
1G PHY Sample Application Guide

<i>Author: John Haechten Microchip Technology Inc.</i>
--

1.0 INTRODUCTION

Most VSC PHYs (100M, 1G, and 10G) require some type of software to control the VSC PHY. This software consists of a known sequence of register accesses to provide a PHY feature or functionality. These features have been tested and verified to provide desired results. Support for a variety of VSC PHYs currently exists in the following areas:

- U-Boot Driver
- Linux[®] Kernel Driver (Open Source)
- User Space API (VSC6802: PHY API and VSC6803: MESA API)

User Space API provides the most complete, feature-rich coverage and control for VSC PHYs. The User Space API expects to have complete control of the PHY, and therefore, if a Linux Driver exists, the Linux Driver should be disabled to prevent conflicts in the control path.

U-Boot Driver is often the desired way to control the VSC PHY during initial boot sequences. In this phase of boot cycle, Diagnostic Power-On Self Test (POST) may be run and the PHY may be configured and receive the necessary system initialization code to continue the boot process.

Linux Kernel Driver is often used to control the PHY. This method can only be used with the Linux operating system (OS). If the User Space API exists in this system, then the Linux Kernel Driver is normally disabled to prevent conflicts.

The API Sample Application was created to provide sample application code, which uses the User Space API (hereafter referred to as the MESA API) to set up and configure the PHY to test the features of each PHY. The options and sequences used within the Sample Application code are provided for reference, and the user may port these sequences into their application. **The Sample Application provides users a way to easily evaluate and test the PHY and MESA API when connecting to a PHY evaluation board.**

From a software application perspective, there are two approaches/options currently available to control and configure the 1G VSC PHYs, and these are:

- MESA API
- Linux Driver

For the 10G VSC PHYs, the only option currently available is MESA API.

This document describes the architecture and usage of the MESA API Sample Application code for 1G PHY. The Sample Application code for 10G PHY also exists for the MESA API but is not covered in this document.

The Sample Application code can be downloaded from GitHub, compiled in a Linux environment as a stand-alone application and executed while connected to a PHY evaluation board. The Sample Application is compiled and linked with a version of the MESA API code that is used to configure the PHY hardware on the evaluation board to test PHY features and functionality. The Sample Application code covered in this document supports many of the VSC 1G PHYs, and this document describes the requirements needed for each evaluation board setup.

The configurations described within this document cover the following VSC 1G PHY Evaluation Board configurations:

- VSC8584 - Viper Family (VSC8582, VSC8584, VSC8575, VSC8562, and VSC8564)
- VSC8574 - Telsa Family (VSC8572, VSC8574, VSC8504, and VSC8552)
- VSC8514 - Elise Family (VSC8514)
- VSC8502 - Nano Family (VSC8501 and VSC8502)

1.1 Sections

This document includes the following topics:

- [Section 2.0, "General Description"](#)
- [Section 3.0, "Sample Application Overview"](#)
- [Section 4.0, "Configuring the PHY Evaluation Board MAC Interface for Testing"](#)
- [Section 5.0, "Configuring the Traffic Generator"](#)
- [Section 6.0, "Configuring and Using the Sample Application"](#)

1.2 References

The following document provides the software options available for each of the mentioned PHYs below and the location where the software can be downloaded:

- *AN3576 - Software for VSC PHYs* (www.microchip.com/DS00003576)

The following reference documents provide additional information about each of the PHYs and the features that may be exposed and exercised using an evaluation board:

- Tesla PHY Family
 - *VSC8504 Data Sheet* (www.microchip.com/VSC8504)
 - *VSC8552 Data Sheet* (www.microchip.com/VSC8552)
 - *VSC8572 Data Sheet* (www.microchip.com/VSC8572)
 - *VSC8574 Data Sheet* (www.microchip.com/VSC8574)
- Viper PHY Family
 - *VSC8562 Data Sheet* (www.microchip.com/VSC8562)
 - *VSC8564 Data Sheet* (www.microchip.com/VSC8564)
 - *VSC8582 Data Sheet* (www.microchip.com/VSC8582)
 - *VSC8575 Data Sheet* (www.microchip.com/VSC8575)
 - *VSC8584 Data Sheet* (www.microchip.com/VSC8584)
- Elise PHY Family
 - *VSC8514 Data Sheet* (www.microchip.com/VSC8514)
- Nano PHY Family
 - *VSC8501 Data Sheet* (www.microchip.com/VSC8501)
 - *VSC8502 Data Sheet* (www.microchip.com/VSC8502)

The following reference documents provide setup and configuration information about each of the above-mentioned PHY family's evaluation boards, which can be used to test the features of the PHY. Normally, the most capable and feature-rich PHY of each family is used to test as many features as possible. These documents can be found in the same location, under the **Documents** tab for the part as the data sheets:

- *VSC8574 Evaluation Board User Guide*
- *VSC8584 Evaluation Board User Guide*
- *VSC8514 Evaluation Board User Guide*
- *VSC8502 Evaluation Board User Guide*

2.0 GENERAL DESCRIPTION

The Sample Application code can be downloaded, compiled, and executed on a laptop PC that is running a Linux type environment. The most common environments are:

- Linux Environment
- VM Linux Environment running on a Windows® PC
- Cygwin Environment

The Sample Application is compiled and linked with a particular version of the MESA API. The examples provided in this application note use MESA v2021.03 for demonstration and reference. There are two repositories that support the VSC PHYs:

- MESA with the latest release: v2021.03 (Growth path)
- PHY API 4.67.05 (Maintenance mode)

This document focuses on the MESA Stand-Alone Sample Application code for the VSC 1G PHY, which was initially released in the MESA v2020.09 release (VSC6803). Directions to obtain VSC6803, the MESA code, can be found in *AN3576*.

This document describes the configuration, compilation and building of an executable image, and the execution of the image and the required hardware.

3.0 SAMPLE APPLICATION OVERVIEW

The following sections describe how to download, configure, compile, and execute the MESA Sample Application code.

3.1 Downloading the Code

The VSC6803 code repository is located on GitHub. Directions to obtain VSC6803, the MESA code, can be found in AN3576.

- Note 1:** Ensure you have the latest software update. Check with the MCHP Application Support team for any updates or patches to the Sample Application (refer to *KB VSC 1G PHY Sample Application Patch Update Description*).
- 2:** When preparing to download the code, please select *only* Released code, and select the latest Released code that is available (recommended). Selecting Released code ensures that the code you are downloading has been tested. Not selecting a Release may default to selecting the TIP of the code based on the time of download, which may not necessarily have been fully tested.

3.2 Configuring the Code for Compilation

Once the code is downloaded, the code must be configured using compile time flags. These flags enable the support within the software for features and functionality that are included in the hardware. The MESA API software, which is being linked with the MESA Sample Application, has been written to control a variety of PHY products with various feature sets. The compilation flags must be matched to the functionality and features of the PHY hardware.

There are two sets of compile time flags: those that configure the MESA Sample Application code and those that configure the MESA API.

The compile time flags listed in [Table 1](#) are used to configure the Sample Application code. The Sample Application code *does not* perform runtime checking. The Sample Application code must be configured correctly, so that the proper Board Support Package code can be included, which was created for a specific PHY evaluation board.

TABLE 1: MESA SAMPLE APPLICATION COMPILATION FLAG

Compilation Flag	Description
ELISE_EVAL_BOARD	Configures Sample Application for VSC8514EV Includes BSP file: vtss_appl_board_elise_eval.c
NANO_EVAL_BOARD	Configures Sample Application for VSC8502EV Includes BSP file: vtss_appl_board_nano_eval.c
TESLA_EVAL_BOARD	Configures Sample Application for VSC8574EV Includes BSP file: vtss_appl_board_tesla_eval.c
VIPER_EVAL_BOARD	Configures Sample Application for VSC8584EV Includes BSP file: vtss_appl_board_viper_eval.c
_INCLUDE_DEBUG_FILE_PRINT_	Includes support for Debug output to Debug File
_INCLUDE_DEBUG_TERM_PRINT_	Includes support for Debug output to Console Term

In addition to compilation flags, the MESA API code, which is the same API code that can be included in the customer's application, also has runtime options to verify the capability of the device. Therefore, to use a particular PHY feature, the compiled time option must be selected to include the code, and the runtime option must be configured to enable the feature. Only compiling the code into the image does not necessarily mean that the API code is executed for the specific PHY. The compile time flags listed in [Table 2](#) are used for compile time configuration of the MESA API code.

TABLE 2: MESA API COMPILATION FLAG

Compilation Flag	Description
VTSS_OPSYS_LINUX = 1	Includes Linux OS Definitions
VTSS_OPT_PORT_COUNT = 4	Total number of ports for this PHY
VTSS_CHIP_CU_PHY	Includes 1G VSC Copper PHY Support
VTSS_FEATURE_EEE	Includes EEE Support
VTSS_SW_OPTION_EEE	Includes EEE Workaround for ATOM12 or LUTON26
VTSS_PHY_OPT_CAP_FE_ONLY	Only allows 10M and 100M speed on 1G-capable PHY
VTSS_PHY_OPT_CAP_GE_ONLY	Only allows 1G capability for 10/100/1000M PHY
VTSS_FEATURE_SERDES_MACRO_SETTINGS	Includes SerDes support in 1G PHY
VTSS_OPT_PHY_TIMESTAMP	Includes 1588 timestamp support
KAT_TEST_ENABLE_1G	Includes MACsec Known Answer Test Utilities
VTSS_FEATURE_MACSEC	Includes MACsec Support

When using the Sample Application and MESA API in conjunction with a PHY evaluation board, the configuration should be targeting the evaluation board features as each board may have different options, depending upon the PHY product on the evaluation board.

For a VSC8502 example, the VSC8502 does not contain a SerDes interface, so the software to control the SerDes is not required to be included in the build. Therefore, `VTSS_FEATURE_SERDES_MACRO_SETTINGS` compilation flag is not required. However, there are run-time options to prevent the code from being executed if it is included. If there are a variety of VSC PHYs supported by this image, if the PHY hardware supports SerDes, then this option may be needed to control the SerDes on any given PHY.

For a VSC8574 example, the VSC8574 contains a SerDes interface, so the `VTSS_FEATURE_SERDES_MACRO_SETTINGS` compilation flag must be defined. The VSC8574 is also capable of 1588 PHY Timestamping, so the `VTSS_OPT_PHY_TIMESTAMP` API compilation flag must be set to include this feature in the MESA API software. The Sample Application Evaluation Board definition `EVAL_BOARD_1588_CAPABLE` is also set to indicate that the evaluation board hardware supports IEEE-1588 Timestamping.

For a VSC8584 example, the VSC8584 contains a SerDes interface, so the `VTSS_FEATURE_SERDES_MACRO_SETTINGS` compilation flag must be defined. The VSC8584 is also capable of 1588 PHY Timestamping, so the `VTSS_OPT_PHY_TIMESTAMP` API compilation flag must be set to include this feature in the MESA API software. The Sample Application Evaluation Board definition `EVAL_BOARD_1588_CAPABLE` is also set to indicate that the evaluation board hardware supports IEEE-1588 Timestamping. The same is true for MACsec Encryption/Decryption. The `VTSS_FEATURE_MACSEC` MESA API compilation flag is set to include the MACsec API code in the build while the `EVAL_BOARD_MACSEC_CAPABLE` Sample Application compilation flag is set to indicate the evaluation board hardware has these capabilities.

The PHY evaluation board may be configured differently depending upon the PHY. In some cases, up to four RJ45 connectors are provided for copper media interfaces, while in other cases, there are up to four SFP cages that allow for evaluation of the fiber media pluggable modules. Both of these media types may be populated on the evaluation board and the Sample Application must be configured to select one type of media interface. The MAC interface connections for SGMII and QSGMII are provided via SMA connectors, while the RGMII MAC interface is provided in another connector type. For configuration of the specific PHY evaluation board (VSC8574EV, VSC8584EV, VSC8514EV, and VSC8502EV), see [Section 1.2, "References"](#) for location of the user guides.

3.3 Creating an Executable File

The code is compiled and an executable file is created using the GNU Compiler Collection, or gcc. The GNU Compiler Collection includes front ends for a variety of languages and options. The MESA Sample Application code, as well as the PHY API that is linked to the MESA Sample Application, is written in Standard C.

The build files are located in the `mesa-2021.03/phy_demo_app1` directory and are listed in [Table 3](#) for each PHY evaluation board.

TABLE 3: MESA API – SAMPLE APPLICATION BUILD FILES

Name of Build File	Sample Application Created
build_elise_vsc8514ev.sh	vtss_api_elise_vsc8514ev (VSC8514)
build_nano_vsc8502ev.sh	vtss_api_nano_vsc8502ev (VSC8501/VSC8502)
build_tesla_vsc8574ev.sh	vtss_api_tesla_vsc8574ev (VSC8574/VSC8572/ VSC8504/VSC8552)
build_viper_vsc8584ev.sh	vtss_api_viper_vsc8584ev (VSC8584/VSC8582/ VSC8575/VSC8562/VSC8564)

The following is an example of creating an executable file for VSC8584EV (vtss_api_viper_vsc8584ev) with Timestamping and MACsec enabled.

```
gcc -g -o vtss_api_viper_vsc8584ev -I ../include -I boards -DVTSS_OPSYS_LINUX=1 -
DVTSS_OPT_PORT_COUNT=4 -DKAT_TEST_ENABLE 1G -DVTSS_CHIP_CU_PHY -
DVIPER_EVAL_BOARD -D_INCLUDE_DEBUG_TERM_PRINT_ -DVTSS_FEATURE_EEE -
DVTSS_SW_OPTION_EEE -DVTSS_FEATURE_SERDES_MACRO_SETTINGS -
DVTSS_OPT_PHY_TIMESTAMP -DVTSS_FEATURE_MACSEC
appl/vtss_appl_board_viper_eval.c appl/vtss_appl_vsc_phy_1g.c
appl/vtss_appl_ts_demo.c appl/vtss_appl_macsec_demo.c
appl/vtss_appl_macsec_kat_test.c
../base/ail/vtss_api.c
../base/ail/vtss_common.c
../base/phy/common/vtss_phy_common.c
../base/phy/ts/vtss_phy_ts_api.c
../base/phy/ts/vtss_phy_ts_util.c
../base/phy/macsec/vtss_macsec_api.c `find
../base/phy/phy_1g -name "*.c"
```

Note 1: The Sample Application Basic configuration is located in the vtss_appl_vsc_phy_1g.c file.

