, u16 *const value)
```

MMD management read function (IEEE 802.3 clause 45)

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
mmd	[IN] MMD address (0-31)
addr	[IN] Register address (0-65535)
value	[OUT] Register value

Returns

Return code.

Definition at line 273 of file vtss_init_api.h.

4.15.3.11 vtss_mmd_read_inc_t

```
typedef vtss_rc(* vtss_mmd_read_inc_t) (const vtss_inst_t inst, const vtss_port_no_t port_no,
const u8 mmd, const u16 addr, u16 *const buf, u8 count)
```

MMD management read increment function (IEEE 802.3 clause 45)

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
mmd	[IN] MMD address (0-31)
addr	[IN] Start register address (0-65535)
buf	[OUT] The register values (pointer provided by user)
count	[IN] Number of register reads (increment register reads)

Returns

Return code.

Definition at line 291 of file vtss_init_api.h.

4.15.3.12 vtss_mmd_write_t

```
typedef vtss_rc(* vtss_mmd_write_t) (const vtss_inst_t inst, const vtss_port_no_t port_no,
const u8 mmd, const u16 addr, const u16 value)
```

MMD management write function (IEEE 802.3 clause 45)

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
mmd	[IN] MMD address (0-31)
addr	[IN] Start register address (0-65535)
buf	[IN] The register value

Returns

Return code.

Definition at line 309 of file vtss_init_api.h.

4.15.4 Enumeration Type Documentation

4.15.4.1 vtss_target_type_t

```
enum vtss_target_type_t
```

Target chip type.

Enumerator

VTSS_TARGET_CU_PHY	Cu PHY family
VTSS_TARGET_10G_PHY	10G PHY family
VTSS_TARGET_SPARX_III_11	SparX-III-11 SME switch
VTSS_TARGET_SERVAL_LITE	Serval Lite CE switch
VTSS_TARGET_SERVAL	Serval CE switch
VTSS_TARGET_SEVILLE	Seville switch
VTSS_TARGET_SPARX_III_10_UM	SparxIII-10 unmanaged switch
VTSS_TARGET_SPARX_III_17_UM	SparxIII-17 unmanaged switch
VTSS_TARGET_SPARX_III_25_UM	SparxIII-25 unmanaged switch
VTSS_TARGET_CARACAL_LITE	Caracal-Lite CE switch
VTSS_TARGET_SPARX_III_10	SparxIII-10 switch
VTSS_TARGET_SPARX_III_18	SparxIII-18 switch
VTSS_TARGET_SPARX_III_24	SparxIII-24 switch
VTSS_TARGET_SPARX_III_26	SparxIII-26 switch
VTSS_TARGET_SPARX_III_10_01	SparxIII-10-01 switch
VTSS_TARGET_CARACAL_1	Caracal-1 CE switch
VTSS_TARGET_CARACAL_2	Caracal-2 CE switch
VTSS_TARGET_JAGUAR_1	Jaguar-1 CE switch
VTSS_TARGET_LYNX_1	LynX-1 CE switch
VTSS_TARGET_E_STAX_III_48	E-StaX-III-48
VTSS_TARGET_E_STAX_III_68	E-StaX-III-68
VTSS_TARGET_E_STAX_III_24_DUAL	Dual E-StaX-III-24

Enumerator

VTSS_TARGET_E_STAX_III_68_DUAL	Dual E-StaX-III-68
VTSS_TARGET_DAYTONA	Daytona FEC OTN Phy
VTSS_TARGET_TALLADEGA	Talladega FEC OTN Phy
VTSS_TARGET_SERVAL_2	Serval-2 CE switch
VTSS_TARGET_LYNX_2	LynX-2 CE switch
VTSS_TARGET_JAGUAR_2	Jaguar-2 CE switch
VTSS_TARGET_SPARX_IV_52	Sparx-IV-52 switch
VTSS_TARGET_SPARX_IV_44	Sparx-IV-44 switch
VTSS_TARGET_SPARX_IV_80	Sparx-IV-80 switch
VTSS_TARGET_SPARX_IV_90	Sparx-IV-80 switch

Definition at line 42 of file vtss_init_api.h.

4.15.4.2 vtss_restart_t

```
enum vtss_restart_t
```

Restart type.

Enumerator

VTSS_RESTART_COLD	Cold: Chip and CPU restart, e.g. power cycling
VTSS_RESTART_COOL	Cool: Chip and CPU restart done by CPU
VTSS_RESTART_WARM	Warm: CPU restart only

Definition at line 601 of file vtss_init_api.h.

4.15.5 Function Documentation

4.15.5.1 vtss_inst_get()

```
vtss_rc vtss_inst_get (
            const vtss_target_type_t target,
            vtss_inst_create_t *const create )
```

Initialize create structure for target.

Parameters

target	[IN] Target name
create	[IN] Create structure

Returns

Return code.

4.15.5.2 vtss_inst_create()

```
vtss_rc vtss_inst_create (
            const vtss_inst_create_t *const create,
            vtss_inst_t *const inst )
```

Create target instance.

Parameters

create	[IN] Create structure
inst	[OUT] Target instance reference.

Returns

Return code.

4.15.5.3 vtss_inst_destroy()

```
vtss_rc vtss_inst_destroy (
            const vtss_inst_t inst )
```

Destroy target instance.

Parameters

inst	[IN] Target instance reference.

Returns

Return code.

4.15.5.4 vtss_init_conf_get()

```
vtss_rc vtss_init_conf_get (
            const vtss_inst_t inst,
            vtss_init_conf_t *const conf )
```

Get default initialization configuration.

Parameters

inst	[IN] Target instance reference
conf	[OUT] Initialization configuration

Returns

Return code.

4.15.5.5 vtss_init_conf_set()

```
vtss_rc vtss_init_conf_set (
            const vtss_inst_t inst,
            const vtss_init_conf_t *const conf )
```

Set initialization configuration.

Parameters

inst	[IN] Target instance reference
conf	[IN] Initialization configuration

Returns

Return code.

4.15.5.6 vtss_restart_conf_end()

```
vtss_rc vtss_restart_conf_end (
            const vtss_inst_t inst )
```

Indicate configuration end. If a warm start has been done, the stored configuration will be applied.

Parameters

inst	[IN] Target instance reference

Returns

Return code.

4.15.5.7 vtss_restart_status_get()

```
vtss_rc vtss_restart_status_get (
             const vtss_inst_t inst,
             vtss_restart_status_t *const status )
```

Get restart status.

Parameters

inst	[IN] Target instance reference
status	[OUT] Restart status

Returns

Return code.

4.15.5.8 vtss_restart_conf_get()

```
vtss_rc vtss_restart_conf_get (
             const vtss_inst_t inst,
             vtss_restart_t *const restart )
```

Get restart configuration (next restart mode)

Parameters

inst	[IN] Target instance reference
restart	[OUT] Restart mode

Returns

Return code.

4.15.5.9 vtss_restart_conf_set()

```
vtss_rc vtss_restart_conf_set (
             const vtss_inst_t inst,
             const vtss_restart_t restart )
```

Set restart configuration (next restart mode)

Parameters

inst	[IN] Target instance reference
restart	[IN] Restart mode

Returns

Return code.

4.16 vtss_api/include/vtss_l2_api.h File Reference

Layer 2 API.

```
#include <vtss/api/types.h>
#include <vtss_security_api.h>
#include "vtss_port_api.h"
#include "vtss_packet_api.h"
#include <vtss/api/l2_types.h>
```

4.16.1 Detailed Description

Layer 2 API.

This header file describes Layer 2 switching functions

4.17 vtss_api/include/vtss_l3_api.h File Reference

L3 routing API.

```
#include <vtss/api/types.h>
```

4.17.1 Detailed Description

L3 routing API.

This header file describes L3 IPv4/IPv6 hardware assisted routing functions.

4.18 vtss_api/include/vtss_mac10g_api.h File Reference

MAC10G API.

```
#include <vtss/api/types.h>
```

4.18.1 Detailed Description

MAC10G API.

4.19 vtss_api/include/vtss_misc_api.h File Reference

Miscellaneous API.

```
#include <vtss/api/types.h>
#include "vtss_init_api.h"
```

Data Structures

- struct vtss_trace_conf_t
  Trace group configuration.
- struct vtss_debug_info_t
  Debug information structure.
- struct vtss_api_lock_t
  API lock structure.
- struct vtss_debug_lock_t
  API debug lock structure.
- struct vtss_os_timestamp_t

Macros

- #define VTSS_CHIP_NO_ALL 0xffffffff
  Special chip number value for showing information from all chips.
- #define VTSS_OS_TIMESTAMP_TYPE vtss_os_timestamp_t
- #define VTSS_OS_TIMESTAMP(timestamp)

Typedefs

- typedef void(* vtss_debug_printf_t) (const char *fmt,...)
  Debug printf function.
- typedef u32(* tod_get_ns_cnt_cb_t) (void)
  If the actual HW does not support time stamping, an external callback function can be set up to do the work.

Enumerations

- enum vtss_trace_layer_t { VTSS_TRACE_LAYER_AIL, VTSS_TRACE_LAYER_CIL, VTSS_TRACE_LAYER_COUNT }
  Trace group layer.
- enum vtss_trace_group_t {
  VTSS_TRACE_GROUP_DEFAULT, VTSS_TRACE_GROUP_PORT, VTSS_TRACE_GROUP_PHY,
  VTSS_TRACE_GROUP_PACKET,
  VTSS_TRACE_GROUP_AFI, VTSS_TRACE_GROUP_QOS, VTSS_TRACE_GROUP_L2, VTSS_TRACE_GROUP_L3,
  VTSS_TRACE_GROUP_SECURITY, VTSS_TRACE_GROUP_EVC, VTSS_TRACE_GROUP_FDMA_NORMAL,
  VTSS_TRACE_GROUP_FDMA_IRQ,
  VTSS_TRACE_GROUP_REG_CHECK, VTSS_TRACE_GROUP_MPLS, VTSS_TRACE_GROUP_HQOS,
  VTSS_TRACE_GROUP_MACSEC,
  VTSS_TRACE_GROUP_VCAP, VTSS_TRACE_GROUP_OAM, VTSS_TRACE_GROUP_TS, VTSS_TRACE_GROUP_COUN
  }

Trace groups.

- enum vtss_trace_level_t { VTSS_TRACE_LEVEL_NONE, VTSS_TRACE_LEVEL_ERROR, VTSS_TRACE_LEVEL_INFO, VTSS_TRACE_LEVEL_DEB VTSS_TRACE_LEVEL_NOISE, VTSS_TRACE_LEVEL_COUNT }

  Trace levels.

- enum vtss_debug_layer_t { VTSS_DEBUG_LAYER_ALL, VTSS_DEBUG_LAYER_AIL, VTSS_DEBUG_LAYER_CIL }

  Debug layer.

- enum vtss_debug_group_t { VTSS_DEBUG_GROUP_ALL, VTSS_DEBUG_GROUP_INIT, VTSS_DEBUG_GROUP_MISC, VTSS_DEBUG_GROUP_POR VTSS_DEBUG_GROUP_PORT_CNT, VTSS_DEBUG_GROUP_PHY, VTSS_DEBUG_GROUP_VLAN, VTSS_DEBUG_GROUP_PVLAN, VTSS_DEBUG_GROUP_MAC_TABLE, VTSS_DEBUG_GROUP_ACL, VTSS_DEBUG_GROUP_QOS, VTSS_DEBUG_GROUP_AGGR, VTSS_DEBUG_GROUP_GLAG, VTSS_DEBUG_GROUP_STP, VTSS_DEBUG_GROUP_MIRROR, VTSS_DEBUG_GROUP_EVC, VTSS_DEBUG_GROUP_ERPS, VTSS_DEBUG_GROUP_EPS, VTSS_DEBUG_GROUP_PACKET, VTSS_DEBUG_GROUP_FDMA, VTSS_DEBUG_GROUP_TS, VTSS_DEBUG_GROUP_PHY_TS, VTSS_DEBUG_GROUP_WM, VTSS_DEBUG_GROUP_LR VTSS_DEBUG_GROUP_IPMC, VTSS_DEBUG_GROUP_STACK, VTSS_DEBUG_GROUP_CMEF, VTSS_DEBUG_GROUP_HOST, VTSS_DEBUG_GROUP_MPLS, VTSS_DEBUG_GROUP_MPLS_OAM, VTSS_DEBUG_GROUP_HQOS, VTSS_DEBUG_GROUP_VXLAT, VTSS_DEBUG_GROUP_OAM, VTSS_DEBUG_GROUP_SER_GPIO, VTSS_DEBUG_GROUP_L3, VTSS_DEBUG_GROUP_AFI, VTSS_DEBUG_GROUP_MACSEC, VTSS_DEBUG_GROUP_COUNT }

  Debug function group.

Functions

- vtss_rc vtss_trace_conf_get (const vtss_trace_group_t group, vtss_trace_conf_t *const conf)

  Get trace configuration.

- vtss_rc vtss_trace_conf_set (const vtss_trace_group_t group, const vtss_trace_conf_t *const conf)

  Set trace configuration.

- void vtss_callout_trace_printf (const vtss_trace_layer_t layer, const vtss_trace_group_t group, const vtss_trace_level_t level, const char *file, const int line, const char *function, const char *format,...)

  Trace callout function.

- void vtss_callout_trace_hex_dump (const vtss_trace_layer_t layer, const vtss_trace_group_t group, const vtss_trace_level_t level, const char *file, const int line, const char *function, const u8 *byte_p, const int byte_cnt)

  Trace hex-dump callout function.

- vtss_rc vtss_debug_info_get (vtss_debug_info_t *const info)

  Get default debug information structure.

- vtss_rc vtss_debug_info_print (const vtss_inst_t inst, const vtss_debug_printf_t prntf, const vtss_debug_info_t *const info)

  Print default information.

- void vtss_callout_lock (const vtss_api_lock_t *const lock)

  Lock API access.

- void vtss_callout_unlock (const vtss_api_lock_t *const lock)

  Unlock API access.

- vtss_rc vtss_debug_lock (const vtss_inst_t inst, const vtss_debug_lock_t *const lock)

  Debug lock API access.

- vtss_rc vtss_debug_unlock (const vtss_inst_t inst, vtss_debug_lock_t *const lock)

Debug unlock API access.

- vtss_rc vtss_intr_cfg (const vtss_inst_t inst, const u32 mask, const BOOL polarity, const BOOL enable)

  Configure interrupt.
- u32 vtss_tod_get_ns_cnt (void)

  Get the current hw nanosec time This function is called from interrupt.
- void vtss_tod_set_ns_cnt_cb (tod_get_ns_cnt_cb_t cb)

  Set an external hw nanosec read function.
- vtss_rc vtss_debug_reg_check_set (const vtss_inst_t inst, const BOOL enable)

  Enable or disable register access checking.

4.19.1 Detailed Description

Miscellaneous API.

This header file describes miscellaneous API functions

4.19.2 Macro Definition Documentation

4.19.2.1 VTSS_OS_TIMESTAMP_TYPE

```
#define VTSS_OS_TIMESTAMP_TYPE vtss_os_timestamp_t
```

The VTSS_OS_TIME_STAMP_TYPE defines the type

Definition at line 1075 of file vtss_misc_api.h.

4.19.2.2 VTSS_OS_TIMESTAMP

```
#define VTSS_OS_TIMESTAMP(
              timestamp )
```

Value:

```
do {                                                                \
    /* Currently no need to lock scheduler, since it's only          */ \
    /* called from a function, where the sceduler is already locked. */ \
    /* cyg_scheduler_lock(__FILE__, __LINE__); */                                \
    (timestamp)->hw_cnt = vtss_tod_get_ns_cnt();                       \
    /* cyg_scheduler_unlock(__FILE__, __LINE__); */                              \
  } while(0);
```

VTSS_OS_TIMESTAMP() provides the implementation that will fill in the timestamp.

Definition at line 1076 of file vtss_misc_api.h.

4.19.3 Typedef Documentation

4.19.3.1 tod_get_ns_cnt_cb_t

```
typedef u32(* tod_get_ns_cnt_cb_t) (void)
```

If the actual HW does not support time stamping, an external callback function can be set up to do the work.

Returns

actual ns counter.

Definition at line 1054 of file vtss_misc_api.h.

4.19.4 Enumeration Type Documentation

4.19.4.1 vtss_trace_layer_t

```
enum vtss_trace_layer_t
```

Trace group layer.

Enumerator

VTSS_TRACE_LAYER_AIL	Application Interface Layer
VTSS_TRACE_LAYER_CIL	Chip Interface Layer
VTSS_TRACE_LAYER_COUNT	Number of layers

Definition at line 43 of file vtss_misc_api.h.

4.19.4.2 vtss_trace_group_t

```
enum vtss_trace_group_t
```

Trace groups.

Enumerator

VTSS_TRACE_GROUP_DEFAULT	Default trace group
VTSS_TRACE_GROUP_PORT	Port control
VTSS_TRACE_GROUP_PHY	PHY control

Enumerator

VTSS_TRACE_GROUP_PACKET	Packet control
VTSS_TRACE_GROUP_AFI	AFI
VTSS_TRACE_GROUP_QOS	Quality of Service
VTSS_TRACE_GROUP_L2	Layer 2
VTSS_TRACE_GROUP_L3	Layer 3
VTSS_TRACE_GROUP_SECURITY	Security
VTSS_TRACE_GROUP_EVC	Ethernet Virtual Connections
VTSS_TRACE_GROUP_FDMA_NORMAL	Frame DMA Extraction and Injection when interrupts/scheduler is enabled
VTSS_TRACE_GROUP_FDMA_IRQ	Frame DMA when interrupts/scheduler is disabled Daytona layers are placed before the PHY layer, othewise they are not shown in the CLI commands
VTSS_TRACE_GROUP_REG_CHECK	Register access errors (must be able to print when interrupts/scheduler is disabled)
VTSS_TRACE_GROUP_MPLS	MPLS
VTSS_TRACE_GROUP_HQOS	Hierarchical Quality of Service
VTSS_TRACE_GROUP_MACSEC	MACSEC control
VTSS_TRACE_GROUP_VCAP	VCAP
VTSS_TRACE_GROUP_OAM	OAM
VTSS_TRACE_GROUP_TS	Timestamping
VTSS_TRACE_GROUP_COUNT	Number of trace groups

Definition at line 52 of file vtss_misc_api.h.

4.19.4.3 vtss_trace_level_t

```
enum vtss_trace_level_t
```

Trace levels.

Enumerator

VTSS_TRACE_LEVEL_NONE	No trace
VTSS_TRACE_LEVEL_ERROR	Error trace
VTSS_TRACE_LEVEL_INFO	Information trace
VTSS_TRACE_LEVEL_DEBUG	Debug trace
VTSS_TRACE_LEVEL_NOISE	More debug information
VTSS_TRACE_LEVEL_COUNT	Number of trace levels

Definition at line 85 of file vtss_misc_api.h.

4.19.4.4 vtss_debug_layer_t

```
enum vtss_debug_layer_t
```

Debug layer.

Enumerator

VTSS_DEBUG_LAYER_ALL	All layers
VTSS_DEBUG_LAYER_AIL	Application Interface Layer
VTSS_DEBUG_LAYER_CIL	Chip Interface Layer

Definition at line 177 of file vtss_misc_api.h.

4.19.4.5 vtss_debug_group_t

```
enum vtss_debug_group_t
```

Debug function group.

Enumerator

VTSS_DEBUG_GROUP_ALL	All groups
VTSS_DEBUG_GROUP_INIT	Initialization
VTSS_DEBUG_GROUP_MISC	Miscellaneous
VTSS_DEBUG_GROUP_PORT	Port configuration
VTSS_DEBUG_GROUP_PORT_CNT	Port counters
VTSS_DEBUG_GROUP_PHY	PHY
VTSS_DEBUG_GROUP_VLAN	VLAN
VTSS_DEBUG_GROUP_PVLAN	PVLAN
VTSS_DEBUG_GROUP_MAC_TABLE	MAC address table
VTSS_DEBUG_GROUP_ACL	ACL
VTSS_DEBUG_GROUP_QOS	QoS
VTSS_DEBUG_GROUP_AGGR	Link aggregation
VTSS_DEBUG_GROUP_GLAG	Global link aggregation
VTSS_DEBUG_GROUP_STP	Spanning Tree
VTSS_DEBUG_GROUP_MIRROR	Mirroring
VTSS_DEBUG_GROUP_EVC	EVC
VTSS_DEBUG_GROUP_ERPS	ERPS
VTSS_DEBUG_GROUP_EPS	EPS
VTSS_DEBUG_GROUP_PACKET	Packet control
VTSS_DEBUG_GROUP_FDMA	FDMA
VTSS_DEBUG_GROUP_TS	TS: TimeStamping
VTSS_DEBUG_GROUP_PHY_TS	PHY_TS: PHY TimeStamping
VTSS_DEBUG_GROUP_WM	WaterMarks
VTSS_DEBUG_GROUP_LRN	LRN:COMMON
VTSS_DEBUG_GROUP_IPMC	IP Multicast
VTSS_DEBUG_GROUP_STACK	Stacking
VTSS_DEBUG_GROUP_CMEF	Congestion Management
VTSS_DEBUG_GROUP_HOST	CE-MAX Host configuration
VTSS_DEBUG_GROUP_MPLS	MPLS
VTSS_DEBUG_GROUP_MPLS_OAM	MPLS OAM
VTSS_DEBUG_GROUP_HQOS	Hierarchical Quality of Service
VTSS_DEBUG_GROUP_VXLAT	VLAN Translation

Enumerator

VTSS_DEBUG_GROUP_OAM	OAM, incl. VOEs/VOP
VTSS_DEBUG_GROUP_SER_GPIO	Serial GPIO configuration
VTSS_DEBUG_GROUP_L3	L3 services
VTSS_DEBUG_GROUP_AFI	Automatic Frame Injector
VTSS_DEBUG_GROUP_MACSEC	802.1AE MacSec
VTSS_DEBUG_GROUP_COUNT	Number of groups

Definition at line 184 of file vtss_misc_api.h.

4.19.5 Function Documentation

4.19.5.1 vtss_trace_conf_get()

```
vtss_rc vtss_trace_conf_get (
            const vtss_trace_group_t group,
            vtss_trace_conf_t *const conf )
```

Get trace configuration.

Parameters

group	[IN] Trace group
conf	[OUT] Trace group configuration.

Returns

Return code.

4.19.5.2 vtss_trace_conf_set()

```
vtss_rc vtss_trace_conf_set (
            const vtss_trace_group_t group,
            const vtss_trace_conf_t *const conf )
```

Set trace configuration.

Parameters

group	[IN] Trace group
conf	[IN] Trace group configuration.

Returns

Return code.

4.19.5.3 vtss_callout_trace_printf()

```
void vtss_callout_trace_printf (
            const vtss_trace_layer_t layer,
            const vtss_trace_group_t group,
            const vtss_trace_level_t level,
            const char * file,
            const int line,
            const char * function,
            const char * format,
             ...  )
```

Trace callout function.

Parameters

layer	[IN] Trace layer
group	[IN] Trace group
level	[IN] Trace level
file	[IN] File name string
line	[IN] Line number in file
function	[IN] Function name string
format	[IN] Print format string

Returns

Nothing.

4.19.5.4 vtss_callout_trace_hex_dump()

```
void vtss_callout_trace_hex_dump (
            const vtss_trace_layer_t layer,
            const vtss_trace_group_t group,
            const vtss_trace_level_t level,
            const char * file,
            const int line,
            const char * function,
            const u8 * byte_p,
            const int byte_cnt )
```

Trace hex-dump callout function.

Parameters

layer	[IN] Trace layer
group	[IN] Trace group
level	[IN] Trace level
file	[IN] The file from where the trace were called.
line	[IN] The line from where the trace were called.
function	[IN] The function from where the trace were called.
byte_p	[IN] Pointer to start of area to print
byte_cnt	[IN] Number of bytes to print

Returns

Nothing.

4.19.5.5 vtss_debug_info_get()

```
vtss_rc vtss_debug_info_get (
            vtss_debug_info_t *const info )
```

Get default debug information structure.

Parameters

info	[OUT] Debug information

Returns

Return code.

4.19.5.6 vtss_debug_info_print()

```
vtss_rc vtss_debug_info_print (
            const vtss_inst_t inst,
            const vtss_debug_printf_t prntf,
            const vtss_debug_info_t *const info )
```

Print default information.

Parameters

inst	[IN] Target instance reference.
prntf	[IN] Debug printf function.
info	[IN] Debug information

Returns

Return code.

4.19.5.7 vtss_callout_lock()

```
void vtss_callout_lock (
             const vtss_api_lock_t *const lock )
```

Lock API access.

Parameters

lock	[IN] Lock information

4.19.5.8 vtss_callout_unlock()

```
void vtss_callout_unlock (
             const vtss_api_lock_t *const lock )
```

Unlock API access.

Parameters

lock	[IN] Lock information

4.19.5.9 vtss_debug_lock()

```
vtss_rc vtss_debug_lock (
             const vtss_inst_t inst,
             const vtss_debug_lock_t *const lock )
```

Debug lock API access.

Parameters

inst	[IN] Target instance reference.
lock	[IN] Lock information

Returns

Return code.

4.19.5.10 vtss_debug_unlock()

```
vtss_rc vtss_debug_unlock (
            const vtss_inst_t inst,
            vtss_debug_lock_t *const lock )
```

Debug unlock API access.

Parameters

inst	[IN] Target instance reference.
lock	[IN] Lock information

Returns

Return code.

4.19.5.11 vtss_intr_cfg()

```
vtss_rc vtss_intr_cfg (
            const vtss_inst_t inst,
            const u32 mask,
            const BOOL polarity,
            const BOOL enable )
```

Configure interrupt.

Parameters

inst	[IN] Target instance reference.
mask	[IN] Interrupt mask - Configures the interrupts for the bits set in the mask.
polarity	[IN] Polarity - Interrupt polarity.
enable	[IN] Enable - 1 = enable, 0 = disable.

Returns

Return code.

4.19.5.12 vtss_tod_get_ns_cnt()

```
u32 vtss_tod_get_ns_cnt (
            void  )
```

Get the current hw nanosec time This function is called from interrupt.

Returns

actual ns counter

4.19.5.13 vtss_tod_set_ns_cnt_cb()

```
void vtss_tod_set_ns_cnt_cb (
             tod_get_ns_cnt_cb_t cb )
```

Set an external hw nanosec read function.

Parameters

cb	pointer to callback function

4.19.5.14 vtss_debug_reg_check_set()

```
vtss_rc vtss_debug_reg_check_set (
             const vtss_inst_t inst,
             const BOOL enable )
```

Enable or disable register access checking.

When enabled, every call to the register read/write callouts (init_conf.reg_read()/write()) will be verified for success. This will slow-down execution, so it is recommended only to use this feature in dedicated debug builds.

In multi-chip targets, only chip number 0 will be verified.

The number of calls will be reference counted, according to the following rules: 1) Calls with enable = FALSE will increase the reference count. 2) Calls with enable = TRUE will decrease the reference count, which cannot go lower than 0. The reference count is initialized to 1 at API instantiation, effectively disabling register access checking. A reference count of 0 enables the feature.

Error indications get printed with a call to VTSS_EG(VTSS_TRACE_GROUP_REG_CHECK, ...), which will request to be interrupt/scheduler disabled tolerant.

Notice that this feature may not be available on all platforms.

This feature will not work if the API is instantiated more than once, since it uses the default instance to find its state.

Parameters

inst	[IN] Target instance reference.
enable	[IN] Enable or disable register access checking (ref. counted).

Returns

Return code.

4.20 vtss_api/include/vtss_mpls_api.h File Reference

MPLS API.

```
#include <vtss/api/types.h>
```

4.20.1 Detailed Description

MPLS API.

This header file describes the MPLS functions

4.21 vtss_api/include/vtss_oam_api.h File Reference

OAM API.

```
#include <vtss/api/types.h>
```

4.21.1 Detailed Description

OAM API.

This header file describes Y.1731/IEEE802.1ag OAM functions.

4.22 vtss_api/include/vtss_oha_api.h File Reference

OHA API.

```
#include <vtss/api/types.h>
```

4.22.1 Detailed Description

OHA API.

4.23 vtss_api/include/vtss_os.h File Reference

OS Layer API.