2: The Sample Appl PTP configuration is located in the vtss_appl_ts_demo.c file.

3: The Sample Appl MACsec configuration is located in the vtss_appl_macsec_demo.c file.

3.4 Connecting to an Evaluation Board

Each of the PHY evaluation boards has a Rabbit daughter board that can interface with a PC either through a direct connection to the PC or through a local area network if configured properly. The latter option requires the user to configure the Rabbit's IP address so as to properly reside on the user's network.

The IP address of the board should be written on the Rabbit network interface daughter board card. The default value should be 10.9.70.193. This IP address must be used to initially access the board for operation or to change its IP address.

3.4.1 CHANGING THE IP ADDRESS OF THE BOARD

To change the IP address of the board:

1. Determine and write down the new unique IP address you want to change the board to.
2. Directly connect an Ethernet cable from a PC to the Rabbit board.

Note: Some older PCs do not support auto-crossover on the Ethernet connection so a cross-over cable may be needed.

3. Launch a DOS command window by clicking Start>Run and typing cmd.
4. In the DOS command window, enter Telnet.
5. In Telnet, connect to the Rabbit board's address using the open command by typing open^10.9.70.xxx, where xxx is the value on your board from the factory (typically 193).

You should get a prompt and be able to type help to get a list of commands available on the Rabbit.

If you are unable to connect, then most likely you will need to change the IP address of the connected PC to have the first three octets similar to the board by following the subsequent steps:

- a) Go to *Windows>Control Panel>Network Connections>Local Area Connection*, and then right click the PC for Properties.
 - b) Under the **General** tab, highlight "Internet Protocol (TCP/IP)" and click Properties.
 - c) Enter the new PC IP address such as 10.9.70.yyy, where yyy is a unique value and *not* the same as the Rabbit board.
 - d) Once complete, return to step 4.
6. Command the board to change its IP address to the new one by typing into Telnet connected to the board. The command is: `set ip <new IP address> <Enter>`, where <new IP address> is in the form xxx.xxx.xxx.xxx. Once you click <Enter>, the IP address will be changed and the Rabbit will save the value and reboot for approximately one minute. The Telnet session will disconnect from the board.
7. Change your PC IP address to the same IP network as the Rabbit board.
8. Establish a Telnet connection to the Rabbit board.
9. Use the following commands to complete configuration of the Rabbit board:
- ```
set netmask xxx.xxx.xxx.xxx
set gateway xxx.xxx.xxx.xxx
save env
```
10. Record and inform Microchip of the new IP address of the board when you return, so that Microchip can connect to and reconfigure the board.
11. Re-label the Rabbit board with the new IP address.

### 3.5 Launching the Sample Application

**Note:** Before launching the Sample Application, make sure to first perform the steps in [Section 3.3, "Creating an Executable File"](#).

The Sample Application provides options that may be selected to test certain PHY API functionalities. The initial configuration of the PHY can be used to test the default functions, which would also be used by an Application to initialize the PHY.

A single option or multiple options may be executed as necessary to test desired functionalities. These options are API functionalities that may result in multiple register read/write sequences, which are also used by a customer application.

To execute the Sample Application on a laptop, the IP address of the PHY evaluation board must be entered to access the PHY evaluation board.

For example, for an IP address of 10.9.70.193:

- For VSC8514EV:
 

```
./vtss_api_elise_vsc8514ev 10.9.70.193
```
- For VSC8502EV:
 

```
./vtss_api_nano_vsc8502ev 10.9.70.193
```
- For VSC8574EV:
 

```
./vtss_api_tesla_vsc8574ev 10.9.70.193
```
- For VSC8584EV:
 

```
./vtss_api_viper_vsc8584ev 10.9.70.193
```

See [Section 6.0, "Configuring and Using the Sample Application"](#) to configure and set up the Sample Application and output.

## 3.6 Command Line Prompts from the Sample Application

The user will be queried about some general setup, Trace Logging and enabling of Debug output to the console in particular. If answering “Y” (or Yes) for these, the user will be prompted to set the trace level, and trace logging will be displayed to the console. The prompts are shown in the following:

```
Change TRACE Logging to TERM Output? (Y/N)
```

```
N
```

```
Enable Live Debug TERM Output? (Y/N)
```

```
N
```

A general PHY status is also displayed as the PHY evaluation board is being configured. Note that a connection depending on end user's particular MAC and media selections must be made before traffic can pass; the MAC or media electrical connections are covered in [Section 4.0, "Configuring the PHY Evaluation Board MAC Interface for Testing"](#).

Once the VSC 1G PHY initialization is complete, a menu with various options is displayed. At this point, the PHY is configured to pass traffic through the PHY.



### 3.6.1 TYPICAL CONSOLE OUTPUT

The following output was captured from the initialization sequence of a VSC8514 Evaluation Board. User inputs are shown in bold text. The Sample Application was executed using the following:

```
./vtss_api_elise_vsc8514ev 10.132.27.22
```

```
//Default AIL TRACE Level : VTSS_TRACE_LEVEL_ERROR
//Default CIL TRACE Level : VTSS_TRACE_LEVEL_ERROR
Change TRACE Logging to TERM Output? (Y/N)
N
//Default Setup Assumptions:
//Eval Board has Power Applied prior to Demo start
//The PHY Initialization Timing delays have been satisfied
//Pwr Supply/Voltages Stable, Ref Clk Stable, Ref & PLL Stable

Enable Live Debug TERM Output? (Y/N)
N
//Comment: Board being initialized
//Comment: Step 1: GET the Default PHY Config from the Instance for Port: 0
//Comment: Step 2: Running PHY vtss_phy_pre_reset on base port of PHY
//Comment: Setup of VSC8514 - Elise Eval Board
//Current Setup: PHY Mode: ANEG Enabled: 1G_FDX: 0x1; 1G_HDX: 0x0
// Config: PHY Mode: ANEG Enabled: FDX: 0x1; HDX: 0x0

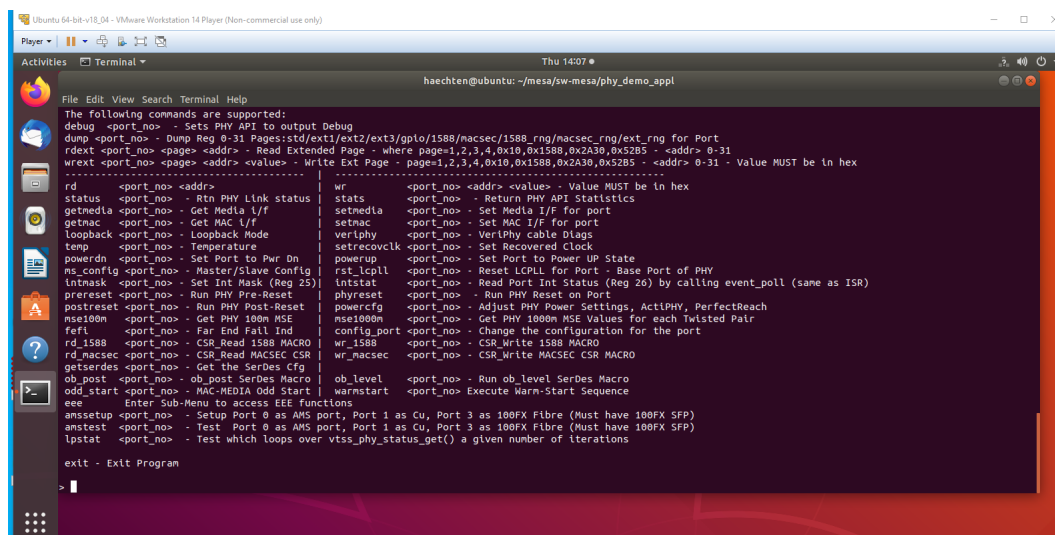
//Comment: miim_read format : miim_read(port_number, address)
//Comment: miim_write format : miim_write(port_number, address, value(hex))

ALL MAC I/F defaulted to: QSGMII
ALL MEDIA I/F defaulted to: COPPER_I/F
//Comment: PHY port reset
//Comment: Step 3: Perform PHY port config on PHY port: 0
//Comment: Step 4: Perform PHY port 1G_config on PHY port: 0
// PHY port 0, Reset Complete; Config Complete, Port config: Manual_Master: 0x0, 1=Master/
0=Slave: 0x0
//Comment: Step 5: Perform PHY port reset on PHY port: 0
//Comment: Step 3: Perform PHY port config on PHY port: 1
//Comment: Step 4: Perform PHY port 1G_config on PHY port: 1
// PHY port 1, Reset Complete; Config Complete, Port config: Manual_Master: 0x0, 1=Master/
0=Slave: 0x0
//Comment: Step 5: Perform PHY port reset on PHY port: 1
//Comment: Step 3: Perform PHY port config on PHY port: 2
//Comment: Step 4: Perform PHY port 1G_config on PHY port: 2
// PHY port 2, Reset Complete; Config Complete, Port config: Manual_Master: 0x0, 1=Master/
0=Slave: 0x0
```

```
//Comment: Step 5: Perform PHY port reset on PHY port: 2
//Comment: Step 3: Perform PHY port config on PHY port: 3
//Comment: Step 4: Perform PHY port 1G_config on PHY port: 3
// PHY port 3, Reset Complete; Config Complete, Port config: Manual_Master: 0x0, 1=Master/
0=Slave: 0x0
//Comment: Step 5: Perform PHY port reset on PHY port: 3
// PHY port 4, Reset Complete; Config Complete, Port config: Manual_Master: 0x0, 1=Master/
0=Slave: 0x0
//Comment: Last_Step: Running PHY vtss_phy_post_reset on base port of PHY
Temp Init RC:x0
Initial Status after Init, populating API fields
Link DOWN! (status=1)
Link: 0x0, Speed: 0x0, FDX: 0x0, Remote_Fault: 0x0, ANEG_Complete: 0x0, Fiber: 0x0,
MDI_Cross: 0x0
Temp Init RC:x0
Initial Status after Init, populating API fields
Link DOWN! (status=1)
Link: 0x0, Speed: 0x0, FDX: 0x0, Remote_Fault: 0x0, ANEG_Complete: 0x0, Fiber: 0x0,
MDI_Cross: 0x0
Temp Init RC:x0
Initial Status after Init, populating API fields
Link DOWN! (status=1)
Link: 0x0, Speed: 0x0, FDX: 0x0, Remote_Fault: 0x0, ANEG_Complete: 0x0, Fiber: 0x0,
MDI_Cross: 0x0
Temp Init RC:x0
Initial Status after Init, populating API fields
Link DOWN! (status=1)
Link: 0x0, Speed: 0x0, FDX: 0x0, Remote_Fault: 0x0, ANEG_Complete: 0x0, Fiber: 0x0,
MDI_Cross: 0x0
```

The following menu then appears (Figure 1):

**FIGURE 1: MENU DISPLAYED**



## 4.0 CONFIGURING THE PHY EVALUATION BOARD MAC INTERFACE FOR TESTING

When using a PHY evaluation board, connecting the MAC interface to a network processor or other type of MAC may be a challenge. Thus, the following connections can be used to connect either (a) two PHY ports back-to-back via the MAC interface using SGMII connections, or (b) two PHY evaluation boards back-to-back to let traffic flow through one of the PHY's line side ports and make the traffic available on another port, either on the same PHY or on the PHY of another evaluation board. These setups are very useful for test and evaluation and for troubleshooting issues.

### 4.1 Configuring the PHY Evaluation Board MAC Interface SGMII to Loopback Traffic Port-To-Port on the Same Evaluation Board

The following connections can be used to connect two PHY ports back-to-back via the MAC interface SGMII connections to flow traffic through one of the PHY's line side ports and the traffic available on another line side port on the same PHY.

The following configuration for the VSC8574EV or VSC8584EV connects Port 0 SGMII MAC interface to Port 1 SGMII MAC interface. Traffic can flow into Port 0 and be seen on the Port 1 output and vice-versa. The PHY must be configured to use the SGMII MAC interface via the Sample Application.

**TABLE 4: SGMII LOOPBACK CONNECTIONS (PORT 0 TO PORT 1)**

| Traffic Gen/Mon ↔ Port 0 ↔ SGMII ↔ Port 1 ↔ Traffic Gen/Monitor |              |               |
|-----------------------------------------------------------------|--------------|---------------|
| J1 (TDIN0+)                                                     | Connected to | J6 (RXDOUT1+) |
| J2 (TDIN0-)                                                     | Connected to | J7 (RXDOUT1-) |
| J4 (RXDOUT0+)                                                   | Connected to | J9 (TXDIN1+)  |
| J5 (RXDOUT0-)                                                   | Connected to | J10 (TXDIN1-) |

**Note 1:** RF cables with SMA connectors should be of the same length.

**TABLE 5: SGMII LOOPBACK CONNECTIONS (PORT 2 TO PORT 3)**

| Traffic Gen/Mon ↔ Port 2 ↔ SGMII ↔ Port 3 ↔ Traffic Gen/Monitor |              |                |
|-----------------------------------------------------------------|--------------|----------------|
| J19 (TXDIN2+)                                                   | connected to | J14 (RXDOUT3+) |
| J20 (TXDIN2-)                                                   | connected to | J15 (RXDOUT3-) |
| J11 (RXDOUT2+)                                                  | connected to | J16 (TXDIN3+)  |
| J12 (RXDOUT2-)                                                  | connected to | J18 (TXDIN3-)  |

**Note 1:** RF cables with SMA connectors should be of the same length.

### 4.2 Configuring the PHY Evaluation Board MAC Interface for QSGMII to Flow Traffic Between Ports on Different Evaluation Boards

There are instances when two evaluation boards may be required to be connected back-to-back. The following connections can be used to connect two ports via the MAC interface QGMII connections to flow traffic through the PHY's line side interfaces on one evaluation board and observe the traffic on the same port on another evaluation board.

The following configuration ([Table 6](#)) can be used for VSC8574EV, VSC8584EV, or VSC8514EV, which all support the QSGMII MAC interface and all have the SMA connectors labeled the same. The PHY must be configured to use the QSGMII MAC interface via the Sample Application.

The VSC8514EV/VSC8574EV/VSC8584EV boards have the same QSGMII MAC interface SMA connections. The following configuration will connect the QSGMII MAC interface from PHY#1 to the QSGMII MAC interface from PHY#2. The PHY#1 (VSC8514EV/VSC8574EV/VSC8584EV) connected via QSGMII to PHY#2 (VSC8514EV/VSC8574EV/VSC8584EV).

**TABLE 6: QSGMII CONNECTIONS FOR BACK-TO-BACK EVALUATION BOARD SETUP**

| Traffic Gen/Mon $\leftrightarrow$ PHY_0 Port 0 $\leftrightarrow$ QSGMII $\leftrightarrow$ PHY_1 Port 0 $\leftrightarrow$ Traffic Gen/Monitor |              |                     |
|----------------------------------------------------------------------------------------------------------------------------------------------|--------------|---------------------|
| PHY_0 J1 (TDIN0+)                                                                                                                            | Connected to | PHY_1 J4 (RXDOUT0+) |
| PHY_0 J2 (TDIN0-)                                                                                                                            | Connected to | PHY_1 J5 (RXDOUT0-) |
| PHY_0 J4 (RXDOUT0+)                                                                                                                          | Connected to | PHY_1 J1 (TXDIN1+)  |
| PHY_0 J5 (RXDOUT0-)                                                                                                                          | Connected to | PHY_1 J2 (TXDIN1-)  |

**Note 1:** QSGMII, therefore all four ports pass traffic across the Port 0 QSGMII MAC interface.

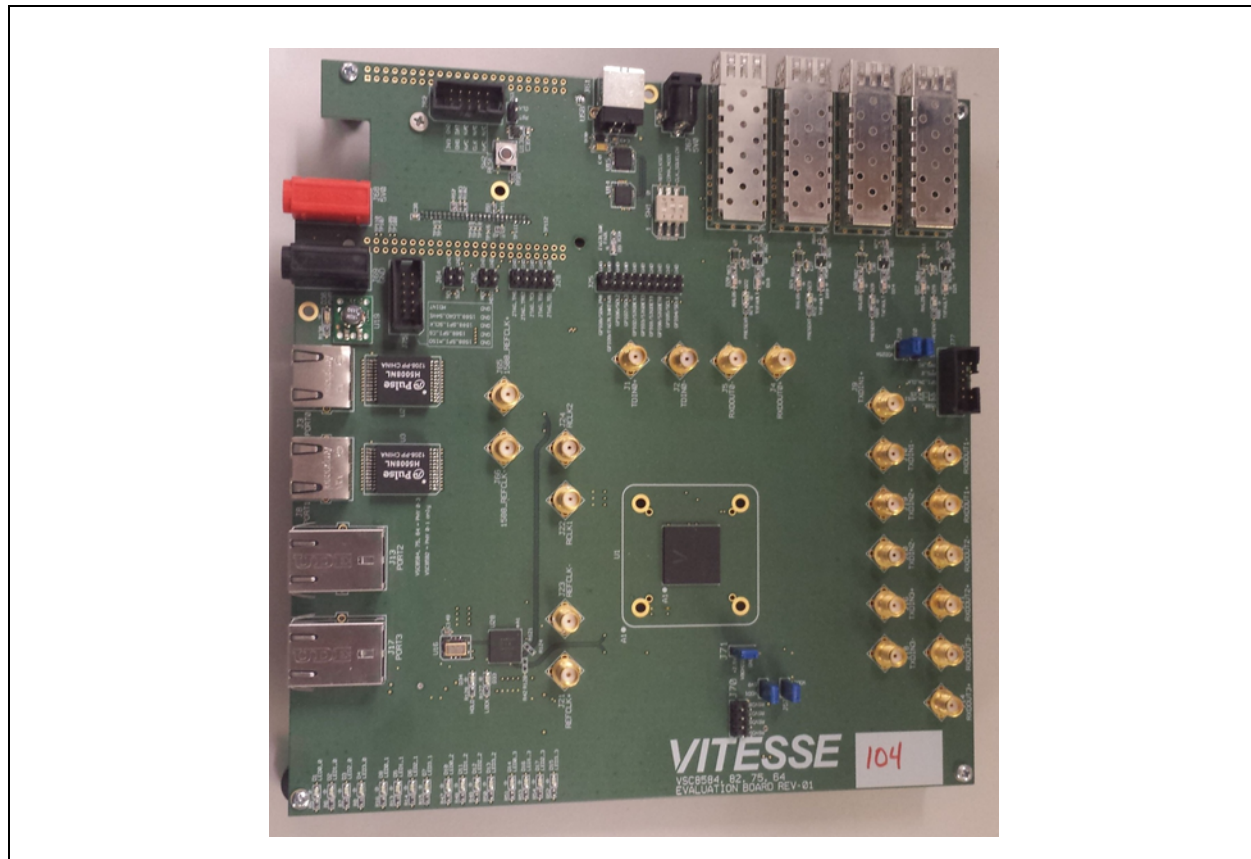
**2:** RF cables with SMA connectors should be of the same length.

## 4.3 Configuring the Evaluation Board for Traffic

The Sample Application will execute on a laptop and read the `PHY_ID` to ensure that the correct PHY/PHY evaluation board type has been accessed.

The VSC8584EV Evaluation Board is shown in [Figure 2](#):

**FIGURE 2: VSC8584EV EVALUATION BOARD**



### 4.3.1 CONFIGURING VSC8584EV FOR PTP TRAFFIC

The following VSC8584EV configuration loops back the traffic on the MAC interface in a port-to-port configuration. This configuration loops the MAC interface data from Port 0 to Port 1 and from Port 2 to Port 3. In this configuration, the PHY MAC interface should be SGMII so that the ports can be looped back in this fashion. In this configuration, the application can exercise the PHY's IEEE-1588 capability. See [Section 4.1, "Configuring the PHY Evaluation Board MAC Interface SGMII to Loopback Traffic Port-To-Port on the Same Evaluation Board"](#) for configuration of the PHY evaluation board MAC interface SGMII to loopback traffic port-to-port.

For the IEEE-1588 configuration, the use of a Cat 5 cable connecting evaluation Ports 1 and 2 together on the Line-Side interface is *not* required. The following flow diagram shows the VSC8584EV interconnection that will exercise the PHY IEEE-1588 capability.

Figure 3 shows an evaluation board configured in this manner without the Traffic Gen or Traffic Monitor. The evaluation board Traffic Flow setup is as follows:

Traffic Gen → Port 0 (Ingress TS) → SGMII → Port 1 (Egress TS) → Monitor

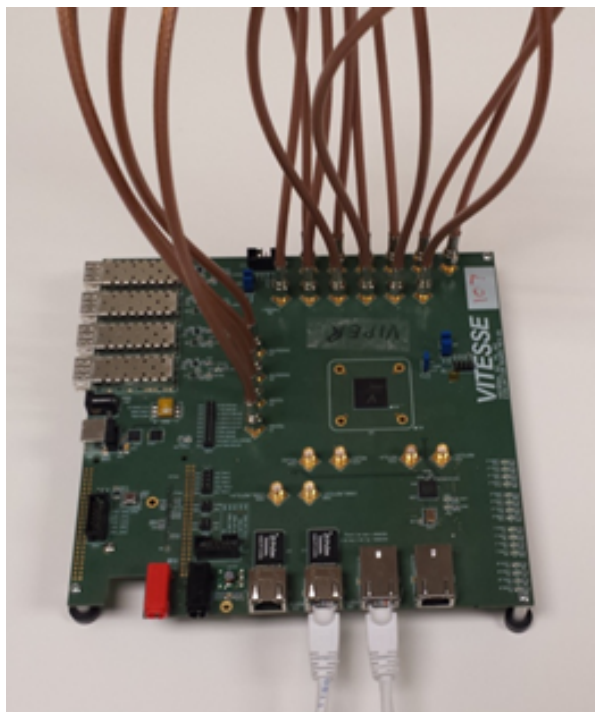
Traffic Gen → Port 2 (Ingress TS) → SGMII → Port 3 (Egress TS) → Monitor

Or:

Traffic Gen → Port 1 (Ingress TS) → SGMII → Port 0 (Egress TS) → Monitor

Traffic Gen → Port 3 (Ingress TS) → SGMII → Port 2 (Egress TS) → Monitor

**FIGURE 3: VSC8584EV EVALUATION BOARD CONFIGURED FOR PTP/MACSEC OPERATION**



**Note:** In Figure 3, for configuring VSC8584EV for PTP traffic:

- a) RF cables with SMA connectors should be of the same length.
- b) Port 0 and Port 3 (RJ45) are connected to Traffic Generator/Monitor.
- c) Cat 5 cables shown in Figure 3 connecting RJ45 Port 1 (J8) and RJ45 Port 2 (J13) are *not* connected for PTP setup.

#### 4.3.2 CONFIGURING VSC8584EV FOR MACSEC

The following VSC8584EV configuration loops back the traffic on the MAC interface in a port-to-port configuration. This configuration loops the MAC interface data from Port 0 to Port 1 and from Port 2 to Port 3. In this configuration, the PHY MAC interface should be SGMII so that the ports can be looped back in this fashion. In this configuration, the application can exercise the PHY's MACsec Encryption and Decryption capability. See [Section 4.1, "Configuring the PHY Evaluation Board MAC Interface SGMII to Loopback Traffic Port-To-Port on the Same Evaluation Board"](#) for configuration of the PHY evaluation board MAC interface SGMII to loopback traffic port-to-port.

# AN4028

---

For the MACsec configuration, the only additional configuration is the use of a Cat 5 cable connecting evaluation Ports 1 and 2 together on the Line-Side interface. The following flow diagram shows the VSC8584EV interconnection that will exercise the PHY Encryption and Decryption capability.

Refer to [Figure 3](#) for an illustration of an evaluation board configured in this manner without the Traffic Gen or Traffic Monitor. The MACsec demonstration setup is as follows:

Traffic Gen → Port 0 → SGMII → Port 1 (Encrypt) → CAT5 Loop → Port 2 (Decrypt) → SGMII → Port 3 → Monitor

Or:

Traffic Gen → Port 3 → SGMII → Port 2 (Encrypt) → CAT5 Loop → Port 1 (Decrypt) → SGMII → Port 0 → Monitor

**Note:** In [Figure 3](#), for configuring VSC8584EV for MACsec:

- a) RF cables with SMA connectors should be of the same length.
- b) Port 0 and Port 3 (RJ45) are connected to Traffic Generator/Monitor.
- c) Cat 5 loopback cable is connected to RJ45 Port 1 (J8) and RJ45 Port 2 (J13).

## 5.0 CONFIGURING THE TRAFFIC GENERATOR

### 5.1 Configuring the Traffic Generator for PTP Traffic

#### 5.1.1 ETH-PTP ENCAPSULATION (ETH-PTP)

To configure the traffic generator for PTP Only 1588 Demonstration setup:

Traffic Gen → Port 0 (Ingress TS) → SGMII → Port 1 (Egress TS) → Monitor

Destination MAC address to be timestamped: 00:00:00:00:00:01

EtherType: IEEE\_PTP\_1588 (0x88F7)

Sample Basic Frame

Ethernet Packet Format: |DA|SA| Etype|Payload|CRC|

```
Sample 96Byte Sync Frame Encap: ETH-PTP:
00 00 00 00 00 01 00 00 00 00 00 02 88 F7 00 02
00 2C 00 00 02 00 00 00 00 00 00 00 00 00 00 00
00 00 AC DE 48 00 00 00 00 00 00 01 00 05 05 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

#### 5.1.2 ETH-IP-PTP ENCAPSULATION (ETH-IPV4-PTP)

To configure the traffic generator for IP encapsulation of PTP for 1588 Demonstration setup, the encapsulation is ETH-IPv4-UDP-PTP.

Traffic Gen → Port 0 (Ingress TS) → SGMII → Port 1 (Egress TS) → Monitor

Destination MAC address to be timestamped: 00:00:00:00:00:01

The four byte IP\_DEST\_ADDR must be: 0a0a0a0a

EtherType: IEEE\_IPV4 (0x0800)

Sample Basic Frame

Ethernet Packet Format: |DA|SA| Etype|Payload|CRC|

```
Sample 96Byte Sync Frame ETH-IPv4-UDP-PTP:
00 00 00 00 00 01 00 10 94 00 00 02 08 00 45 00
00 14 00 00 00 00 FF 11 E6 6D C0 55 01 02 0A 0A
0A 0A 00 00 01 3F 00 00 28 C2 00 02 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 7F 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

#### 5.1.3 ETH-IP-PTP ENCAPSULATION (ETH-IPV4-PTP) WITH VLAN

To configure the traffic generator for IP encapsulation of PTP for 1588 Demonstration setup, the encapsulation is ETH-IPv4-UDP-PTP.