```
#include <vtss_os_linux.h>
```

4.23.1 Detailed Description

OS Layer API.

This header file includes the OS specific header file

4.24 vtss_api/include/vtss_os_custom.h File Reference

OS custom header file.

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

Macros

- #define uint unsigned int
- #define ulong unsigned long
- #define VTSS_MSLEEP(msec) <your function>
- #define VTSS_MTIMER_START(pTimer, msec) <your impl>
- #define VTSS_MTIMER_TIMEOUT(pTimer) <your impl>
- #define VTSS_MTIMER_CANCEL(pTimer) <your impl>
- #define VTSS_DIV64(dividend, divisor) <your impl>
- #define VTSS_MOD64(dividend, divisor) <your impl>
- #define VTSS_LABS(arg) <your impl>
- #define VTSS_LLABS(arg) <your impl>
- #define VTSS_OS_CTZ(val32) <your impl>
- #define VTSS_OS_CTZ64(val64) <your impl>
- #define VTSS_OS_MALLOC(size, flags) <your impl>
- #define VTSS_OS_FREE(ptr, flags) <your impl>
- #define VTSS_OS_RAND() <your impl>

Typedefs

- typedef int vtss_mtimer_t

4.24.1 Detailed Description

OS custom header file.

This file is a skeleton to be replaced by for customer specific OS.

4.24.2 Macro Definition Documentation

4.24.2.1 uint

```
#define uint unsigned int
```

Define API uint type - unsigned 16 bits

Definition at line 36 of file vtss_os_custom.h.

4.24.2.2 ulong

```
#define ulong unsigned long
```

Define API ulong - unsigned 32 bits

Definition at line 37 of file vtss_os_custom.h.

4.24.2.3 VTSS_MSLEEP

```
#define VTSS_MSLEEP(
              msec ) <your function>
```

Your function for sleeping for "msec" milli seconds

Definition at line 40 of file vtss_os_custom.h.

4.24.2.4 VTSS_MTIMER_START

```
#define VTSS_MTIMER_START(
              pTimer,
              msec ) <your impl>
```

Start the timer (pTimer) with a timeout after a number of milliseconds

Definition at line 44 of file vtss_os_custom.h.

4.24.2.5 VTSS_MTIMER_TIMEOUT

```
#define VTSS_MTIMER_TIMEOUT(
              pTimer ) <your impl>
```

Check if timer has timed out (Return TRUE in case of timeout else FALSE).

Definition at line 45 of file vtss_os_custom.h.

4.24.2.6 VTSS_MTIMER_CANCEL

```
#define VTSS_MTIMER_CANCEL(
              pTimer ) <your impl>
```

Stop the timer

Definition at line 46 of file vtss_os_custom.h.

4.24.2.7 VTSS_DIV64

```
#define VTSS_DIV64(
              dividend,
              divisor ) <your impl>
```

support for 64 bit division

Definition at line 48 of file vtss_os_custom.h.

4.24.2.8 VTSS_MOD64

```
#define VTSS_MOD64(
              dividend,
              divisor ) <your impl>
```

support for 64 bit division

Definition at line 49 of file vtss_os_custom.h.

4.24.2.9 VTSS_LABS

```
#define VTSS_LABS(
              arg ) <your impl>
```

long to abs

Definition at line 50 of file vtss_os_custom.h.

4.24.2.10 VTSS_LLABS

```
#define VTSS_LLABS(
              arg ) <your impl>
```

long long to abs

Definition at line 51 of file vtss_os_custom.h.

4.24.2.11 VTSS_OS_CTZ

```
#define VTSS_OS_CTZ(
              val32 ) <your impl>
```

Count trailing zeros of a 32-bit unsigned. Requirements/examples: VTSS_OS_CTZ(0x00000001) = 0 VTSS_OS_CTZ(0x80000000) = 31 VTSS_OS_CTZ(0x00000000) >= 32 (if result is taken as unsigned; Most implementations return -1, and (u32)(-1) >= 32).

__builtin_ctz() is included in GCC 3.2.2 and later according to http://en.wikipedia.org/wiki/Find_first_set.

Definition at line 62 of file vtss_os_custom.h.

4.24.2.12 VTSS_OS_CTZ64

```
#define VTSS_OS_CTZ64(
              val64 ) <your impl>
```

Count trailing zeros of a 64-bit unsigned. Requirements/examples: VTSS_OS_CTZ64(0x00000000_00000001) = 0 VTSS_OS_CTZ64(0x00000000_80000000) = 31 VTSS_OS_CTZ64(0x00000001_00000000) = 32 VTSS_OS_CTZ64(0x80000000_00000000) = 63 VTSS_OS_CTZ64(0x00000000_00000000) >= 64 (if result is taken as unsigned; Most implementations return -1, and (u32)(-1) >= 64).

Definition at line 73 of file vtss_os_custom.h.

4.24.2.13 VTSS_OS_MALLOC

```
#define VTSS_OS_MALLOC(
              size,
              flags ) <your impl>
```

Request OS to allocate size bytes of memory.

The first argument is the number of bytes that must be allocated. Type is size_t.

The second argument is a mask of flags that the implementation must obey. Type is vtss_mem_flags_t.

The returned pointer should be at least 8-byte aligned, to make it suitable for a struct.

Definition at line 87 of file vtss_os_custom.h.

4.24.2.14 VTSS_OS_FREE

```
#define VTSS_OS_FREE(
              ptr,
              flags ) <your impl>
```

Request OS to free memory previously allocated with VTSS_OS_MALLOC().

The first argument is the pointer previously obtained with a call to VTSS_OS_MALLOC(). Type is void *.

The second argument is a mask of flags identical to those passed to VTSS_OS_MALLOC() when the memory was requested.

Definition at line 101 of file vtss_os_custom.h.

4.24.2.15 VTSS_OS_RAND

```
#define VTSS_OS_RAND( ) <your impl>
```

Wrap of call to rand() defined in stdlib.h

Definition at line 106 of file vtss_os_custom.h.

4.24.3 Typedef Documentation

4.24.3.1 vtss_mtimer_t

```
typedef int vtss_mtimer_t
```

Timer

Definition at line 43 of file vtss_os_custom.h.

4.25 vtss_api/include/vtss_os_ecos.h File Reference

eCos OS API

```
#include "vtss/api/types.h"
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <sys/types.h>
#include <cyg/kernel/kapi.h>
#include <cyg/hal/hal_cache.h>
#include <cyg/hal/hal_arch.h>
#include <cyg/hal/hal_endian.h>
```

Data Structures

- struct vtss_timeofday_t

  Time of day structure.

Macros

- #define VTSS_MSLEEP(msec) HAL_DELAY_US(msec∗1000)
- #define VTSS_NSLEEP(nsec) HAL_DELAY_US((nsec)/1000)
- #define VTSS_MTIMER_START(pTimer, msec) ∗pTimer = cyg_current_time() + ((msec)/10) + 1
- #define VTSS_MTIMER_TIMEOUT(pTimer) (cyg_current_time() > ∗(pTimer))
- #define VTSS_MTIMER_CANCEL(pTimer)
- #define VTSS_TIME_OF_DAY(tod) (tod.sec = (cyg_current_time() / CYGNUM_HAL_RTC_DENOMINAT↩ OR))
- #define VTSS_DIV64(dividend, divisor) ((dividend) / (divisor))
- #define VTSS_MOD64(dividend, divisor) ((dividend) % (divisor))
- #define VTSS_LABS(arg) labs(arg)
- #define VTSS_LLABS(arg) llabs(arg)
- #define VTSS_OS_CTZ(val32) ((val32) == 0 ? 32 : __builtin_ctz(val32))
- #define VTSS_OS_CTZ64(val64) ((val64) == 0 ? 64 : __builtin_ctzll(val64))
- #define VTSS_OS_MALLOC(size, flags) vtss_callout_malloc(size, flags)
- #define VTSS_OS_FREE(ptr, flags) vtss_callout_free(ptr, flags)
- #define VTSS_OS_RAND() rand()
- #define VTSS_OS_REORDER_BARRIER() HAL_REORDER_BARRIER()
- #define VTSS_OS_COMPILER_ATTRIBUTE_ALIGNED(x) __attribute ((aligned(x)))
- #define VTSS_OS_DCACHE_LINE_SIZE_BYTES HAL_DCACHE_LINE_SIZE
- #define VTSS_OS_DCACHE_INVALIDATE(virt_addr, size) HAL_DCACHE_INVALIDATE(virt_addr, size)
- #define VTSS_OS_DCACHE_FLUSH(virt_addr, size) HAL_DCACHE_STORE(virt_addr, size)
- #define VTSS_OS_VIRT_TO_PHYS(addr) (u32)CYGARC_PHYSICAL_ADDRESS(addr)
- #define VTSS_OS_BIG_ENDIAN

  VTSS_OS_BIG_ENDIAN: If undefined, we're running little endian. If defined we're running big endian.
- #define VTSS_OS_NTOHL(x) (x)
- #define VTSS_OS_SCHEDULER_FLAGS cyg_uint32 __attribute__((unused))
- #define VTSS_OS_SCHEDULER_LOCK(flags) cyg_scheduler_lock(__FILE__, __LINE__)
- #define VTSS_OS_SCHEDULER_UNLOCK(flags) cyg_scheduler_unlock(__FILE__, __LINE__)
- #define VTSS_OS_INTERRUPT_FLAGS NOT_NEEDED
- #define VTSS_OS_INTERRUPT_DISABLE(flags) NOT_NEEDED
- #define VTSS_OS_INTERRUPT_RESTORE(flags) NOT_NEEDED

Typedefs

- typedef cyg_tick_count_t vtss_mtimer_t

Functions

- long long int llabs (long long int val)

  Obtain the absolute value of a long long integer.
- void * vtss_callout_malloc (size_t size, vtss_mem_flags_t flags)

  Callout to allocate memory.
- void vtss_callout_free (void *ptr, vtss_mem_flags_t flags)

  Callout to free memory.

4.25.1 Detailed Description

eCos OS API

This header file describes OS functions for eCos

4.25.2 Macro Definition Documentation

4.25.2.1 VTSS_MSLEEP

```
#define VTSS_MSLEEP(
              msec ) HAL_DELAY_US(msec*1000)
```

Sleep for "msec" milliseconds

Definition at line 50 of file vtss_os_ecos.h.

4.25.2.2 VTSS_NSLEEP

```
#define VTSS_NSLEEP(
              nsec ) HAL_DELAY_US((nsec)/1000)
```

Sleep for "nsec" nanoseconds

Definition at line 51 of file vtss_os_ecos.h.

4.25.2.3 VTSS_MTIMER_START

```
#define VTSS_MTIMER_START(
              pTimer,
              msec ) *pTimer = cyg_current_time() + ((msec)/10) + 1
```

Starting Timer

Definition at line 54 of file vtss_os_ecos.h.

4.25.2.4 VTSS_MTIMER_TIMEOUT

```
#define VTSS_MTIMER_TIMEOUT(
              pTimer ) (cyg_current_time() > *(pTimer))
```

Timer timeout

Definition at line 55 of file vtss_os_ecos.h.

4.25.2.5 VTSS_MTIMER_CANCEL

```
#define VTSS_MTIMER_CANCEL(
              pTimer )
```

No action in this implementation.

Definition at line 56 of file vtss_os_ecos.h.

4.25.2.6 VTSS_TIME_OF_DAY

```
#define VTSS_TIME_OF_DAY(
              tod ) (tod.sec = (cyg_current_time() / CYGNUM_HAL_RTC_DENOMINATOR))
```

Time of day macro

Definition at line 63 of file vtss_os_ecos.h.

4.25.2.7 VTSS_DIV64

```
#define VTSS_DIV64(
              dividend,
              divisor ) ((dividend) / (divisor))
```

support for 64 bit division

Definition at line 73 of file vtss_os_ecos.h.

4.25.2.8 VTSS_MOD64

```
#define VTSS_MOD64(
              dividend,
              divisor ) ((dividend) % (divisor))
```

support for 64 bit division

Definition at line 74 of file vtss_os_ecos.h.

4.25.2.9 VTSS_LABS

```
#define VTSS_LABS(
              arg ) labs(arg)
```

long to abs

Definition at line 75 of file vtss_os_ecos.h.

4.25.2.10 VTSS_LLABS

```
#define VTSS_LLABS(
              arg ) llabs(arg)
```

long long to abs

Definition at line 76 of file vtss_os_ecos.h.

4.25.2.11 VTSS_OS_CTZ

```
#define VTSS_OS_CTZ(
              val32 ) ((val32) == 0 ?  32 :  __builtin_ctz(val32))
```

Count trailing zeros of a 32-bit unsigned. Requirements/examples: VTSS_OS_CTZ(0x00000001) = 0 VTSS_OS_CTZ(0x80000000) = 31 VTSS_OS_CTZ(0x00000000) >= 32 (if result is taken as unsigned; Most implementations return -1, and (u32)(-1) >= 32).

__builtin_ctz() is included in GCC 3.2.2 and later according to http://en.wikipedia.org/wiki/Find_first_set.

Note: __builtin_ctz() is undefined for zero input values.

Definition at line 89 of file vtss_os_ecos.h.

4.25.2.12 VTSS_OS_CTZ64

```
#define VTSS_OS_CTZ64(
              val64 ) ((val64) == 0 ?  64 :  __builtin_ctzll(val64))
```

Count trailing zeros of a 64-bit unsigned. Requirements/examples: VTSS_OS_CTZ64(0x00000000_00000001) = 0 VTSS_OS_CTZ64(0x00000000_80000000) = 31 VTSS_OS_CTZ64(0x00000001_00000000) = 32 VTSS_OS_CTZ64(0x80000000_00000000) = 63 VTSS_OS_CTZ64(0x00000000_00000000) >= 64 (if result is taken as unsigned; Most implementations return -1, and (u32)(-1) >= 64).

Note: __builtin_ctzll() is undefined for zero input values.

Definition at line 102 of file vtss_os_ecos.h.

4.25.2.13 VTSS_OS_MALLOC

```
#define VTSS_OS_MALLOC(
              size,
              flags ) vtss_callout_malloc(size, flags)
```

Request OS to allocate size bytes of memory.

The first argument is the number of bytes that must be allocated. Type is size_t.

The second argument is a mask of flags that the implementation must obey. Type is vtss_mem_flags_t.

The returned pointer should be at least 8-byte aligned, to make it suitable for a struct.

Definition at line 135 of file vtss_os_ecos.h.

4.25.2.14 VTSS_OS_FREE

```
#define VTSS_OS_FREE(
              ptr,
              flags ) vtss_callout_free(ptr, flags)
```

Request OS to free memory previously allocated with VTSS_OS_MALLOC().

The first argument is the pointer previously obtained with a call to VTSS_OS_MALLOC(). Type is void *.

The second argument is a mask of flags identical to those passed to VTSS_OS_MALLOC() when the memory was requested.

Definition at line 149 of file vtss_os_ecos.h.

4.25.2.15 VTSS_OS_RAND

```
#define VTSS_OS_RAND( ) rand()
```

Wrap of call to rand() defined in stdlib.h

Definition at line 154 of file vtss_os_ecos.h.

4.25.2.16 VTSS_OS_REORDER_BARRIER

```
#define VTSS_OS_REORDER_BARRIER( ) HAL_REORDER_BARRIER()
```

The compiler may swap instructions to optimize performance of the final code (size- or speed-wise). When configuration of hardware is involved, it may not always be valid to swap two statements. Consider for instance the following two writes to two FDMA registers: Write the source address to the FDMA Enable the FDMA. To the compiler, these two writes can be executed in any order and still semantically yield the correct result, had it been normal RAM they were written to. But since they are written to actual hardware, it is crucial that they are executed in the correct order. The VTSS_OS_REORDER_BARRIER() macro should implement code that ensures that the compiler doesn't optimize across the barrier.

Definition at line 173 of file vtss_os_ecos.h.

4.25.2.17 VTSS_OS_COMPILER_ATTRIBUTE_ALIGNED

```
#define VTSS_OS_COMPILER_ATTRIBUTE_ALIGNED(
             x ) __attribute ((aligned(x)))
```

In some special cases, it is of utmost importance that a certain variable has a certain memory alignment. Applications for this is e.g. placing variables on cache-line boundaries.

Definition at line 180 of file vtss_os_ecos.h.

4.25.2.18 VTSS_OS_DCACHE_LINE_SIZE_BYTES

```
#define VTSS_OS_DCACHE_LINE_SIZE_BYTES HAL_DCACHE_LINE_SIZE
```

The number of bytes one DCache-line is made up of.

Definition at line 189 of file vtss_os_ecos.h.

4.25.2.19 VTSS_OS_DCACHE_INVALIDATE

```
#define VTSS_OS_DCACHE_INVALIDATE(
              virt_addr,
              size ) HAL_DCACHE_INVALIDATE(virt_addr, size)
```

Invalidate @size bytes at virtual address @virt_addr of the DCache. After invalidation, the invalidated area will be fetched from RAM.

Definition at line 195 of file vtss_os_ecos.h.

4.25.2.20 VTSS_OS_DCACHE_FLUSH

```
#define VTSS_OS_DCACHE_FLUSH(
              virt_addr,
              size ) HAL_DCACHE_STORE(virt_addr, size)
```

Force a write of @size bytes of dirty cache lines to RAM starting at virtual address @virt_addr.

Definition at line 201 of file vtss_os_ecos.h.

4.25.2.21 VTSS_OS_VIRT_TO_PHYS

```
#define VTSS_OS_VIRT_TO_PHYS(
              addr ) (u32)CYGARC_PHYSICAL_ADDRESS(addr)
```

Macro that implements the conversion from a virtual to a physical address. In OSs with a flat memory layout, this could be as simple as (u32)(addr).

Definition at line 208 of file vtss_os_ecos.h.

4.25.2.22 VTSS_OS_BIG_ENDIAN

```
#define VTSS_OS_BIG_ENDIAN
```

VTSS_OS_BIG_ENDIAN: If undefined, we're running little endian. If defined we're running big endian.

We're running big endian

Definition at line 221 of file vtss_os_ecos.h.

4.25.2.23 VTSS_OS_NTOHL

```
#define VTSS_OS_NTOHL(
              x ) (x)
```

Convert from network to host order

Definition at line 222 of file vtss_os_ecos.h.

4.25.2.24 VTSS_OS_SCHEDULER_FLAGS

```
#define VTSS_OS_SCHEDULER_FLAGS cyg_uint32 __attribute__((unused))
```

VTSS_OS_SCHEDULER_FLAGS VTSS_OS_SCHEDULER_LOCK(flags) VTSS_OS_SCHEDULER_UNLOCK(flags) These functions are called by API code that consists of a user-level part and a deferred interrupt handler part. Only the user-level part will call the VTSS_OS_SCHEDULER_LOCK()/UNLOCK() functions, since it is assumed that the deferred interrupt handler will have atomic access throughout its execution. Each module within the API that contains such functionality will have an option to call either the VTSS_OS_SCHEDULER_(UN)LOCK() functions or the VTSS_OS_INTERRUPT_DISABLE()/RESTORE() functions. The **attribute**((unused)) ensures that we don't get compiler warnings.

Definition at line 248 of file vtss_os_ecos.h.

4.25.2.25 VTSS_OS_SCHEDULER_LOCK

```
#define VTSS_OS_SCHEDULER_LOCK(
              flags ) cyg_scheduler_lock(__FILE__, __LINE__)
```

Lock scheduler.

Definition at line 252 of file vtss_os_ecos.h.

4.25.2.26 VTSS_OS_SCHEDULER_UNLOCK

```
#define VTSS_OS_SCHEDULER_UNLOCK(
              flags ) cyg_scheduler_unlock(__FILE__, __LINE__)
```

Unlock scheduler.

Definition at line 256 of file vtss_os_ecos.h.

4.25.2.27 VTSS_OS_INTERRUPT_FLAGS

```
#define VTSS_OS_INTERRUPT_FLAGS NOT_NEEDED
```

VTSS_OS_INTERRUPT_FLAGS VTSS_OS_INTERRUPT_DISABLE(flags) VTSS_OS_INTERRUPT_RESTORE(flags) These functions are called by API code that consists of a user-level part and an interrupt handler part executing directly in interrupt context. Only the user-level part will call the VTSS_OS_INTERRUPT_DISABLE()/RESTORE() functions, since it is assumed that the interrupt handler will have atomic access throughout its execution. Each module within the API that contains such functionality will have an option to call either the VTSS_OS_SCHED↩ ULER_(UN)LOCK() functions or the VTSS_OS_INTERRUPT_DISABLE()/RESTORE() functions. Not needed in eCos, since all interrupt handlers will be called in deferred context.

Definition at line 272 of file vtss_os_ecos.h.

4.25.2.28 VTSS_OS_INTERRUPT_DISABLE

```
#define VTSS_OS_INTERRUPT_DISABLE(
              flags ) NOT_NEEDED
```

Disable interrupts.

Definition at line 276 of file vtss_os_ecos.h.

4.25.2.29 VTSS_OS_INTERRUPT_RESTORE

```
#define VTSS_OS_INTERRUPT_RESTORE(
              flags ) NOT_NEEDED
```

Enable interrupts.

Definition at line 280 of file vtss_os_ecos.h.

4.25.3 Typedef Documentation

4.25.3.1 vtss_mtimer_t

```
typedef cyg_tick_count_t vtss_mtimer_t
```

Timer

Definition at line 53 of file vtss_os_ecos.h.

4.25.4 Function Documentation

4.25.4.1 llabs()

```
long long int llabs (
             long long int val )
```

Obtain the absolute value of a long long integer.

Parameters

val	[IN] The value to convert to absolute value.

Returns

The absolute value of val.

4.25.4.2 vtss_callout_malloc()

```
void* vtss_callout_malloc (
            size_t size,
            vtss_mem_flags_t flags )
```

Callout to allocate memory.

[IN]/[OUT] seen from called function.

Parameters

size	[IN] Number of bytes to allocate.
flags	[IN] See vtss_mem_flags_t for details.

Returns

Pointer to allocated area.

4.25.4.3 vtss_callout_free()

```
void vtss_callout_free (
            void * ptr,
            vtss_mem_flags_t flags )
```

Callout to free memory.

[IN]/[OUT] seen from called function.

Parameters

ptr	[IN] Pointer previously obtained with call to vtss_callout_malloc().
flags	[IN] See vtss_mem_flags_t for details.

4.26 vtss_api/include/vtss_os_linux.h File Reference

Linux OS API.

```
#include <endian.h>
#include <asm/byteorder.h>
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>
#include <errno.h>
#include <time.h>
#include <sys/time.h>
```

Data Structures

- struct vtss_mtimer_t

  Timer structure.
- struct vtss_timeofday_t

  Time of day structure.

Macros

- #define VTSS_OS_BIG_ENDIAN

  VTSS_OS_BIG_ENDIAN: If undefined, we're running little endian. If defined we're running big endian.
- #define VTSS_OS_NTOHL(x) __be32_to_cpu(x)
- #define VTSS_NSLEEP(nsec)
- #define VTSS_MSLEEP(msec)
- #define VTSS_MTIMER_START(timer, msec)
- #define VTSS_MTIMER_TIMEOUT(timer) (gettimeofday(&((timer)->now),NULL)==0 && timercmp(&((timer)->now),&((timer)->timeout),>))
- #define VTSS_MTIMER_CANCEL(timer)
- #define VTSS_TIME_OF_DAY(tod)
- #define VTSS_OS_SCHEDULER_FLAGS int
- #define VTSS_OS_SCHEDULER_LOCK(flags) do {flags = flags;} while (0);
- #define VTSS_OS_SCHEDULER_UNLOCK(flags) do {flags = flags;} while (0);
- #define VTSS_DIV64(dividend, divisor) ((dividend) / (divisor))
- #define VTSS_MOD64(dividend, divisor) ((dividend) % (divisor))
- #define VTSS_LABS(arg) labs(arg)
- #define VTSS_LLABS(arg) llabs(arg)
- #define VTSS_OS_CTZ(val32) ((val32) == 0 ? 32 : __builtin_ctzl((unsigned long)val32))
- #define VTSS_OS_CTZ64(val64)
- #define VTSS_OS_MALLOC(size, flags) malloc(size)
- #define VTSS_OS_FREE(ptr, flags) free(ptr)
- #define VTSS_OS_RAND() rand()

4.26.1 Detailed Description

Linux OS API.

This header file describes OS functions for Linux

4.26.2 Macro Definition Documentation

4.26.2.1 VTSS_OS_BIG_ENDIAN

```
#define VTSS_OS_BIG_ENDIAN
```

VTSS_OS_BIG_ENDIAN: If undefined, we're running little endian. If defined we're running big endian.

We're running big endian

Definition at line 46 of file vtss_os_linux.h.

4.26.2.2 VTSS_OS_NTOHL

```
#define VTSS_OS_NTOHL(
              x ) __be32_to_cpu(x)
```

Convert a 32-bit value from network to host order

Definition at line 49 of file vtss_os_linux.h.

4.26.2.3 VTSS_NSLEEP

```
#define VTSS_NSLEEP(
              nsec )
```

Value:

```
{                                       \
    struct timespec ts;                                         \
    ts.tv_sec = 0;                                              \
    ts.tv_nsec = nsec;                                          \
    while(nanosleep(&ts, &ts) == -1 && errno == EINTR) {        \
    }                                                           \
}
```

Sleep for

Parameters

nsec	nanoseconds

Definition at line 69 of file vtss_os_linux.h.

4.26.2.4 VTSS_MSLEEP

```
#define VTSS_MSLEEP(
              msec )
```

Value:

```
{                                   \
    struct timespec ts;                                     \
    ts.tv_sec = msec / 1000;                                \
    ts.tv_nsec = (msec % 1000) * 1000000;                   \
    while(nanosleep(&ts, &ts) == -1 && errno == EINTR) {    \
    }                                                       \
}
```

Sleep for

Parameters

msec	milliseconds

Definition at line 78 of file vtss_os_linux.h.

4.26.2.5 VTSS_MTIMER_START

```
#define VTSS_MTIMER_START(
              timer,
              msec )
```

Value:

```
{ \
    (void) gettimeofday(&((timer)->timeout),NULL);   \
    (timer)->timeout.tv_usec+=msec*1000; \
    if ((timer)->timeout.tv_usec>=1000000) { (timer)->timeout.tv_sec+=(timer)->timeout.tv_usec/1000000; (
      timer)->timeout.tv_usec%=1000000; } \
}
```

Start timer

Definition at line 93 of file vtss_os_linux.h.

4.26.2.6 VTSS_MTIMER_TIMEOUT

```
#define VTSS_MTIMER_TIMEOUT(
              timer ) (gettimeofday(&((timer)->now),NULL)==0 && timercmp(&((timer)->now),&((timer)->timeout),
```

Timer timeout

Definition at line 103 of file vtss_os_linux.h.

4.26.2.7 VTSS_MTIMER_CANCEL

```
#define VTSS_MTIMER_CANCEL(
              timer )
```

No action in this implementation. Time of day struct

Definition at line 105 of file vtss_os_linux.h.

4.26.2.8 VTSS_TIME_OF_DAY

```
#define VTSS_TIME_OF_DAY(
              tod )
```

Value:

```
{ \
    struct timeval tve; \
    (void)gettimeofday(&tve,NULL);  \
    tod.sec = tve.tv_sec; \
}
```

Time of day macro

Definition at line 112 of file vtss_os_linux.h.

4.26.2.9 VTSS_OS_SCHEDULER_FLAGS

```
#define VTSS_OS_SCHEDULER_FLAGS int
```

VTSS_OS_SCHEDULER_FLAGS VTSS_OS_SCHEDULER_LOCK(flags) VTSS_OS_SCHEDULER_UNLOCK(flags) These functions are called by API code that consists of a user-level part and a deferred interrupt handler part. Only the user-level part will call the VTSS_OS_SCHEDULER_LOCK()/UNLOCK() functions, since it is assumed that the deferred interrupt handler will have atomic access throughout its execution. Each module within the API that contains such functionality will have an option to call either the VTSS_OS_SCHEDULER_(UN)LOCK() functions or the VTSS_OS_INTERRUPT_DISABLE()/RESTORE() functions.

Definition at line 138 of file vtss_os_linux.h.

4.26.2.10 VTSS_OS_SCHEDULER_LOCK

```
#define VTSS_OS_SCHEDULER_LOCK(
              flags ) do {flags = flags;} while (0);
```

Lock scheduler

Definition at line 139 of file vtss_os_linux.h.

4.26.2.11 VTSS_OS_SCHEDULER_UNLOCK

```
#define VTSS_OS_SCHEDULER_UNLOCK(
              flags ) do {flags = flags;} while (0);
```

Unlock scheduler

Definition at line 140 of file vtss_os_linux.h.

4.26.2.12 VTSS_DIV64

```
#define VTSS_DIV64(
              dividend,
              divisor ) ((dividend) / (divisor))
```

VTSS_DIV64 - perform 64/32 bit division yelding 32 bit (at least) output

Definition at line 145 of file vtss_os_linux.h.

4.26.2.13 VTSS_MOD64

```
#define VTSS_MOD64(
              dividend,
              divisor ) ((dividend) % (divisor))
```

VTSS_MOD64 - perform 64/32 bit modulus yelding 32 bit (at least) output

Definition at line 149 of file vtss_os_linux.h.

4.26.2.14 VTSS_LABS

```
#define VTSS_LABS(
              arg ) labs(arg)
```

VTSS_LABS - perform abs() on long

Definition at line 153 of file vtss_os_linux.h.

4.26.2.15 VTSS_LLABS

```
#define VTSS_LLABS(
              arg ) llabs(arg)
```

VTSS_LLABS - perform abs() on long long

Definition at line 158 of file vtss_os_linux.h.

4.26.2.16 VTSS_OS_CTZ

```
#define VTSS_OS_CTZ(
              val32 ) ((val32) == 0 ?  32 :  __builtin_ctzl((unsigned long)val32))
```

VTSS_OS_CTZ(val32)

Count trailing zeros of a 32-bit unsigned. Requirements/examples: VTSS_OS_CTZ(0x00000001) = 0 VTSS_OS_CTZ(0x80000000) = 31 VTSS_OS_CTZ(0x00000000) >= 32 (if result is taken as unsigned; Most implementations return -1, and (u32)(-1) >= 32).

Parameters

val32	The value to decode

Returns

Number of trailing zeroes - or - the bit index of the lowest bit set in the input given.

Note

__builtin_ctz() is included in GCC 3.2.2 and later according to http://en.wikipedia.org/wiki/Find_first_set.

Note: __builtin_ctzl() is undefined for zero input values.

Definition at line 370 of file vtss_os_linux.h.

4.26.2.17 VTSS_OS_CTZ64

```
#define VTSS_OS_CTZ64(
              val64 )
```

Value:

```
({                                  \
    u32 _r = VTSS_OS_CTZ((u32)(val64));                    \
    (val64) == 0 ? 64 :                                    \
    _r < 32 ? _r : 32 + VTSS_OS_CTZ((u32)((val64) >> 32)); \
})
```

Count trailing zeros of a 64-bit unsigned. Requirements/examples: VTSS_OS_CTZ64(0x00000000_00000001) = 0 VTSS_OS_CTZ64(0x00000000_80000000) = 31 VTSS_OS_CTZ64(0x00000001_00000000) = 32 VTSS_OS_CTZ64(0x80000000_00000000) = 63 VTSS_OS_CTZ64(0x00000000_00000000) >= 64 (if result is taken as unsigned; Most implementations return -1, and (u32)(-1) >= 64).

Definition at line 381 of file vtss_os_linux.h.

4.26.2.18 VTSS_OS_MALLOC

```
#define VTSS_OS_MALLOC(
              size,
              flags ) malloc(size)
```

Request OS to allocate size bytes of memory.

The first argument is the number of bytes that must be allocated. Type is size_t.

The second argument is a mask of flags that the implementation must obey. Type is vtss_mem_flags_t.

The returned pointer should be at least 8-byte aligned, to make it suitable for a struct.

Definition at line 425 of file vtss_os_linux.h.

4.26.2.19 VTSS_OS_FREE

```
#define VTSS_OS_FREE(
              ptr,
              flags ) free(ptr)
```

Request OS to free memory previously allocated with VTSS_OS_MALLOC().

The first argument is the pointer previously obtained with a call to VTSS_OS_MALLOC(). Type is void *.

The second argument is a mask of flags identical to those passed to VTSS_OS_MALLOC() when the memory was requested.

Definition at line 443 of file vtss_os_linux.h.

4.26.2.20 VTSS_OS_RAND

```
#define VTSS_OS_RAND( ) rand()
```

Wrap of call to rand() defined in stdlib.h

Definition at line 449 of file vtss_os_linux.h.

4.27 vtss_api/include/vtss_otn_api.h File Reference

OTN API.

```
#include <vtss/api/types.h>
```

4.27.1 Detailed Description

OTN API.

4.28 vtss_api/include/vtss_packet_api.h File Reference

Packet API.

```
#include <vtss/api/types.h>
#include <vtss_l2_api.h>
```

4.28.1 Detailed Description

Packet API.

This header file describes CPU Rx/Tx packet functions.

4.29 vtss_api/include/vtss_pcs_10gbase_r_api.h File Reference

PCS_10BASE_R API.