Traffic Gen → Port 0 (Ingress TS) → SGMII → Port 1 (Egress TS) → Monitor

Destination MAC address to be timestamped: 00:00:00:00:00:01

The four byte IP\_DEST\_ADDR must be: 0a0a0a0a

EtherType: VLAN 0x8100

VLAN = 01

EtherType: IEEE\_IPV4 (0x0800)

## Sample Basic Frame

Ethernet Packet Format: |DA|SA| Etype|Payload|CRC|

```
Sample 96Byte Sync Frame ETH-IPv4-UDP-PTP w/VLAN V_ID=01:
00 00 00 00 00 01 00 00 00 00 00 02 81 00 00 01
08 00 45 00 00 14 00 00 00 00 FF 11 A6 C4 01 00
00 01 0A 0A 0A 0A 00 00 01 3F 00 00 28 C2 00 02
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 7F
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

### 5.1.4 ETH-IP-PTP ENCAPSULATION (ETH-IPV6-PTP)

To configure the traffic generator for IP encapsulation of PTP for 1588 Demonstration setup, the encapsulation is ETH-IPv6-UDP-PTP.

Traffic Gen → Port 0 (Ingress TS) → SGMII → Port 1 (Egress TS) → Monitor

Destination MAC address to be timestamped: 00:00:00:00:00:01

The 16 byte IP\_DEST\_ADDR must be:

```
A0 00 00 0A
B0 00 00 0B
C0 00 00 0C
00 00 00 01
```

EtherType: IEEE\_IPV6 (0x86DD)

## Sample Basic Frame

Ethernet Packet Format: |DA|SA| Etype|Payload|CRC|

```
Sample 128Byte Sync Frame ETH-IPv6-UDP-PTP:
00 00 00 00 00 01 00 10 94 00 00 02 86 DD 60 00
00 00 00 25 11 FF 20 00 00 00 00 00 00 00 00 00
00 00 00 00 00 02 A0 00 00 0A B0 00 00 0B C0 00
00 0C 00 00 00 01 00 00 01 3F 00 10 DE 89 00 02
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 7F
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

### 5.1.5 ETH-IP-PTP ENCAPSULATION (ETH-IPV6-PTP) WITH VLAN

To configure the traffic generator for IP encapsulation of PTP for 1588 Demonstration setup, the encapsulation is ETH-IPv6-UDP-PTP.

Traffic Gen → Port 0 (Ingress TS) → SGMII → Port 1 (Egress TS) → Monitor

Destination MAC address to be timestamped: 00:00:00:00:00:01

The 16 byte IP\_DEST\_ADDR must be:

```
A0 00 00 0A
B0 00 00 0B
C0 00 00 0C
00 00 00 01
```

EtherType: IEEE\_IPV6 (0x86DD)

VLAN = 01



## Sample Basic Frame

Ethernet Packet Format: |DA|SA|Etype|Payload|CRC|

```

Sample 128Byte Sync Frame:
ETH-IPv6-UDP-PTP w/VLAN VLAN_ID=01:
00 00 00 00 00 01 00 10 94 00 00 02 81 00 00 01
86 DD 60 00 00 00 00 25 11 FF 20 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 02 A0 00 00 0A B0 00
00 0B C0 00 00 0C 00 00 00 01 00 00 01 3F 00 10
CE 66 00 02 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 7F 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

```

## 5.2 Configuring the Evaluation Board and Traffic Generator for MACsec Traffic

### 5.2.1 CONFIGURING THE TRAFFIC GENERATOR FOR MACSEC

To configure the traffic generator for ETH-IPv4 traffic for the demonstration of MACsec encapsulation show Encryption/Decryption capabilities. Once the MACsec use case has been enabled, Port 1 and Port 2 are configured for MACsec operation. The PHY encrypts data upon egress of the PHY port and it decrypts data upon ingress into the PHY port. Ingress traffic must be of IPv4 traffic (EtherType = 0x0800) to be encrypted. The encrypted traffic that flows between Port 1 and Port 2 will have the designation for EtherType corresponding to IEEE 802.1AE (EtherType = 0x88e5) and the data is encrypted. Once the data is decrypted, the traffic will return to IPv4 traffic (EtherType = 0x0800).

The following sample IPv4 and IEEE 802.1AE packet descriptions aid the user in identifying packet elements:

Sample of Unencrypted IPv4 packet (etherType=0x0800):

Ethernet Packet Format: |DA|SA|Etype|Payload|CRC|

Ethernet Type = Etype

Sample 96Byte Sync Frame ETH-IPv4-UDP-PTP:

```

00 00 00 00 00 01 00 10 94 00 00 02 08 00 45 00
00 14 00 00 00 00 FF 11 E6 6D C0 55 01 02 0A 0A
0A 0A 00 00 01 3F 00 00 28 C2 00 02 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 7F 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

```

## 6.0 CONFIGURING AND USING THE SAMPLE APPLICATION

The Microchip MESA API Software provides a software interface to control the PHY features/functionality of VSC PHYs by programming the PHY for the feature, instead of single register access. The MESA API Sample Application provides the user a way to program the PHY functionality and provides a Command Line Interface (CLI) for ease of operation and demonstration. Many of the PHY features controlled by the MESA API are exposed via the CLI in the Sample Application, which allows the user to configure and reconfigure the PHY features without recompiling the code.

### 6.1 Understanding the Basics of MESA PHY API

To understand the MESA API Sample Application, the user must first have a basic understanding of the MESA API.

#### 6.1.1 MESA API BASICS

The MESA API provides a logical interface to access and control PHY features and functionality. The API uses a block of memory, called the PHY instance or `PHY_INST`, as a container to store configurations and status. Each time an API is used to configure the PHY, the API software writes the values to the PHY registers and also retains these values in the `PHY_INST`. When values are being read from the PHY (that is, PHY status), the status may also be retained in the `PHY_INST`.

This memory block is allocated at the beginning of operations on the PHY. The `PHY_INST` is created (memory allocate) by the MESA API `vtss_inst_create()`. The `PHY_INST` reference (handle) must be maintained by the application and used each time the PHY is being accessed.

The API abstracts the PHY chip using the `PHY_INST`, which contains:

- I/O layer hooks for register read/write callout functions
- Configuration parameters, status, and initialization configurations
- Channel modes
- Mutex lock/unlock to ensure single-threading within a single `PHY_INST`

After the instance creation, initialization configuration of the instance *should be updated* with I/O layer read/write callout functions and channel modes.

The update activates the channel in configured modes. With this static initialization alone, traffic should pass through the device.

The application may have a single or multiple `PHY_INST`. The size of the `PHY_INST` is determined at compile time. The compilation flag, `VTSS_OPT_PORT_COUNT`, is used to indicate the maximum number of ports that will be allocated for any `PHY_INST` that is created.

- The smallest `PHY_INST` is over a single PHY (not per PHY port). In this case, the value of `VTSS_OPT_PORT_COUNT` would be the maximum ports on the PHY.
- A single `PHY_INST` can have multiple PHYs (`PHY_INST` size determined at compile time). In this case, the value of `VTSS_OPT_PORT_COUNT` would be the sum of all the ports across all PHYs planned to be within the `PHY_INST`.
- The Sample Application is default configured for the maximum ports on a single PHY.

When updating the configuration in the `PHY_INST`, it is highly recommended to always use a “Get/Set” combination. Most configuration APIs have a `_get` and a `_set` option.

- All configurations passed into the API are archived in the `PHY_INST` before being programmed into the PHY HW registers.
- By using a `_get` API function, the data structure gets initialized/prepopulated with the existing values in the `PHY_INST`. Application updates fewer parameters.
- By using a `_set` API function, the data structure passed into the API updates the `PHY_INST` and then applies those settings to the HW registers.

To create a `PHY_INST`, the sequence is: `vtss_inst_get()` followed by `vtss_inst_create()`. This ensures that if the `PHY_INST` already exists, the application will retrieve it, and if it does not exist, then the `PHY_INST` will be created.

Once a `PHY_INST` is created, then initialization begins. The `init_config` portion contains the accessor functions used to access or control the PHY. These functions may be MDIO (MIIM or MMD), or SPI, or both and using `vtss_init_conf_get()` and `vtss_init_conf_set()`.

The initialization functions are unprotected by a mutex, so the application must call these before any other API functions:

- `vtss_inst_get()`, `vtss_inst_create()`
- `vtss_init_conf_get()`, `vtss_init_conf_set()`
- `vtss_port_map_set()`
- *Other API functions are protected*

The application must implement lock/unlock mutex functions:

- `vtss_callout_lock()`
  - called by macro `VTSS_ENTER()`
- `vtss_callout_unlock()`
  - called by macro `VTSS_EXIT`

The functions may be empty for single-threaded applications. The Sample Application code provided is expected to be a single-threaded application, therefore mutexes are not employed.

## 6.2 Using the Sample Application to Configure the PHY Evaluation Board for Basic Traffic

### 6.2.1 CREATING, INITIALIZING, AND CONFIGURING PHY INSTANCE

The Sample Application will retrieve the `PHY_INST` if there is one, and create a new one and get the `init_conf` from the `PHY_INST`.

```
vtss_inst_get(VTSS_TARGET_CU_PHY, &create);
vtss_inst_create(&create, &inst);
vtss_init_conf_get(inst, &init_conf);
```

### 6.2.2 POPULATING BOARD STRUCTURE, FILLING OUT MDIO READ/WRITE ACCESSOR FUNCTIONS, `init_conf`, AND MORE

The Sample Application updates the MIIM accessor functions and configures the type of restart.

```
init_conf.miim_read = miim_read; // Set pointer, MIIM read func for this board.
init_conf.miim_write = miim_write; // Set pointer, MIIM write func for this board.
init_conf.warm_start_enable = TRUE;
init_conf.restart_info_src = VTSS_RESTART_INFO_SRC_CU_PHY;
init_conf.restart_info_port = 0;
```

### 6.2.3 SAVING THE BOARD ACCESSOR FUNCTIONS AND RESTARTING IN PHY INSTANCE

The Sample Application will save the `init_conf` and the `restart_conf`.

```
vtss_restart_conf_set(inst, VTSS_RESTART_WARM);
vtss_init_conf_set(inst, &init_conf)
```

### 6.2.4 SETTING UP TRACE LOGGING CONFIGURATION

The Sample Application will generate some console output. The output may vary slightly depending upon the PHY evaluation board being used and the Logging/Trace settings.

The configuration being applied by the Sample Application may be changed by editing the file: `vtss_appl_vsc_phy_1g.c` to customize for a particular setup.

MESA API includes Trace macros for debugging. The application must implement Log/Trace function.

- `vtss_callout_trace_printf` - Prints or logs a trace message.
- `vtss_callout_trace_hex_dump` - Prints or logs a dump of data in hexadecimal.

Trace logging is very useful in debugging and is used for:

- Verifying proper API call sequencing within the application
- Determining which APIs are being used (missing API calls)
- Unexpected branching or returns

Trace levels are:

- VTSS\_TRACE\_LEVEL\_NONE
- VTSS\_TRACE\_LEVEL\_ERROR
- VTSS\_TRACE\_LEVEL\_INFO
- VTSS\_TRACE\_LEVEL\_DEBUG
- VTSS\_TRACE\_LEVEL\_NOISE

Trace levels may be changed for each trace group dynamically.

- `vtss_trace_conf_get()`
- `vtss_trace_conf_set()`

Trace may be excluded at compile time using: Set `VTSS_OPT_TRACE` to 0.

## 6.2.5 PHY API CONFIGURATION SEQUENCE

The Sample Application will get the `PHY_Config`, `PHY_1g_Config`, and `Reset_Cfg` from the `PHY_INST`.

```
vtss_phy_conf_get(inst, port_no, &vtss_phy_conf);
vtss_phy_conf_1g_get(inst, port_no, &phy_1g);
vtss_phy_reset_get(inst, port_no, &phy_resetCfg);
```

## 6.2.6 ADJUSTING PHY\_CONFIG, CONFIG\_1G, AND RESET\_CFG FOR BOARD/SETUP

The Sample Application will modify the `PHY_Config`, `PHY_1g_Config`, and `Reset_Cfg` retrieved from the `PHY_INST`, and select the MAC/MEDIA interface configuration and the mode of operation.

- `phy_resetCfg`: Select MAC interface, media interface, and so on
- `vtss_phy_conf`: Select: ANEG parameters, Force Speed, MDI/MDIX, Fast Link Fail, SigDet, MAC\_if\_PCS, MEDIA\_if\_PCS, Clk\_Skew, and so on
- `phy_1g`: Select: Master/Slave Manual Selection

## 6.2.7 APPLYING PHY\_CONFIG, PHY\_1G\_CONFIG, AND RESET\_CFG

The Sample Application will set the `PHY_Config` and `PHY_1g_Config` in the `PHY_INST`, so that immediately after PHY Reset, the proper configuration will be applied.

```
vtss_phy_conf_set(inst, port_no, &vtss_phy_conf);
vtss_phy_conf_1g_set(inst, port_no, &phy_1g);
```

## 6.2.8 INITIALIZING THE PHY

The Sample Application will execute the PHY Pre-Reset function, which applies the initialization scripts to optimize the PHY settings and update the micro-patch. After completion, the PHY Reset is applied with `Reset_Cfg` that has been updated, and selection of MAC/media interface configuration is applied. Once the PHY Reset is complete, the PHY configuration and mode will automatically be applied within the API.

Apply `init_scripts` to optimize settings and update micro-patch.

```
vtss_phy_pre_reset(inst, base_port_no)
```

Apply MAC/media interface selections and reset the PHY port (`Reg0.15=1`).

**Note:** `vtss_phy_reset()` has an implied application for `PHY_Config` and `PHY_1g_Config`.

```
for (port_no = VTSS_PORT_NO_START; port_no < VTSS_PORTS; port_no++) {
 vtss_phy_reset(inst, port_no, &phy_resetCfg); // Do port reset
}
```

Release COMA mode, and make PHY External interface active.

```
vtss_phy_post_reset(inst, base_port_no);
```

## 6.2.9 CONFIGURING SYNCE (OPTIONAL)

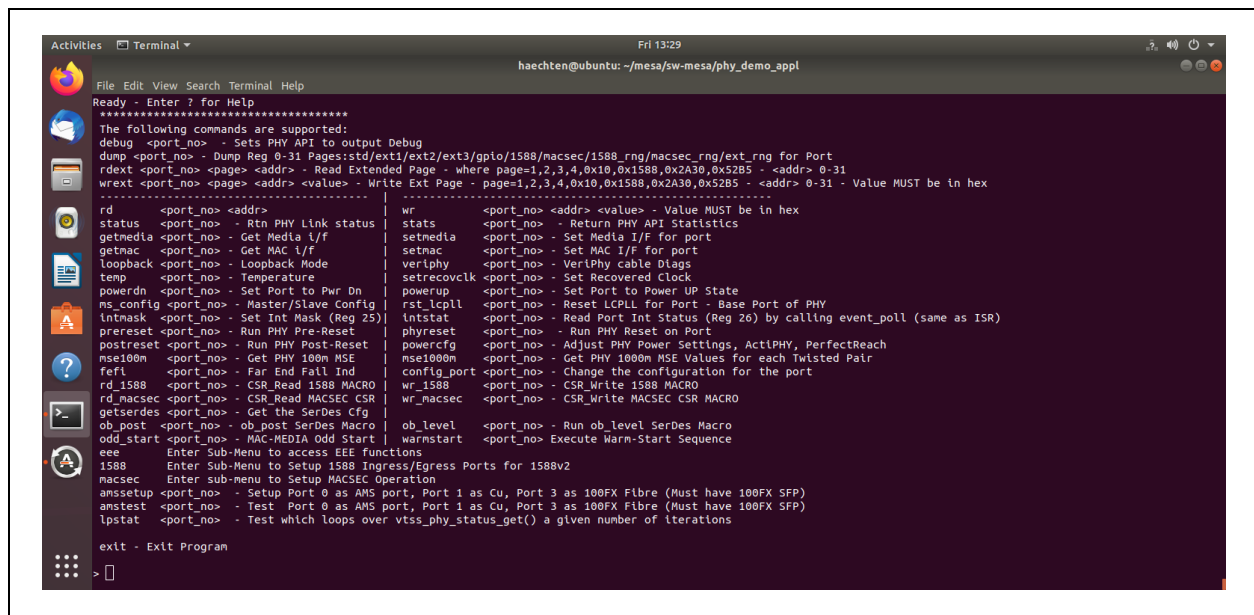
The Sample Application may be configured to retrieve the PHY Recovered Clock configuration using the `_get` API function from the `PHY_INST`, modify the Recovered Clock configuration, and then apply the new Recovered Clock settings using the `_set` API function.

```
vtss_phy_clock_conf_get()
vtss_phy_clock_conf_set()
```

## 6.2.10 LAUNCHING THE SAMPLE APPLICATION

Refer to [Section 3.5, "Launching the Sample Application"](#). After the Sample Application runs through the initialization sequence, a page similar to the one shown below appears on the console.

**FIGURE 4: PHY SAMPLE APPLICATION MAIN MENU**



## 6.3 Using the Sample Application to Configure the PHY Evaluation Board for PTP Traffic

The PTP Sample Application is located in the `vtss_appl_ts_demo.c` file.

For VSC8574 and VSC8584 examples, both PHYs contain a SerDes interface, so the `VTSS_FEATURE_SERDES_MACRO_SETTINGS` compilation flag must be defined. Both PHYs are capable of 1588 PHY Timestamping, hence the `VTSS_OPT_PHY_TIMESTAMP` API compilation flag must be set to include this feature in the MESA API software.

The Sample Application Evaluation Board definition `EVAL_BOARD_1588_CAPABLE` is also set to indicate that the evaluation board hardware supports IEEE-1588 Timestamping.

## 6.3.1 LAUNCHING THE PTP SAMPLE APPLICATION

Prior to configuring the PHY evaluation board for PTP traffic, ensure that basic configuration of the PHY has been performed on *all* ports of the PHY. See [Section 3.0, "Sample Application Overview"](#) and follow the steps to complete basic initialization of the PHY. Once *all* the ports of the VSC 1G PHY initialization is complete, a menu with various options appears. To access the sub-menu for timestamp, the PHY must support IEEE-1588 PTP. These include the following 1G PHYs:

- 1G Tesla PHYs: VSC8574 and VSC8572
- 1G Viper PHYs: VSC8584, VSC8582, and VSC8575

Refer to [Section 5.1, "Configuring the Traffic Generator for PTP Traffic"](#) and configure the Traffic Generator accordingly. Configure the PHY evaluation board according to [Section 4.1, "Configuring the PHY Evaluation Board MAC Interface SGMII to Loopback Traffic Port-To-Port on the Same Evaluation Board"](#), which loops back traffic port-to-port as:

Traffic Gen → Port 0 (Ingress TS) → SGMII → Port 1 (Egress TS) → Monitor

Prior to enabling 1588 operation, ensure that traffic can pass in both directions, from Port 0 to Port 1 and from Port 1 to Port 0. Once the traffic flow from the ingress at Port 0 to the egress at Port 1 and the traffic flow from the ingress at Port 1 to the egress at Port 0 can be verified, configure the customer evaluation board for the IEEE-1588 operation desired.

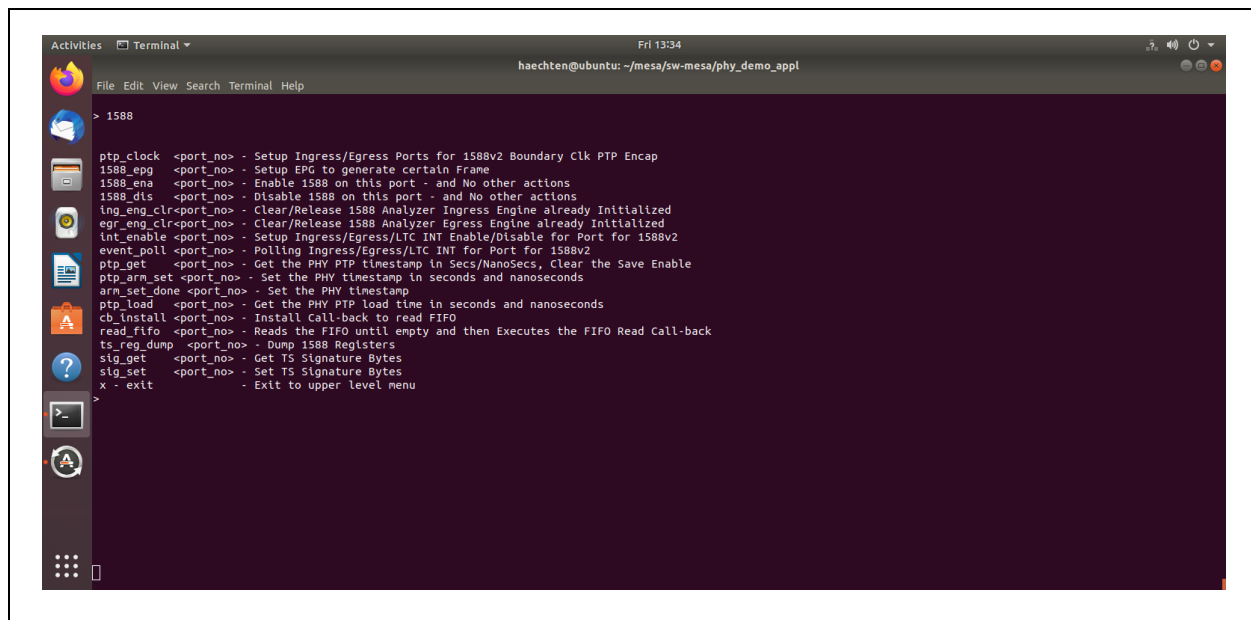
The MESA 1G PHY Sample Application code has an IEEE-1588 CLI menu that appears when the IEEE-1588 Option is selected.

The Sample Application code for the IEEE-1588 code is located in the `phy_demo_appl/appl/vtss_appl_ts_demo.c` file, and the menu can be found in the `vtss_appl_ts_demo_menu` function.

**Note:** When configuring for 1588 Operation, the ports that are being configured for 1588 must have *link-up*.

An example of the `vtss_appl_ts_demo_menu` function is shown in [Figure 5](#):

**FIGURE 5: VTSS\_APPL\_TS\_DEMO\_MENU EXAMPLE**



A single option or multiple options may be executed as necessary to test desired functionalities. These options are API functionalities that may result in multiple register read/write sequences, which are also used by a customer application.

The `vtss_appl_ts_demo_menu` has a `ptp_clock` menu option that executes `vtss_ptp_sample_clock()`. This sequences the initialization of the IEEE-1588 Block for the port selected, and the sequence is:

- [PHY API Configuration Sequence for PTP Block](#)
- [Initializing the Ingress and Egress Timestamp Engines](#)
- [Initializing the Ingress and Egress Flows](#)
- [Initializing the Ingress and Egress Actions](#)

Additional configuration may be needed for a particular setup and must be selected from the CLI menu to accomplish the following:

- [Initializing the Signature Used in Timestamp FIFO](#)
- [Initializing/Setting the Local Time Counter](#)
- [Saving and Retrieving the Local Time Counter](#)
- [Enabling and Polling for IEEE-1588 Events](#)
- [Installing the TS FIFO Read Callback](#)
- [Emptying TS FIFO After TS Event](#)

#### 6.3.1.1 PHY API Configuration Sequence for PTP Block

The Sample Application will execute the `vtss_ptp_sample_clock()` function to initialize the IEEE-1588 Block. The user must configure software configuration to match the hardware configuration, as there are multiple options available. The configuration of the `vtss_phy_ts_init_conf_t` structure must match the hardware configuration on the board.

This structure has the following parameters that must match the hardware configuration:

- `clk_freq`
- `clk_src`

The `clk_freq` is the IEEE-1588 Input Clk Frequency to the chip. This is normally the value of `VTSS_PHY_TS_CLOCK_FREQ_250M` or `VTSS_PHY_TS_CLOCK_FREQ_125M`, with the higher frequency normally indicating higher accuracy timestamping is possible.

The `clk_src` is the IEEE-1588 Input Clk Source to the chip. This is normally a value of `VTSS_PHY_TS_CLOCK_SRC_INTERNAL` or `VTSS_PHY_TS_CLOCK_SRC_EXTERNAL`.

The 1G PHYs have an option for selecting an internal clock `VTSS_PHY_TS_CLOCK_SRC_INTERNAL`, which runs at a clock frequency of 250 MHz; therefore the `clk_freq` is set to `VTSS_PHY_TS_CLOCK_FREQ_250M`. The internal source is selected for the Sample Application for ease of configuration as this removes the requirement for an external clock source.

When selecting `VTSS_PHY_TS_CLOCK_SRC_EXTERNAL`, the evaluation board has SMA connectors for an External Clk Src input source and common values are 250 MHz (`VTSS_PHY_TS_CLOCK_FREQ_250M`) and 125 MHz (`VTSS_PHY_TS_CLOCK_FREQ_125M`).

The 10G PHYs do not have an option for an internal clock, therefore it has to be supplied by an external source. Both 1G and 10G PHYs support external 1588 input clock sources, and all evaluation boards support an external IEEE-1588 input clock source.

An example of this configuration is shown below:

```
conf.clk_freq = VTSS_PHY_TS_CLOCK_FREQ_250M; /* reference clock frequency */
conf.clk_src = VTSS_PHY_TS_CLOCK_SRC_INTERNAL; /* Clock Source */
conf.rx_ts_pos = VTSS_PHY_TS_RX_TIMESTAMP_POS_IN_PTP; /* Rx ts in PTP reserved bytes */
conf.rx_ts_len = VTSS_PHY_TS_RX_TIMESTAMP_LEN_30BIT; /* Rx ts update nsec part */
conf.tx_fifo_mode = VTSS_PHY_TS_FIFO_MODE_NORMAL; /* Ts pushed out on the SPI i/f */
conf.tx_ts_len = VTSS_PHY_TS_FIFO_TIMESTAMP_LEN_10BYTE; /* 10byte Full Tx timestamp */
conf.tc_op_mode = VTSS_PHY_TS_TC_OP_MODE_B; /* TC operating mode is Mode B */
conf.auto_clear_ls = FALSE; /* Load and Save of LTC are auto cleared */
conf.macsec_ena = FALSE; /* MACsec is enabled or disabled */
conf.chk_ing_modified = FALSE;
```

This structure is populated and passed into the `vtss_phy_ts_init` API function.

```
vtss_phy_ts_init(inst, port_no, &conf)
vtss_phy_ts_mode_set (inst, port_no, enable);
```

#### 6.3.1.2 Initializing the Ingress and Egress Timestamp Engines

The Sample Application initializes the ingress and egress timestamp engines (that is, selecting the engine, setting the encapsulation type, and performing `start_flow`, `end_flow`, and `flow match` type).