```
#include <vtss/api/types.h>
```

4.29.1 Detailed Description

PCS_10BASE_R API.

4.30 vtss_api/include/vtss_phy_10g_api.h File Reference

10G PHY API

```
#include <vtss/api/types.h>
#include <vtss_misc_api.h>
```

Data Structures

- struct vtss_sublayer_status_t
  10G Phy link and fault status
- struct vtss_phy_10g_polarity_inv_t
  10G Phy Polarity inversion
- struct vtss_phy_10g_clk_src_t
  10G Phy CLOCK Source Selection
- struct ib_par_cfg
  Generalized data structure for IB parameters.
- struct vtss_phy_10g_ib_conf_t
  10G Phy IB configuration
- struct vtss_phy_10g_ib_status_t
  10G Phy IB configuration
- struct vtss_phy_10g_apc_conf_t
  10G Phy APC configuration
- struct vtss_phy_10g_apc_status_t
  10G Phy APC status
- struct vtss_phy_10g_serdes_status_t
  10G Phy SERDES status
- struct vtss_phy_10g_jitter_conf_t
  10G Phy Optimisation of jitter performance
- struct vtss_phy_10g_mode_t
  10G Phy operating mode
- struct vtss_phy_10g_init_parm_t
  10G Phy Initialization configuration
- struct vtss_phy_10g_rxckout_conf_t
  10G Phy RXCKOUT config data
- struct vtss_phy_10g_txckout_conf_t

10G Phy TXCKOUT config data

- struct vtss_phy_10g_srefclk_mode_t

  10G Phy srefclk config data
- struct vtss_phy_10g_ckout_conf_t

  10G Phy CKOUT config data
- struct vtss_phy_10g_sckout_conf_t

  10G Phy SCKOUT config data
- struct vtss_phy_10g_line_clk_conf_t

  10G Phy Line clock config data
- struct vtss_phy_10g_host_clk_conf_t

  10G Phy Host clock config data
- struct vtss_phy_10g_lane_sync_conf_t

  10G Phy Lane SYNC Configuration
- struct vtss_phy_10g_ob_status_t

  10G Phy OB status
- struct vtss_phy_10g_status_t

  10G Phy link and fault status for all sublayers
- struct vtss_phy_10g_clause_37_adv_t

  Advertisement control data for Clause 37 aneg.
- struct vtss_phy_10g_clause_37_status_t

  Clause 37 Auto-negotiation status.
- struct vtss_phy_10g_clause_37_cmn_status_t

  Clause 37 Auto-negotiation status for line and host.
- struct vtss_phy_10g_clause_37_control_t

  Clause 37 control struct.
- struct vtss_phy_10g_loopback_t

  10G Phy system and network loopbacks
- struct vtss_phy_pcs_cnt_t

  10G Phy PCS counters
- struct vtss_phy_10g_cnt_t

  10G Phy Sublayer counters
- struct vtss_phy_10g_auto_failover_conf_t

  10G PHY Automatic Failover configuration
- struct vtss_phy_10g_vscope_conf_t
- struct vtss_phy_10g_ib_storage_t

  VSCOPE fast scan storage.
- struct vtss_phy_10g_vscope_scan_conf_t

  VSCOPE scan configuration.
- struct vtss_phy_10g_vscope_scan_status_t
- struct vtss_phy_10g_pcs_prbs_gen_conf_t
- struct vtss_phy_10g_pcs_prbs_mon_conf_t
- struct vtss_phy_10g_prbs_gen_conf_t
- struct vtss_phy_10g_prbs_mon_conf_t

  10G PHY prbs monitor Configuration
- struct vtss_phy_10g_pkt_gen_conf_t

  10G PHY Packet generator configuration
- struct vtss_phy_10g_pkt_mon_conf_t

  10G PHY Packet Monitor configuration
- struct vtss_phy_10g_timestamp_val_t

  10G PHY timestamp value array(holder)
- struct vtss_phy_10g_id_t

10G Phy part number and revision

- struct vtss_gpio_10g_gpio_mode_t

  GPIO configured mode.

- struct vtss_phy_10g_fw_status_t

  Firmware status.

Macros

- #define BOOLEAN_STORAGE_COUNT 6
- #define UNSIGNED_STORAGE_COUNT 5
- #define PHASE_POINTS 128
- #define AMPLITUDE_POINTS 64
- #define VTSS_PHY_10G_ONE_LINE_ACTIVE 0x08
- #define VTSS_PHY_10G_MACSEC_DISABLED 0x04
- #define VTSS_PHY_10G_TIMESTAMP_DISABLED 0x02
- #define VTSS_PHY_10G_MACSEC_KEY_128 0x01
- #define VTSS_10G_PHY_GPIO_MAX 12
- #define VTSS_10G_PHY_GPIO_MAL_MAX 40
- #define VTSS_PHY_10G_LINK_LOS_EV 0x00000001

  Event source identification mask values.

- #define VTSS_PHY_10G_RX_LOL_EV 0x00000002
- #define VTSS_PHY_10G_TX_LOL_EV 0x00000004
- #define VTSS_PHY_10G_LOPC_EV 0x00000008
- #define VTSS_PHY_10G_HIGH_BER_EV 0x00000010
- #define VTSS_PHY_10G_MODULE_STAT_EV 0x00000020
- #define VTSS_PHY_10G_PCS_RECEIVE_FAULT_EV 0x00000040
- #define VTSS_PHY_EWIS_SEF_EV 0x00000080
- #define VTSS_PHY_EWIS_FPLM_EV 0x00000100
- #define VTSS_PHY_EWIS_FAIS_EV 0x00000200
- #define VTSS_PHY_EWIS_LOF_EV 0x00000400
- #define VTSS_PHY_EWIS_RDIL_EV 0x00000800
- #define VTSS_PHY_EWIS_AISL_EV 0x00001000
- #define VTSS_PHY_EWIS_LCDP_EV 0x00002000
- #define VTSS_PHY_EWIS_PLMP_EV 0x00004000
- #define VTSS_PHY_EWIS_AISP_EV 0x00008000
- #define VTSS_PHY_EWIS_LOPP_EV 0x00010000
- #define VTSS_PHY_EWIS_UNEQP_EV 0x00020000
- #define VTSS_PHY_EWIS_FEUNEQP_EV 0x00040000
- #define VTSS_PHY_EWIS_FERDIP_EV 0x00080000
- #define VTSS_PHY_EWIS_REIL_EV 0x00100000
- #define VTSS_PHY_EWIS_REIP_EV 0x00200000
- #define VTSS_PHY_EWIS_B1_NZ_EV 0x00400000
- #define VTSS_PHY_EWIS_B2_NZ_EV 0x00800000
- #define VTSS_PHY_EWIS_B3_NZ_EV 0x01000000
- #define VTSS_PHY_EWIS_REIL_NZ_EV 0x02000000
- #define VTSS_PHY_EWIS_REIP_NZ_EV 0x04000000
- #define VTSS_PHY_EWIS_B1_THRESH_EV 0x08000000
- #define VTSS_PHY_EWIS_B2_THRESH_EV 0x10000000
- #define VTSS_PHY_EWIS_B3_THRESH_EV 0x20000000
- #define VTSS_PHY_EWIS_REIL_THRESH_EV 0x40000000
- #define VTSS_PHY_EWIS_REIP_THRESH_EV 0x80000000
- #define VTSS_PHY_10G_RX_LOS_EV 0x00000001
- #define VTSS_PHY_10G_RX_LOL_EV 0x00000002

- #define VTSS_PHY_10G_TX_LOL_EV 0x00000004
- #define VTSS_PHY_10G_RX_CHAR_DEC_CNT_THRESH_EV 0x00000010
- #define VTSS_PHY_10G_TX_CHAR_ENC_CNT_THRESH_EV 0x00000020
- #define VTSS_PHY_10G_RX_BLK_DEC_CNT_THRESH_EV 0x00000040
- #define VTSS_PHY_10G_TX_BLK_ENC_CNT_THRESH_EV 0x00000080
- #define VTSS_PHY_10G_RX_SEQ_CNT_THRESH_EV 0x00000100
- #define VTSS_PHY_10G_TX_SEQ_CNT_THRESH_EV 0x00000200
- #define VTSS_PHY_10G_FEC_UNFIXED_CNT_THRESH_EV 0x00000400
- #define VTSS_PHY_10G_FEC_FIXED_CNT_THRESH_EV 0x00000800
- #define VTSS_PHY_10G_HIGHBER_EV 0x00001000
- #define VTSS_PHY_10G_RX_LINK_STAT_EV 0x00002000
- #define VTSS_PHY_10G_RX_LINK_STAT_EV 0x00002000
- #define VTSS_PHY_10G_GPIO_INT_AGG0_EV 0x00004000
- #define VTSS_PHY_10G_GPIO_INT_AGG1_EV 0x00008000
- #define VTSS_PHY_10G_GPIO_INT_AGG2_EV 0x00010000
- #define VTSS_PHY_10G_GPIO_INT_AGG3_EV 0x00020000
- #define VTSS_PHY_1G_LINE_AUTONEG_RESTART_EV 0x00040000
- #define VTSS_PHY_1G_HOST_AUTONEG_RESTART_EV 0x00080000
- #define VTSS_PHY_10G_LINE_MAC_LOCAL_FAULT_EV 0x00100000
- #define VTSS_PHY_10G_HOST_MAC_LOCAL_FAULT_EV 0x00400000
- #define VTSS_PHY_10G_LINE_MAC_REMOTE_FAULT_EV 0x00800000
- #define VTSS_PHY_10G_HOST_MAC_REMOTE_FAULT_EV 0x01000000

Typedefs

- typedef u16 vtss_gpio_10g_no_t
- typedef enum ckout_sel_ ckout_sel_t

  10G Phy CKOUTs Enum
- typedef u32 vtss_32_cntr_t
- typedef u32 vtss_gpio_no_t

  GPIO configured mode.
- typedef u32 vtss_phy_10g_event_t
- typedef u32 vtss_phy_10g_extnd_event_t

Enumerations

- enum oper_mode_t {
  VTSS_PHY_LAN_MODE, VTSS_PHY_WAN_MODE, VTSS_PHY_1G_MODE, VTSS_PHY_LAN_SYNCE_MODE,
  VTSS_PHY_WAN_SYNCE_MODE, VTSS_PHY_LAN_MIXED_SYNCE_MODE, VTSS_PHY_WAN_MIXED_SYNCE_MODE,
  VTSS_PHY_REPEATER_MODE }

  10G Phy operating mode enum type
- enum vtss_wrefclk_t { VTSS_WREFCLK_155_52, VTSS_WREFCLK_622_08 }

  Modes for WAN reference clock.
- enum vtss_phy_interface_mode {
  VTSS_PHY_XAUI_XFI, VTSS_PHY_XGMII_XFI, VTSS_PHY_RXAUI_XFI, VTSS_PHY_SGMII_LANE_0_XFI,
  VTSS_PHY_SGMII_LANE_3_XFI, VTSS_PHY_SFI_XFI }

  Phy Interface modes.
- enum vtss_recvrd_t { VTSS_RECVRD_RXCLKOUT, VTSS_RECVRD_TXCLKOUT }

  Modes for recovered clock.
- enum vtss_recvrdclk_cdr_div_t { VTSS_RECVRDCLK_CDR_DIV_64, VTSS_RECVRDCLK_CDR_DIV_66 }

  Modes for recovered clock divisor.

- enum vtss_srefclk_div_t { VTSS_SREFCLK_DIV_64, VTSS_SREFCLK_DIV_66, VTSS_SREFCLK_DIV_16 }

  Modes for Synch-E recovered clock.
- enum vtss_wref_clk_div_t { VTSS_WREFCLK_NONE, VTSS_WREFCLK_DIV_16 }

  Modes for WREFCLK clock divisor.
- enum apc_ib_regulator_t { VTSS_APC_IB_SFP_PLUS_ZR, VTSS_APC_IB_BACKPLANE }

  APC Rx regulator mode.
- enum ddr_mode_t { VTSS_DDR_MODE_A, VTSS_DDR_MODE_K, VTSS_DDR_MODE_M }

  Interleave mode.
- enum clk_mstr_t { VTSS_CLK_MSTR_INTERNAL, VTSS_CLK_MSTR_EXTERNAL }

  Clock master.
- enum vtss_rptr_rate_t { VTSS_RPTR_RATE_NONE, VTSS_RPTR_RATE_10_3125, VTSS_RPTR_RATE_9_9532, VTSS_RPTR_RATE_11_3, VTSS_RPTR_RATE_10_5187, VTSS_RPTR_RATE_1_25, VTSS_RPTR_RATE_10_709, VTSS_RPTR_RATE_11_095727, VTSS_RPTR_RATE_11_05 }

  Repeater Data rate.
- enum vtss_phy_10g_media_t { VTSS_MEDIA_TYPE_SR, VTSS_MEDIA_TYPE_SR2, VTSS_MEDIA_TYPE_DAC, VTSS_MEDIA_TYPE_ZR, VTSS_MEDIA_TYPE_KR, VTSS_MEDIA_TYPE_SR_SC, VTSS_MEDIA_TYPE_SR2_SC, VTSS_MEDIA_TYPE_DAC_SC, VTSS_MEDIA_TYPE_ZR_SC, VTSS_MEDIA_TYPE_ZR2_SC, VTSS_MEDIA_TYPE_KR_SC, VTSS_MEDIA_TYPE_NONE }

  10G Phy Media type
- enum vtss_phy_6g_link_partner_distance_t { VTSS_6G_LINK_SHORT_RANGE, VTSS_6G_LINK_LONG_RANGE }

  6G serdes link partner distance selection
- enum vtss_phy_10g_ib_apc_op_mode_t { VTSS_IB_APC_AUTO, VTSS_IB_APC_MANUAL, VTSS_IB_APC_FREEZE, VTSS_IB_APC_RESET, VTSS_IB_APC_RESTART, VTSS_IB_APC_NONE }

  10G SERDES APC operation
- enum vtss_channel_t { VTSS_CHANNEL_AUTO, VTSS_CHANNEL_0, VTSS_CHANNEL_1, VTSS_CHANNEL_2, VTSS_CHANNEL_3 }

  Channel modes - Auto is recommended.
- enum vtss_recvrd_clkout_t { VTSS_RECVRD_CLKOUT_DISABLE, VTSS_RECVRD_CLKOUT_LINE_SIDE_RX_CLOCK, VTSS_RECVRD_CLKOUT_LINE_SIDE_TX_CLOCK }

  Modes for (rx/tx) recovered clock output.
- enum vtss_phy_10g_srefclk_freq_t { VTSS_PHY_10G_SREFCLK_156_25, VTSS_PHY_10G_SREFCLK_125_00, VTSS_PHY_10G_SREFCLK_155_52, VTSS_PHY_10G_SREFCLK_INVALID }

  10G Phy sref clock input frequency
- enum vtss_phy_10g_ckout_freq_t { VTSS_PHY_10G_CLK_FULL_RATE, VTSS_PHY_10G_CLK_DIVIDE_BY_2, VTSS_PHY_10G_CLK_INVALID }

  10G Phy clock frequency
- enum vtss_ckout_data_sel_t { VTSS_CKOUT_LINE0_TX_CLOCK, VTSS_CKOUT_LINE1_TX_CLOCK, VTSS_CKOUT_LINE2_TX_CLOCK, VTSS_CKOUT_LINE3_TX_CLOCK, VTSS_CKOUT_HOST0_TX_CLOCK, VTSS_CKOUT_HOST1_TX_CLOCK, VTSS_CKOUT_HOST2_TX_CLOCK, VTSS_CKOUT_HOST3_TX_CLOCK, VTSS_CKOUT_LINE0_RECVRD_CLOCK, VTSS_CKOUT_LINE1_RECVRD_CLOCK, VTSS_CKOUT_LINE2_RECVRD_CLO VTSS_CKOUT_LINE3_RECVRD_CLOCK, VTSS_CKOUT_HOST0_RECVRD_CLOCK, VTSS_CKOUT_HOST1_RECVRD_CLOCK, VTSS_CKOUT_HOST2_RECVRD_ VTSS_CKOUT_HOST3_RECVRD_CLOCK, VTSS_CKOUT_HOST_PLL_CLOCK, VTSS_CKOUT_LINE_PLL_CLOCK, VTSS_CKOUT_CSR_CLOCK, VTSS_CKOUT_LTC_CLOCK, VTSS_CKOUT_DF2F_CLOCK, VTSS_CKOUT_F2DF_CLOCK, VTSS_CKOUT_DEBUG1, VTSS_CKOUT_DEBUG2, VTSS_CKOUT_OSCILLATOR_OUTPUT }

Modes for recovered clock output.

- enum vtss_phy_10g_squelch_src_t {
  VTSS_CKOUT_SQUELCH_SRC_GPIO0, VTSS_CKOUT_SQUELCH_SRC_GPIO1, VTSS_CKOUT_SQUELCH_SRC_GPIO2
  VTSS_CKOUT_SQUELCH_SRC_GPIO3,
  VTSS_CKOUT_SQUELCH_SRC_GPIO4, VTSS_CKOUT_SQUELCH_SRC_GPIO5, VTSS_CKOUT_SQUELCH_SRC_GPIO6
  VTSS_CKOUT_SQUELCH_SRC_GPIO7,
  VTSS_CKOUT_SQUELCH_SRC_LINK_LINE0, VTSS_CKOUT_SQUELCH_SRC_LINK_LINE1, VTSS_CKOUT_SQUELCH_S
  VTSS_CKOUT_SQUELCH_SRC_LINK_LINE3,
  VTSS_CKOUT_SQUELCH_SRC_LINK_HOST0, VTSS_CKOUT_SQUELCH_SRC_LINK_HOST1, VTSS_CKOUT_SQUELCH
  VTSS_CKOUT_SQUELCH_SRC_LINK_HOST3,
  VTSS_CKOUT_SQUELCH_SRC_LOS_LINE0, VTSS_CKOUT_SQUELCH_SRC_LOS_LINE1, VTSS_CKOUT_SQUELCH_SI
  VTSS_CKOUT_SQUELCH_SRC_LOS_LINE3,
  VTSS_CKOUT_SQUELCH_SRC_LOS_HOST0, VTSS_CKOUT_SQUELCH_SRC_LOS_HOST1, VTSS_CKOUT_SQUELCH_
  VTSS_CKOUT_SQUELCH_SRC_LOS_HOST3,
  VTSS_CKOUT_SQUELCH_SRC_LINK_LINE0_KR, VTSS_CKOUT_SQUELCH_SRC_LINK_LINE1_KR,
  VTSS_CKOUT_SQUELCH_SRC_LINK_LINE2_KR, VTSS_CKOUT_SQUELCH_SRC_LINK_LINE3_KR,
  VTSS_CKOUT_SQUELCH_SRC_LINK_HOST0_KR, VTSS_CKOUT_SQUELCH_SRC_LINK_HOST1_KR,
  VTSS_CKOUT_SQUELCH_SRC_LINK_HOST2_KR, VTSS_CKOUT_SQUELCH_SRC_LINK_HOST3_KR,
  VTSS_CKOUT_NO_SQUELCH }

  squelch control source
- enum vtss_phy_10g_clk_sel_t {
  VTSS_PHY_10G_LINE0_RECVRD_CLOCK, VTSS_PHY_10G_LINE1_RECVRD_CLOCK, VTSS_PHY_10G_LINE2_RECVR
  VTSS_PHY_10G_LINE3_RECVRD_CLOCK,
  VTSS_PHY_10G_HOST0_RECVRD_CLOCK, VTSS_PHY_10G_HOST1_RECVRD_CLOCK, VTSS_PHY_10G_HOST2_REC
  VTSS_PHY_10G_HOST3_RECVRD_CLOCK,
  VTSS_PHY_10G_SREFCLK, VTSS_PHY_10G_SYNC_DISABLE = 15 }

  Modes of recovered clocks for ckout and sckout pins.
- enum vtss_phy_10g_recvrd_clk_sel_t {
  VTSS_PHY_10G_USE_LINE0_RECVRD_CLOCK, VTSS_PHY_10G_USE_LINE1_RECVRD_CLOCK,
  VTSS_PHY_10G_USE_LINE2_RECVRD_CLOCK, VTSS_PHY_10G_USE_LINE3_RECVRD_CLOCK,
  VTSS_PHY_10G_USE_HOST0_RECVRD_CLOCK, VTSS_PHY_10G_USE_HOST1_RECVRD_CLOCK,
  VTSS_PHY_10G_USE_HOST2_RECVRD_CLOCK, VTSS_PHY_10G_USE_HOST3_RECVRD_CLOCK,
  VTSS_PHY_10G_USE_SREFCLK_CLOCK, VTSS_PHY_10G_USE_DEFAULT_RECVRD_CLOCK }

  Modes of recovered clock selection.
- enum ckout_sel_ { **CKOUT0**, **CKOUT1**, **CKOUT2**, **CKOUT3** }

  10G Phy CKOUTs Enum
- enum vtss_phy_10g_sckout_freq_t { VTSS_PHY_10G_SCKOUT_156_25, VTSS_PHY_10G_SCKOUT_125_00,
  VTSS_PHY_10G_SCKOUT_INVALID }

  10G Phy sckout clock input frequency
- enum vtss_phy_10g_rx_macro_t { VTSS_PHY_10G_RX_MACRO_LINE, VTSS_PHY_10G_RX_MACRO_HOST,
  VTSS_PHY_10G_RX_MACRO_SREFCLK }

  10G Phy Rx MACRO Configuration
- enum vtss_phy_10g_tx_macro_t { VTSS_PHY_10G_TX_MACRO_LINE, VTSS_PHY_10G_TX_MACRO_HOST,
  VTSS_PHY_10G_TX_MACRO_SCKOUT }

  10G Phy tx MACRO Configuration
- enum vtss_phy_10g_clause_37_remote_fault_t { VTSS_PHY_10G_CLAUSE_37_RF_LINK_OK, VTSS_PHY_10G_CLAUSE_3
  VTSS_PHY_10G_CLAUSE_37_RF_LINK_FAILURE, VTSS_PHY_10G_CLAUSE_37_RF_AUTONEG_ERROR
  }

  Auto-negotiation remote fault type.
- enum vtss_lb_type_t {
  VTSS_LB_NONE, VTSS_LB_SYSTEM_XS_SHALLOW, VTSS_LB_SYSTEM_XS_DEEP, VTSS_LB_SYSTEM_PCS_SHALLO
  VTSS_LB_SYSTEM_PCS_DEEP, VTSS_LB_SYSTEM_PMA, VTSS_LB_NETWORK_XS_SHALLOW,
  VTSS_LB_NETWORK_XS_DEEP,
  VTSS_LB_NETWORK_PCS, VTSS_LB_NETWORK_WIS, VTSS_LB_NETWORK_PMA, VTSS_LB_H2,
  VTSS_LB_H3, VTSS_LB_H4, VTSS_LB_H5, VTSS_LB_H6,
  VTSS_LB_L0, VTSS_LB_L1, VTSS_LB_L2, VTSS_LB_L3,
  VTSS_LB_L2C }

10G loopback types

- enum vtss_phy_10g_power_t { VTSS_PHY_10G_POWER_ENABLE, VTSS_PHY_10G_POWER_DISABLE }

  10G Phy power setting

- enum vtss_phy_10g_failover_mode_t { VTSS_PHY_10G_PMA_TO_FROM_XAUI_NORMAL, VTSS_PHY_10G_PMA_TO_FROM_XAUI_CROSSED, VTSS_PHY_10G_PMA_0_TO_FROM_XAUI_0_TO_XAUI_1, VTSS_PHY_10G_PMA_0_TO_FROM_XAUI_1_TO_XAUI_0, VTSS_PHY_10G_PMA_1_TO_FROM_XAUI_0_TO_XAUI_1, VTSS_PHY_10G_PMA_1_TO_FROM_XAUI_1_TO_XAUI_0 }

  10G Phy Failover Mode Setting

- enum vtss_phy_10g_auto_failover_event_t { VTSS_PHY_10G_AUTO_FAILOVER_EVENT_PCS_LINK_STATUS, VTSS_PHY_10G_AUTO_FAILOVER_EVENT_SERDES VTSS_PHY_10G_AUTO_FAILOVER_EVENT_WIS_LOF, VTSS_PHY_10G_AUTO_FAILOVER_EVENT_GPIO, VTSS_PHY_10G_AUTO_FAILOVER_EVENT_NONE }

  10G Phy Automatic Failover Event Setting

- enum vtss_phy_10g_auto_failover_filter_t { VTSS_PHY_10G_AUTO_FAILOVER_FILTER_NONE, VTSS_PHY_10G_AUTO_FAILOVER_FILTER_CNT_B2316, VTSS_PHY_10G_AUTO_FAILOVER_FILTER_CNT_B70, VTSS_PHY_10G_AUTO_FAILOVER_FILTER_CNT_A2316, VTSS_PHY_10G_AUTO_FAILOVER_FILTER_CNT_A70 }

  10G PHY Automatic Failover Filter

- enum vtss_phy_10g_vscope_scan_t { VTSS_PHY_10G_FAST_SCAN, VTSS_PHY_10G_FAST_SCAN_PLUS, VTSS_PHY_10G_QUICK_SCAN, VTSS_PHY_10G_FULL_SCAN }

  VSCOPE scan types.

- enum vtss_phy_10g_pkt_mon_rst_t { VTSS_PHY_10G_PKT_MON_RST_ALL, VTSS_PHY_10G_PKT_MON_RST_GOOD, VTSS_PHY_10G_PKT_MON_RST_BA VTSS_PHY_10G_PKT_MON_RST_FRAG, VTSS_PHY_10G_PKT_MON_RST_LFAULT, VTSS_PHY_10G_PKT_MON_RST_BER, VTSS_PHY_10G_PKT_MON_RST_N }

  10G PHY Packet monitor configuration

- enum vtss_phy_10g_type_t { **VTSS_PHY_TYPE_10G_NONE** = 0, **VTSS_PHY_TYPE_8484** = 8484, **VTSS_PHY_TYPE_8486** = 8486, **VTSS_PHY_TYPE_8487** = 8487, **VTSS_PHY_TYPE_8488** = 8488, **VTSS_PHY_TYPE_8489** = 8489, **VTSS_PHY_TYPE_8489_15** = 848915, **VTSS_PHY_TYPE_8490** = 8490, **VTSS_PHY_TYPE_8491** = 8491, **VTSS_PHY_TYPE_8256** = 8256, **VTSS_PHY_TYPE_8257** = 8257, **VTSS_PHY_TYPE_8258** = 8258, **VTSS_PHY_TYPE_8254** = 8254 }

  10g PHY type

- enum vtss_phy_10g_family_t { **VTSS_PHY_FAMILY_10G_NONE**, **VTSS_PHY_FAMILY_XAUI_XGMII_XFI**, **VTSS_PHY_FAMILY_XAUI_XFI**, **VTSS_PHY_FAMILY_VENICE**, **VTSS_PHY_FAMILY_MALIBU** }

  10G PHY family

- enum vtss_10g_phy_gpio_t { VTSS_10G_PHY_GPIO_NOT_INITIALIZED, VTSS_10G_PHY_GPIO_OUT, VTSS_10G_PHY_GPIO_IN, VTSS_10G_PHY_GPIO_WIS_INT, VTSS_10G_PHY_GPIO_1588_LOAD_SAVE, VTSS_10G_PHY_GPIO_1588_1PPS_0, VTSS_10G_PHY_GPIO_1588_1PPS_ VTSS_10G_PHY_GPIO_1588_1PPS_2, VTSS_10G_PHY_GPIO_1588_1PPS_3, VTSS_10G_PHY_GPIO_PCS_RX_FAULT, VTSS_10G_PHY_GPIO_SET_I2C_MAS VTSS_10G_PHY_GPIO_TX_ENABLE, VTSS_10G_PHY_GPIO_LINE_PLL_STATUS, VTSS_10G_PHY_GPIO_HOST_PLL_STATUS, VTSS_10G_PHY_GPIO_RCO VTSS_10G_PHY_GPIO_CHAN_INT_0, VTSS_10G_PHY_GPIO_CHAN_INT_1, VTSS_10G_PHY_GPIO_1588_INT, VTSS_10G_PHY_GPIO_TS_FIFO_EMPTY, VTSS_10G_PHY_GPIO_AGG_INT_0, VTSS_10G_PHY_GPIO_AGG_INT_1, VTSS_10G_PHY_GPIO_AGG_INT_2, VTSS_10G_PHY_GPIO_AGG_INT_3, VTSS_10G_PHY_GPIO_PLL_INT_0,

VTSS_10G_PHY_GPIO_PLL_INT_1, VTSS_10G_PHY_GPIO_SET_I2C_SLAVE, VTSS_10G_PHY_GPIO_CRSS_INT, VTSS_10G_PHY_GPIO_LED,
VTSS_10G_PHY_GPIO_DRIVE_LOW, VTSS_10G_PHY_GPIO_DRIVE_HIGH }

GPIO configured mode.

- enum vtss_gpio_10g_gpio_intr_sgnl_t {
VTSS_10G_GPIO_INTR_SGNL_I2C_MSTR_DATA_OUT, VTSS_10G_GPIO_INTR_SGNL_I2C_MSTR_CLK_OUT, VTSS_10G_GPIO_INTR_SGNL_LED_TX, VTSS_10G_GPIO_INTR_SGNL_LED_RX, VTSS_10G_GPIO_INTR_SGNL_RX_ALARM, VTSS_10G_GPIO_INTR_SGNL_TX_ALARM, VTSS_10G_GPIO_INTR_SGNL_ VTSS_10G_GPIO_INTR_SGNL_LINE_LINK, VTSS_10G_GPIO_INTR_SGNL_LINE_KR_8b10b_2GPIO, VTSS_10G_GPIO_INTR_SGNL_LINE_KR_10b_2GPIO, VTSS_10G_GPIO_INTR_SGNL_ROSI_PULSE, VTSS_10G_GPIO_INTR_SGNL_ROSI_SDATA, VTSS_10G_GPIO_INTR_SGNL_ROSI_SCLK, VTSS_10G_GPIO_INTR_SGNL_TOSI_PULSE, VTSS_10G_GPIO_INTR_SGN VTSS_10G_GPIO_INTR_SGNL_LINE_PCS1G_LINK, VTSS_10G_GPIO_INTR_SGNL_LINE_PCS_RX_STAT, VTSS_10G_GPIO_INTR_SGNL_CLIENT_PCS1G_LINK, VTSS_10G_GPIO_INTR_SGNL_HOST_PCS_RX_STAT, VTSS_10G_GPIO_INTR_SGNL_HOST_SD10G_IB_SIG, VTSS_10G_GPIO_INTR_SGNL_LINE_SD10G_IB_SIG, VTSS_10G_GPIO_INTR_SGNL_HPCS_INTR, VTSS_10G_GPIO_INTR_SGNL_LPCS_INTR, VTSS_10G_GPIO_INTR_SGNL_CLIENT_PCS1G_INTR, VTSS_10G_GPIO_INTR_SGNL_LINE_PCS1G_INTR, VTSS_10G_GPIO_INTR_SGNL_WIS_INT0, VTSS_10G_GPIO_INTR_SGNL_HOST_PMA_INT, VTSS_10G_GPIO_INTR_SGNL_LINE_PMA_INT, VTSS_10G_GPIO_INTR_SGNL_DATA_ACT_TX, VTSS_10G_GPIO_INTR_SGNL_DATA_ACT_RX, VTSS_10G_GPIO_INTR_SGNL_HDATA_ACT_TX, VTSS_10G_GPIO_INTR_SGNL_HDATA_ACT_RX, VTSS_10G_GPIO_INTR_SGNL_XGMII_PAUS_EGR, VTSS_10G_GPIO_INTR_SGNL_XGMII_PAUS_ING, VTSS_10G_GPIO_INTR_SGNL_RX_PCS_PAUS, VTSS_10G_GPIO_INTR_SGNL_TX_PCS_PAUS, VTSS_10G_GPIO_INTR_SGNL_RX_WIS_PAUS, VTSS_10G_GPIO_INTR_SGNL_TX_WIS_PAUS, VTSS_10G_GPIO_INTR_SGNL_ETH_CHAN_DIS, VTSS_10G_GPIO_INTR_SGNL_MACSEC_1588_SFD_LANE, VTSS_10G_GPIO_INTR_SGNL_LINE_S_TX_FAULT, VTSS_10G_GPIO_INTR_SGNL_LPCS1G_LATENCY0_OR_EWIS_BIT VTSS_10G_GPIO_INTR_SGNL_LPCS1G_LATENCY1_OR_EWIS_BIT1, VTSS_10G_GPIO_INTR_SGNL_LPCS1G_CHAR_ VTSS_10G_GPIO_INTR_SGNL_LPCS1G_CHAR_POS1_OR_EWIS_WORD0, VTSS_10G_GPIO_INTR_SGNL_LPCS1G_CH VTSS_10G_GPIO_INTR_SGNL_LPCS1G_CHAR_POS3_OR_EWIS_WORD2, VTSS_10G_GPIO_INTR_SGNL_MACSEC_IG VTSS_10G_GPIO_INTR_SGNL_MACSEC_IGR_PRED_VAR1, VTSS_10G_GPIO_INTR_SGNL_KR_ACTV_2GPIO, VTSS_10G_GPIO_INTR_SGNL_DFT_TX_2GPIO, VTSS_10G_GPIO_INTR_SGNL_RESERVED, VTSS_10G_GPIO_INTR_SGNL_EXE_LST_2GPIO_0, VTSS_10G_GPIO_INTR_SGNL_EXE_LST_2GPIO_1, VTSS_10G_GPIO_INTR_SGNL_EXE_LST_2GPIO_2, VTSS_10G_GPIO_INTR_SGNL_EXE_LST_2GPIO_3, VTSS_10G_GPIO_INTR_SGNL_EXE_LST_2GPIO_4, VTSS_10G_GPIO_INTR_SGNL_LINK_HCD_2GPIO_0, VTSS_10G_GPIO_INTR_SGNL_LINK_HCD_2GPIO_1, VTSS_10G_GPIO_INTR_SGNL_LINK_HCD_2GPIO_2, VTSS_10G_GPIO_INTR_SGNL_ETH_1G_ENA, VTSS_10G_GPIO_INTR_SGNL_H_KR_8b10b_2GIPO, VTSS_10G_GPIO_INTR_SGNL_H_KR_10Gb_2GPIO, VTSS_10G_GPIO_INTR_SGNL_H_KR_ACTV_2GPIO, VTSS_10G_GPIO_INTR_SGNL_NONE }

  GPIO internal signal types.

- enum vtss_gpio_10g_chan_intrpt_t {
VTSS_10G_GPIO_INTRPT_WIS0, VTSS_10G_GPIO_INTRPT_WIS1, VTSS_10G_GPIO_INTRPT_LPCS10G, VTSS_10G_GPIO_INTRPT_HPCS10G, VTSS_10G_GPIO_INTRPT_LPCS1G, VTSS_10G_GPIO_INTRPT_HPCS1G, VTSS_10G_GPIO_INTRPT_MSEC_EGR, VTSS_10G_GPIO_INTRPT_MSEC_IGR, VTSS_10G_GPIO_INTRPT_LMAC, VTSS_10G_GPIO_INTRPT_HMAC, VTSS_10G_GPIO_INTRPT_FCBUF, VTSS_10G_GPIO_INTRPT_LIGR_FIFO, VTSS_10G_GPIO_INTRPT_LEGR_FIFO, VTSS_10G_GPIO_INTRPT_HEGR_FIFO, VTSS_10G_GPIO_INTRPT_LPMA, VTSS_10G_GPIO_INTRPT_HPMA }

  GPIO Channel level interrupts.

- enum vtss_gpio_10g_aggr_intrpt_t {
VTSS_10G_GPIO_AGGR_INTRPT_CH0_INTR0_EN, VTSS_10G_GPIO_AGGR_INTRPT_CH0_INTR1_EN, VTSS_10G_GPIO_AGGR_INTRPT_CH1_INTR0_EN, VTSS_10G_GPIO_AGGR_INTRPT_CH1_INTR1_EN, VTSS_10G_GPIO_AGGR_INTRPT_CH2_INTR0_EN, VTSS_10G_GPIO_AGGR_INTRPT_CH2_INTR1_EN, VTSS_10G_GPIO_AGGR_INTRPT_CH3_INTR0_EN, VTSS_10G_GPIO_AGGR_INTRPT_CH3_INTR1_EN, VTSS_10G_GPIO_AGGR_INTRPT_IP1588_0_INTR0_EN, VTSS_10G_GPIO_AGGR_INTRPT_IP1588_0_INTR1_EN, VTSS_10G_GPIO_AGGR_INTRPT_IP1588_0_INTR2_EN, VTSS_10G_GPIO_AGGR_INTRPT_IP1588_0_INTR3_EN, VTSS_10G_GPIO_AGGR_INTRPT_IP1588_1_INTR0_EN, VTSS_10G_GPIO_AGGR_INTRPT_IP1588_1_INTR1_EN, VTSS_10G_GPIO_AGGR_INTRPT_IP1588_1_INTR2_EN, VTSS_10G_GPIO_AGGR_INTRPT_IP1588_1_INTR3_EN,

VTSS_10G_GPIO_AGGR_INTRPT_LCPLL_0_INTR_EN, VTSS_10G_GPIO_AGGR_INTRPT_LCPLL_1_INTR_EN, VTSS_10G_GPIO_AGGR_INTRPT_EXP4_INTR_EN, VTSS_10G_GPIO_AGGR_INTRPT_CLK_MUX_INTR_EN, VTSS_10G_GPIO_AGGR_INTRPT_GPIO_INTR_EN }

GPIO Channel level interrupts.

- enum vtss_gpio_10g_input_t { **VTSS_10G_GPIO_INPUT_NONE**, VTSS_10G_GPIO_INPUT_LINE_LOPC, VTSS_10G_GPIO_INPUT_HOST_LOPC }

  GPIO Channel level interrupts.

Functions

- vtss_rc vtss_phy_10g_mode_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_mode_t *const mode)

  Get the Phy operating mode.
- vtss_rc vtss_phy_10g_init (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_init_parm_t *const init_conf)

  Identify PHY and initialize software accordingly.
- vtss_rc vtss_phy_10g_mode_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_mode_t *const mode)

  Identify, Reset and set the operating mode of the PHY.
- vtss_rc vtss_phy_10g_ib_conf_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_ib_conf_t *const ib_conf, BOOL is_host)

  Configure Input buffer .
- vtss_rc vtss_phy_10g_ib_conf_get (const vtss_inst_t inst, const vtss_port_no_t port_no, const BOOL is_host, vtss_phy_10g_ib_conf_t *const ib_conf)

  Get configuration of Input buffer .
- vtss_rc vtss_phy_10g_ib_status_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_ib_status_t *const ib_status)

  Get status of Input buffer .
- vtss_rc vtss_phy_10g_apc_conf_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_apc_conf_t *const apc_conf, const BOOL is_host)

  Configure APC .
- vtss_rc vtss_phy_10g_apc_conf_get (const vtss_inst_t inst, const vtss_port_no_t port_no, const BOOL is↩_host, vtss_phy_10g_apc_conf_t *const apc_conf)

  Get configuration of APC .
- vtss_rc vtss_phy_10g_apc_status_get (const vtss_inst_t inst, const vtss_port_no_t port_no, const BOOL is↩_host, vtss_phy_10g_apc_status_t *const apc_status)

  Get status of APC.
- vtss_rc vtss_phy_10g_apc_restart (const vtss_inst_t inst, const vtss_port_no_t port_no, const BOOL is_host)

  Restart of APC - Debug function only.
- vtss_rc vtss_phy_10g_jitter_conf_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_jitter_conf_t *const jitter_conf, BOOL is_host)

  Configure optimised jitter.
- vtss_rc vtss_phy_10g_jitter_conf_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_jitter_conf_t *jitter_conf, BOOL is_host)

  Gets current Jitter configuration.
- vtss_rc vtss_phy_10g_jitter_status_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_jitter_conf_t *const jitter_conf, BOOL is_host)

  Jitter status.
- vtss_rc vtss_phy_10g_synce_clkout_get (const vtss_inst_t inst, const vtss_port_no_t port_no, BOOL *const synce_clkout)

  Get the status of recovered clock from PHY. (recommended to use vtss_phy_10g_rxckout_get instead)

- vtss_rc vtss_phy_10g_synce_clkout_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const BOOL synce_clkout)

  Enable or Disable the recovered clock from PHY. (recommended to use vtss_phy_10g_rxckout_set instead)
- vtss_rc vtss_phy_10g_xfp_clkout_get (const vtss_inst_t inst, const vtss_port_no_t port_no, BOOL *const xfp_clkout)

  Get the status of RXCLKOUT/TXCLKOUT from PHY. (recommended to use vtss_phy_10g_txckout_get instead)
- vtss_rc vtss_phy_10g_xfp_clkout_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const BOOL xfp_clkout)

  Enable or Disable the RXCLKOUT/TXCLKOUT from PHY. (recommended to use vtss_phy_10g_txckout_set instead)
- vtss_rc vtss_phy_10g_rxckout_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_rxckout_conf_t *const rxckout)

  Get the rx recovered clock output configuration.
- vtss_rc vtss_phy_10g_rxckout_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_rxckout_conf_t *const rxckout)

  Set the rx recovered clock output configuration.
- vtss_rc vtss_phy_10g_txckout_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_txckout_conf_t *const txckout)

  Get the status of tx recovered clock output configuration.
- vtss_rc vtss_phy_10g_txckout_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_txckout_conf_t *const txckout)

  Set the tx recovered clock output configuration.
- vtss_rc vtss_phy_10g_srefclk_conf_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_srefclk_mode_t *const srefclk)

  Get the configuration of srefclk setting
  Avaliable for PHY family VENICE
  This function should not be used any more, instead use the API function vtss_phy_10g_mode_get, see the parameter documentation for that function.
- vtss_rc vtss_phy_10g_srefclk_conf_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_srefclk_mode_t *const srefclk)

  Set the configuration of srefclk setting. Avaliable for PHY family VENICE
  This function should not be used any more, instead use the API function vtss_phy_10g_mode_set, see the parameter documentation for that function.
- vtss_rc vtss_phy_10g_sckout_conf_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_sckout_conf_t *const sckout)

  Set the configuration of sckout setting. Avaliable for PHY family MALIBU
  .
- vtss_rc vtss_phy_10g_ckout_conf_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_ckout_conf_t *const ckout)

  Set the configuration of ckout setting. Avaliable for PHY family MALIBU
  .
- vtss_rc vtss_phy_10g_line_clk_conf_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_line_clk_conf_t *const line_clk)

  Set the configuration of sckout setting. Avaliable for PHY family MALIBU
  .
- vtss_rc vtss_phy_10g_host_clk_conf_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_host_clk_conf_t *const host_clk)

  Set the configuration of sckout setting. Avaliable for PHY family MALIBU
  .
- vtss_rc vtss_phy_10g_line_recvrd_clk_conf_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_line_clk_conf_t *const line_clk)

  Set the configuration of line clk recovered setting. Avaliable for PHY family MALIBU
  .
- vtss_rc vtss_phy_10g_host_recvrd_clk_conf_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_host_clk_conf_t *const host_clk)

Set the configuration of host clk recovered setting. Avaliable for PHY family MALIBU
.

- vtss_rc vtss_phy_10g_lane_sync_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_lane_sync_conf_t *const lane_sync)

  Set the configuration of lane sync setting. Avaliable for PHY family MALIBU
  .
- vtss_rc vtss_phy_10g_debug_register_dump (const vtss_inst_t inst, const vtss_debug_printf_t pr, BOOL clear, const vtss_port_no_t port_no)

  Set the configuration of 10G PHY Dump setting. Avaliable for PHY family Venice & Malibu
  .
- vtss_rc vtss_phy_10g_status_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_status_t *const status)

  Get the link and fault status of the PHY sublayers.
- vtss_rc vtss_phy_10g_serdes_status_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_serdes_status_t *const status)

  Get the status of PHY including sub layers.
- vtss_rc vtss_phy_10g_reset (const vtss_inst_t inst, const vtss_port_no_t port_no)

  Reset the phy. Phy is reset to default values.
- vtss_rc vtss_phy_10g_clause_37_status_get (const vtss_inst_t inst, vtss_port_no_t port_no, vtss_phy_10g_clause_37_cmn_st *const status)

  Get clause 37 status.
- vtss_rc vtss_phy_10g_clause_37_control_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_clause_37_control_t *const control)

  Get clause 37 control configuration from software.
- vtss_rc vtss_phy_10g_clause_37_control_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_clause_37_control_t *const control)

  Set clause 37 control configuration.
- vtss_rc vtss_phy_10g_loopback_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_loopback_t *const loopback)

  Enable/Disable a phy network or system loopback.
  Only one loopback mode can be active at the same time.
- vtss_rc vtss_phy_10g_loopback_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_loopback_t *const loopback)

  Get loopback settings.
- vtss_rc vtss_phy_10g_cnt_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_cnt_t *const cnt)

  Get counters.
- vtss_rc vtss_phy_10g_power_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_power_t *const power)

  Get the power settings.
- vtss_rc vtss_phy_10g_power_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_power_t *const power)

  Set the power settings.
- BOOL vtss_phy_10G_is_valid (const vtss_inst_t inst, const vtss_port_no_t port_no)

  Gives a True/False value if the Phy is supported by the API
  Only Vitesse phys are supported. vtss_phy_10g_mode_set() must be applied.
- vtss_rc vtss_phy_10g_failover_set (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_failover_mode_t *const mode)

  Set the failover mode.
- vtss_rc vtss_phy_10g_failover_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_failover_mode_t *const mode)

  Get the failover mode.
- vtss_rc vtss_phy_10g_auto_failover_set (const vtss_inst_t inst, vtss_phy_10g_auto_failover_conf_t *const mode)

Set the automatic failover mode.

- vtss_rc vtss_phy_10g_auto_failover_get (const vtss_inst_t inst, vtss_phy_10g_auto_failover_conf_t *const mode)

  Get the Automatic failover mode Configuration.

- vtss_rc vtss_phy_10g_vscope_conf_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_vscope_conf_t *const conf)

  set VSCOPE fast scan configuration

- vtss_rc vtss_phy_10g_vscope_conf_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_vscope_conf_t *const conf)

  get VSCOPE fast scan configuration

- vtss_rc vtss_phy_10g_vscope_scan_status_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_vscope_scan_status_t *const conf)

  set VSCOPE fast scan configuration

- vtss_rc vtss_phy_10g_pcs_prbs_gen_conf_set (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_pcs_prbs_gen_conf_t *const conf, const BOOL line)

  Set PCS-prbs generator Configuration.

- vtss_rc vtss_phy_10g_pcs_prbs_gen_conf_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_pcs_prbs_gen_conf_t *const conf, const BOOL line)

  Get PCS-prbs generator Configuration.

- vtss_rc vtss_phy_10g_pcs_prbs_mon_conf_set (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_pcs_prbs_mon_conf_t *const conf, const BOOL line)

  Set PCS-prbs monitor Configuration.

- vtss_rc vtss_phy_10g_pcs_prbs_mon_conf_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_pcs_prbs_mon_conf_t *const conf, const BOOL line)

  Get PCS-prbs monitor Configuration.

- vtss_rc vtss_phy_10g_pcs_prbs_mon_status_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_pcs_prbs_mon_conf_t *const conf, const BOOL line)

  Get PCS-prbs monitor status.

- vtss_rc vtss_phy_10g_prbs_gen_conf (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_prbs_gen_conf_t *const conf)

  set prbs generator Configuration

- vtss_rc vtss_phy_10g_prbs_gen_conf_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_prbs_gen_conf_t *const conf, BOOL line)

  get prbs generator Configuration

- vtss_rc vtss_phy_10g_prbs_mon_conf (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_prbs_mon_conf_t *const conf)

  set prbs generator Configuration

- vtss_rc vtss_phy_10g_prbs_mon_conf_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_prbs_mon_conf_t *const conf, BOOL line)

  prbs generator Configuration get

- vtss_rc vtss_phy_10g_prbs_mon_status_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_prbs_mon_conf_t *const mon_status, BOOL line, BOOL reset)

  prbs Checker Status get

- vtss_rc vtss_phy_10g_pkt_gen_conf (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_pkt_gen_conf_t *const conf)

  Set Packet generation Configuration.

- vtss_rc vtss_phy_10g_pkt_mon_conf (const vtss_inst_t inst, const vtss_port_no_t port_no, BOOL ts_rd, vtss_phy_10g_pkt_mon_conf_t *const conf, vtss_phy_10g_timestamp_val_t *const conf_ts)

  Set Packet Monitor Configuration.

- vtss_rc vtss_phy_10g_pkt_mon_counters_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_pkt_mon_conf_t *const conf)

  Set/Get Packet mon Counters.

- vtss_rc vtss_phy_10g_id_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_id_t *const phy_id)

  Read the Phy Id.
- vtss_rc vtss_phy_10g_gpio_mode_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_gpio_10g_no_t gpio_no, const vtss_gpio_10g_gpio_mode_t *const mode)

  Set GPIO mode. There is only one set og GPIO per PHY chip - not per port.
- vtss_rc vtss_phy_10g_gpio_mode_get (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_gpio_10g_no_t gpio_no, vtss_gpio_10g_gpio_mode_t *const mode)

  Get GPIO mode.
- vtss_rc vtss_phy_10g_gpio_read (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_gpio_10g_no_t gpio_no, BOOL *const value)

  Read from GPIO input pin.
- vtss_rc vtss_phy_10g_gpio_write (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_gpio_10g_no_t gpio_no, const BOOL value)

  Write to GPIO output pin.
- vtss_rc vtss_phy_10g_event_enable_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_event_t ev_mask, const BOOL enable)

  Enabling / Disabling of events.
- vtss_rc vtss_phy_10g_event_enable_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_event_t *const ev_mask)

  Get Enabling of events.
- vtss_rc vtss_phy_10g_extended_event_enable_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_extnd_event_t *const ex_ev_mask)

  Get Enabling of events.
- vtss_rc vtss_phy_10g_event_poll (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_event_t *const ev_mask)

  Polling for active events.
- vtss_rc vtss_phy_10g_pcs_status_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_extnd_event_t *const ex_events)

  poll and clear PCS STICKY Register
- vtss_rc vtss_phy_10g_extended_event_poll (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_extnd_event_t *const ex_events)

  Polling for active events.
- vtss_rc vtss_phy_10g_extended_event_enable_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_phy_10g_extnd_event_t ex_ev_mask, const BOOL extnd_enable)

  Enabling / Disabling of events.
- vtss_rc vtss_phy_10g_poll_1sec (const vtss_inst_t inst)

  Function is called once a second.
- vtss_rc vtss_phy_10g_edc_fw_status_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_phy_10g_fw_status_t *const status)

  Internal microprocessor status.
- vtss_rc vtss_phy_10g_fc_buffer_reset (const vtss_inst_t inst, const vtss_port_no_t port_no)

  debug function for PHY 10G FC buffer reset
- vtss_rc vtss_phy_10g_csr_read (const vtss_inst_t inst, const vtss_port_no_t port_no, const u32 dev, const u32 addr, u32 *const value)

  CSR register read.
- vtss_rc vtss_phy_10g_csr_write (const vtss_inst_t inst, const vtss_port_no_t port_no, const u32 dev, const u32 addr, const u32 value)

  CSR register write.
- vtss_rc vtss_phy_warm_start_10g_failed_get (const vtss_inst_t inst, const vtss_port_no_t port_no)

  Function for checking if any issue were seen during warm-start.
- vtss_rc vtss_phy_10g_sgmii_mode_set (const vtss_inst_t inst, const vtss_port_no_t port_no, BOOL enable)

  Enables Pass through mode in 10G PHY.

- vtss_rc vtss_phy_10g_i2c_read (const vtss_inst_t inst, const vtss_port_no_t port_no, const u16 addr, u16 *value)

  read from i2c device
- vtss_rc vtss_phy_10g_i2c_write (const vtss_inst_t inst, const vtss_port_no_t port_no, const u16 addr, const u16 *value)

  Write to i2c device.
- vtss_rc vtss_phy_10g_get_user_data (const vtss_inst_t inst, const vtss_port_no_t port_no, void **user_data)

  Gets generic pointer in vtss_state structure.

4.30.1 Detailed Description

10G PHY API

This header file describes 10G PHY control functions

4.30.2 Macro Definition Documentation

4.30.2.1 BOOLEAN_STORAGE_COUNT

```
#define BOOLEAN_STORAGE_COUNT 6
```

BOOL parameters to be stored during Vscope Scan

Definition at line 1894 of file vtss_phy_10g_api.h.

4.30.2.2 UNSIGNED_STORAGE_COUNT

```
#define UNSIGNED_STORAGE_COUNT 5
```

UNSIGNED parameters to be stored during Vscope Scan

Definition at line 1895 of file vtss_phy_10g_api.h.

4.30.2.3 PHASE_POINTS

```
#define PHASE_POINTS 128
```

phase points range from 0-127

Definition at line 1915 of file vtss_phy_10g_api.h.

4.30.2.4 AMPLITUDE_POINTS

```
#define AMPLITUDE_POINTS 64
```

amplitude points range from 0-63

Definition at line 1916 of file vtss_phy_10g_api.h.

4.30.2.5 VTSS_PHY_10G_ONE_LINE_ACTIVE

```
#define VTSS_PHY_10G_ONE_LINE_ACTIVE 0x08
```

Bit indicating PHY vith only one line interface

Definition at line 2261 of file vtss_phy_10g_api.h.

4.30.2.6 VTSS_PHY_10G_MACSEC_DISABLED

```
#define VTSS_PHY_10G_MACSEC_DISABLED 0x04
```

Bit indicating that macsec is disabled

Definition at line 2262 of file vtss_phy_10g_api.h.

4.30.2.7 VTSS_PHY_10G_TIMESTAMP_DISABLED

```
#define VTSS_PHY_10G_TIMESTAMP_DISABLED 0x02
```

Bit indicating that timestamp feature is disabled

Definition at line 2263 of file vtss_phy_10g_api.h.

4.30.2.8 VTSS_PHY_10G_MACSEC_KEY_128

```
#define VTSS_PHY_10G_MACSEC_KEY_128 0x01
```

Bit indicating that only 128 bit macsec encryption key is supported, otherwise it is 128/256 key

Definition at line 2264 of file vtss_phy_10g_api.h.

4.30.2.9 VTSS_10G_PHY_GPIO_MAX

```
#define VTSS_10G_PHY_GPIO_MAX 12
```

Max value of gpio_no parameter

Definition at line 2496 of file vtss_phy_10g_api.h.

4.30.2.10 VTSS_10G_PHY_GPIO_MAL_MAX

```
#define VTSS_10G_PHY_GPIO_MAL_MAX 40
```

Max value of gpio_no parameter,Malibu

Definition at line 2498 of file vtss_phy_10g_api.h.

4.30.2.11 VTSS_PHY_10G_LINK_LOS_EV

```
#define VTSS_PHY_10G_LINK_LOS_EV 0x00000001
```

Event source identification mask values.

PHY Link Los interrupt - only on 8486

Definition at line 2563 of file vtss_phy_10g_api.h.

4.30.2.12 VTSS_PHY_10G_RX_LOL_EV [1/2]

```
#define VTSS_PHY_10G_RX_LOL_EV 0x00000002
```

PHY RXLOL interrupt - only on 8488

PHY RX LOL interrupt - 8256 specific

Definition at line 2603 of file vtss_phy_10g_api.h.

4.30.2.13 VTSS_PHY_10G_TX_LOL_EV [1/2]

```
#define VTSS_PHY_10G_TX_LOL_EV 0x00000004
```

PHY TXLOL interrupt - only on 8488

PHY TX LOL interrupt - 8256 specific

Definition at line 2604 of file vtss_phy_10g_api.h.

4.30.2.14 VTSS_PHY_10G_LOPC_EV

```
#define VTSS_PHY_10G_LOPC_EV 0x00000008
```

PHY LOPC interrupt - only on 8488

Definition at line 2566 of file vtss_phy_10g_api.h.

4.30.2.15 VTSS_PHY_10G_HIGH_BER_EV

```
#define VTSS_PHY_10G_HIGH_BER_EV 0x00000010
```

PHY HIGH_BER interrupt - only on 8488

Definition at line 2567 of file vtss_phy_10g_api.h.

4.30.2.16 VTSS_PHY_10G_MODULE_STAT_EV

```
#define VTSS_PHY_10G_MODULE_STAT_EV 0x00000020
```

PHY MODULE_STAT interrupt - only on 8488

Definition at line 2568 of file vtss_phy_10g_api.h.

4.30.2.17 VTSS_PHY_10G_PCS_RECEIVE_FAULT_EV

```
#define VTSS_PHY_10G_PCS_RECEIVE_FAULT_EV 0x00000040
```

PHY PCS_RECEIVE_FAULT interrupt - only on 8488

Definition at line 2569 of file vtss_phy_10g_api.h.

4.30.2.18 VTSS_PHY_EWIS_SEF_EV

```
#define VTSS_PHY_EWIS_SEF_EV 0x00000080
```

SEF has changed state - only for 8488

Definition at line 2571 of file vtss_phy_10g_api.h.

4.30.2.19 VTSS_PHY_EWIS_FPLM_EV

```
#define VTSS_PHY_EWIS_FPLM_EV 0x00000100
```

far-end (PLM-P) / (LCDP) - only for 8488

Definition at line 2572 of file vtss_phy_10g_api.h.

4.30.2.20 VTSS_PHY_EWIS_FAIS_EV

```
#define VTSS_PHY_EWIS_FAIS_EV 0x00000200
```

far-end (AIS-P) / (LOP) - only for 8488

Definition at line 2573 of file vtss_phy_10g_api.h.

4.30.2.21 VTSS_PHY_EWIS_LOF_EV

```
#define VTSS_PHY_EWIS_LOF_EV 0x00000400
```

Loss of Frame (LOF) - only for 8488

Definition at line 2574 of file vtss_phy_10g_api.h.

4.30.2.22 VTSS_PHY_EWIS_RDIL_EV

```
#define VTSS_PHY_EWIS_RDIL_EV 0x00000800
```

Line Remote Defect Indication (RDI-L) - only for 8488

Definition at line 2575 of file vtss_phy_10g_api.h.

4.30.2.23 VTSS_PHY_EWIS_AISL_EV

```
#define VTSS_PHY_EWIS_AISL_EV 0x00001000
```

Line Alarm Indication Signal (AIS-L) - only for 8488

Definition at line 2576 of file vtss_phy_10g_api.h.

4.30.2.24 VTSS_PHY_EWIS_LCDP_EV

```
#define VTSS_PHY_EWIS_LCDP_EV 0x00002000
```

Loss of Code-group Delineation (LCD-P) - only for 8488

Definition at line 2577 of file vtss_phy_10g_api.h.

4.30.2.25 VTSS_PHY_EWIS_PLMP_EV

```
#define VTSS_PHY_EWIS_PLMP_EV 0x00004000
```

Path Label Mismatch (PLMP) - only for 8488

Definition at line 2578 of file vtss_phy_10g_api.h.

4.30.2.26 VTSS_PHY_EWIS_AISP_EV

```
#define VTSS_PHY_EWIS_AISP_EV 0x00008000
```

Path Alarm Indication Signal (AIS-P) - only for 8488

Definition at line 2579 of file vtss_phy_10g_api.h.

4.30.2.27 VTSS_PHY_EWIS_LOPP_EV

```
#define VTSS_PHY_EWIS_LOPP_EV 0x00010000
```

Path Loss of Pointer (LOP-P) - only for 8488

Definition at line 2580 of file vtss_phy_10g_api.h.

4.30.2.28 VTSS_PHY_EWIS_UNEQP_EV

```
#define VTSS_PHY_EWIS_UNEQP_EV 0x00020000
```

Unequiped Path (UNEQ-P) - only for 8488

Definition at line 2581 of file vtss_phy_10g_api.h.

4.30.2.29 VTSS_PHY_EWIS_FEUNEQP_EV

```
#define VTSS_PHY_EWIS_FEUNEQP_EV 0x00040000
```

Far-end Unequiped Path (UNEQ-P) - only for 8488

Definition at line 2582 of file vtss_phy_10g_api.h.

4.30.2.30 VTSS_PHY_EWIS_FERDIP_EV

```
#define VTSS_PHY_EWIS_FERDIP_EV 0x00080000
```

Far-end Path Remote Defect Identifier (RDI-P) - only for 8488

Definition at line 2583 of file vtss_phy_10g_api.h.

4.30.2.31 VTSS_PHY_EWIS_REIL_EV

```
#define VTSS_PHY_EWIS_REIL_EV 0x00100000
```

Line Remote Error Indication (REI-L) - only for 8488

Definition at line 2584 of file vtss_phy_10g_api.h.

4.30.2.32 VTSS_PHY_EWIS_REIP_EV

```
#define VTSS_PHY_EWIS_REIP_EV 0x00200000
```

Path Remote Error Indication (REI-P) - only for 8488

Definition at line 2585 of file vtss_phy_10g_api.h.

4.30.2.33 VTSS_PHY_EWIS_B1_NZ_EV

```
#define VTSS_PHY_EWIS_B1_NZ_EV 0x00400000
```

PMTICK B1 BIP (B1_ERR_CNT) not zero - only for 8488

Definition at line 2586 of file vtss_phy_10g_api.h.

4.30.2.34 VTSS_PHY_EWIS_B2_NZ_EV

```
#define VTSS_PHY_EWIS_B2_NZ_EV 0x00800000
```

PMTICK B2 BIP (B1_ERR_CNT) not zero - only for 8488

Definition at line 2587 of file vtss_phy_10g_api.h.

4.30.2.35 VTSS_PHY_EWIS_B3_NZ_EV

```
#define VTSS_PHY_EWIS_B3_NZ_EV 0x01000000
```

PMTICK B3 BIP (B1_ERR_CNT) not zero - only for 8488

Definition at line 2588 of file vtss_phy_10g_api.h.

4.30.2.36 VTSS_PHY_EWIS_REIL_NZ_EV

```
#define VTSS_PHY_EWIS_REIL_NZ_EV 0x02000000
```

PMTICK REI-L (REIL_ERR_CNT) not zero - only for 8488

Definition at line 2589 of file vtss_phy_10g_api.h.

4.30.2.37 VTSS_PHY_EWIS_REIP_NZ_EV

```
#define VTSS_PHY_EWIS_REIP_NZ_EV 0x04000000
```

PMTICK REI-P (REIP_ERR_CNT) not zero - only for 8488

Definition at line 2590 of file vtss_phy_10g_api.h.

4.30.2.38 VTSS_PHY_EWIS_B1_THRESH_EV

```
#define VTSS_PHY_EWIS_B1_THRESH_EV 0x08000000
```

B1_THRESH_ERR - only for 8488

Definition at line 2591 of file vtss_phy_10g_api.h.

4.30.2.39 VTSS_PHY_EWIS_B2_THRESH_EV

```
#define VTSS_PHY_EWIS_B2_THRESH_EV 0x10000000
```

B2_THRESH_ERR - only for 8488

Definition at line 2592 of file vtss_phy_10g_api.h.

4.30.2.40 VTSS_PHY_EWIS_B3_THRESH_EV

```
#define VTSS_PHY_EWIS_B3_THRESH_EV 0x20000000
```

B3_THRESH_ERR - only for 8488

Definition at line 2593 of file vtss_phy_10g_api.h.

4.30.2.41 VTSS_PHY_EWIS_REIL_THRESH_EV

```
#define VTSS_PHY_EWIS_REIL_THRESH_EV 0x40000000
```

REIL_THRESH_ERR - only for 8488

Definition at line 2594 of file vtss_phy_10g_api.h.

4.30.2.42 VTSS_PHY_EWIS_REIP_THRESH_EV

```
#define VTSS_PHY_EWIS_REIP_THRESH_EV 0x80000000
```

REIp_THRESH_ERR - only for 8488

Definition at line 2595 of file vtss_phy_10g_api.h.

4.30.2.43 VTSS_PHY_10G_RX_LOS_EV

```
#define VTSS_PHY_10G_RX_LOS_EV 0x00000001
```

PHY RX LOS interrupt - 8256 specific

Definition at line 2602 of file vtss_phy_10g_api.h.

4.30.2.44 VTSS_PHY_10G_RX_LOL_EV [2/2]

```
#define VTSS_PHY_10G_RX_LOL_EV 0x00000002
```

PHY RXLOL interrupt - only on 8488

PHY RX LOL interrupt - 8256 specific

Definition at line 2603 of file vtss_phy_10g_api.h.

4.30.2.45 VTSS_PHY_10G_TX_LOL_EV [2/2]

```
#define VTSS_PHY_10G_TX_LOL_EV 0x00000004
```

PHY TXLOL interrupt - only on 8488

PHY TX LOL interrupt - 8256 specific

Definition at line 2604 of file vtss_phy_10g_api.h.

4.30.2.46 VTSS_PHY_10G_RX_CHAR_DEC_CNT_THRESH_EV

```
#define VTSS_PHY_10G_RX_CHAR_DEC_CNT_THRESH_EV 0x00000010
```

PHY RX character decode error - 8256 specific

Definition at line 2606 of file vtss_phy_10g_api.h.

4.30.2.47 VTSS_PHY_10G_TX_CHAR_ENC_CNT_THRESH_EV