The options for `engine_id` are:

- `VTSS_PHY_TS_PTP_ENGINE_ID_0`
- `VTSS_PHY_TS_PTP_ENGINE_ID_1`
- `VTSS_PHY_TS_PTP_ENGINE_ID_2A`
- `VTSS_PHY_TS_PTP_ENGINE_ID_2B`

Common encapsulation types are:

- `VTSS_PHY_TS_ENCAP_ETH_PTP`
- `VTSS_PHY_TS_ENCAP_ETH_PTP`

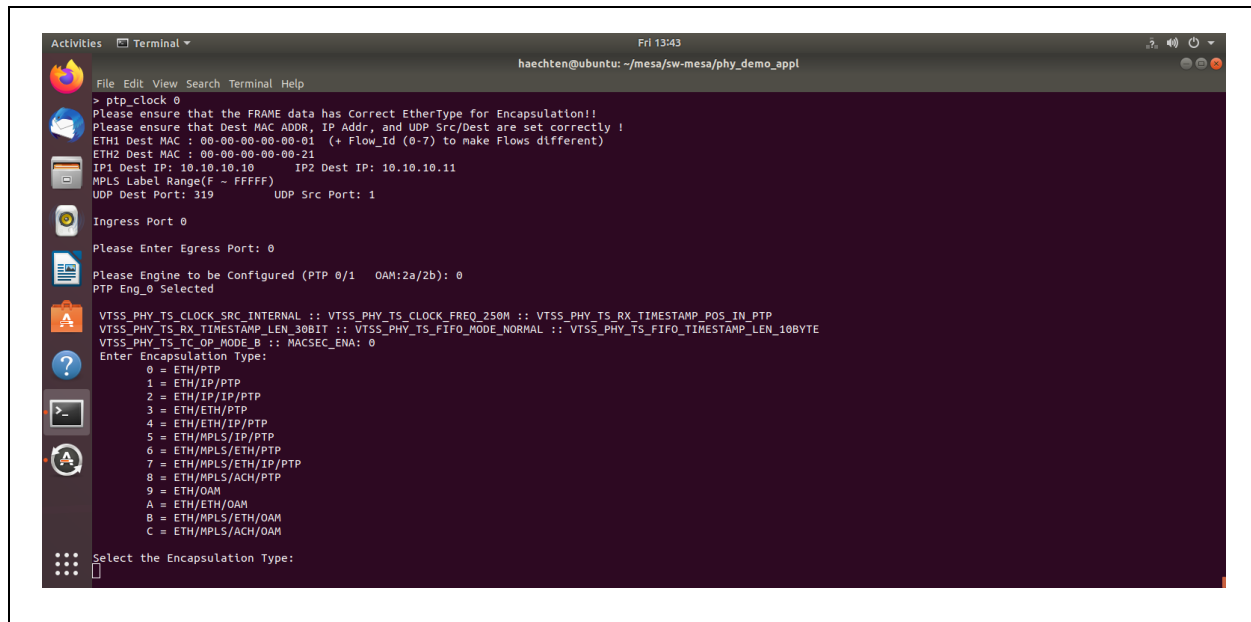
The `start_flow` and `end_flow` have a range of 0 to 7, which indicates the maximum size allocated for flows.

```
rc = vtss_phy_ts_ingress_engine_init(inst, ing_port_no,
 eng_id,
 VTSS_PHY_TS_ENCAP_ETH_IP_PTP,
 start_flow,
 end_flow,
 VTSS_PHY_TS_ENG_FLOW_MATCH_STRICT);

rc = vtss_phy_ts_egress_engine_init(inst, egr_port_no,
 eng_id,
 VTSS_PHY_TS_ENCAP_ETH_IP_PTP,
 start_flow,
 end_flow,
 VTSS_PHY_TS_ENG_FLOW_MATCH_STRICT);
```

An example of the Sample Application output is shown in showing operator selection of port, encapsulation, and so on.

**FIGURE 6: SAMPLE APPLICATION 1588 PROMPTS**





### 6.3.1.3 Initializing the Ingress and Egress Flows

Once the ingress and egress engines have been initialized, the Sample Application will then configure the Ingress and Egress 1588 Analyzer. The Sample Application has code examples for each of the encapsulation types. Each engine can be configured for only one encapsulation type. For every engine/encapsulation type, up to eight flows can be configured to program the analyzer within the PHY to match a particular traffic pattern or type of packet to be timestamped.

The 1588 block is shared across two ports, so there is channel mapping (that is, mapping the functionality for each flow for Ch0, Ch1, or both).

For example, for common encapsulation types, the following Sample Application functions are used to configure the flows within the PHY:

```
VTSS_PHY_TS_ENCAP_ETH_PTP - vtss_1588_sample_flows_ep
```

```
VTSS_PHY_TS_ENCAP_ETH_PTP - vtss_1588_sample_flows_eip
```

The Sample Application uses the following API functions to configure the analyzer flows. The flows may be symmetrical (that is, ingress and egress flows match the same traffic pattern) or asymmetrical (that is, ingress and egress traffic patterns do not match the same flows), using the following APIs:

- vtss\_phy\_ts\_ingress\_engine\_conf\_get
- vtss\_phy\_ts\_ingress\_engine\_conf\_set
- vtss\_phy\_ts\_egress\_engine\_conf\_get
- vtss\_phy\_ts\_egress\_engine\_conf\_set

If the flow configuration is symmetrical, then the ingress and egress flow configuration can be set to be the same.

### 6.3.1.4 Initializing the Ingress and Egress Actions

Once the ingress and egress flow configurations have been updated, the Sample Application will then configure the Ingress and Egress 1588 Actions. The actions indicate what is performed with a packet that matches all the analyzer settings. The Sample Application has code examples for available actions. The actions taken are determined based on `clk_mode` and `delaym_type`.

The 1588 block is shared across two ports, so there is channel mapping for the actions to be taken, mapping the functionality for each flow for Ch0, Ch1, or both.

The Sample Application has code examples for selecting the actions based on the following:

- `clk_mode`
  - VTSS\_PHY\_TS\_PTP\_CLOCK\_MODE\_BC1STEP
  - VTSS\_PHY\_TS\_PTP\_CLOCK\_MODE\_BC2STEP
  - VTSS\_PHY\_TS\_PTP\_CLOCK\_MODE\_TC1STEP
  - VTSS\_PHY\_TS\_PTP\_CLOCK\_MODE\_TC2STEP
  - VTSS\_PHY\_TS\_PTP\_DELAY\_COMP\_ENGINE
- `delaym_type`
  - VTSS\_PHY\_TS\_PTP\_DELAYM\_P2P
  - VTSS\_PHY\_TS\_PTP\_DELAYM\_E2E

The Sample Application uses the following API functions to configure the actions. The actions may be symmetrical (that is, ingress and egress actions match the same traffic pattern), or asymmetrical (ingress and egress traffic do not match the same packets), using the following APIs:

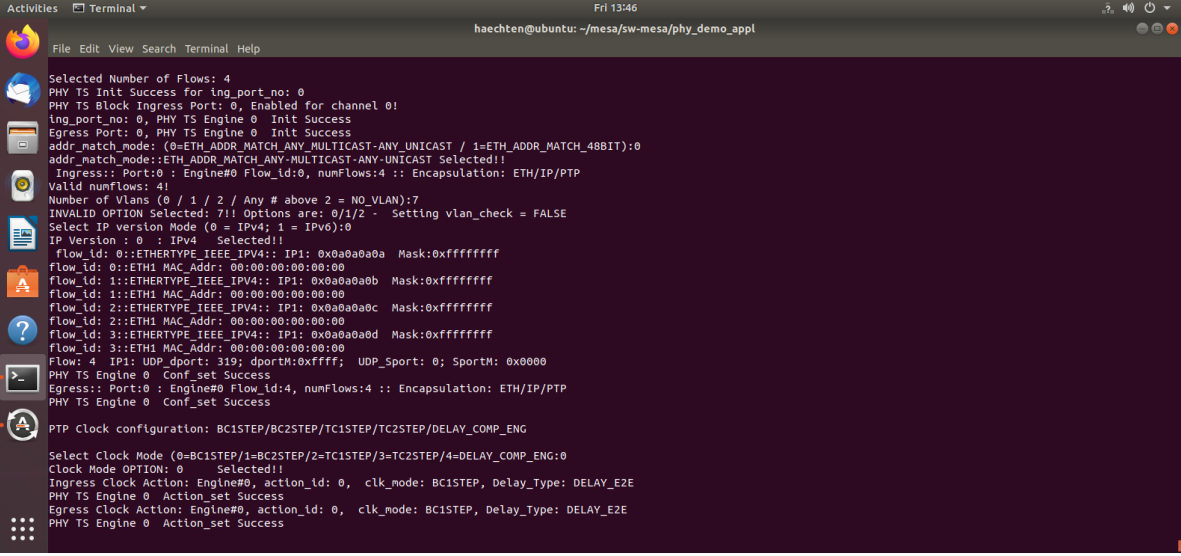
- vtss\_phy\_ts\_ingress\_engine\_action\_get
- vtss\_phy\_ts\_ingress\_engine\_action\_set
- vtss\_phy\_ts\_egress\_engine\_action\_get
- vtss\_phy\_ts\_egress\_engine\_action\_set

If the action configuration is symmetrical, then the ingress and egress actions can be set to be the same.

An example of the Sample Application output is shown in [Figure 7](#), which shows the number of flows being configured. The flow configuration and selection of MAC address, IP1 address, and more are automatically incremented in the Sample Application to ensure that they are unique.

Sample clock actions are also shown in the figure, with the `clk_mode` and `delaym_type` configured. This helps the user verify the configuration and compare this configuration to the traffic the user is attempting to match.

**FIGURE 7: EXAMPLE SAMPLE APPLICATION OUTPUT**



```
Selected Number of Flows: 4
PHY TS Init Success for lng_port_no: 0
PHY TS Block Ingress Port: 0, Enabled for channel 0!
lng_port_no: 0, PHY TS Engine 0 Init Success
Egress Port: 0, PHY TS Engine 0 Init Success
addr_match_mode: (0=ETH_ADDR_MATCH_ANY_MULTICAST-ANY_UNICAST / 1=ETH_ADDR_MATCH_48BIT):0
addr_match_mode::ETH_ADDR_MATCH_ANY_MULTICAST-ANY_UNICAST Selected!!
Ingress:: Port:0 : Engine#0 Flow_id:0, numFlows:4 :: Encapsulation: ETH/IP/PTP
Valid numFlows: 4!
Number of VLANs (0 / 1 / 2 / Any # above 2 = NO_VLAN):7
INVALID OPTION Selected: 7! Options are: 0/1/2 - Setting vlan_check = FALSE
Select IP version Mode (0 = IPv4; 1 = IPv6):0
IP Version : 0 : IPv4 Selected!!
flow_id: 0::ETHERTYPE_IP IPv4:: IP1: 0x0a0a0a0a Mask:0xffffffff
flow_id: 0::ETH1 MAC_Addr: 00:00:00:00:00:00
flow_id: 1::ETHERTYPE_IP IPv4:: IP1: 0x0a0a0a0b Mask:0xffffffff
flow_id: 1::ETH1 MAC_Addr: 00:00:00:00:00:00
flow_id: 2::ETHERTYPE_IP IPv4:: IP1: 0x0a0a0a0c Mask:0xffffffff
flow_id: 2::ETH1 MAC_Addr: 00:00:00:00:00:00
flow_id: 3::ETHERTYPE_IP IPv4:: IP1: 0x0a0a0a0d Mask:0xffffffff
flow_id: 3::ETH1 MAC_Addr: 00:00:00:00:00:00
Flow: 4 IP1: UDP_dport: 319; dportM:0xffff; UDP_Sport: 0; SportM: 0x0000
PHY TS Engine 0 Conf_set Success
Egress:: Port:0 : Engine#0 Flow_id:4, numFlows:4 :: Encapsulation: ETH/IP/PTP
PHY TS Engine 0 Conf_set Success
PTP Clock configuration: BC1STEP/BC2STEP/TC1STEP/TC2STEP/DELAY_COMP_ENG
Select Clock Mode (0=BC1STEP/1=BC2STEP/2=TC1STEP/3=TC2STEP/4=DELAY_COMP_ENG):0
clock Mode OPTION: 0 Selected!!
Ingress Clock Action: Engine#0, action_id: 0, clk_mode: BC1STEP, Delay_Type: DELAY_EZE
PHY TS Engine 0 Action_set Success
Egress Clock Action: Engine#0, action_id: 0, clk_mode: BC1STEP, Delay_Type: DELAY_EZE
PHY TS Engine 0 Action_set Success
```

### 6.3.1.5 Initializing the Signature Used in Timestamp FIFO

The Sample Application uses the following API functions to configure the timestamp signature using the following APIs:

- `vtss_phy_ts_fifo_sig_get`
- `vtss_phy_ts_fifo_sig_set`

### 6.3.1.6 Initializing/Setting the Local Time Counter

The Sample Application provides a sequence to use the following API functions to configure and set the Local Time Counter (LTC):

- `vtss_phy_ts_ptptime_arm`
- `vtss_phy_ts_ptptime_set`
- L/S Pin Pulse occurs in the Hardware
- `vtss_phy_ts_ptptime_set_done`

### 6.3.1.7 Saving and Retrieving the Local Time Counter

The Sample Application provides a sequence to use the following API functions to configure to get the LTC value and read these register values:

- `vtss_phy_ts_ptptime_arm`
- L/S Pin Pulse occurs in the Hardware
- `vtss_phy_ts_ptptime_get`

### 6.3.1.8 Enabling and Polling for IEEE-1588 Events

The Sample Application uses the following API functions to configure the 1588 Events (Interrupts) and polling of events:

- `vtss_phy_ts_event_enable`
- `vtss_phy_ts_event_poll`

### 6.3.1.9 Installing the TS FIFO Read Callback

The Sample Application uses the `vtss_phy_ts_fifo_read_install` API function to install the Read Callback function, which is called when reading the timestamp FIFO value.

#### 6.3.1.10 Emptying TS FIFO After TS Event

The Sample application uses the `vtss_phy_ts_fifo_empty` API function to empty the TS FIFO when called and to call the registered TS FIFO read callback.

#### 6.3.2 USING THE SAMPLE APPLICATION TO CONFIGURE THE PHY EVALUATION BOARD FOR IEEE-1588 PTP BOUNDARY CLOCK ETH-PTP TRAFFIC

Refer to [Section 6.3, "Using the Sample Application to Configure the PHY Evaluation Board for PTP Traffic"](#) and configure the PHY evaluation board such that Port 0 is configured for ingress and Port 1 is configured for egress. Both are configured for Boundary Clock and 1588 Operation. Encapsulation for this case is ETH-PTP. The PHY applies the timestamp to the ingress data (4-byte field) and, upon egress of the PHY Port 1, it timestamps the data using the 10-byte timestamp.