```
#define VTSS_PHY_10G_TX_CHAR_ENC_CNT_THRESH_EV 0x00000020
```

PHY TX character encode error count - 8256 specific

Definition at line 2607 of file vtss_phy_10g_api.h.

4.30.2.48 VTSS_PHY_10G_RX_BLK_DEC_CNT_THRESH_EV

```
#define VTSS_PHY_10G_RX_BLK_DEC_CNT_THRESH_EV 0x00000040
```

PHY RX block decode error count - 8256 specific

Definition at line 2608 of file vtss_phy_10g_api.h.

4.30.2.49 VTSS_PHY_10G_TX_BLK_ENC_CNT_THRESH_EV

```
#define VTSS_PHY_10G_TX_BLK_ENC_CNT_THRESH_EV 0x00000080
```

PHY TX block encode error count- 8256 specific

Definition at line 2609 of file vtss_phy_10g_api.h.

4.30.2.50 VTSS_PHY_10G_RX_SEQ_CNT_THRESH_EV

```
#define VTSS_PHY_10G_RX_SEQ_CNT_THRESH_EV 0x00000100
```

PHY RX sequencing error count - 8256 specific

Definition at line 2610 of file vtss_phy_10g_api.h.

4.30.2.51 VTSS_PHY_10G_TX_SEQ_CNT_THRESH_EV

```
#define VTSS_PHY_10G_TX_SEQ_CNT_THRESH_EV 0x00000200
```

PHY TX sequencing error count - 8256 specific

Definition at line 2611 of file vtss_phy_10g_api.h.

4.30.2.52 VTSS_PHY_10G_FEC_UNFIXED_CNT_THRESH_EV

```
#define VTSS_PHY_10G_FEC_UNFIXED_CNT_THRESH_EV 0x00000400
```

PHY KR-FEC uncorrectable block count interrupt - 8256 specific

Definition at line 2612 of file vtss_phy_10g_api.h.

4.30.2.53 VTSS_PHY_10G_FEC_FIXED_CNT_THRESH_EV

```
#define VTSS_PHY_10G_FEC_FIXED_CNT_THRESH_EV 0x00000800
```

PHY KR-FEC corrected threshold - 8256 specific

Definition at line 2613 of file vtss_phy_10g_api.h.

4.30.2.54 VTSS_PHY_10G_HIGHBER_EV

```
#define VTSS_PHY_10G_HIGHBER_EV 0x00001000
```

PHY high bit Error - 8256 specific

Definition at line 2614 of file vtss_phy_10g_api.h.

4.30.2.55 VTSS_PHY_10G_RX_LINK_STAT_EV [1/2]

```
#define VTSS_PHY_10G_RX_LINK_STAT_EV 0x00002000
```

PHY Link status up/down interrupt - 8256 specific

Definition at line 2616 of file vtss_phy_10g_api.h.

4.30.2.56 VTSS_PHY_10G_RX_LINK_STAT_EV [2/2]

```
#define VTSS_PHY_10G_RX_LINK_STAT_EV 0x00002000
```

PHY Link status up/down interrupt - 8256 specific

Definition at line 2616 of file vtss_phy_10g_api.h.

4.30.2.57 VTSS_PHY_10G_GPIO_INT_AGG0_EV

```
#define VTSS_PHY_10G_GPIO_INT_AGG0_EV 0x00004000
```

PHY GPIO interrupt on Aggregator0 - 8256 specific

Definition at line 2617 of file vtss_phy_10g_api.h.

4.30.2.58 VTSS_PHY_10G_GPIO_INT_AGG1_EV

```
#define VTSS_PHY_10G_GPIO_INT_AGG1_EV 0x00008000
```

PHY GPIO interrupt on Aggregator1 - 8256 specific

Definition at line 2618 of file vtss_phy_10g_api.h.

4.30.2.59 VTSS_PHY_10G_GPIO_INT_AGG2_EV

```
#define VTSS_PHY_10G_GPIO_INT_AGG2_EV 0x00010000
```

PHY GPIO interrupt on Aggregator2 - 8256 specific

Definition at line 2619 of file vtss_phy_10g_api.h.

4.30.2.60 VTSS_PHY_10G_GPIO_INT_AGG3_EV

```
#define VTSS_PHY_10G_GPIO_INT_AGG3_EV 0x00020000
```

PHY GPIO interrupt on Aggregator3 - 8256 specific

Definition at line 2620 of file vtss_phy_10g_api.h.

4.30.2.61 VTSS_PHY_1G_LINE_AUTONEG_RESTART_EV

```
#define VTSS_PHY_1G_LINE_AUTONEG_RESTART_EV 0x00040000
```

PHY 1G Line side Autoneg restart event

Definition at line 2621 of file vtss_phy_10g_api.h.

4.30.2.62 VTSS_PHY_1G_HOST_AUTONEG_RESTART_EV

```
#define VTSS_PHY_1G_HOST_AUTONEG_RESTART_EV 0x00080000
```

PHY 1G Host side Autoneg restart event - 8256 specific

Definition at line 2622 of file vtss_phy_10g_api.h.

4.30.2.63 VTSS_PHY_10G_LINE_MAC_LOCAL_FAULT_EV

```
#define VTSS_PHY_10G_LINE_MAC_LOCAL_FAULT_EV 0x00100000
```

PHY 10G LINE MAC local fault event

Definition at line 2625 of file vtss_phy_10g_api.h.

4.30.2.64 VTSS_PHY_10G_HOST_MAC_LOCAL_FAULT_EV

```
#define VTSS_PHY_10G_HOST_MAC_LOCAL_FAULT_EV 0x00400000
```

PHY 10G HOST MAC local fault event

Definition at line 2626 of file vtss_phy_10g_api.h.

4.30.2.65 VTSS_PHY_10G_LINE_MAC_REMOTE_FAULT_EV

```
#define VTSS_PHY_10G_LINE_MAC_REMOTE_FAULT_EV 0x00800000
```

PHY 10G LINE MAC remote fault event

Definition at line 2627 of file vtss_phy_10g_api.h.

4.30.2.66 VTSS_PHY_10G_HOST_MAC_REMOTE_FAULT_EV

```
#define VTSS_PHY_10G_HOST_MAC_REMOTE_FAULT_EV 0x01000000
```

PHY 10G HOST MAC remote fault event

Definition at line 2628 of file vtss_phy_10g_api.h.

4.30.3 Typedef Documentation

4.30.3.1 vtss_gpio_10g_no_t

```
typedef u16 vtss_gpio_10g_no_t
```

GPIO type for 10G ports

Definition at line 412 of file vtss_phy_10g_api.h.

4.30.3.2 ckout_sel_t

```
typedef enum ckout_sel_ ckout_sel_t
```

10G Phy CKOUTs Enum

Malibu Only

4.30.3.3 vtss_32_cntr_t

```
typedef u32 vtss_32_cntr_t
```

32-bit counter

Definition at line 2175 of file vtss_phy_10g_api.h.

4.30.3.4 vtss_gpio_no_t

```
typedef u32 vtss_gpio_no_t
```

GPIO configured mode.

GPIO type for 1G ports

Definition at line 2300 of file vtss_phy_10g_api.h.

4.30.3.5 vtss_phy_10g_event_t

```
typedef u32 vtss_phy_10g_event_t
```

The type definition to contain the above defined evant mask

Definition at line 2597 of file vtss_phy_10g_api.h.

4.30.3.6 vtss_phy_10g_extnd_event_t

```
typedef u32 vtss_phy_10g_extnd_event_t
```

The type definition to contain the above defined extended event mask

Definition at line 2631 of file vtss_phy_10g_api.h.

4.30.4 Enumeration Type Documentation

4.30.4.1 oper_mode_t

```
enum oper_mode_t
```

10G Phy operating mode enum type

Enumerator

VTSS_PHY_LAN_MODE	LAN mode: Single clock (XREFCK=156,25 MHz), no recovered clock output
VTSS_PHY_WAN_MODE	WAN mode: 848X: Dual clock (XREFCK=156,25 MHz, WREFCK=155,52 MHz), no recovered clock output Venice: Single clock (XREFCK), no recovered clock output
VTSS_PHY_1G_MODE	8488: 1G pass-through mode Venice: 1G mode, Single clock (XREFCK=156,25 MHz), no recovered clock output For 1G operation, customer should select VTSS_MEDIA_TYPE_SR for all media applications and specify the operation is in 1G mode

VTSS_PHY_LAN_SYNCE_MODE	LAN SyncE:
	if hl_clk_synth == 1:
	8488: Single clock (XREFCK=156,25 MHz), recovered clock output enabled
	Venice: Single clock (XREFCK=156,25 MHz), recovered clock output enabled
	if hl_clk_synth == 0:
	8488: Dual clock (XREFCK=156,25 MHz, SREFCK=156,25 MHz), recovered clock output enabled
	Venice: Dual clock (XREFCK=156,25 MHz, SREFCK=156,25 MHz), recovered clock output enabled

VTSS_PHY_WAN_SYNCE_MODE	WAN SyncE:
	if hl_clk_synth == 1:
	8488: Single clock (WREFCK=155,52 MHz or 622,08 MHz), recovered clock output enabled
	Venice: Single clock (XREFCK=156,25 MHz), recovered clock output enabled
	if hl_clk_synth == 0:
	8488: Dual clock (WREFCK=155,52 MHz or 622,08 MHz, SREFCK=155,52 MHz), recovered clock output enabled
	Venice: Dual clock (XREFCK=156,25 MHz, SREFCK=155,52 MHz), recovered clock output enabled

VTSS_PHY_LAN_MIXED_SYNCE_MODE	8488: Channels are in different modes, channel being configured is in LAN Venice: Same as VTSS_PHY_LAN_SYNCE_MODE
VTSS_PHY_WAN_MIXED_SYNCE_MODE	8488: Channels are in different modes, channel being configured is in WAN Venice: Same as VTSS_PHY_WAN_SYNCE_MODE
VTSS_PHY_REPEATER_MODE	Malibu: Repeater mode,better jitter performance

Definition at line 45 of file vtss_phy_10g_api.h.

4.30.4.2 vtss_wrefclk_t

```
enum vtss_wrefclk_t
```

Modes for WAN reference clock.

Enumerator

VTSS_WREFCLK_155_52	WREFCLK = 155.52Mhz - WAN ref clock
VTSS_WREFCLK_622_08	WREFCLK = 622.08Mhz - WAN ref clock

Definition at line 80 of file vtss_phy_10g_api.h.

4.30.4.3 vtss_phy_interface_mode

`enum vtss_phy_interface_mode`

Phy Interface modes.

Enumerator

VTSS_PHY_XAUI_XFI	XAUI <-> XFI - Interface mode.
VTSS_PHY_XGMII_XFI	XGMII <-> XFI - Interface mode. Only for VSC8486
VTSS_PHY_RXAUI_XFI	RXAUI <-> XFI - Interface mode. Only for Venice
VTSS_PHY_SGMII_LANE_0_XFI	SGMII <-> XFI - LANE 0. Only for Venice
VTSS_PHY_SGMII_LANE_3_XFI	SGMII <-> XFI - LANE 3. Only for Venice
VTSS_PHY_SFI_XFI	SFI <-> XFI - Interface mode. Only for Malibu

Definition at line 94 of file vtss_phy_10g_api.h.

4.30.4.4 vtss_recvrd_t

`enum vtss_recvrd_t`

Modes for recovered clock.

Enumerator

VTSS_RECVRD_RXCLKOUT	RXCLKOUT is used for recovered clock
VTSS_RECVRD_TXCLKOUT	TXCLKOUT is used for recovered clock

Definition at line 104 of file vtss_phy_10g_api.h.

4.30.4.5 vtss_recvrdclk_cdr_div_t

`enum vtss_recvrdclk_cdr_div_t`

Modes for recovered clock divisor.

Enumerator

VTSS_RECVRDCLK_CDR_DIV_64	recovered clock is /64
VTSS_RECVRDCLK_CDR_DIV_66	recovered clock is /66

Definition at line 110 of file vtss_phy_10g_api.h.

4.30.4.6 vtss_srefclk_div_t

`enum vtss_srefclk_div_t`

Modes for Synch-E recovered clock.

Enumerator

VTSS_SREFCLK_DIV_64	SREFCLK/64 ,valid for LAN,WAN
VTSS_SREFCLK_DIV_66	SREFCLK/66 ,valid for LAN
VTSS_SREFCLK_DIV_16	SREFCLK/16 ,valid for WAN

Definition at line 116 of file vtss_phy_10g_api.h.

4.30.4.7 vtss_wref_clk_div_t

`enum vtss_wref_clk_div_t`

Modes for WREFCLK clock divisor.

Enumerator

VTSS_WREFCLK_NONE	NA
VTSS_WREFCLK_DIV_16	WREFCLK/16

Definition at line 124 of file vtss_phy_10g_api.h.

4.30.4.8 apc_ib_regulator_t

`enum apc_ib_regulator_t`

APC Rx regulator mode.

Enumerator

VTSS_APC_IB_SFP_PLUS_ZR	SFP+ ZR module.
VTSS_APC_IB_BACKPLANE	Backplane application.

Definition at line 130 of file vtss_phy_10g_api.h.

4.30.4.9 ddr_mode_t

`enum ddr_mode_t`

Interleave mode.

Enumerator

VTSS_DDR_MODE_A	Interleave mode with A alignment symbol based byte re-ordering
VTSS_DDR_MODE_K	Interleave mode with K coma based byte re-ordering
VTSS_DDR_MODE_M	Interleave mode with A alignment and 8b10b decoding disabled

Definition at line 136 of file vtss_phy_10g_api.h.

4.30.4.10 clk_mstr_t

`enum clk_mstr_t`

Clock master.

Enumerator

VTSS_CLK_MSTR_INTERNAL	Master clock is internal
VTSS_CLK_MSTR_EXTERNAL	Master clock is external

Definition at line 143 of file vtss_phy_10g_api.h.

4.30.4.11 vtss_rptr_rate_t

`enum vtss_rptr_rate_t`

Repeater Data rate.

Enumerator

VTSS_RPTR_RATE_NONE	None
VTSS_RPTR_RATE_10_3125	LAN rate=10.3125 Gbps,
VTSS_RPTR_RATE_9_9532	WAN rate=9.9532 Gbps
VTSS_RPTR_RATE_11_3	rate=11.3 Gbps,clock 171Mhz
VTSS_RPTR_RATE_10_5187	Fiber channel rate=10.51875 Gbps,
VTSS_RPTR_RATE_1_25	1G rate=1.25Gbps
VTSS_RPTR_RATE_10_709	OTU2 rate= 10.709 Gbps
VTSS_RPTR_RATE_11_095727	OTU2E rate = 11.095727 Gbps
VTSS_RPTR_RATE_11_05	OTU1E rate = 11.05 Gbps

Definition at line 149 of file vtss_phy_10g_api.h.

4.30.4.12 vtss_phy_10g_media_t

```
enum vtss_phy_10g_media_t
```

10G Phy Media type

Enumerator

VTSS_MEDIA_TYPE_SR	SR,10GBASE-SR
VTSS_MEDIA_TYPE_SR2	SR,10GBASE-SR
VTSS_MEDIA_TYPE_DAC	DAC,Direct attach cable
VTSS_MEDIA_TYPE_ZR	ZR,10GBASE-ZR
VTSS_MEDIA_TYPE_KR	KR,10GBASE-KR
VTSS_MEDIA_TYPE_SR_SC	SR,10GBASE-SR with smart control
VTSS_MEDIA_TYPE_SR2_SC	SR,10GBASE-SR with smart control
VTSS_MEDIA_TYPE_DAC_SC	DAC,Direct attach cable with smart control
VTSS_MEDIA_TYPE_ZR_SC	ZR,10GBASE-ZR with smart control
VTSS_MEDIA_TYPE_ZR2_SC	ZR,10GBASE-ZR with smart control with ld_lev_ini:40
VTSS_MEDIA_TYPE_KR_SC	KR,10GBASE-KR with smart control
VTSS_MEDIA_TYPE_NONE	None

Definition at line 170 of file vtss_phy_10g_api.h.

4.30.4.13 vtss_phy_6g_link_partner_distance_t

```
enum vtss_phy_6g_link_partner_distance_t
```

6G serdes link partner distance selection

Enumerator

VTSS_6G_LINK_SHORT_RANGE	distance between 6G macro and serdes macro of link partner is less (direct connection)
VTSS_6G_LINK_LONG_RANGE	distance between 6G macro and serdes macro of link parter is more (connection via backplanes)

Definition at line 186 of file vtss_phy_10g_api.h.

4.30.4.14 vtss_phy_10g_ib_apc_op_mode_t

`enum vtss_phy_10g_ib_apc_op_mode_t`

10G SERDES APC operation

Enumerator

VTSS_IB_APC_AUTO	AUTO Operation
VTSS_IB_APC_MANUAL	Manual operation
VTSS_IB_APC_FREEZE	Freeze
VTSS_IB_APC_RESET	Reset
VTSS_IB_APC_RESTART	Restart APC
VTSS_IB_APC_NONE	None

Definition at line 197 of file vtss_phy_10g_api.h.

4.30.4.15 vtss_channel_t

`enum vtss_channel_t`

Channel modes - Auto is recommended.

Enumerator

VTSS_CHANNEL_AUTO	Automatically detects the channel id based on the phy order. The phys be setup in the consecutive order, from the lowest MDIO to highest MDIO address
VTSS_CHANNEL_0	Channel id is hardcoded to 0
VTSS_CHANNEL_1	Channel id is hardcoded to 1
VTSS_CHANNEL_2	Channel id is hardcoded to 2
VTSS_CHANNEL_3	Channel id is hardcoded to 3

Definition at line 321 of file vtss_phy_10g_api.h.

4.30.4.16 vtss_recvrd_clkout_t

enum vtss_recvrd_clkout_t

Modes for (rx/tx) recovered clock output.

Enumerator

VTSS_RECVRD_CLKOUT_DISABLE	recovered clock output is disabled
VTSS_RECVRD_CLKOUT_LINE_SIDE_RX_CLOCK	recovered clock output is derived from Lineside Rx clock
VTSS_RECVRD_CLKOUT_LINE_SIDE_TX_CLOCK	recovered clock output is derived from Lineside Tx clock

Definition at line 699 of file vtss_phy_10g_api.h.

4.30.4.17 vtss_phy_10g_srefclk_freq_t

enum vtss_phy_10g_srefclk_freq_t

10G Phy sref clock input frequency

Enumerator

VTSS_PHY_10G_SREFCLK_156_25	156,25 MHz
VTSS_PHY_10G_SREFCLK_125_00	125,00 MHz
VTSS_PHY_10G_SREFCLK_155_52	155,52 MHz
VTSS_PHY_10G_SREFCLK_INVALID	Other values are not allowed

Definition at line 779 of file vtss_phy_10g_api.h.

4.30.4.18 vtss_phy_10g_ckout_freq_t

enum vtss_phy_10g_ckout_freq_t

10G Phy clock frequency

Malibu only

Enumerator

VTSS_PHY_10G_CLK_FULL_RATE	LAN:332.25 MHz, WAN:311.04MHz, 1G:125MHz
VTSS_PHY_10G_CLK_DIVIDE_BY↩ _2	LAN:161.12 MHz, WAN:155.52MHz, 1G:62.5MHz
VTSS_PHY_10G_CLK_INVALID	Other values are not allowed

Definition at line 831 of file vtss_phy_10g_api.h.

4.30.4.19 vtss_ckout_data_sel_t

enum vtss_ckout_data_sel_t

Modes for recovered clock output.

Applicable to Malibu only

Enumerator

VTSS_CKOUT_LINE0_TX_CLOCK	Line0 Transmit clock
VTSS_CKOUT_LINE1_TX_CLOCK	Line1 Transmit clock
VTSS_CKOUT_LINE2_TX_CLOCK	Line2 Transmit clock
VTSS_CKOUT_LINE3_TX_CLOCK	Line3 Transmit clock
VTSS_CKOUT_HOST0_TX_CLOCK	Host0 Transmit clock
VTSS_CKOUT_HOST1_TX_CLOCK	Host1 Transmit clock
VTSS_CKOUT_HOST2_TX_CLOCK	Host2 Transmit clock
VTSS_CKOUT_HOST3_TX_CLOCK	Host3 Transmit clock
VTSS_CKOUT_LINE0_RECVRD_CLOCK	Line0 Recovered clock
VTSS_CKOUT_LINE1_RECVRD_CLOCK	Line1 Recovered clock
VTSS_CKOUT_LINE2_RECVRD_CLOCK	Line2 Recovered clock
VTSS_CKOUT_LINE3_RECVRD_CLOCK	Line3 Recovered clock
VTSS_CKOUT_HOST0_RECVRD_CLOCK	Host0 Recovered clock
VTSS_CKOUT_HOST1_RECVRD_CLOCK	Host1 Recovered clock
VTSS_CKOUT_HOST2_RECVRD_CLOCK	Host2 Recovered clock
VTSS_CKOUT_HOST3_RECVRD_CLOCK	Host3 Recovered clock
VTSS_CKOUT_HOST_PLL_CLOCK	Host PLL clock
VTSS_CKOUT_LINE_PLL_CLOCK	Line PLL clock
VTSS_CKOUT_CSR_CLOCK	CSR clock
VTSS_CKOUT_LTC_CLOCK	LTC clock
VTSS_CKOUT_DF2F_CLOCK	Df2f clock
VTSS_CKOUT_F2DF_CLOCK	F2df clock
VTSS_CKOUT_DEBUG1	Debug1
VTSS_CKOUT_DEBUG2	Debug2
VTSS_CKOUT_OSCILLATOR_OUTPUT	Oscillator output

Definition at line 841 of file vtss_phy_10g_api.h.

4.30.4.20 vtss_phy_10g_squelch_src_t

enum vtss_phy_10g_squelch_src_t

squelch control source

Applicable to Malibu only

Enumerator

VTSS_CKOUT_SQUELCH_SRC_GPIO0	GPIO0 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_GPIO1	GPIO1 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_GPIO2	GPIO2 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_GPIO3	GPIO3 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_GPIO4	GPIO4 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_GPIO5	GPIO5 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_GPIO6	GPIO6 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_GPIO7	GPIO7 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_LINE0	Link status from Line0 source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_LINE1	Link status from Line1 source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_LINE2	Link status from Line2 source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_LINE3	Link status from Line3 source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_HOST0	Link status from Host0 source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_HOST1	Link status from Host1 source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_HOST2	Link status from Host2 source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_HOST3	Link status from Host3 source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LOS_LINE0	Serdes LOS from Line0 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LOS_LINE1	Serdes LOS from Line1 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LOS_LINE2	Serdes LOS from Line2 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LOS_LINE3	Serdes LOS from Line3 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LOS_HOST0	Serdes LOS from Host0 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LOS_HOST1	Serdes LOS from Host1 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LOS_HOST2	Serdes LOS from Host2 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LOS_HOST3	Serdes LOS from Host3 as source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_LINE0_KR	Link status from Line0 KR source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_LINE1_KR	Link status from Line1 KR source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_LINE2_KR	Link status from Line2 KR source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_LINE3_KR	Link status from Line3 KR source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_HOST0_KR	Link status from Host0 KR source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_HOST1_KR	Link status from Host1 KR source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_HOST2_KR	Link status from Host2 KR source of auto squelch
VTSS_CKOUT_SQUELCH_SRC_LINK_HOST3_KR	Link status from Host3 KR source of auto squelch
VTSS_CKOUT_NO_SQUELCH	No squelch(32-63)

Definition at line 873 of file vtss_phy_10g_api.h.

4.30.4.21 vtss_phy_10g_clk_sel_t

```
enum vtss_phy_10g_clk_sel_t
```

Modes of recovered clocks for ckout and sckout pins.

Applicable to Malibu only

Enumerator

VTSS_PHY_10G_LINE0_RECVRD_CLOCK	Line0 Recovered clock
VTSS_PHY_10G_LINE1_RECVRD_CLOCK	Line1 Recovered clock
VTSS_PHY_10G_LINE2_RECVRD_CLOCK	Line2 Recovered clock
VTSS_PHY_10G_LINE3_RECVRD_CLOCK	Line3 Recovered clock
VTSS_PHY_10G_HOST0_RECVRD_CLOCK	Host0 Recovered clock
VTSS_PHY_10G_HOST1_RECVRD_CLOCK	Host1 Recovered clock
VTSS_PHY_10G_HOST2_RECVRD_CLOCK	Host2 Recovered clock
VTSS_PHY_10G_HOST3_RECVRD_CLOCK	Host3 Recovered clock
VTSS_PHY_10G_SREFCLK	SREFCLK
VTSS_PHY_10G_SYNC_DISABLE	Sync Disable 9-15

Definition at line 914 of file vtss_phy_10g_api.h.

4.30.4.22 vtss_phy_10g_recvrd_clk_sel_t

```
enum vtss_phy_10g_recvrd_clk_sel_t
```

Modes of recovered clock selection.

Applicable to Malibu only

Enumerator

VTSS_PHY_10G_USE_LINE0_RECVRD_CLOCK	All lines using Line0 Recovered clock
VTSS_PHY_10G_USE_LINE1_RECVRD_CLOCK	All lines using Line1 Recovered clock
VTSS_PHY_10G_USE_LINE2_RECVRD_CLOCK	All lines using Line2 Recovered clock
VTSS_PHY_10G_USE_LINE3_RECVRD_CLOCK	All lines using Line3 Recovered clock
VTSS_PHY_10G_USE_HOST0_RECVRD_CLOCK	All lines using Host0 Recovered clock
VTSS_PHY_10G_USE_HOST1_RECVRD_CLOCK	All lines using Host1 Recovered clock
VTSS_PHY_10G_USE_HOST2_RECVRD_CLOCK	All lines using Host2 Recovered clock
VTSS_PHY_10G_USE_HOST3_RECVRD_CLOCK	All lines using Host3 Recovered clock
VTSS_PHY_10G_USE_SREFCLK_CLOCK	All lines using SREFCLK
VTSS_PHY_10G_USE_DEFAULT_RECVRD_CL↩ OCK	Use Recvrd Clk from the respcective line LineX Uses recovered clock from LineX only

Definition at line 932 of file vtss_phy_10g_api.h.

4.30.4.23 ckout_sel_

```
enum ckout_sel_
```

10G Phy CKOUTs Enum

Malibu Only

Definition at line 951 of file vtss_phy_10g_api.h.

4.30.4.24 vtss_phy_10g_sckout_freq_t

enum vtss_phy_10g_sckout_freq_t

10G Phy sckout clock input frequency

Enumerator

VTSS_PHY_10G_SCKOUT_156_25	156,25 MHz
VTSS_PHY_10G_SCKOUT_125_00	125,00 MHz
VTSS_PHY_10G_SCKOUT_INVALID	Other values are not allowed

Definition at line 972 of file vtss_phy_10g_api.h.

4.30.4.25 vtss_phy_10g_rx_macro_t

enum vtss_phy_10g_rx_macro_t

10G Phy Rx MACRO Configuration

Malibu Only

Enumerator

VTSS_PHY_10G_RX_MACRO_LINE	Rx MACRO Line
VTSS_PHY_10G_RX_MACRO_HOST	Rx MACRO Host
VTSS_PHY_10G_RX_MACRO_SREFCLK	Rx MACRO SREFCLK

Definition at line 1110 of file vtss_phy_10g_api.h.

4.30.4.26 vtss_phy_10g_tx_macro_t

enum vtss_phy_10g_tx_macro_t

10G Phy tx MACRO Configuration

Malibu Only

Enumerator

VTSS_PHY_10G_TX_MACRO_LINE	Tx MACRO Line
VTSS_PHY_10G_TX_MACRO_HOST	Tx MACRO Host
VTSS_PHY_10G_TX_MACRO_SCKOUT	Tx MACRO SREFCLK

Definition at line 1120 of file vtss_phy_10g_api.h.

4.30.4.27 vtss_phy_10g_clause_37_remote_fault_t

enum vtss_phy_10g_clause_37_remote_fault_t

Auto-negotiation remote fault type.

Advertisement Word (Refer to IEEE 802.3 Clause 37): MSB LSB D15 D14 D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0 +—+—+—+—+—+—+—+—+—+—+—+—+—+—+—+—+ | NP | Ack| RF2| R↩ F1|rsvd|rsvd|rsvd| PS2| PS1| HD | FD |rsvd|rsvd|rsvd|rsvd|rsvd| +—+—+—+—+—+—+—+—+—+—+—+—+—+—+—+—+ —+—+—+—+—+

Enumerator

VTSS_PHY_10G_CLAUSE_37_RF_LINK_OK	Link OK
VTSS_PHY_10G_CLAUSE_37_RF_OFFLINE	Off line
VTSS_PHY_10G_CLAUSE_37_RF_LINK_FAILURE	Link failure
VTSS_PHY_10G_CLAUSE_37_RF_AUTONEG_ERROR	Autoneg error

Definition at line 1497 of file vtss_phy_10g_api.h.

4.30.4.28 vtss_lb_type_t

enum vtss_lb_type_t

10G loopback types

Enumerator

VTSS_LB_NONE	No looback
VTSS_LB_SYSTEM_XS_SHALLOW	System Loopback B, XAUI -> XS -> XAUI 4x800E.13, Venice: H2
VTSS_LB_SYSTEM_XS_DEEP	System Loopback C, XAUI -> XS -> XAUI 4x800F.2, Venice: N.A.
VTSS_LB_SYSTEM_PCS_SHALLOW	System Loopback E, XAUI -> PCS FIFO -> XAUI 3x8005.2, Venice: N.A.
VTSS_LB_SYSTEM_PCS_DEEP	System Loopback G, XAUI -> PCS -> XAUI 3x0000.14, Venice: H3
VTSS_LB_SYSTEM_PMA	System Loopback J, XAUI -> PMA -> XAUI 1x0000.0, Venice: H4
VTSS_LB_NETWORK_XS_SHALLOW	Network Loopback D, XFI -> XS -> XFI 4x800F.1, Venice: N.A.
VTSS_LB_NETWORK_XS_DEEP	Network Loopback A, XFI -> XS -> XFI 4x0000.1 4x800E.13=0, Venice: L1

Enumerator

VTSS_LB_NETWORK_PCS	Network Loopback F, XFI -> PCS -> XFI 3x8005.3, Venice: L2
VTSS_LB_NETWORK_WIS	Network Loopback H, XFI -> WIS -> XFI 2xE600.0, Venice: N.A.
VTSS_LB_NETWORK_PMA	Network Loopback K, XFI -> PMA -> XFI 1x8000.8, Venice: L3
VTSS_LB_H2	Host Loopback 2, 40-bit XAUI-PHY interface Mirror XAUI data
VTSS_LB_H3	Host Loopback 3, 64-bit PCS after the gearbox FF00 repeating IEEE PCS system loopback
VTSS_LB_H4	Host Loopback 4, 64-bit WIS FF00 repeating IEEE WIS system loopback
VTSS_LB_H5	Host Loopback 5, 1-bit SFP+ after SerDes Mirror XAUI data IEEE PMA system loopback
VTSS_LB_H6	Host Loopback 6, 32-bit XAUI-PHY interface Mirror XAUI data
VTSS_LB_L0	Line Loopback 0, 4-bit XAUI before SerDes Mirror SFP+ data
VTSS_LB_L1	Line Loopback 1, 4-bit XAUI after SerDes Mirror SFP+ data IEEE PHY-XS network loopback
VTSS_LB_L2	Line Loopback 2, 64-bit XGMII after FIFO Mirror SFP+ data
VTSS_LB_L3	Line Loopback 3, 64-bit PMA interface Mirror SFP+ data

Definition at line 1598 of file vtss_phy_10g_api.h.

4.30.4.29 vtss_phy_10g_power_t

```
enum vtss_phy_10g_power_t
```

10G Phy power setting

Enumerator

VTSS_PHY_10G_POWER_ENABLE	Enable Phy power for all sublayers
VTSS_PHY_10G_POWER_DISABLE	Disable Phy power for all sublayers

Definition at line 1699 of file vtss_phy_10g_api.h.

4.30.4.30 vtss_phy_10g_failover_mode_t

```
enum vtss_phy_10g_failover_mode_t
```

10G Phy Failover Mode Setting

Enumerator

VTSS_PHY_10G_PMA_TO_FROM_XAUI_NORMAL	PMA_0/1 to XAUI_0/1. 8487: XAUI 0 to PMA 0
VTSS_PHY_10G_PMA_TO_FROM_XAUI_CROS↩ SED	PMA_0/1 to XAUI_1/0. 8487: XAUI 1 to PMA 0

Enumerator

VTSS_PHY_10G_PMA_0_TO_FROM_XAUI_0_TO_XAUI_1	PMA 0 to/from XAUI 0 and to XAUI 1
VTSS_PHY_10G_PMA_0_TO_FROM_XAUI_1_TO_XAUI_0	PMA 0 to/from XAUI 1 and to XAUI 0
VTSS_PHY_10G_PMA_1_TO_FROM_XAUI_0_TO_XAUI_1	PMA 1 to/from XAUI 0 and to XAUI 1. VSC8487:N/A
VTSS_PHY_10G_PMA_1_TO_FROM_XAUI_1_TO_XAUI_0	PMA 1 to/from XAUI 1 and to XAUI 0. VSC8487:N/A

Definition at line 1749 of file vtss_phy_10g_api.h.

4.30.4.31 vtss_phy_10g_auto_failover_event_t

enum vtss_phy_10g_auto_failover_event_t

10G Phy Automatic Failover Event Setting

Enumerator

VTSS_PHY_10G_AUTO_FAILOVER_EVENT_PCS_LINK_STATUS	PCS link status
VTSS_PHY_10G_AUTO_FAILOVER_EVENT_SERDES_LOS	LOS from SERDES
VTSS_PHY_10G_AUTO_FAILOVER_EVENT_WIS_LOF	LOF from Line WIS
VTSS_PHY_10G_AUTO_FAILOVER_EVENT_GPIO	External GPIO input
VTSS_PHY_10G_AUTO_FAILOVER_EVENT_NONE	Manual switching to be done

Definition at line 1788 of file vtss_phy_10g_api.h.

4.30.4.32 vtss_phy_10g_auto_failover_filter_t

enum vtss_phy_10g_auto_failover_filter_t

10G PHY Automatic Failover Filter

Enumerator

VTSS_PHY_10G_AUTO_FAILOVER_FILTER_NONE	No filter configuration
VTSS_PHY_10G_AUTO_FAILOVER_FILTER_CNT_B2316	False condition, upper 8 bits of 24-bit threshold
VTSS_PHY_10G_AUTO_FAILOVER_FILTER_CNT_B70	False condition, lower 8 bits of 24-bit threshold
VTSS_PHY_10G_AUTO_FAILOVER_FILTER_CNT_A2316	True condition, upper 8 bits of 24-bit threshold
VTSS_PHY_10G_AUTO_FAILOVER_FILTER_CNT_A70	True condition, lower 8 bits of 24-bit threshold

Definition at line 1798 of file vtss_phy_10g_api.h.

4.30.4.33 vtss_phy_10g_vscope_scan_t

enum vtss_phy_10g_vscope_scan_t

VSCOPE scan types.

Enumerator

VTSS_PHY_10G_FAST_SCAN	selects the fast scan feature
VTSS_PHY_10G_FAST_SCAN_PLUS	selects the fast scan feature with diagonal points
VTSS_PHY_10G_QUICK_SCAN	selects the quick scan feature
VTSS_PHY_10G_FULL_SCAN	selects the full scan freature

Definition at line 1848 of file vtss_phy_10g_api.h.

4.30.4.34 vtss_phy_10g_pkt_mon_rst_t

enum vtss_phy_10g_pkt_mon_rst_t

10G PHY Packet monitor configuration

Enumerator

VTSS_PHY_10G_PKT_MON_RST_ALL	Reset all counters
VTSS_PHY_10G_PKT_MON_RST_GOOD	Reset good crc counter
VTSS_PHY_10G_PKT_MON_RST_BAD	Reset bad crc counter
VTSS_PHY_10G_PKT_MON_RST_FRAG	Reset Fragment counter
VTSS_PHY_10G_PKT_MON_RST_LFAULT	Reset local fault counter
VTSS_PHY_10G_PKT_MON_RST_BER	Reset Ber counter
VTSS_PHY_10G_PKT_MON_RST_NONE	None

Definition at line 2165 of file vtss_phy_10g_api.h.

4.30.4.35 vtss_10g_phy_gpio_t

enum vtss_10g_phy_gpio_t

GPIO configured mode.

Enumerator

VTSS_10G_PHY_GPIO_NOT_INITIALIZED	This GPIO pin has has been initialized by a call to API from application. aregisters contain power-up default value
VTSS_10G_PHY_GPIO_OUT	Output enabled
VTSS_10G_PHY_GPIO_IN	Input enabled

Enumerator

VTSS_10G_PHY_GPIO_WIS_INT	Output WIS interrupt channel 0 or 1 (depending on port_no) enabled
VTSS_10G_PHY_GPIO_1588_LOAD_SAVE	Input interrupt generated on falling edge Input interrupt generated on raising edge Input interrupt generated on raising and falling edge Input 1588 load/save function
VTSS_10G_PHY_GPIO_1588_1PPS_0	Output 1588 1PPS from channel 0 function
VTSS_10G_PHY_GPIO_1588_1PPS_1	Output 1588 1PPS from channel 1 function
VTSS_10G_PHY_GPIO_1588_1PPS_2	Output 1588 1PPS from channel 2 function
VTSS_10G_PHY_GPIO_1588_1PPS_3	Output 1588 1PPS from channel 3 function
VTSS_10G_PHY_GPIO_PCS_RX_FAULT	PCS_RX_FAULT (from channel 0 or 1) is transmitted on GPIO
VTSS_10G_PHY_GPIO_SET_I2C_MASTER	Used in communicating with I2C slave, like SPP+
VTSS_10G_PHY_GPIO_TX_ENABLE	Transmit enable , MALIBU
VTSS_10G_PHY_GPIO_LINE_PLL_STATUS	Line PLL Status , MALIBU
VTSS_10G_PHY_GPIO_HOST_PLL_STATUS	Host PLL Status , MALIBU
VTSS_10G_PHY_GPIO_RCOMP_BUSY	RCOMP busy Status , MALIBU
VTSS_10G_PHY_GPIO_CHAN_INT_0	Interrupt 0 from channel , MALIBU
VTSS_10G_PHY_GPIO_CHAN_INT_1	Interrupt 1 from channel , MALIBU
VTSS_10G_PHY_GPIO_1588_INT	1588 Interrupt , MALIBU
VTSS_10G_PHY_GPIO_TS_FIFO_EMPTY	TS FIFO empty , MALIBU
VTSS_10G_PHY_GPIO_AGG_INT_0	Aggregated interrupt 0 , MALIBU
VTSS_10G_PHY_GPIO_AGG_INT_1	Aggregated interrupt 1 , MALIBU
VTSS_10G_PHY_GPIO_AGG_INT_2	Aggregated interrupt 2 , MALIBU
VTSS_10G_PHY_GPIO_AGG_INT_3	Aggregated interrupt 3 , MALIBU
VTSS_10G_PHY_GPIO_PLL_INT_0	Interrupt 0 from PLL , MALIBU
VTSS_10G_PHY_GPIO_PLL_INT_1	Interrupt 0 from PLL , MALIBU
VTSS_10G_PHY_GPIO_SET_I2C_SLAVE	I2C Slave set , MALIBU
VTSS_10G_PHY_GPIO_CRSS_INT	Cross Connect Interrupt , MALIBU
VTSS_10G_PHY_GPIO_LED	LED Setting , MALIBU
VTSS_10G_PHY_GPIO_DRIVE_LOW	GPIO output to LOW , MALIBU
VTSS_10G_PHY_GPIO_DRIVE_HIGH	GPIO output to HIGH , MALIBU

Definition at line 2306 of file vtss_phy_10g_api.h.

4.30.4.36 vtss_gpio_10g_gpio_intr_sgnl_t

```
enum vtss_gpio_10g_gpio_intr_sgnl_t
```

GPIO internal signal types.

Enumerator

VTSS_10G_GPIO_INTR_SGNL_I2C_MSTR_CLK↩ _OUT	GPIO outupt I2C master data out
VTSS_10G_GPIO_INTR_SGNL_LED_TX	GPIO outupt I2C master clock out
VTSS_10G_GPIO_INTR_SGNL_LED_RX	LED transmit

Enumerator

VTSS_10G_GPIO_INTR_SGNL_RX_ALARM	LED receive
VTSS_10G_GPIO_INTR_SGNL_TX_ALARM	RX Alarm
VTSS_10G_GPIO_INTR_SGNL_HOST_LINK	TX Alarm
VTSS_10G_GPIO_INTR_SGNL_LINE_LINK	Host Link status
VTSS_10G_GPIO_INTR_SGNL_LINE_KR_8b10b_2GPIO	Line Link status
VTSS_10G_GPIO_INTR_SGNL_LINE_KR_10b_2GPIO	KR 8b10b
VTSS_10G_GPIO_INTR_SGNL_ROSI_PULSE	KR 10b
VTSS_10G_GPIO_INTR_SGNL_ROSI_SDATA	ROSI Pulse
VTSS_10G_GPIO_INTR_SGNL_ROSI_SCLK	ROSI sdata
VTSS_10G_GPIO_INTR_SGNL_TOSI_PULSE	ROSI sclock
VTSS_10G_GPIO_INTR_SGNL_TOSI_SCLK	TOSI Pulse
VTSS_10G_GPIO_INTR_SGNL_LINE_PCS1G_LINK	TOSI sclock
VTSS_10G_GPIO_INTR_SGNL_LINE_PCS_RX_STAT	Line PCS1G link status
VTSS_10G_GPIO_INTR_SGNL_CLIENT_PCS1G_LINK	Line PCS RX link status
VTSS_10G_GPIO_INTR_SGNL_HOST_PCS_RX_STAT	Client PCS1G link status
VTSS_10G_GPIO_INTR_SGNL_HOST_SD10G_IB_SIG	Host PCS RX status
VTSS_10G_GPIO_INTR_SGNL_LINE_SD10G_IB_SIG	Host SERDES 10G 1B signal
VTSS_10G_GPIO_INTR_SGNL_HPCS_INTR	Line SERDES 10G 1B signal
VTSS_10G_GPIO_INTR_SGNL_LPCS_INTR	HPCS interrupt
VTSS_10G_GPIO_INTR_SGNL_CLIENT_PCS1G_INTR	LPCS interrupt
VTSS_10G_GPIO_INTR_SGNL_LINE_PCS1G_INTR	Client PCS1G interrupt
VTSS_10G_GPIO_INTR_SGNL_WIS_INT0	Line PCS1G interrupt
VTSS_10G_GPIO_INTR_SGNL_HOST_PMA_INT	WIS interrupt 0
VTSS_10G_GPIO_INTR_SGNL_LINE_PMA_INT	Host PMA interrupt
VTSS_10G_GPIO_INTR_SGNL_DATA_ACT_TX	Line PMA interrupt
VTSS_10G_GPIO_INTR_SGNL_DATA_ACT_RX	TX data activity
VTSS_10G_GPIO_INTR_SGNL_HDATA_ACT_TX	RX data activity
VTSS_10G_GPIO_INTR_SGNL_HDATA_ACT_RX	Host TX data activity
VTSS_10G_GPIO_INTR_SGNL_XGMII_PAUS_EGR	Host RX data activity
VTSS_10G_GPIO_INTR_SGNL_XGMII_PAUS_ING	XGMI pause egress
VTSS_10G_GPIO_INTR_SGNL_RX_PCS_PAUS	XGMI pause ingress
VTSS_10G_GPIO_INTR_SGNL_TX_PCS_PAUS	PCS RX Pause
VTSS_10G_GPIO_INTR_SGNL_RX_WIS_PAUS	PCS TX Pause
VTSS_10G_GPIO_INTR_SGNL_TX_WIS_PAUS	WIS RX Pause
VTSS_10G_GPIO_INTR_SGNL_ETH_CHAN_DIS	WIS TX Pause
VTSS_10G_GPIO_INTR_SGNL_MACSEC_1588_SFD_LANE	Ethernet channel disable
VTSS_10G_GPIO_INTR_SGNL_LINE_S_TX_FAULT	MACSEC,1588 SFD lane

Enumerator

VTSS_10G_GPIO_INTR_SGNL_LPCS1G_LATEN↩ CY0_OR_EWIS_BIT0	TX fault
VTSS_10G_GPIO_INTR_SGNL_LPCS1G_LATEN↩ CY1_OR_EWIS_BIT1	LPCS1G latency 0 in case of 1G mode or EWIS BIT 0
VTSS_10G_GPIO_INTR_SGNL_LPCS1G_CHAR↩ _POS0_OR_EWIS_BIT2	LPCS1G latency 0 in case of 1G mode or EWIS BIT 1
VTSS_10G_GPIO_INTR_SGNL_LPCS1G_CHAR↩ _POS1_OR_EWIS_WORD0	LPCS1G Char pos 0 in case of 1G mode or EWIS BIT 2
VTSS_10G_GPIO_INTR_SGNL_LPCS1G_CHAR↩ _POS2_OR_EWIS_WORD1	LPCS1G Char pos 1 in case of 1G mode or EWIS word 0
VTSS_10G_GPIO_INTR_SGNL_LPCS1G_CHAR↩ _POS3_OR_EWIS_WORD2	LPCS1G Char pos 2 in case of 1G mode or EWIS word 1
VTSS_10G_GPIO_INTR_SGNL_MACSEC_IGR_↩ PRED_VAR0	LPCS1G Char pos 3 in case of 1G mode or EWIS word 2
VTSS_10G_GPIO_INTR_SGNL_MACSEC_IGR_↩ PRED_VAR1	Macsec ingress predictor var 0
VTSS_10G_GPIO_INTR_SGNL_KR_ACTV_2GPIO	Macsec ingress predictor var 1
VTSS_10G_GPIO_INTR_SGNL_DFT_TX_2GPIO	KR activity
VTSS_10G_GPIO_INTR_SGNL_RESERVED	DFT transmit
VTSS_10G_GPIO_INTR_SGNL_EXE_LST_2GPI↩ O_0	Reserved for future use
VTSS_10G_GPIO_INTR_SGNL_EXE_LST_2GPI↩ O_1	EXE LST to GPIO 0
VTSS_10G_GPIO_INTR_SGNL_EXE_LST_2GPI↩ O_2	EXE LST to GPIO 1
VTSS_10G_GPIO_INTR_SGNL_EXE_LST_2GPI↩ O_3	EXE LST to GPIO 2
VTSS_10G_GPIO_INTR_SGNL_EXE_LST_2GPI↩ O_4	EXE LST to GPIO 3
VTSS_10G_GPIO_INTR_SGNL_LINK_HCD_2GPI↩ O_0	EXE LST to GPIO 4
VTSS_10G_GPIO_INTR_SGNL_LINK_HCD_2GPI↩ O_1	Link HCD to GPIO 0
VTSS_10G_GPIO_INTR_SGNL_LINK_HCD_2GPI↩ O_2	Link HCD to GPIO 1
VTSS_10G_GPIO_INTR_SGNL_ETH_1G_ENA	Link HCD to GPIO 2
VTSS_10G_GPIO_INTR_SGNL_H_KR_8b10b_2↩ GIPO	Ethernet 1G enable
VTSS_10G_GPIO_INTR_SGNL_H_KR_10Gb_2G↩ PIO	KR 8b10b to GPIO
VTSS_10G_GPIO_INTR_SGNL_H_KR_ACTV_2G↩ PIO	KR10Gb to GPIO
VTSS_10G_GPIO_INTR_SGNL_NONE	KR activity to GPIO

Definition at line 2347 of file vtss_phy_10g_api.h.

4.30.4.37 vtss_gpio_10g_chan_intrpt_t

```
enum vtss_gpio_10g_chan_intrpt_t
```

GPIO Channel level interrupts.

Internal signals supported on Malibu

Enumerator

VTSS_10G_GPIO_INTRPT_WIS1	WIS interrupt 0
VTSS_10G_GPIO_INTRPT_LPCS10G	WIS interrupt 1
VTSS_10G_GPIO_INTRPT_HPCS10G	LPCS 10G interrupt
VTSS_10G_GPIO_INTRPT_LPCS1G	HPCS 10G interrupt
VTSS_10G_GPIO_INTRPT_HPCS1G	LPCS 1G interrupt
VTSS_10G_GPIO_INTRPT_MSEC_EGR	HPCS 1G interrupt
VTSS_10G_GPIO_INTRPT_MSEC_IGR	Macsec Egress interrupt
VTSS_10G_GPIO_INTRPT_LMAC	Macsec Ingress interrupt
VTSS_10G_GPIO_INTRPT_HMAC	Line MAC interrupt
VTSS_10G_GPIO_INTRPT_FCBUF	Host MAC interrupt
VTSS_10G_GPIO_INTRPT_LIGR_FIFO	FC Buffer interrupt
VTSS_10G_GPIO_INTRPT_LEGR_FIFO	Line ingress FIFO interrupt
VTSS_10G_GPIO_INTRPT_HEGR_FIFO	Line egress FIFO interrupt
VTSS_10G_GPIO_INTRPT_LPMA	Host egress FIFO interrupt
VTSS_10G_GPIO_INTRPT_HPMA	Line PMA interrupt

Definition at line 2419 of file vtss_phy_10g_api.h.

4.30.4.38 vtss_gpio_10g_aggr_intrpt_t

```
enum vtss_gpio_10g_aggr_intrpt_t
```

GPIO Channel level interrupts.

Channel Level Interrupts

Enumerator

VTSS_10G_GPIO_AGGR_INTRPT_CH0_INTR1_EN	CH0_INTR0_EN
VTSS_10G_GPIO_AGGR_INTRPT_CH1_INTR0_EN	CH0_INTR1_EN
VTSS_10G_GPIO_AGGR_INTRPT_CH1_INTR1_EN	CH1_INTR0_EN
VTSS_10G_GPIO_AGGR_INTRPT_CH2_INTR0_EN	CH1_INTR1_EN
VTSS_10G_GPIO_AGGR_INTRPT_CH2_INTR1_EN	CH2_INTR0_EN
VTSS_10G_GPIO_AGGR_INTRPT_CH3_INTR0_EN	CH2_INTR1_EN
VTSS_10G_GPIO_AGGR_INTRPT_CH3_INTR1_EN	CH3_INTR0_EN
VTSS_10G_GPIO_AGGR_INTRPT_IP1588_0_INTR0_EN	CH3_INTR1_EN
VTSS_10G_GPIO_AGGR_INTRPT_IP1588_0_INTR1_EN	IP1588_0_INTR0_EN
VTSS_10G_GPIO_AGGR_INTRPT_IP1588_0_INTR2_EN	IP1588_0_INTR1_EN
VTSS_10G_GPIO_AGGR_INTRPT_IP1588_0_INTR3_EN	IP1588_0_INTR2_EN
VTSS_10G_GPIO_AGGR_INTRPT_IP1588_1_INTR0_EN	IP1588_0_INTR3_EN
VTSS_10G_GPIO_AGGR_INTRPT_IP1588_1_INTR1_EN	TS_FIFO empty channel 0
VTSS_10G_GPIO_AGGR_INTRPT_IP1588_1_INTR2_EN	TS_FIFO empty channel 1

Enumerator

VTSS_10G_GPIO_AGGR_INTRPT_IP1588_1_INTR3_EN	TS_FIFO empty channel 2
VTSS_10G_GPIO_AGGR_INTRPT_LCPLL_0_INTR_EN	TS_FIFO empty channel 3
VTSS_10G_GPIO_AGGR_INTRPT_LCPLL_1_INTR_EN	LCPLL_0_INTR_EN
VTSS_10G_GPIO_AGGR_INTRPT_EXP4_INTR_EN	LCPLL_1_INTR_EN
VTSS_10G_GPIO_AGGR_INTRPT_CLK_MUX_INTR_EN	EXP4_INTR_EN
VTSS_10G_GPIO_AGGR_INTRPT_GPIO_INTR_EN	CLK_MUX_INTR_EN

Definition at line 2442 of file vtss_phy_10g_api.h.

4.30.4.39 vtss_gpio_10g_input_t

```
enum vtss_gpio_10g_input_t
```

GPIO Channel level interrupts.

Aggregated Interrupts

Enumerator

VTSS_10G_GPIO_INPUT_LINE_LOPC	Input that doesn't need any extra configuration
VTSS_10G_GPIO_INPUT_HOST_LOPC	LOPC from SFP on LINE

Definition at line 2470 of file vtss_phy_10g_api.h.

4.30.5 Function Documentation

4.30.5.1 vtss_phy_10g_mode_get()

```
vtss_rc vtss_phy_10g_mode_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_mode_t *const mode )
```

Get the Phy operating mode.

Prior using this API, vtss_phy_10g_mode_set() or vtss_phy_10g_init() has to be called atleast once on this port/channel

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
mode	[IN] Mode configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.2 vtss_phy_10g_init()

```
vtss_rc vtss_phy_10g_init (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_init_parm_t *const init_conf )
```

Identify PHY and initialize software accordingly.

This API initializes the mode to 10G LAN. It supports AUTO and MANUAL channel Configuration. For AUTO channel Assignment the API should be called in the channel order (0 -> 3) For MANUAL channel Assignment the API can be called in any order, Application must provide the correct channel number specific to a port while calling Manual channel Assignment.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
init_conf	[IN] Configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.3 vtss_phy_10g_mode_set()

```
vtss_rc vtss_phy_10g_mode_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_mode_t *const mode )
```

Identify, Reset and set the operating mode of the PHY.

This API is supposed be called on base channel/port first and later for alternate channels/ports

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
mode	[IN] Mode configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.4 vtss_phy_10g_ib_conf_set()

```
vtss_rc vtss_phy_10g_ib_conf_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_ib_conf_t *const ib_conf,
            BOOL is_host )
```

Configure Input buffer .

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
ib_conf	[IN] IB configuration.
is_host	[IN] Direction to be configured.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.5 vtss_phy_10g_ib_conf_get()

```
vtss_rc vtss_phy_10g_ib_conf_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const BOOL is_host,
            vtss_phy_10g_ib_conf_t *const ib_conf )
```

Get configuration of Input buffer .

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
is_host	[IN] Direction to be configured.
ib_conf	[OUT] IB configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.6 vtss_phy_10g_ib_status_get()

```
vtss_rc vtss_phy_10g_ib_status_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_ib_status_t *const ib_status )
```

Get status of Input buffer .

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
ib_status	[OUT] IB status.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.7 vtss_phy_10g_apc_conf_set()

```
vtss_rc vtss_phy_10g_apc_conf_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_apc_conf_t *const apc_conf,
            const BOOL is_host )
```

Configure APC .

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
apc_conf	[IN] APC configuration.
is_host	[IN] Configuration side.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.8 vtss_phy_10g_apc_conf_get()

```
vtss_rc vtss_phy_10g_apc_conf_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const BOOL is_host,
            vtss_phy_10g_apc_conf_t *const apc_conf )
```

Get configuration of APC .

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
is_host	[IN] Configuration side.
apc_conf	[OUT] APC configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.9 vtss_phy_10g_apc_status_get()

```
vtss_rc vtss_phy_10g_apc_status_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const BOOL is_host,
            vtss_phy_10g_apc_status_t *const apc_status )
```

Get status of APC.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
is_host	[IN] Configuration side.
apc_status	[OUT] APC status.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.10 vtss_phy_10g_apc_restart()

```
vtss_rc vtss_phy_10g_apc_restart (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const BOOL is_host )
```

Restart of APC - Debug function only.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
is_host	[IN] Configuration side.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.11 vtss_phy_10g_jitter_conf_set()

```
vtss_rc vtss_phy_10g_jitter_conf_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_jitter_conf_t *const jitter_conf,
            BOOL is_host )
```

Configure optimised jitter.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
jitter_conf	[IN] Jitter configuration.
is_host	[IN] Direction to be configured.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.12 vtss_phy_10g_jitter_conf_get()

```
vtss_rc vtss_phy_10g_jitter_conf_get (
            const vtss_inst_t inst,
```

```
            const vtss_port_no_t port_no,
            vtss_phy_10g_jitter_conf_t * jitter_conf,
            BOOL is_host )
```

Gets current Jitter configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
jitter_conf	[IN] Jitter configuration.
is_host	[IN] Direction to be configured.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.13 vtss_phy_10g_jitter_status_get()

```
vtss_rc vtss_phy_10g_jitter_status_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_jitter_conf_t *const jitter_conf,
            BOOL is_host )
```

Jitter status.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
jitter_conf	[IN] Jitter configuration status.
is_host	[IN] Direction.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.14 vtss_phy_10g_synce_clkout_get()

```
vtss_rc vtss_phy_10g_synce_clkout_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            BOOL *const synce_clkout )
```

Get the status of recovered clock from PHY. (recommended to use vtss_phy_10g_rxckout_get instead)

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
synce_clkout	[IN] Recovered clock configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.15 vtss_phy_10g_synce_clkout_set()

```
vtss_rc vtss_phy_10g_synce_clkout_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const BOOL synce_clkout )
```

Enable or Disable the recovered clock from PHY. (recommended to use vtss_phy_10g_rxckout_set instead)

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
synce_clkout	[IN] Recovered clock to be enabled or disabled.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.16 vtss_phy_10g_xfp_clkout_get()

```
vtss_rc vtss_phy_10g_xfp_clkout_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            BOOL *const xfp_clkout )
```

Get the status of RXCLKOUT/TXCLKOUT from PHY. (recommended to use vtss_phy_10g_txckout_get instead)

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
xfp_clkout	[IN] XFP clock configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.17 vtss_phy_10g_xfp_clkout_set()

```
vtss_rc vtss_phy_10g_xfp_clkout_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const BOOL xfp_clkout )
```

Enable or Disable the RXCLKOUT/TXCLKOUT from PHY. (recommended to use vtss_phy_10g_txckout_set instead)

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
xfp_clkout	[IN] XFP clock to be enabled or disabled.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.18 vtss_phy_10g_rxckout_get()

```
vtss_rc vtss_phy_10g_rxckout_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_rxckout_conf_t *const rxckout )
```

Get the rx recovered clock output configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
rxckout	[OUT] RXCKOUT clock configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.19 vtss_phy_10g_rxckout_set()

```
vtss_rc vtss_phy_10g_rxckout_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_rxckout_conf_t *const rxckout )
```

Set the rx recovered clock output configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
rxckout	[IN] RXCKOUT clock configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.20 vtss_phy_10g_txckout_get()

```
vtss_rc vtss_phy_10g_txckout_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_txckout_conf_t *const txckout )
```

Get the status of tx recovered clock output configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
txckout	[OUT] TXCKOUT clock configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.21 vtss_phy_10g_txckout_set()

```
vtss_rc vtss_phy_10g_txckout_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_txckout_conf_t *const txckout )
```

Set the tx recovered clock output configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
txckout	[IN] TXCKOUT clock configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.22 vtss_phy_10g_srefclk_conf_get()

```
vtss_rc vtss_phy_10g_srefclk_conf_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_srefclk_mode_t *const srefclk )
```

Get the configuration of srefclk setting
Avaliable for PHY family VENICE
This function should not be used any more, instead use the API function vtss_phy_10g_mode_get, see the parameter documentation for that function.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
srefclk	[OUT] srefclk configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.23 vtss_phy_10g_srefclk_conf_set()

```
vtss_rc vtss_phy_10g_srefclk_conf_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_srefclk_mode_t *const srefclk )
```

Set the configuration of srefclk setting. Avaliable for PHY family VENICE
This function should not be used any more, instead use the API function vtss_phy_10g_mode_set, see the parameter documentation for that function.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
srefclk	[IN] srefclk configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.24 vtss_phy_10g_sckout_conf_set()

```
vtss_rc vtss_phy_10g_sckout_conf_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_sckout_conf_t *const sckout )
```

Set the configuration of sckout setting. Avaliable for PHY family MALIBU
.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
sckout	[IN] sckout configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.25 vtss_phy_10g_ckout_conf_set()

```
vtss_rc vtss_phy_10g_ckout_conf_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_ckout_conf_t *const ckout )
```

Set the configuration of ckout setting. Avaliable for PHY family MALIBU
.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
ckout	[IN] ckout configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.26 vtss_phy_10g_line_clk_conf_set()

```
vtss_rc vtss_phy_10g_line_clk_conf_set (
             const vtss_inst_t inst,
             const vtss_port_no_t port_no,
             const vtss_phy_10g_line_clk_conf_t *const line_clk )
```

Set the configuration of sckout setting. Avaliable for PHY family MALIBU
.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
line_clk	[IN] line_clk configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.27 vtss_phy_10g_host_clk_conf_set()

```
vtss_rc vtss_phy_10g_host_clk_conf_set (
             const vtss_inst_t inst,
             const vtss_port_no_t port_no,
             const vtss_phy_10g_host_clk_conf_t *const host_clk )
```

Set the configuration of sckout setting. Avaliable for PHY family MALIBU
.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
host_clk	[IN] host_clk configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.28 vtss_phy_10g_line_recvrd_clk_conf_set()

```
vtss_rc vtss_phy_10g_line_recvrd_clk_conf_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_line_clk_conf_t *const line_clk )
```

Set the configuration of line clk recovered setting. Avaliable for PHY family MALIBU
.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
line_clk	[IN] line_recvrd_clk configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.29 vtss_phy_10g_host_recvrd_clk_conf_set()

```
vtss_rc vtss_phy_10g_host_recvrd_clk_conf_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_host_clk_conf_t *const host_clk )
```

Set the configuration of host clk recovered setting. Avaliable for PHY family MALIBU
.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
host_clk	[IN] host_clk configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.30 vtss_phy_10g_lane_sync_set()

```
vtss_rc vtss_phy_10g_lane_sync_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_lane_sync_conf_t *const lane_sync )
```

Set the configuration of lane sync setting. Avaliable for PHY family MALIBU
.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
lane_sync	[IN] ckout configuration.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.31 vtss_phy_10g_debug_register_dump()

```
vtss_rc vtss_phy_10g_debug_register_dump (
            const vtss_inst_t inst,
            const vtss_debug_printf_t pr,
            BOOL clear,
            const vtss_port_no_t port_no )
```

Set the configuration of 10G PHY Dump setting. Avaliable for PHY family Venice & Malibu
.

Parameters

inst	[IN] Target instance reference.
pr	[IN] Print function.
clear	[IN] set for clearing the counters
port_no	[IN] Port number.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.32 vtss_phy_10g_status_get()

```
vtss_rc vtss_phy_10g_status_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_status_t *const status )
```

Get the link and fault status of the PHY sublayers.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
status	[IN] Status of all sublayers

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.33 vtss_phy_10g_serdes_status_get()

```
vtss_rc vtss_phy_10g_serdes_status_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_serdes_status_t *const status )
```

Get the status of PHY including sub layers.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
status	[IN] Status of PLL,SUB layers

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.34 vtss_phy_10g_reset()

```
vtss_rc vtss_phy_10g_reset (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no )
```

Reset the phy. Phy is reset to default values.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.35 vtss_phy_10g_clause_37_status_get()