To test this configuration, ingress traffic must be an `IEEE_PTP_1588` traffic (EtherType = 0x88F7) to be timestamped. Refer to [Section 5.1, "Configuring the Traffic Generator for PTP Traffic"](#) and [Section 5.1.1, "ETH-PTP Encapsulation \(ETH-PTP\)"](#).

Traffic Gen → Port 0 (Ingress TS) → SGMII → Port 1 (Egress TS) → Monitor

Verify or monitor the traffic coming out of Port 1.

#### 6.3.3 USING THE SAMPLE APPLICATION TO CONFIGURE THE PHY EVALUATION BOARD FOR IEEE-1588 PTP BOUNDARY CLOCK ETH-IP-UDP-PTP TRAFFIC

Refer to [Section 6.3, "Using the Sample Application to Configure the PHY Evaluation Board for PTP Traffic"](#) and configure the PHY evaluation board such that Port 0 is configured symmetrically for ingress/egress and Port 1 is configured symmetrically for ingress/egress. Both are configured for Boundary Clock and 1588 Operation. Encapsulation for this case is ETH-IP-UDP-PTP. The PHY applies the timestamp to the ingress data (4-byte field) and, upon egress of the PHY Port 1, it timestamps the data using the 10-byte timestamp.

To test this configuration, ingress traffic must be an `IEEE_IPV4` traffic (EtherType = 0x0800) to be timestamped. The 4-byte `IP_SRC_ADDR` for the PTP packet must be 55443322. Refer to [Section 5.1, "Configuring the Traffic Generator for PTP Traffic"](#) and [Section 5.1.2, "ETH-IP-PTP Encapsulation \(ETH-IPv4-PTP\)"](#).

Traffic Gen → Port 0 (Ingress TS) → SGMII → Port 1 (Egress TS) → Monitor

Verify or monitor the traffic coming out of Port 1.

#### 6.3.4 USING THE SAMPLE APPLICATION TO CONFIGURE THE PHY EVALUATION BOARD FOR IEEE-1588 PTP TRANSPARENT CLOCK ETH-PTP TRAFFIC

Refer to [Section 6.3, "Using the Sample Application to Configure the PHY Evaluation Board for PTP Traffic"](#) and configure the PHY evaluation board such that Port 0 is configured for ingress and Port 1 is configured for egress. Both are configured for Transparent Clock and 1588 Operation. Encapsulation for this case is ETH-PTP. The PHY applies the timestamp to the ingress data (4-byte field) and, upon egress of the PHY Port 1, it timestamps the data by updating the correction field.

To test this configuration, ingress traffic must be of an `IEEE_PTP_1588` traffic (EtherType = 0x88F7) to be timestamped. Refer to [Section 5.1, "Configuring the Traffic Generator for PTP Traffic"](#) and [Section 5.1.1, "ETH-PTP Encapsulation \(ETH-PTP\)"](#).

Traffic Gen → Port 0 (Ingress TS) → SGMII → Port 1 (Egress TS) → Monitor

Verify or monitor the traffic coming out of Port 1.

## 6.3.5 USING THE SAMPLE APPLICATION TO CONFIGURE THE PHY EVALUATION BOARD FOR IEEE-1588 PTP TRANSPARENT CLOCK ETH-IP-UDP-PTP TRAFFIC

Refer to [Section 6.3, "Using the Sample Application to Configure the PHY Evaluation Board for PTP Traffic"](#) and configure the PHY evaluation board such that Port 0 is configured for ingress and Port 1 is configured for egress. Both are configured for Transparent Clock and 1588 Operation. Encapsulation for this case is ETH-IP-UDP-PTP. The PHY applies the timestamp to the ingress data (4-byte field) and, upon egress of the PHY Port 1, it timestamps the data by updating the correction field.

To test this configuration, ingress traffic must be an `IEEE_IPV4` traffic (EtherType = 0x0800) to be timestamped. The 4-byte `IP_SRC_ADDR` for the PTP packet must be 55443322. Refer to [Section 5.1, "Configuring the Traffic Generator for PTP Traffic"](#) and [Section 5.1.2, "ETH-IP-PTP Encapsulation \(ETH-IPv4-PTP\)"](#).

Traffic Gen → Port 0 (Ingress TS) → SGMII → Port 1 (Egress TS) → Monitor

Verify or monitor the traffic coming out of Port 1.

## 6.4 Using the Sample Application to Configure the PHY Evaluation Board for MACsec Traffic

The MACsec Sample Application uses the MESA API code to configure the PHY for MACsec operation. The Sample Application can be configured to use 128-bit or 256-bit encryption keys and makes use of static keys embedded within the software. There is currently no means to update a key within the Sample Application, like KeySec, which means that at the time of packet number rollover, traffic will stop. This software can be easily modified with more keys added, if that is desired.

The MACsec Sample Application is located in the `vtss_appl_macsec_demo.c` file, and the user should become familiar with it before implementation in their application.

The MCHP MESA API software provides a software interface to control the PHY features/functionality for MACsec operation. The MESA API programs the MACsec hardware functionality within the PHY and provides the user an interface that allows ease of control and operation for demonstration of MACsec features.

In a system environment, the application may be running on a local or on a remote platform CPU and may include Inside Secure's MAC Security Key Agreement Protocol (MKA). A Remote Authentication Dial-In User Service (RADIUS) server running an Extensible Authentication Protocol (EAP) can also be used for authentication. This document describes the MESA API software package, which supports multiple MCHP VSC PHYs, however this document describes the setup and evaluation of MACsec using the customer evaluation board VSC8584EV, which is a 1Gbit PHY that is capable of MACsec operation.

The VSC8584 contains a SerDes interface so the `VTSS_FEATURE_SERDES_MACRO_SETTINGS` compilation flag must be defined. This PHY is capable of 1588 PHY timestamping and MACsec encryption/decryption, so the API compilation flags should be set to indicate `VTSS_OPT_PHY_MACSEC`, `VTSS_OPT_PHY_TIMESTAMP`, or both if timestamping is also desired. These flags may be set to include these features when compiling the MESA API software.

The Sample Application evaluation board definition `EVAL_BOARD_MACSEC_CAPABLE` is also set to indicate that the VSC8584EV evaluation board hardware supports MACsec encryption/decryption. The definition `EVAL_BOARD_1588_CAPABLE` may also be set to indicate that the VSC8584EV evaluation board hardware supports IEEE-1588 timestamping, if the user desires to include IEEE-1588 functionality in the build.

### 6.4.1 LAUNCHING THE MACSEC SAMPLE APPLICATION

Prior to configuring the PHY evaluation board for MACsec traffic, ensure that the basic configuration of the PHY has been performed on *all* ports of the PHY. If MACsec and PTP are to be used together, the order of initialization is important and the MACsec block must be configured first. Upon completion of MACsec configuration, IEEE-1588 block configuration is then performed.

Order of initialization when MACsec/PTP are used together:

1. Basic PHY initialization of *all* ports
2. Initialization of MACsec block
3. Initialization of IEEE-1588 block

See [Section 3.0, "Sample Application Overview"](#) and follow the steps to complete basic initialization of the PHY. Once *all* the ports of the VSC 1G PHY initialization is complete, a menu with various options appears. To access the sub-menu for MACsec, the PHY must support MACsec encryption, which means it must be one of the following PHYs:

- 1G Viper PHYs: VSC8584, VSC8582, VSC8564, VSC8562

If PTP is to also be configured, the user would exit the MACsec menu, and then access the sub-menu for IEEE-1588 timestamp block initialization. The following 1G PHYs include IEEE-1588 and MACsec:

- 1G Viper PHYs: VSC8584 or VSC8582

Refer to [Section 5.2, "Configuring the Evaluation Board and Traffic Generator for MACsec Traffic"](#) and configure the Traffic Generator accordingly. The PHY evaluation board should be configured following [Section 4.1, "Configuring the PHY Evaluation Board MAC Interface SGMII to Loopback Traffic Port-To-Port on the Same Evaluation Board"](#), which will loopback traffic port-to-port as follows:

Traffic Gen → Port 0 → SGMII → Port 1 (Encrypt) → CAT5 Loop → Port 2 (Decrypt) → SGMII → Port 3 → Monitor

Or:

Traffic Gen → Port 3 → SGMII → Port 2 (Encrypt) → CAT5 Loop → Port 1 (Decrypt) → SGMII → Port 0 → Monitor

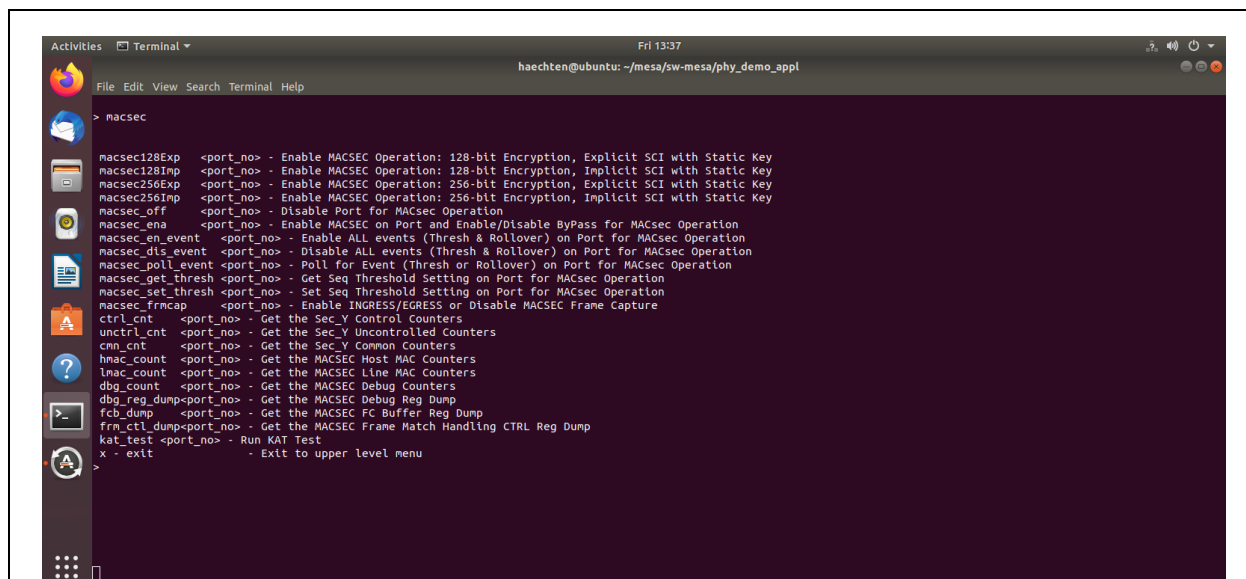
Prior to enabling MACsec or 1588 operation, ensure that traffic can pass in both directions, from Port 0 to Port 1 and from Port 1 to Port 0. If Ports 2 and 3 are to be used, also confirm basic traffic flow between these ports as well.

When enabling MACsec, the Cat 5 cable connecting Ports 1 and 2 should be added. Ensure that traffic can pass in both directions, from Port 0 to Port 3.

Once traffic flow from the ingress at Port 0 to the egress at Port 3, and traffic flow from the ingress at Port 3 to the egress at Port 0 can be verified, configure the customer evaluation board for the IEEE-1588 operation if so desired.

The MESA 1G PHY Sample Application code has an IEEE-1588 CLI menu, which will appear when the IEEE-1588 option is selected. An example of this menu is shown in [Figure 8](#).

**FIGURE 8: CLI MENU**



## 6.4.1.1 PHY API Configuration Sequence for MACsec Block

The Sample Application code for the MACsec code is located in the `phy_demo_appl/appl/vtss_appl_macsec_demo.c` file, and the menu can be found in the `vtss_appl_macsec_demo_menu` function.

**Note:** When configuring for MACsec operation, the ports that are being configured for MACsec must have *link-up*.

A single option or multiple options may be performed as necessary to test the desired functionalities. These options are API functionalities that may result in multiple register read/write sequences, which are also used by a customer application.

The `vtss_appl_macsec_demo_menu` has the following menu options to perform the MACsec initialization sequence using the MESA API:

- `macsec128Exp` - Enable MACsec operation: 128-bit encryption, explicit SCI with static key
- `macsec128Imp` - Enable MACsec operation: 128-bit encryption, implicit SCI with static key
- `macsec256Exp` - Enable MACsec operation: 256-bit encryption, explicit SCI with static key
- `macsec256Imp` - Enable MACsec operation: 256-bit encryption, implicit SCI with static key

Each of these selections calls the same Sample Application function, `vtss_single_secy_sample_system_create()`, with flags set for the type of encryption, that is, 128 bit or 256 bit.

This starts an automated sequence of the initialization of the MACsec block for the port selected. The following sequence is executed:

- [PHY API Configuration Sequence for Creating Basic SecY](#)
- [Initialization of the MACsec Tx Channel](#)
  - Initializing the Tx Secure Channel (SC)
  - Initializing the Tx Encryption Key (Sak)
  - Initializing the Tx Secure Association (SA)
  - Activating the Tx Secure Association (SA)
- [Initialization of the MACsec Rx Channel](#)
  - Initializing the Rx Secure Channel (SC)
  - Initializing the Rx Encryption Key (Sak)
  - Initializing the Rx Secure Association (SA)
  - Activating the Rx Secure Association (SA)

Additional configuration may be needed for a particular setup and must be selected from the CLI menu to accomplish the following:

- Manually Enable/Disable MACsec on a Port
- Manually Enable/Disable MACsec Events (Threshold/Rollover)
- Enabling and Polling for MACsec Events
- Displaying MACsec Counters

## 6.4.1.2 PHY API Configuration Sequence for Creating Basic SecY

The Sample Application executes the `vtss_single_secy_sample_system_create()` function to initialize the MACsec block. The port being configured will be connected to a link partner port, and the link partner port must also have a configuration identical to the local port.