```
vtss_rc vtss_phy_10g_clause_37_status_get (
            const vtss_inst_t inst,
            vtss_port_no_t port_no,
            vtss_phy_10g_clause_37_cmn_status_t *const status )
```

Get clause 37 status.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
status	[OUT] Clause 37 status of the line and host link.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.36 vtss_phy_10g_clause_37_control_get()

```
vtss_rc vtss_phy_10g_clause_37_control_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_clause_37_control_t *const control )
```

Get clause 37 control configuration from software.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
control	[OUT] Clause 37 configuration,control.line,control.host are 'in' parameters.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.37 vtss_phy_10g_clause_37_control_set()

```
vtss_rc vtss_phy_10g_clause_37_control_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_clause_37_control_t *const control )
```

Set clause 37 control configuration.

Clause 37 can be configured independently on HOST,LINE 1G PCSs 1G speed is only supported in 1000-X aneg

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
control	[OUT] Clause 37 configuration. Same configuration is applied to Host and Line interface.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.38 vtss_phy_10g_loopback_set()

```
vtss_rc vtss_phy_10g_loopback_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_loopback_t *const loopback )
```

Enable/Disable a phy network or system loopback.
Only one loopback mode can be active at the same time.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
loopback	[IN] Loopback settings. When disabling a loopback, the lb_type is ignored, i.e. the active loopback is disabled.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.
Error conditions: Loopback not supported for the PHY Attempt to enable loopback while loopback is already active Attempt to disable loopback while no loopback is active

4.30.5.39 vtss_phy_10g_loopback_get()

```
vtss_rc vtss_phy_10g_loopback_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_loopback_t *const loopback )
```

Get loopback settings.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
loopback	[OUT] Current loopback settings.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.40 vtss_phy_10g_cnt_get()

```
vtss_rc vtss_phy_10g_cnt_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_cnt_t *const cnt )
```

Get counters.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
cnt	[OUT] Phy counters

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.41 vtss_phy_10g_power_get()

```
vtss_rc vtss_phy_10g_power_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_power_t *const power )
```

Get the power settings.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
power	[OUT] power settings

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.42 vtss_phy_10g_power_set()

```
vtss_rc vtss_phy_10g_power_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_power_t *const power )
```

Set the power settings.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
power	[IN] power settings

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.43 vtss_phy_10G_is_valid()

```
BOOL vtss_phy_10G_is_valid (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no )
```

Gives a True/False value if the Phy is supported by the API
Only Vitesse phys are supported. vtss_phy_10g_mode_set() must be applied.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.

Returns

TRUE : Phy is supported.
FALSE : Phy is not supported.

4.30.5.44 vtss_phy_10g_failover_set()

```
vtss_rc vtss_phy_10g_failover_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_failover_mode_t *const mode )
```

Set the failover mode.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number. (Use any port within the phy).
mode	[IN] Failover mode

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.45 vtss_phy_10g_failover_get()

```
vtss_rc vtss_phy_10g_failover_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_failover_mode_t *const mode )
```

Get the failover mode.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
mode	[OUT] failover mode

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.46 vtss_phy_10g_auto_failover_set()

```
vtss_rc vtss_phy_10g_auto_failover_set (
            const vtss_inst_t inst,
            vtss_phy_10g_auto_failover_conf_t *const mode )
```

Set the automatic failover mode.

Parameters

inst	[IN] Target instance reference.
mode	[IN] Automatic Failover mode

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.47 vtss_phy_10g_auto_failover_get()

```
vtss_rc vtss_phy_10g_auto_failover_get (
            const vtss_inst_t inst,
            vtss_phy_10g_auto_failover_conf_t *const mode )
```

Get the Automatic failover mode Configuration.

Parameters

inst	[IN] Target instance reference.
mode	[OUT] failover mode

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.48 vtss_phy_10g_vscope_conf_set()

```
vtss_rc vtss_phy_10g_vscope_conf_set (
            const vtss_inst_t inst,
```

```
            const vtss_port_no_t port_no,
            const vtss_phy_10g_vscope_conf_t *const conf )
```

set VSCOPE fast scan configuration

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
conf	[IN] VSCOPE fast scan configuration

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.49 vtss_phy_10g_vscope_conf_get()

```
vtss_rc vtss_phy_10g_vscope_conf_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_vscope_conf_t *const conf )
```

get VSCOPE fast scan configuration

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
conf	[OUT] VSCOPE fast scan configuration

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.50 vtss_phy_10g_vscope_scan_status_get()

```
vtss_rc vtss_phy_10g_vscope_scan_status_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_vscope_scan_status_t *const conf )
```

set VSCOPE fast scan configuration

\ brief VSCOPE fast scan status

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
conf	[IN] VSCOPE fast scan configuration

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.51 vtss_phy_10g_pcs_prbs_gen_conf_set()

```
vtss_rc vtss_phy_10g_pcs_prbs_gen_conf_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_pcs_prbs_gen_conf_t *const conf,
            const BOOL line )
```

Set PCS-prbs generator Configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
conf	[IN] Pcs-Prbs generator configuration
line	[IN] Direction

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.52 vtss_phy_10g_pcs_prbs_gen_conf_get()

```
vtss_rc vtss_phy_10g_pcs_prbs_gen_conf_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_pcs_prbs_gen_conf_t *const conf,
            const BOOL line )
```

Get PCS-prbs generator Configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
conf	[OUT] Pcs-Prbs generator configuration
line	[IN] Direction

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.53 vtss_phy_10g_pcs_prbs_mon_conf_set()

```
vtss_rc vtss_phy_10g_pcs_prbs_mon_conf_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_pcs_prbs_mon_conf_t *const conf,
            const BOOL line )
```

Set PCS-prbs monitor Configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
conf	[IN] Pcs-Prbs monitor configuration
line	[IN] Direction

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.54 vtss_phy_10g_pcs_prbs_mon_conf_get()

```
vtss_rc vtss_phy_10g_pcs_prbs_mon_conf_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_pcs_prbs_mon_conf_t *const conf,
            const BOOL line )
```

Get PCS-prbs monitor Configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
conf	[OUT] Pcs-Prbs monitor configuration
line	[IN] Direction

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.55 vtss_phy_10g_pcs_prbs_mon_status_get()

```
vtss_rc vtss_phy_10g_pcs_prbs_mon_status_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_pcs_prbs_mon_conf_t *const conf,
            const BOOL line )
```

Get PCS-prbs monitor status.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
conf	[OUT] Pcs-Prbs monitor status
line	[IN] Direction

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.56 vtss_phy_10g_prbs_gen_conf()

```
vtss_rc vtss_phy_10g_prbs_gen_conf (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_prbs_gen_conf_t *const conf )
```

set prbs generator Configuration

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
conf	[IN] Prbs configuration

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.57 vtss_phy_10g_prbs_gen_conf_get()

```
vtss_rc vtss_phy_10g_prbs_gen_conf_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_prbs_gen_conf_t *const conf,
            BOOL line )
```

get prbs generator Configuration

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
conf	[OUT] Prbs configuration
line	[IN] Direction in which Prbs generator configuration is requested

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.58 vtss_phy_10g_prbs_mon_conf()

```
vtss_rc vtss_phy_10g_prbs_mon_conf (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_prbs_mon_conf_t *const conf )
```

set prbs generator Configuration

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
conf	[IN] Prbs Monitor configuration

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.59 vtss_phy_10g_prbs_mon_conf_get()

```
vtss_rc vtss_phy_10g_prbs_mon_conf_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_prbs_mon_conf_t *const conf,
            BOOL line )
```

prbs generator Configuration get

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
conf	[IN] Prbs Monitor configuration
line	[IN] Direction in which Prbs Monitor configuration is requested

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.60 vtss_phy_10g_prbs_mon_status_get()

```
vtss_rc vtss_phy_10g_prbs_mon_status_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_prbs_mon_conf_t *const mon_status,
            BOOL line,
            BOOL reset )
```

prbs Checker Status get

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
mon_status	[OUT] Prbs Monitor status
line	[IN] Direction in which Prbs Monitor status is requested
reset	[IN] Resets prbs counters before retreiving the status

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.61 vtss_phy_10g_pkt_gen_conf()

```
vtss_rc vtss_phy_10g_pkt_gen_conf (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_pkt_gen_conf_t *const conf )
```

Set Packet generation Configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
conf	[IN] Packet generator configuration

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.62 vtss_phy_10g_pkt_mon_conf()

```
vtss_rc vtss_phy_10g_pkt_mon_conf (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            BOOL ts_rd,
            vtss_phy_10g_pkt_mon_conf_t *const conf,
            vtss_phy_10g_timestamp_val_t *const conf_ts )
```

Set Packet Monitor Configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
ts_rd	[IN] Flag to indicate that timestamp fifo is also to be read.
conf	Packet monitor configuration
conf_ts	[OUT] Timestamp value array.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.63 vtss_phy_10g_pkt_mon_counters_get()

```
vtss_rc vtss_phy_10g_pkt_mon_counters_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_pkt_mon_conf_t *const conf )
```

Set/Get Packet mon Counters.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
conf	Packet monitor configuration

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.64 vtss_phy_10g_id_get()

```
vtss_rc vtss_phy_10g_id_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_id_t *const phy_id )
```

Read the Phy Id.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
phy_id	[OUT] The part number and revision.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.65 vtss_phy_10g_gpio_mode_set()

```
vtss_rc vtss_phy_10g_gpio_mode_set (
             const vtss_inst_t inst,
             const vtss_port_no_t port_no,
             const vtss_gpio_10g_no_t gpio_no,
             const vtss_gpio_10g_gpio_mode_t *const mode )
```

Set GPIO mode. There is only one set og GPIO per PHY chip - not per port.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number that identify the PHY chip.
gpio_no	[IN] GPIO pin number < VTSS_10G_PHY_GPIO_MAX.
mode	[IN] GPIO mode.

Returns

Return code.

4.30.5.66 vtss_phy_10g_gpio_mode_get()

```
vtss_rc vtss_phy_10g_gpio_mode_get (
             const vtss_inst_t inst,
             const vtss_port_no_t port_no,
             const vtss_gpio_10g_no_t gpio_no,
             vtss_gpio_10g_gpio_mode_t *const mode )
```

Get GPIO mode.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number that identify the PHY chip.
gpio_no	[IN] GPIO pin number.
mode	[OUT] GPIO mode.

Returns

Return code.

4.30.5.67 vtss_phy_10g_gpio_read()

```
vtss_rc vtss_phy_10g_gpio_read (
             const vtss_inst_t inst,
```

```
            const vtss_port_no_t port_no,
            const vtss_gpio_10g_no_t gpio_no,
            BOOL *const value )
```

Read from GPIO input pin.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
gpio_no	[IN] GPIO pin number.
value	[OUT] TRUE if pin is high, FALSE if it is low.

Returns

Return code.

4.30.5.68 vtss_phy_10g_gpio_write()

```
vtss_rc vtss_phy_10g_gpio_write (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_gpio_10g_no_t gpio_no,
            const BOOL value )
```

Write to GPIO output pin.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
gpio_no	[IN] GPIO pin number.
value	[IN] TRUE to set pin high, FALSE to set pin low.

Returns

Return code.

4.30.5.69 vtss_phy_10g_event_enable_set()

```
vtss_rc vtss_phy_10g_event_enable_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_phy_10g_event_t ev_mask,
            const BOOL enable )
```

Enabling / Disabling of events.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
ev_mask	[IN] Mask containing events that are enabled/disabled
enable	[IN] Enable/disable of event

Returns

Return code.

4.30.5.70 vtss_phy_10g_event_enable_get()

```
vtss_rc vtss_phy_10g_event_enable_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_event_t *const ev_mask )
```

Get Enabling of events.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
ev_mask	[OUT] Mask containing events that are enabled

Returns

Return code.

4.30.5.71 vtss_phy_10g_extended_event_enable_get()

```
vtss_rc vtss_phy_10g_extended_event_enable_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_extnd_event_t *const ex_ev_mask )
```

Get Enabling of events.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
ex_ev_mask	[OUT] Mask containing extended events that are enabled

Returns

Return code.

4.30.5.72 vtss_phy_10g_event_poll()

```
vtss_rc vtss_phy_10g_event_poll (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_event_t *const ev_mask )
```

Polling for active events.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
ev_mask	[OUT] Mask containing events that are active

Returns

Return code.

4.30.5.73 vtss_phy_10g_pcs_status_get()

```
vtss_rc vtss_phy_10g_pcs_status_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_extnd_event_t *const ex_events )
```

poll and clear PCS STICKY Register

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
ex_events	[OUT] Event mask containing events that are active

Returns

Return code.

4.30.5.74 vtss_phy_10g_extended_event_poll()

```
vtss_rc vtss_phy_10g_extended_event_poll (
             const vtss_inst_t inst,
             const vtss_port_no_t port_no,
             vtss_phy_10g_extnd_event_t *const ex_events )
```

Polling for active events.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
ex_events	[OUT] Event mask containing events that are active

Returns

Return code.

4.30.5.75 vtss_phy_10g_extended_event_enable_set()

```
vtss_rc vtss_phy_10g_extended_event_enable_set (
             const vtss_inst_t inst,
             const vtss_port_no_t port_no,
             const vtss_phy_10g_extnd_event_t ex_ev_mask,
             const BOOL extnd_enable )
```

Enabling / Disabling of events.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
ex_ev_mask	[IN] Mask containing exetnded events that are enabled/disabled
extnd_enable	[IN] Enable/disable of event

Returns

Return code.

4.30.5.76 vtss_phy_10g_poll_1sec()

```
vtss_rc vtss_phy_10g_poll_1sec (
             const vtss_inst_t inst )
```

Function is called once a second.

Parameters

inst	[IN] Target instance reference.

Returns

Return code.

4.30.5.77 vtss_phy_10g_edc_fw_status_get()

```
vtss_rc vtss_phy_10g_edc_fw_status_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_phy_10g_fw_status_t *const status )
```

Internal microprocessor status.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
status	[OUT] Status of the EDC FW running on the internal CPU

Returns

Return code.

4.30.5.78 vtss_phy_10g_fc_buffer_reset()

```
vtss_rc vtss_phy_10g_fc_buffer_reset (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no )
```

debug function for PHY 10G FC buffer reset

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Any phy port with the chip

Returns

VTSS_RC_OK - success of fc buffer reset

4.30.5.79 vtss_phy_10g_csr_read()

```
vtss_rc vtss_phy_10g_csr_read (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const u32 dev,
            const u32 addr,
            u32 *const value )
```

CSR register read.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
dev	[IN] Device id (or MMD)
addr	[IN] Addr of the register, 16 or 32 bit
value	[OUT] Return value of the register

Returns

Return code.

4.30.5.80 vtss_phy_10g_csr_write()

```
vtss_rc vtss_phy_10g_csr_write (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const u32 dev,
            const u32 addr,
            const u32 value )
```

CSR register write.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
dev	[IN] Device id (or MMD)
addr	[IN] Addr of the register, 16 or 32 bit
value	[IN] Value to be written

Returns

Return code.

4.30.5.81 vtss_phy_warm_start_10g_failed_get()

```
vtss_rc vtss_phy_warm_start_10g_failed_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no )
```

Function for checking if any issue were seen during warm-start.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] The port in question.

Returns

Return code. VTSS_RC_OK if not errors ware seen during warm-start else VTSS_RC_ERROR.

4.30.5.82 vtss_phy_10g_sgmii_mode_set()

```
vtss_rc vtss_phy_10g_sgmii_mode_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            BOOL enable )
```

Enables Pass through mode in 10G PHY.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
enable	[IN] Enables SGMII mode.

Returns

VTSS_RC_OK on success.
VTSS_RC_ERROR on error.

4.30.5.83 vtss_phy_10g_i2c_read()

```
vtss_rc vtss_phy_10g_i2c_read (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const u16 addr,
            u16 * value )
```

read from i2c device

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
addr	[IN] Addr of the SFP ROM register
value	[OUT] Return Value of the register

Returns

Return code.

4.30.5.84 vtss_phy_10g_i2c_write()

```
vtss_rc vtss_phy_10g_i2c_write (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const u16 addr,
            const u16 * value )
```

Write to i2c device.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
addr	[IN] Addr of the SFP ROM register
value	[IN] value to be writter to register

Returns

Return code.

4.30.5.85 vtss_phy_10g_get_user_data()

```
vtss_rc vtss_phy_10g_get_user_data (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            void ** user_data )
```

Gets generic pointer in vtss_state structure.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
user_data	[OUT] Gets value in generic pointer

Returns

Return code.

4.31 vtss_api/include/vtss_phy_api.h File Reference

PHY API.

```
#include <vtss/api/types.h>
#include <vtss/api/port.h>
#include <vtss/api/phy.h>
#include <vtss_misc_api.h>
```

4.31.1 Detailed Description

PHY API.

This header file describes PHY control functions

4.32 vtss_api/include/vtss_phy_ts_api.h File Reference

PHY TimeStamping API.

```
#include <vtss/api/types.h>
#include <vtss/api/port.h>
#include "vtss_misc_api.h"
```

4.32.1 Detailed Description

PHY TimeStamping API.

This header file describes PTP/OAM TimeStamping API functions

4.33 vtss_api/include/vtss_port_api.h File Reference

Port API.

```
#include <vtss/api/port.h>
```

Enumerations

- enum vtss_miim_controller_t { VTSS_MIIM_CONTROLLERS, VTSS_MIIM_CONTROLLER_NONE = -1 }
  MII management controller.

Functions

- vtss_rc vtss_port_mmd_read (const vtss_inst_t inst, const vtss_port_no_t port_no, const u8 mmd, const u16 addr, u16 *const value)

  Read value from MMD register.
- vtss_rc vtss_port_mmd_read_inc (const vtss_inst_t inst, const vtss_port_no_t port_no, const u8 mmd, const u16 addr, u16 *const buf, u8 count)

  Read values (a number of 16 bit values) from MMD register.
- vtss_rc vtss_port_mmd_write (const vtss_inst_t inst, const vtss_port_no_t port_no, const u8 mmd, const u16 addr, const u16 value)

  Write value to MMD register.
- vtss_rc vtss_port_mmd_masked_write (const vtss_inst_t inst, const vtss_port_no_t port_no, const u8 mmd, const u16 addr, const u16 value, const u16 mask)

  Read, modify and write value to MMD register.
- vtss_rc vtss_mmd_read (const vtss_inst_t inst, const vtss_chip_no_t chip_no, const vtss_miim_controller_t miim_controller, const u8 miim_addr, const u8 mmd, const u16 addr, u16 *const value)

  Direct MMD read (Clause 45, bypassing port map)
- vtss_rc vtss_mmd_write (const vtss_inst_t inst, const vtss_chip_no_t chip_no, const vtss_miim_controller_t miim_controller, const u8 miim_addr, const u8 mmd, const u16 addr, const u16 value)

  Direct MMD write (Clause 45, bypassing port map)

4.33.1 Detailed Description

Port API.

4.33.2 Enumeration Type Documentation

4.33.2.1 vtss_miim_controller_t

```
enum vtss_miim_controller_t
```

MII management controller.

Enumerator

VTSS_MIIM_CONTROLLERS	Number of MIIM controllers
VTSS_MIIM_CONTROLLER_NONE	Unassigned MIIM controller

Definition at line 42 of file vtss_port_api.h.

4.33.3 Function Documentation

4.33.3.1 vtss_port_mmd_read()

```
vtss_rc vtss_port_mmd_read (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const u8 mmd,
            const u16 addr,
            u16 *const value )
```

Read value from MMD register.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number connected to MMD.
mmd	[IN] MMD number.
addr	[IN] PHY register address.
value	[OUT] PHY register value.

Returns

Return code.

4.33.3.2 vtss_port_mmd_read_inc()

```
vtss_rc vtss_port_mmd_read_inc (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const u8 mmd,
            const u16 addr,
            u16 *const buf,
            u8 count )
```

Read values (a number of 16 bit values) from MMD register.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number connected to MMD.
mmd	[IN] MMD number.
addr	[IN] PHY register address.
buf	[OUT] PHY register values.
count	[IN] number of values to read.

Returns

Return code.

4.33.3.3 vtss_port_mmd_write()

```
vtss_rc vtss_port_mmd_write (
             const vtss_inst_t inst,
             const vtss_port_no_t port_no,
             const u8 mmd,
             const u16 addr,
             const u16 value )
```

Write value to MMD register.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number connected to MMD.
mmd	[IN] MMD number.
addr	[IN] PHY register address.
value	[IN] PHY register value.

Returns

Return code.

4.33.3.4 vtss_port_mmd_masked_write()

```
vtss_rc vtss_port_mmd_masked_write (
             const vtss_inst_t inst,
             const vtss_port_no_t port_no,
             const u8 mmd,
             const u16 addr,
             const u16 value,
             const u16 mask )
```

Read, modify and write value to MMD register.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number connected to MMD.
mmd	[IN] MMD number.
addr	[IN] PHY register address.
value	[IN] PHY register value.
mask	[IN] PHY register mask, only enabled bits are changed.

Returns

Return code.

4.33.3.5 vtss_mmd_read()

```
vtss_rc vtss_mmd_read (
            const vtss_inst_t inst,
            const vtss_chip_no_t chip_no,
            const vtss_miim_controller_t miim_controller,
            const u8 miim_addr,
            const u8 mmd,
            const u16 addr,
            u16 *const value )
```

Direct MMD read (Clause 45, bypassing port map)

Parameters

inst	[IN] Target instance reference.
chip_no	[IN] Chip number (if multi-chip instance).
miim_controller	[IN] MIIM Controller Instance
miim_addr	[IN] MIIM Device Address
mmd	[IN] MMD number.
addr	[IN] MIIM Register Address
value	[OUT] Register value read

Returns

Return code.

4.33.3.6 vtss_mmd_write()

```
vtss_rc vtss_mmd_write (
            const vtss_inst_t inst,
            const vtss_chip_no_t chip_no,
            const vtss_miim_controller_t miim_controller,
            const u8 miim_addr,
            const u8 mmd,
            const u16 addr,
            const u16 value )
```

Direct MMD write (Clause 45, bypassing port map)

Parameters

inst	[IN] Target instance reference.
chip_no	[IN] Chip number (if multi-chip instance).
miim_controller	[IN] MIIM Controller Instance
miim_addr	[IN] MIIM Device Address
mmd	[IN] MMD number.
addr	[IN] MIIM Register Address
value	[IN] Register value to write

Returns

Return code.

4.34 vtss_api/include/vtss_qos_api.h File Reference

QoS API.

```
#include <vtss/api/types.h>
```

4.34.1 Detailed Description

QoS API.

This header file describes Quality of Service functions

4.35 vtss_api/include/vtss_rab_api.h File Reference

RAB API.

```
#include <vtss/api/types.h>
```

4.35.1 Detailed Description

RAB API.

4.36 vtss_api/include/vtss_security_api.h File Reference

Security API.

```
#include <vtss/api/types.h>
```

4.36.1 Detailed Description

Security API.

This header file describes security functions

4.37 vtss_api/include/vtss_sfi4_api.h File Reference

SFI4 API.

```
#include <vtss/api/types.h>
```

4.37.1 Detailed Description

SFI4 API.

4.38 vtss_api/include/vtss_sync_api.h File Reference

Synchronization API.

```
#include "vtss/api/types.h"
```

4.38.1 Detailed Description

Synchronization API.

This header file describes synchronization functions

4.39 vtss_api/include/vtss_tfi5_api.h File Reference

TFI5 API.

```
#include <vtss/api/types.h>
```

4.39.1 Detailed Description

TFI5 API.

4.40 vtss_api/include/vtss_ts_api.h File Reference

TimeStamping API.

```
#include <vtss/api/types.h>
```

4.40.1 Detailed Description

TimeStamping API.

This header file describes PTP/OAM TimeStamping API functions and associated types.

4.41 vtss_api/include/vtss_upi_api.h File Reference

Define UPI API interface.

```
#include <vtss/api/types.h>
```

4.41.1 Detailed Description

Define UPI API interface.

4.42 vtss_api/include/vtss_wis_api.h File Reference

eWIS layer API

```
#include <vtss/api/types.h>
```

Data Structures

- struct vtss_ewis_tti_s
  Trail Trace Identifier type.
- struct vtss_ewis_fault_cons_act_s
  eWIS fault mask configuration, i.e set up which defects trigger the Fault condition
- struct vtss_ewis_aisl_cons_act_s
  eWIS AIS-L consequent actions
- struct vtss_ewis_rdil_cons_act_s
  eWIS RDI-L consequent actions
- struct vtss_ewis_cons_act_s
  eWIS consequent actions
- struct vtss_ewis_line_force_mode_s
  eWIS line force mode
- struct vtss_ewis_line_tx_force_mode_s
  eWIS line TX force mode
- struct vtss_ewis_path_force_mode_s
  eWIS path force modes
- struct vtss_ewis_force_mode_s
  eWIS force modes
- struct vtss_ewis_perf_mode_s
  eWIS Mode(Bit/Block) for the Performence Monitoring Counters

- struct vtss_ewis_status_s

  eWIS status
- struct vtss_ewis_defects_s

  eWIS defects
- struct vtss_ewis_perf_s

  eWIS performance primitives. These data are assumed to be read once every sec. The counters holds increments compared to previous read. The namings and definitions are taken from ITU-T rec G.783.
- struct vtss_ewis_counter_s

  eWIS performance counters. These counters are free running counters that wraps to zero.
- struct vtss_ewis_test_conf_s

  eWIS test configuration
- struct vtss_ewis_test_status_s

  eWIS test status
- struct vtss_ewis_tx_oh_s

  WIS transmitted overhead data. only a few oh bytes can be set dynamically. These OH bytes are not configurable from the API: H4 : multiframe indicator M0/M1: STS-1 Line Remote Error Indication (REI) G1: Path status.
- struct vtss_ewis_tx_passthru_s

  eWIS overhead passthru configuration.
- struct vtss_ewis_counter_threshold_s

  eWIS performance counter thresholds.
- struct vtss_ewis_static_conf_s

  eWIS static configuration data,
- struct vtss_ewis_sl_conf_s

  signal label configuration
- struct vtss_ewis_conf_s

  eWIS configuration primitives, used to hold all the configuration parameters in the internal state in the API.

Macros

- #define VTSS_EWIS_SEF_EV 0x00000001

  WIS interrupt events.
- #define VTSS_EWIS_FPLM_EV 0x00000002
- #define VTSS_EWIS_FAIS_EV 0x00000004
- #define VTSS_EWIS_LOF_EV 0x00000008
- #define VTSS_EWIS_LOS_EV 0x00000010
- #define VTSS_EWIS_RDIL_EV 0x00000020
- #define VTSS_EWIS_AISL_EV 0x00000040
- #define VTSS_EWIS_LCDP_EV 0x00000080
- #define VTSS_EWIS_PLMP_EV 0x00000100
- #define VTSS_EWIS_AISP_EV 0x00000200
- #define VTSS_EWIS_LOPP_EV 0x00000400
- #define VTSS_EWIS_MODULE_EV 0x00000800
- #define VTSS_EWIS_TXLOL_EV 0x00001000
- #define VTSS_EWIS_RXLOL_EV 0x00002000
- #define VTSS_EWIS_LOPC_EV 0x00004000
- #define VTSS_EWIS_UNEQP_EV 0x00008000
- #define VTSS_EWIS_FEUNEQP_EV 0x00010000
- #define VTSS_EWIS_FERDIP_EV 0x00020000
- #define VTSS_EWIS_REIL_EV 0x00040000
- #define VTSS_EWIS_REIP_EV 0x00080000
- #define VTSS_EWIS_HIGH_BER_EV 0x00100000
- #define VTSS_EWIS_PCS_RECEIVE_FAULT_PEND 0x00200000

- #define VTSS_EWIS_B1_NZ_EV 0x00400000
- #define VTSS_EWIS_B2_NZ_EV 0x00800000
- #define VTSS_EWIS_B3_NZ_EV 0x01000000
- #define VTSS_EWIS_REIL_NZ_EV 0x02000000
- #define VTSS_EWIS_REIP_NZ_EV 0x04000000
- #define VTSS_EWIS_B1_THRESH_EV 0x08000000
- #define VTSS_EWIS_B2_THRESH_EV 0x10000000
- #define VTSS_EWIS_B3_THRESH_EV 0x20000000
- #define VTSS_EWIS_REIL_THRESH_EV 0x40000000
- #define VTSS_EWIS_REIP_THRESH_EV 0x80000000

Typedefs

- typedef struct vtss_ewis_tti_s vtss_ewis_tti_t

  Trail Trace Identifier type.
- typedef struct vtss_ewis_fault_cons_act_s vtss_ewis_fault_cons_act_t

  eWIS fault mask configuration, i.e set up which defects trigger the Fault condition
- typedef struct vtss_ewis_aisl_cons_act_s vtss_ewis_aisl_cons_act_t

  eWIS AIS-L consequent actions
- typedef struct vtss_ewis_rdil_cons_act_s vtss_ewis_rdil_cons_act_t

  eWIS RDI-L consequent actions
- typedef struct vtss_ewis_cons_act_s vtss_ewis_cons_act_t

  eWIS consequent actions
- typedef struct vtss_ewis_line_force_mode_s vtss_ewis_line_force_mode_t

  eWIS line force mode
- typedef struct vtss_ewis_line_tx_force_mode_s vtss_ewis_line_tx_force_mode_t

  eWIS line TX force mode
- typedef struct vtss_ewis_path_force_mode_s vtss_ewis_path_force_mode_t

  eWIS path force modes
- typedef struct vtss_ewis_force_mode_s vtss_ewis_force_mode_t

  eWIS force modes
- typedef struct vtss_ewis_perf_mode_s vtss_ewis_perf_mode_t

  eWIS Mode(Bit/Block) for the Performence Monitoring Counters
- typedef struct vtss_ewis_status_s vtss_ewis_status_t

  eWIS status
- typedef struct vtss_ewis_defects_s vtss_ewis_defects_t

  eWIS defects
- typedef struct vtss_ewis_perf_s vtss_ewis_perf_t

  eWIS performance primitives. These data are assumed to be read once every sec. The counters holds increments compared to previous read. The namings and definitions are taken from ITU-T rec G.783.
- typedef struct vtss_ewis_counter_s vtss_ewis_counter_t

  eWIS performance counters. These counters are free running counters that wraps to zero.
- typedef enum vtss_ewis_test_pattern_s vtss_ewis_test_pattern_t

  eWIS test pattern mode types.
- typedef struct vtss_ewis_test_conf_s vtss_ewis_test_conf_t

  eWIS test configuration
- typedef struct vtss_ewis_test_status_s vtss_ewis_test_status_t

  eWIS test status
- typedef struct vtss_ewis_tx_oh_s vtss_ewis_tx_oh_t

  WIS transmitted overhead data. only a few oh bytes can be set dynamically. These OH bytes are not configurable from the API: H4 : multiframe indicator M0/M1: STS-1 Line Remote Error Indication (REI) G1: Path status.

- typedef struct vtss_ewis_tx_passthru_s vtss_ewis_tx_oh_passthru_t

  eWIS overhead passthru configuration.
- typedef struct vtss_ewis_counter_threshold_s vtss_ewis_counter_threshold_t

  eWIS performance counter thresholds.
- typedef struct vtss_ewis_static_conf_s vtss_ewis_static_conf_t

  eWIS static configuration data,
- typedef struct vtss_ewis_sl_conf_s vtss_ewis_sl_conf_t

  signal label configuration
- typedef struct vtss_ewis_conf_s vtss_ewis_conf_t

  eWIS configuration primitives, used to hold all the configuration parameters in the internal state in the API.
- typedef u64 vtss_ewis_event_t

Enumerations

- enum vtss_ewis_tti_mode_t { TTI_MODE_1, TTI_MODE_16, TTI_MODE_64, **TTI_MODE_MAX** }

  Trail Trace Identifier mode types.
- enum vtss_ewis_perf_cntr_mode_t { VTSS_EWIS_PERF_MODE_BIT, VTSS_EWIS_PERF_MODE_BLOCK }

  eWIS Mode(Bit/Block) for the Performence Monitoring Counters
- enum vtss_ewis_mode_t { **VTSS_WIS_OPERMODE_DISABLE**, VTSS_WIS_OPERMODE_WIS_MODE, VTSS_WIS_OPERMODE_STS192, VTSS_WIS_OPERMODE_STM64, VTSS_WIS_OPERMODE_MAX }

  eWIS operational mode types
- enum vtss_ewis_test_pattern_s { VTSS_WIS_TEST_MODE_DISABLE, VTSS_WIS_TEST_MODE_SQUARE_WAVE, VTSS_WIS_TEST_MODE_PRBS31, VTSS_WIS_TEST_MODE_MIXED_FREQUENCY, VTSS_WIS_TEST_MODE_MAX }

  eWIS test pattern mode types.
- enum vtss_ewis_prbs31_err_inj_t { EWIS_PRBS31_SINGLE_ERR, EWIS_PRBS31_SAT_ERR, **EWIS_P↩ RBS31_MODE_MAX** }

  test error injection types

Functions

- vtss_rc vtss_ewis_event_enable (const vtss_inst_t inst, const vtss_port_no_t port_no, const BOOL enable, const vtss_ewis_event_t ev_mask)

  Enable event generation for a specific event type or group of events.
- vtss_rc vtss_ewis_event_poll (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_event_t *const status)

  Polling function called at by interrupt or periodically.
- vtss_rc vtss_ewis_event_poll_without_mask (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_event_t *const status)

  Polling function called at by interrupt or periodically.
- vtss_rc vtss_ewis_event_force (const vtss_inst_t inst, const vtss_port_no_t port_no, const BOOL enable, const vtss_ewis_event_t ev_force)

  Forces one or more WIS events to occur (simulated events)
- vtss_rc vtss_ewis_static_conf_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_static_conf_t *const stat_conf)

  Get eWIS static configuration.

- vtss_rc vtss_ewis_force_conf_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_ewis_force_mode_t *const force_conf)

  Set WIS force mode configuration. The WIS can be forced to induce a particular condition, These can be configured in vtss_ewis_force_mode_t. AIS,RDI can be forced in Tx and Rx directions and force UNEQ and RDI in path layer.
- vtss_rc vtss_ewis_force_conf_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_force_mode_t *const force_conf)

  Get WIS force mode configuration.
- vtss_rc vtss_ewis_tx_oh_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_ewis_tx_oh_t *const tx_oh)

  Set WIS transmitted overhead bytes. Supports insertion of various Section, line and path overhead Bytes.
- vtss_rc vtss_ewis_tx_oh_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_tx_oh_t *const tx_oh)

  Get configured WIS transmitted overhead bytes.
- vtss_rc vtss_ewis_tx_oh_passthru_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_ewis_tx_oh_passthru_t *const tx_oh_passthru)

  Set eWIS overhead passthru configuration.
- vtss_rc vtss_ewis_tx_oh_passthru_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_tx_oh_passthru_t *const tx_oh_passthru)

  Get eWIS overhead passthru configuration.
- vtss_rc vtss_ewis_mode_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_ewis_mode_t *const mode)

  Set eWIS mode.
- vtss_rc vtss_ewis_mode_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_mode_t *const mode)

  Get WIS mode.
- vtss_rc vtss_ewis_reset (const vtss_inst_t inst, const vtss_port_no_t port_no)

  Reset WIS block.
- vtss_rc vtss_ewis_cons_act_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_ewis_cons_act_t *const cons_act)

  Set consequent actions, i.e. how to handle AIS-L insertion and RDI_L backreporting.
- vtss_rc vtss_ewis_cons_act_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_cons_act_t *const cons_act)

  Get the configured consequent actions.
- vtss_rc vtss_ewis_section_txti_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_ewis_tti_t *const txti)

  Set section transmitted Trail Trace Identifier. The transmitted trace identifier is aligned according to the specification, i.e 16 byte mode: In the first byte MSB = 1, and all other bytes MSB = 0. 64 byte mode: this is a text string terminated with CR/LF.
- vtss_rc vtss_ewis_section_txti_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_tti_t *const txti)

  Get the configured section transmitted Trail Trace Identifier.
- vtss_rc vtss_ewis_exp_sl_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_ewis_sl_conf_t *const sl)

  Set expected Signal label. The signal label is only configurable in SONET/SDH mode, in this mode the path overhead is not terminated, it is only monitored, i.e. only expected signal label is configurable.
- vtss_rc vtss_ewis_path_txti_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_ewis_tti_t *const txti)

  *Set Path Transmitted Trail Trace Identifier. The transmitted trace identifier is aligned according to the specification, i.e 16 byte mode: In the first byte MSB = 1, and all other bytes MSB = 0. 64 byte mode: this is a text string terminated with CR/LF. *.*
- vtss_rc vtss_ewis_path_txti_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_tti_t *const txti)

  Get the configured Path Transmitted Trail Trace Identifier.

- vtss_rc vtss_ewis_test_mode_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_ewis_test_conf_t *const test_mode)

  Set WIS test mode.
- vtss_rc vtss_ewis_test_mode_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_test_conf_t *const test_mode)

  Get eWIS test mode.
- vtss_rc vtss_ewis_prbs31_err_inj_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_ewis_prbs31_err_inj_t *const inj)

  Inject eWIS PRBS31 errors.
- vtss_rc vtss_ewis_test_counter_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_test_status_t *const test_status)

  Get eWIS test counter.
- vtss_rc vtss_ewis_defects_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_defects_t *const def)

  Get eWIS defects. Reports the currect status of the defects.
- vtss_rc vtss_ewis_status_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_status_t *const status)

  Get eWIS fault and link status.
- vtss_rc vtss_ewis_section_acti_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_tti_t *const acti)

  Get section received (accepted) Trail Trace Identifier. The received trace identifier is aligned according to the specification, i.e 16 byte mode: In the first byte MSB = 1, and all other bytes MSB = 0. (see G.707 section 9.2.2.2) No CRC checksum verification is done in the API. 64 byte mode: this is a text string terminated with CR/LF.
- vtss_rc vtss_ewis_path_acti_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_tti_t *const acti)

  Get path received (accepted) Trail Trace Identifier.
- vtss_rc vtss_ewis_counter_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_counter_t *const counter)

  Get free running eWIS counters.
- vtss_rc vtss_ewis_perf_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_perf_t *const perf)

  Get eWIS counters per second (performance primitives). By default the source of PMTICK event (generation of 1 second) is configured to be internal. The values are accumulate for a period of 1 second and are then updated in the associated registers.
- vtss_rc vtss_ewis_counter_threshold_set (const vtss_inst_t inst, const vtss_port_no_t port_no, const vtss_ewis_counter_threshold_t *const threshold)

  Set eWIS error counter thresholds per second. The PHY generates an interrupt once the error counter exceeds the configured threshold values. The units is frames per second. The threshold value configuration is possible for B1,B2,B3,REIP,REIL errors. Eg: a threshold value of 0 gives one interrupt for every 1 second if the error counter is atleast 1.
- vtss_rc vtss_ewis_counter_threshold_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_counter_threshold_t *const threshold)

  Get the configured eWIS error counter thresholds.
- vtss_rc vtss_ewis_perf_mode_set (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_perf_mode_t const *perf_mode)

  Set the eWIS performance block counter modes.
- vtss_rc vtss_ewis_perf_mode_get (const vtss_inst_t inst, const vtss_port_no_t port_no, vtss_ewis_perf_mode_t *const perf_mode)

  Get the eWIS performance block counter modes.

4.42.1 Detailed Description

eWIS layer API

4.42.2 Macro Definition Documentation

4.42.2.1 VTSS_EWIS_SEF_EV

```
#define VTSS_EWIS_SEF_EV 0x00000001
```

WIS interrupt events.

Note

These interrupts are not used for 8487-15/8488-15. There are seperate type vtss_phy_10g_event_t defined in vtss_phy_10g_api.h for these chips.SEF has changed state

Definition at line 338 of file vtss_wis_api.h.

4.42.2.2 VTSS_EWIS_FPLM_EV

```
#define VTSS_EWIS_FPLM_EV 0x00000002
```

far-end (PLM-P) / (LCDP)

Definition at line 339 of file vtss_wis_api.h.

4.42.2.3 VTSS_EWIS_FAIS_EV

```
#define VTSS_EWIS_FAIS_EV 0x00000004
```

far-end (AIS-P) / (LOP)

Definition at line 340 of file vtss_wis_api.h.

4.42.2.4 VTSS_EWIS_LOF_EV

```
#define VTSS_EWIS_LOF_EV 0x00000008
```

Loss of Frame (LOF)

Definition at line 341 of file vtss_wis_api.h.

4.42.2.5 VTSS_EWIS_LOS_EV

```
#define VTSS_EWIS_LOS_EV 0x00000010
```

Loss of Signal (LOS)

Definition at line 342 of file vtss_wis_api.h.

4.42.2.6 VTSS_EWIS_RDIL_EV

```
#define VTSS_EWIS_RDIL_EV 0x00000020
```

Line Remote Defect Indication (RDI-L)

Definition at line 343 of file vtss_wis_api.h.

4.42.2.7 VTSS_EWIS_AISL_EV

```
#define VTSS_EWIS_AISL_EV 0x00000040
```

Line Alarm Indication Signal (AIS-L)

Definition at line 344 of file vtss_wis_api.h.

4.42.2.8 VTSS_EWIS_LCDP_EV

```
#define VTSS_EWIS_LCDP_EV 0x00000080
```

Loss of Code-group Delineation (LCD-P)

Definition at line 345 of file vtss_wis_api.h.

4.42.2.9 VTSS_EWIS_PLMP_EV

```
#define VTSS_EWIS_PLMP_EV 0x00000100
```

Path Label Mismatch (PLMP)

Definition at line 346 of file vtss_wis_api.h.

4.42.2.10 VTSS_EWIS_AISP_EV

```
#define VTSS_EWIS_AISP_EV 0x00000200
```

Path Alarm Indication Signal (AIS-P)

Definition at line 347 of file vtss_wis_api.h.

4.42.2.11 VTSS_EWIS_LOPP_EV

```
#define VTSS_EWIS_LOPP_EV 0x00000400
```

Path Loss of Pointer (LOP-P)

Definition at line 348 of file vtss_wis_api.h.

4.42.2.12 VTSS_EWIS_MODULE_EV

```
#define VTSS_EWIS_MODULE_EV 0x00000800
```

GPIO pin state being driven by optics module

Definition at line 349 of file vtss_wis_api.h.

4.42.2.13 VTSS_EWIS_TXLOL_EV

```
#define VTSS_EWIS_TXLOL_EV 0x00001000
```

PMA CMU Loss of Lock

Definition at line 350 of file vtss_wis_api.h.

4.42.2.14 VTSS_EWIS_RXLOL_EV

```
#define VTSS_EWIS_RXLOL_EV 0x00002000
```

PMA CRU Loss of Lock

Definition at line 351 of file vtss_wis_api.h.

4.42.2.15 VTSS_EWIS_LOPC_EV

```
#define VTSS_EWIS_LOPC_EV 0x00004000
```

Loss of Optical Carrier (LOPC)

Definition at line 352 of file vtss_wis_api.h.

4.42.2.16 VTSS_EWIS_UNEQP_EV

```
#define VTSS_EWIS_UNEQP_EV 0x00008000
```

Unequiped Path (UNEQ-P)

Definition at line 353 of file vtss_wis_api.h.

4.42.2.17 VTSS_EWIS_FEUNEQP_EV

```
#define VTSS_EWIS_FEUNEQP_EV 0x00010000
```

Far-end Unequiped Path (UNEQ-P)

Definition at line 354 of file vtss_wis_api.h.

4.42.2.18 VTSS_EWIS_FERDIP_EV

```
#define VTSS_EWIS_FERDIP_EV 0x00020000
```

Far-end Path Remote Defect Identifier (RDI-P)

Definition at line 355 of file vtss_wis_api.h.

4.42.2.19 VTSS_EWIS_REIL_EV

```
#define VTSS_EWIS_REIL_EV 0x00040000
```

Line Remote Error Indication (REI-L)

Definition at line 356 of file vtss_wis_api.h.

4.42.2.20 VTSS_EWIS_REIP_EV

```
#define VTSS_EWIS_REIP_EV 0x00080000
```

Path Remote Error Indication (REI-P)

Definition at line 357 of file vtss_wis_api.h.

4.42.2.21 VTSS_EWIS_HIGH_BER_EV

```
#define VTSS_EWIS_HIGH_BER_EV 0x00100000
```

PCS high bit error rate (BER)

Definition at line 358 of file vtss_wis_api.h.

4.42.2.22 VTSS_EWIS_PCS_RECEIVE_FAULT_PEND

```
#define VTSS_EWIS_PCS_RECEIVE_FAULT_PEND 0x00200000
```

PCS Receive fault

Definition at line 360 of file vtss_wis_api.h.

4.42.2.23 VTSS_EWIS_B1_NZ_EV

```
#define VTSS_EWIS_B1_NZ_EV 0x00400000
```

PMTICK B1 BIP (B1_ERR_CNT) not zero

Definition at line 362 of file vtss_wis_api.h.

4.42.2.24 VTSS_EWIS_B2_NZ_EV

```
#define VTSS_EWIS_B2_NZ_EV 0x00800000
```

PMTICK B2 BIP (B1_ERR_CNT) not zero

Definition at line 363 of file vtss_wis_api.h.

4.42.2.25 VTSS_EWIS_B3_NZ_EV

```
#define VTSS_EWIS_B3_NZ_EV 0x01000000
```

PMTICK B3 BIP (B1_ERR_CNT) not zero

Definition at line 364 of file vtss_wis_api.h.

4.42.2.26 VTSS_EWIS_REIL_NZ_EV

```
#define VTSS_EWIS_REIL_NZ_EV 0x02000000
```

PMTICK REI-L (REIL_ERR_CNT) not zero

Definition at line 365 of file vtss_wis_api.h.

4.42.2.27 VTSS_EWIS_REIP_NZ_EV

```
#define VTSS_EWIS_REIP_NZ_EV 0x04000000
```

PMTICK REI-P (REIP_ERR_CNT) not zero

Definition at line 366 of file vtss_wis_api.h.

4.42.2.28 VTSS_EWIS_B1_THRESH_EV

```
#define VTSS_EWIS_B1_THRESH_EV 0x08000000
```

B1_THRESH_ERR

Definition at line 368 of file vtss_wis_api.h.

4.42.2.29 VTSS_EWIS_B2_THRESH_EV

```
#define VTSS_EWIS_B2_THRESH_EV 0x10000000
```

B2_THRESH_ERR

Definition at line 369 of file vtss_wis_api.h.

4.42.2.30 VTSS_EWIS_B3_THRESH_EV

```
#define VTSS_EWIS_B3_THRESH_EV 0x20000000
```

B3_THRESH_ERR

Definition at line 370 of file vtss_wis_api.h.

4.42.2.31 VTSS_EWIS_REIL_THRESH_EV

```
#define VTSS_EWIS_REIL_THRESH_EV 0x40000000
```

REIL_THRESH_ERR

Definition at line 371 of file vtss_wis_api.h.

4.42.2.32 VTSS_EWIS_REIP_THRESH_EV

```
#define VTSS_EWIS_REIP_THRESH_EV 0x80000000
```

REIp_THRESH_ERR

Definition at line 372 of file vtss_wis_api.h.

4.42.3 Typedef Documentation

4.42.3.1 vtss_ewis_static_conf_t

```
typedef struct vtss_ewis_static_conf_s vtss_ewis_static_conf_t
```

eWIS static configuration data,

Note

This is specific to 8487/8488-15 and should not be used for Daytona.

4.42.3.2 vtss_ewis_event_t

```
typedef u64 vtss_ewis_event_t
```

Int events: Single event or 'OR' multiple events above

Definition at line 396 of file vtss_wis_api.h.

4.42.4 Enumeration Type Documentation

4.42.4.1 vtss_ewis_tti_mode_t

```
enum vtss_ewis_tti_mode_t
```

Trail Trace Identifier mode types.

Enumerator

TTI_MODE_1	one byte trace identifier
TTI_MODE_16	16 bytes trace identifier
TTI_MODE_64	64 bytes trace identifier

Definition at line 47 of file vtss_wis_api.h.

4.42.4.2 vtss_ewis_perf_cntr_mode_t

`enum vtss_ewis_perf_cntr_mode_t`

eWIS Mode(Bit/Block) for the Performence Monitoring Counters

Enumerator

VTSS_EWIS_PERF_MODE_BIT	Bit mode of the perf monitor counter
VTSS_EWIS_PERF_MODE_BLOCK	Block mode of the perf monitor counter

Definition at line 123 of file vtss_wis_api.h.

4.42.4.3 vtss_ewis_mode_t

`enum vtss_ewis_mode_t`

eWIS operational mode types

Enumerator

VTSS_WIS_OPERMODE_WIS_MODE	WIS mode disabled
VTSS_WIS_OPERMODE_STS192	WIS mode enabled
VTSS_WIS_OPERMODE_STM64	WIS mode SONET - STS192
VTSS_WIS_OPERMODE_MAX	WIS mode SDH - STM64 WIS mode Invalid

Definition at line 138 of file vtss_wis_api.h.

4.42.4.4 vtss_ewis_test_pattern_s

`enum vtss_ewis_test_pattern_s`

eWIS test pattern mode types.

Enumerator

VTSS_WIS_TEST_MODE_DISABLE	Disable test
VTSS_WIS_TEST_MODE_SQUARE_WAVE	Enable squarevave generator, Only valid for test generator
VTSS_WIS_TEST_MODE_PRBS31	Enable prbs31 generator / analyzer (not supported in Daytona)
VTSS_WIS_TEST_MODE_MIXED_FREQUENCY	Enable mixed frequency generator / analyzer
VTSS_WIS_TEST_MODE_MAX	Test mode Invalid

Definition at line 197 of file vtss_wis_api.h.

4.42.4.5 vtss_ewis_prbs31_err_inj_t

```
enum vtss_ewis_prbs31_err_inj_t
```

test error injection types

Enumerator

EWIS_PRBS31_SINGLE_ERR	Inject a single bit error (=> error counter incrementing by 3
EWIS_PRBS31_SAT_ERR	Force the PRBS31 pattern error counter to a value of 65528 (close to saturation)

Definition at line 278 of file vtss_wis_api.h.

4.42.5 Function Documentation

4.42.5.1 vtss_ewis_event_enable()

```
vtss_rc vtss_ewis_event_enable (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const BOOL enable,
            const vtss_ewis_event_t ev_mask )
```

Enable event generation for a specific event type or group of events.

Note

Not applicable for 8487/8488-15

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
enable	[IN] Enable or disable events
ev_mask	[IN] Event type(s) to control (mask)

Returns

Return code.

4.42.5.2 vtss_ewis_event_poll()

```
vtss_rc vtss_ewis_event_poll (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_event_t *const status )
```

Polling function called at by interrupt or periodically.

Note

Interrupt status will be cleared on read. Not applicable for 8487/8488-15

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
status	[OUT] Event status, bit set indicates corresponding event/interrupt has detected

Returns

Return code.

4.42.5.3 vtss_ewis_event_poll_without_mask()

```
vtss_rc vtss_ewis_event_poll_without_mask (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_event_t *const status )
```

Polling function called at by interrupt or periodically.

Note

Interrupt status will be cleared on read. Not applicable for 8487/8488-15

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
status	[OUT] Event status, bit set indicates corresponding event/interrupt has detected irrespective of the mask register

Returns

Return code.

4.42.5.4 vtss_ewis_event_force()

```
vtss_rc vtss_ewis_event_force (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const BOOL enable,
            const vtss_ewis_event_t ev_force )
```

Forces one or more WIS events to occur (simulated events)

Note

useful in debugging.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number
enable	[IN] Enable or disable events
ev_force	[IN] Mask defining which events are forces

Returns

Return code.

4.42.5.5 vtss_ewis_static_conf_get()

```
vtss_rc vtss_ewis_static_conf_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_static_conf_t *const stat_conf )
```

Get eWIS static configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
stat_conf	[OUT] Get eWIS Static configuration, i.e configuration that is set up at initialization, and not changed afterwards.

Returns

Return code.

4.42.5.6 vtss_ewis_force_conf_set()

```
vtss_rc vtss_ewis_force_conf_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_ewis_force_mode_t *const force_conf )
```

Set WIS force mode configuration. The WIS can be forced to induce a particular condition, These can be configured in vtss_ewis_force_mode_t. AIS,RDI can be forced in Tx and Rx directions and force UNEQ and RDI in path layer.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
force_conf	[IN] Set force mode.

Returns

Return code.

4.42.5.7 vtss_ewis_force_conf_get()

```
vtss_rc vtss_ewis_force_conf_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_force_mode_t *const force_conf )
```

Get WIS force mode configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
force_conf	[OUT] Get force mode configuration.

Returns

Return code.

4.42.5.8 vtss_ewis_tx_oh_set()

```
vtss_rc vtss_ewis_tx_oh_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_ewis_tx_oh_t *const tx_oh )
```

Set WIS transmitted overhead bytes. Supports insertion of various Section, line and path overhead Bytes.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
tx_oh	[IN] Transmitted overhead byte values

Returns

Return code.

4.42.5.9 vtss_ewis_tx_oh_get()

```
vtss_rc vtss_ewis_tx_oh_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_tx_oh_t *const tx_oh )
```

Get configured WIS transmitted overhead bytes.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
tx_oh	[OUT] Transmitted overhead byte values

Returns

Return code.

4.42.5.10 vtss_ewis_tx_oh_passthru_set()

```
vtss_rc vtss_ewis_tx_oh_passthru_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_ewis_tx_oh_passthru_t *const tx_oh_passthru )
```

Set eWIS overhead passthru configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
tx_oh_passthru	[IN] Transmitted overhead passthrough configuration

Returns

Return code.

4.42.5.11 vtss_ewis_tx_oh_passthru_get()

```
vtss_rc vtss_ewis_tx_oh_passthru_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_tx_oh_passthru_t *const tx_oh_passthru )
```

Get eWIS overhead passthru configuration.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
tx_oh_passthru	[OUT] Transmitted overhead passthrough configuration

Returns

Return code.

4.42.5.12 vtss_ewis_mode_set()

```
vtss_rc vtss_ewis_mode_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_ewis_mode_t *const mode )
```

Set eWIS mode.

Note

Should not used for 8487-15/8488-15. The mode configuration is enabled by calling vtss_phy_10g_mode_set in the case of 8487-15. In Daytona this is useful in setting the WIS block to operate in multiple modes.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
mode	[IN] Set WIS mode (Disable, WIS, STS192, STM64). sts192 (full Sonet/SDH termination is only supported in Daytona)

Returns

Return code.

4.42.5.13 vtss_ewis_mode_get()

```
vtss_rc vtss_ewis_mode_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_mode_t *const mode )
```

Get WIS mode.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
mode	[OUT] Get WIS mode (Disable, WIS, STS192, STM64).

Returns

Return code.

4.42.5.14 vtss_ewis_reset()

```
vtss_rc vtss_ewis_reset (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no )
```

Reset WIS block.

Note

Useful only for 8487-17/8488-15.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.

Returns

Return code.

4.42.5.15 vtss_ewis_cons_act_set()

```
vtss_rc vtss_ewis_cons_act_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_ewis_cons_act_t *const cons_act )
```

Set consequent actions, i.e. how to handle AIS-L insertion and RDI_L backreporting.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
cons_act	[IN] pointer to consequent actions.

Returns

Return code.

4.42.5.16 vtss_ewis_cons_act_get()

```
vtss_rc vtss_ewis_cons_act_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_cons_act_t *const cons_act )
```

Get the configured consequent actions.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
cons_act	[OUT] pointer to consequent actions.

Returns

Return code.

4.42.5.17 vtss_ewis_section_txti_set()

```
vtss_rc vtss_ewis_section_txti_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_ewis_tti_t *const txti )
```

Set section transmitted Trail Trace Identifier. The transmitted trace identifier is aligned according to the specification, i.e 16 byte mode: In the first byte MSB = 1, and all other bytes MSB = 0. 64 byte mode: this is a text string terminated with CR/LF.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
txti	[IN] pointer to transmitted tti.

Returns

Return code.

4.42.5.18 vtss_ewis_section_txti_get()

```
vtss_rc vtss_ewis_section_txti_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_tti_t *const txti )
```

Get the configured section transmitted Trail Trace Identifier.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
txti	[OUT] pointer to transmitted tti.

Returns

Return code.

4.42.5.19 vtss_ewis_exp_sl_set()

```
vtss_rc vtss_ewis_exp_sl_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_ewis_sl_conf_t *const sl )
```

Set expected Signal label. The signal label is only configurable in SONET/SDH mode, in this mode the path overhead is not terminated, it is only monitored, i.e. only expected signal label is configurable.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
sl	[IN] pointer to expected signal label.

Returns

Return code.

4.42.5.20 vtss_ewis_path_txti_set()

```
vtss_rc vtss_ewis_path_txti_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_ewis_tti_t *const txti )
```

Set Path Transmitted Trail Trace Identifier. The transmitted trace identifier is aligned according to the specification, i.e 16 byte mode: In the first byte MSB = 1, and all other bytes MSB = 0. 64 byte mode: this is a text string terminated with CR/LF. ∗.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
txti	[IN] pointer to transmitted tti.

Returns

Return code.

4.42.5.21 vtss_ewis_path_txti_get()

```
vtss_rc vtss_ewis_path_txti_get (
            const vtss_inst_t inst,
```

```
const vtss_port_no_t port_no,
vtss_ewis_tti_t *const txti )
```

Get the configured Path Transmitted Trail Trace Identifier.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
txti	[OUT] pointer to transmitted tti.

Returns

Return code.

4.42.5.22 vtss_ewis_test_mode_set()

```
vtss_rc vtss_ewis_test_mode_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_ewis_test_conf_t *const test_mode )
```

Set WIS test mode.

Note

This is useful for debugging purpose.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
test_mode	[IN] Set WIS test mode (loopback and test patterns).

Returns

Return code.

Test pattern setup is applied to both TX (test generator) and RX (test analyzer)

4.42.5.23 vtss_ewis_test_mode_get()

```
vtss_rc vtss_ewis_test_mode_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_test_conf_t *const test_mode )
```

Get eWIS test mode.

Note

This is useful for debugging purpose.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
test_mode	[OUT] Get eWIS test mode (loopback and test patterns).

Returns

Return code.

4.42.5.24 vtss_ewis_prbs31_err_inj_set()

```
vtss_rc vtss_ewis_prbs31_err_inj_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_ewis_prbs31_err_inj_t *const inj )
```

Inject eWIS PRBS31 errors.

Note

This is useful for debugging purpose.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
inj	[IN] Defines the type of error injected.

Returns

Return code.

Test pattern setup is applied to both TX (test generator) and RX (test analyzer)

4.42.5.25 vtss_ewis_test_counter_get()

```
vtss_rc vtss_ewis_test_counter_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_test_status_t *const test_status )
```

Get eWIS test counter.

Note

This is useful for debugging purpose.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
test_status	[OUT] Get eWIS test status (test pattern error counter, clear on read).

Returns

Return code.

Test pattern error counter is only used in prbs31 mode. In mixed frequency mode, the normal performance counters are maintained.

4.42.5.26 vtss_ewis_defects_get()

```
vtss_rc vtss_ewis_defects_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_defects_t *const def )
```

Get eWIS defects. Reports the currect status of the defects.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
def	[OUT] pointer to defect status structure.

Returns

Return code.

4.42.5.27 vtss_ewis_status_get()

```
vtss_rc vtss_ewis_status_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_status_t *const status )
```

Get eWIS fault and link status.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
status	[OUT] pointer to status structure.

Returns

Return code.

4.42.5.28 vtss_ewis_section_acti_get()

```
vtss_rc vtss_ewis_section_acti_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_tti_t *const acti )
```

Get section received (accepted) Trail Trace Identifier. The received trace identifier is aligned according to the specification, i.e 16 byte mode: In the first byte MSB = 1, and all other bytes MSB = 0. (see G.707 section 9.2.2.2) No CRC checksum verification is done in the API. 64 byte mode: this is a text string terminated with CR/LF.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
acti	[OUT] pointer to accepted tti.

Returns

Return code.

4.42.5.29 vtss_ewis_path_acti_get()

```
vtss_rc vtss_ewis_path_acti_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_tti_t *const acti )
```

Get path received (accepted) Trail Trace Identifier.

The received trace identifier is aligned according to the specification, i.e 16 byte mode: In the first byte MSB = 1, and all other bytes MSB = 0. (see G.707 section 9.2.2.2) No CRC checksum verification is done in the API. 64 byte mode: this is a text string terminated with CR/LF.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
acti	[OUT] pointer to accepted TTI.

Returns

Return code.

4.42.5.30 vtss_ewis_counter_get()

```
vtss_rc vtss_ewis_counter_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_counter_t *const counter )
```

Get free running eWIS counters.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
counter	[OUT] pointer to counter structure.

Returns

Return code.

4.42.5.31 vtss_ewis_perf_get()

```
vtss_rc vtss_ewis_perf_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_perf_t *const perf )
```

Get eWIS counters per second (performance primitives). By default the source of PMTICK event (generation of 1 second) is configured to be internal. The values are accumulate for a period of 1 second and are then updated in the associated registers.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
perf	[OUT] pointer to performance primitive structure.

Returns

Return code.

4.42.5.32 vtss_ewis_counter_threshold_set()

```
vtss_rc vtss_ewis_counter_threshold_set (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            const vtss_ewis_counter_threshold_t *const threshold )
```

Set eWIS error counter thresholds per second. The PHY generates an interrupt once the error counter exceeds the configured threshold values. The units is frames per second. The threshold value configuration is possible for B1,B2,B3,REIP,REIL errors. Eg: a threshold value of 0 gives one interrupt for every 1 second if the error counter is atleast 1.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
threshold	[IN] pointer to counter threshold structure.

Returns

Return code.

4.42.5.33 vtss_ewis_counter_threshold_get()

```
vtss_rc vtss_ewis_counter_threshold_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_counter_threshold_t *const threshold )
```

Get the configured eWIS error counter thresholds.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
threshold	[OUT] pointer to eWIS error counters threshold structure.

Returns

Return code.

4.42.5.34 vtss_ewis_perf_mode_set()

```
vtss_rc vtss_ewis_perf_mode_set (
            const vtss_inst_t inst,
```

```
            const vtss_port_no_t port_no,
            vtss_ewis_perf_mode_t const * perf_mode )
```

Set the eWIS performance block counter modes.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
perf_mode	[IN] Pointer to the modes of the all performance counters.

Returns

Return code.

4.42.5.35 vtss_ewis_perf_mode_get()

```
vtss_rc vtss_ewis_perf_mode_get (
            const vtss_inst_t inst,
            const vtss_port_no_t port_no,
            vtss_ewis_perf_mode_t *const perf_mode )
```

Get the eWIS performance block counter modes.

Parameters

inst	[IN] Target instance reference.
port_no	[IN] Port number.
perf_mode	[OUT] Pointer to the modes of the all performance counters.

Returns

Return code.

4.43 vtss_api/include/vtss_xaui_api.h File Reference

XAUI API.

```
#include <vtss/api/types.h>
```

4.43.1 Detailed Description

XAUI API.

4.44 vtss_api/include/vtss_xfi_api.h File Reference

XFI API.

```
#include <vtss/api/types.h>
```

4.44.1 Detailed Description

XFI API.

Index