The initial API function, `vtss_macsec_init_set()`, is called to set up the basic configuration and to enable or disable bypass. This is followed by the default configuration for the actions to be taken when a frame does not match the MACsec parameters for encryption/decryption.

The SecY is then created or added to the configuration that is followed by the configuration of the pattern matching for egress/ingress for the controlled and uncontrolled port. Once all the SecY configuration parameters are initialized, the SecY controlled port settings are pushed down to the PHY hardware.

Initialize the MACsec block:

```
vtss_macsec_init_get()
vtss_macsec_init_set()
```

Get/Set the default rules for all non-matched traffic (Bypass/Drop):

```
vtss_macsec_default_action_get()
vtss_macsec_default_action_set()
```

Add the SecY configuration for this port:

```
vtss_macsec_secy_conf_add()
```

Associate the pattern for the controlled *egress* port:

```
vtss_macsec_pattern_set(VTSS_MACSEC_MATCH_ACTION_CONTROLLED_PORT)
```

Associate the pattern for the controlled *ingress* port:

```
vtss_macsec_pattern_set(VTSS_MACSEC_MATCH_ACTION_CONTROLLED_PORT)
```

Associate the pattern for the uncontrolled *egress* port:

```
vtss_macsec_pattern_set(VTSS_MACSEC_MATCH_ACTION_UNCONTROLLED_PORT)
```

Associate the Pattern for the uncontrolled *ingress* port:

```
vtss_macsec_pattern_set(VTSS_MACSEC_MATCH_ACTION_UNCONTROLLED_PORT)
```

Modify the SecY (Enable/Disable):

```
vtss_macsec_secy_controlled_get()
vtss_macsec_secy_controlled_set()
```

#### 6.4.1.3 Initialization of the MACsec Tx Channel

The Sample Application initializes the egress (Tx), creating the secure channel, creating the secure association, and installing the encryption key, and activating the secure association. The sequence and MESA API function calls are shown in the following:

- Add a new Tx secure channel

```
vtss_macsec_tx_sc_set
```

- Update the Hash key and Sak before being installed using `_set`. The Sample Application code is:

```
sak_update_hash_key
```

- Install the Hash key/Sak in the egress hardware registers

```
vtss_macsec_tx_sa_set
```

- Activate the Tx SA in the egress hardware

```
vtss_macsec_tx_sa_activate
```

#### 6.4.1.4 Initialization of the MACsec Rx Channel

The Sample Application adds or initializes the ingress (Rx), creating the secure channel, creating the secure association, and installing the encryption key, and activating the secure association. The Sample Application then creates a second secure association, which is used when the current secure association expires or rolls over. The sequence and MESA API function calls are shown in the following:

- Add a new Rx secure channel

```
vtss_macsec_rx_sc_add
```

Add two Rx SAs. The first Rx SA is the active SA, and the second Rx SA is the alternate used when a key rollover occurs.

Install the Hash key/Sak in the ingress hardware registers

- Update the Hash key and Sak before being installed using `_set`. The Sample Application code is:

```
sak_update_hash_key
```

- Add the first Rx SA and then activate the SA and the second Rx SA (alternate) when a key rollover occurs

```
vtss_macsec_rx_sc_add
```

```
vtss_macsec_rx_sa_activate
```

Install the Hash key/Sak in the ingress hardware registers



- Update the Hash key and Sak before being installed using `_set`. The Sample Application code is:

```
sak_update_hash_key
```

- Add the first Rx SA and then activate the SA and the second Rx SA (alternate) when a key rollover occurs

```
vtss_macsec_rx_sa_set
```

```
vtss_macsec_rx_sa_activate
```

## 6.4.2 TYPICAL CONSOLE OUTPUT

The following output was captured from the initialization sequence of a VSC8514 evaluation board. The user input is shown in bold. The Sample Application was executed using the following:

Determine whether to configure for 128-bit or 256-bit encryption. Two ports must be configured, Ports 1 and 2.

Determine whether to configure for implicit SCI or explicit SCI. (See [Note 1.](#))

Determine confidentiality (Invoke Encryption/Decryption mode). (See [Note 2.](#))

**Note 1:** Implicit SCI means that the Destination Port MAC addresses inside the packet and `Port_No` are used by the peer as the SCI for matching.

For implicit configuration, Destination Port MAC address = 00 00 00 00 00 BB in packet data. This demo configures for SecTag = 8 bytes in size for this configuration.

Explicit SCI means that the MAC Address and `Port_No` are defined explicitly in the SecTag, which is used by the peer as the SCI for matching. For the purpose of this testing, a `MAC_ADDR` value of 00 00 00 00 00 AA and `Port_No` 0001 is used.

For explicit configuration, Destination Port MAC address = 00 00 00 00 00 BB in packet data. This demo configures for SecTag = 16 bytes in size for this configuration.

- 2:** Confidentiality = 1 indicates to encrypt/decrypt the data. Add the SecTag and ICV.  
Confidentiality = 0 indicates to *not* encrypt/decrypt the data, but to add the SecTag and ICV.

See the `macsec_port_conf.always_include_sci` setting in demo for configuring the SecTag size.

### 6.4.2.1 Example Output for 256-Bit Encryption, Implicit-SCI

At the prompt, enter:

```
>macsec256Imp 1
```

When prompted for Confidentiality, enter **1**. The display will output the implicit SCI, key information, and more for the user to see the parameters selected.

When the main menu is redisplayed, PHY Port 1 will be configured for MACsec operation.

Traffic can flow from Port 0 to Port 3, and the monitor will indicate that the traffic has been encrypted.

Traffic Gen → Port 0 → SGMII → Port 1 (Encrypt) → CAT5 Loop → Port 2 (Decrypt) → SGMII → Port 3 → Monitor

The PHY encrypts on Egress and decrypts on ingress. Therefore, the data will ingress with EtherType = 0800 through Port 0 and will be looped back on the MAC interface to the Port 1 MAC interface. The data will then be encrypted by the PHY on egress. The PHY will add bytes for the SecTag and ICV Values. For this test, the data will show an increase in packet size of 24 bytes (8 bytes for SecTag (Implicit-SCI contained in SecTag) and 16 bytes for ICV) and EtherType = 88e5.

The console output shows the following:

```
EGRESS CTRL Pattern Set Complete - Pattern Match Set to: VTSS_MACSEC_MATCH_ETYPE = 0800
INGRESS CTRL Pattern Set Complete - Pattern Match Set to: VTSS_MACSEC_MATCH_ETYPE = 88e5
```

This configuration is set up to *only* match the EtherType.

- On egress, if the pattern matches an EtherType = 0800 (IPv4), the packet will be encrypted.
- On ingress, if the pattern matches an EtherType = 88e5 (802.1ae), the packet will be decrypted.



Using the monitor, examine the Ethernet packet. Check the following:

- Ingress EtherType = 0800 (Port 0)
- Egress EtherType = 88e5 (Port 1)
- Ingress Packet size = "x" (example, 128 bytes)
- Egress Packet size = "x+24" (example, 152 bytes)

#### 6.4.2.2 Example for 256-Bit Encryption, Implicit-SCI, Encryption/Decryption on Port 1 and Port 2

Enable Port 2 so the data will be decrypted. At the prompt, enter:

```
>macsec256Imp 2
```

When prompted for Confidentiality, enter 1. The display will output the key information for the user to see the static key being selected.

When the main menu is redisplayed, PHY Port 2 will be configured for MACsec operation.

The following setup is configured and is ready for traffic. Traffic can flow from Port 0 to Port 3 or from Port 3 to Port 0.

Traffic Gen → Port 0 → SGMII → Port 1 (Encrypt) → CAT5 Loop → Port 2 (Decrypt) → SGMII → Port 3 → Monitor  
 Monitor ← Port 0 ← SGMII ← (Decrypt) Port 1 ← CAT5 Loop ← (Encrypt) Port 2 ← SGMII ← Port 3 ← Traffic Gen

The PHY encrypts on egress and decrypts on ingress. Therefore, the data will Ingress with EtherType = 0800 through Port 0 and will be looped back on the MAC interface to the Port 1 MAC interface. The data will then be encrypted by the PHY on egress. The PHY will add bytes for the SecTag and ICV values. For this test, the data will show an increase in packet size of 24 bytes (8 bytes for SecTag (Explicit-SCI contained in SecTag) and 16 bytes for ICV) and EtherType = 88e5.

This data is then looped back into Port 2 of the PHY. The data will then be decrypted by the PHY on ingress of Port 2. The PHY will remove the bytes added for the SecTag and ICV. The data will ingress to the MAC interface with EtherType = 0800 and will be looped back to Port 3 MAC interface and egress out Port 3 as the same data that was inputted into Port 0.

The console output shows the following:

```
EGRESS CTRL Pattern Set Complete - Pattern Match Set to: VTSS_MACSEC_MATCH_ETYPE = 0800
INGRESS CTRL Pattern Set Complete - Pattern Match Set to: VTSS_MACSEC_MATCH_ETYPE = 88e5
```

This configuration is set up to *only* match the EtherType.

- On egress, if the pattern matches an EtherType = 0800 (IPv4), the packet will be encrypted.
- On ingress, if the pattern matches an EtherType = 88e5 (802.1ae), the packet will be decrypted.

Generate IPv4 Traffic (EtherType = 0800) and transmit into Port 0 of the PHY. Verify that the Port 3 data output matches that input into Port 0.

Generate IPv4 Traffic (EtherType = 0800) and transmit into Port 3 of the PHY. Verify that the Port 0 data output matches that input into Port 0.

Using the monitor, examine the Ethernet packet. Check the following:

- Ingress EtherType = 0800 (Port 0)
- Egress EtherType = 0800 (Port 3)
- Ingress Packet size = "x" (example, 128 bytes)
- Egress Packet size (Port 3) = "x" (example, 128 bytes)

To disable the MACsec operation for the ports, enter:

```
>macsec_off 1
>macsec_off 2
```

## 6.4.2.3 Example Output for 256-Bit Encryption, Explicit-SCI

At the prompt, enter:

```
>macsec256Exp 1
```

When prompted for Confidentiality, enter **1**. The display will output the explicit SCI, key information, and more for the user to see the parameters selected.

When the main menu is redisplayed, PHY Port 1 will be configured for MACsec operation.

Traffic can flow from Port 0 to Port 3, and the monitor will indicate that the traffic has been encrypted.

Traffic Gen → Port 0 → SGMII → Port 1 (Encrypt) → CAT5 Loop → Port 2 (Decrypt) → SGMII → Port 3 → Monitor

The PHY encrypts on egress and decrypts on ingress. Therefore, the data will ingress with EtherType = 0800 through Port 0 and will be looped back on the MAC interface to the Port 1 MAC interface. The data will then be encrypted by the PHY on egress. The PHY will add bytes for the SecTag and ICV values. For this test, the data will show an increase in packet size of 32 bytes (16 bytes for SecTag (Explicit-SCI) and 16 bytes for ICV) and EtherType = 88e5.

The console output shows the following:

```
EGRESS CTRL Pattern Set Complete - Pattern Match Set to: VTSS_MACSEC_MATCH_ETYPE = 0800
INGRESS CTRL Pattern Set Complete - Pattern Match Set to: VTSS_MACSEC_MATCH_ETYPE = 88e5
```

This configuration is set up to *only* match the EtherType.

- On egress, if the pattern matches an EtherType = 0800 (IPv4), the packet will be encrypted.
- On ingress, if the pattern matches an EtherType = 88e5 (802.1ae), the packet will be decrypted.

Using the monitor, examine the Ethernet packet. Check the following:

- Ingress EtherType = 0800 (Port 0)
- Egress EtherType = 88e5 (Port 1)
- Ingress Packet size = "x" (example, 128 bytes)
- Egress Packet size = "x+32" (example, 160 bytes)

## 6.4.2.4 Example for 256-Bit Encryption, Explicit-SCI, Encryption/Decryption on Port 1 and Port 2

Enable Port 2 so the data will be decrypted. At the prompt, enter:

```
>macsec256Exp 2
```

When prompted for Confidentiality, enter **1**. The display will output the key information for the user to see the static key being selected.

When the main menu is redisplayed, PHY Port 2 will be configured for MACsec operation.

The following setup is configured and is ready for traffic. Traffic can flow from Port 0 to Port 3 or from Port 3 to Port 0.

Traffic Gen → Port 0 → SGMII → Port 1 (Encrypt) → CAT5 Loop → Port 2 (Decrypt) → SGMII → Port 3 → Monitor

Monitor ← Port 0 ← SGMII ← (Decrypt) Port 1 ← CAT5 Loop ← (Encrypt) Port 2 ← SGMII ← Port 3 ← Traffic Gen

The PHY encrypts on egress and decrypts on ingress. Therefore, the data will ingress with EtherType = 0800 through Port 0 and will be looped back on the MAC interface to the Port 1 MAC interface. The data will then be encrypted by the PHY on egress. The PHY will add bytes for the SecTag and ICV values. For this test, the data will show an increase in packet size of 32 bytes (16 bytes for SecTag (Explicit-SCI contained in SecTag) and 16 bytes for ICV) and EtherType = 88e5.

This data is then looped back into Port 2 of the PHY. The data will then be decrypted by the PHY on ingress of Port 2. The PHY will remove the bytes added for the SecTag and ICV. The data will ingress to the MAC interface with EtherType = 0800 and will be looped back to Port 3 MAC interface and egress out Port 3 as the same data that was inputted into Port 0.

The console output shows the following:

```
EGRESS CTRL Pattern Set Complete - Pattern Match Set to: VTSS_MACSEC_MATCH_ETYPE = 0800
INGRESS CTRL Pattern Set Complete - Pattern Match Set to: VTSS_MACSEC_MATCH_ETYPE = 88e5
```

This configuration is set up to *only* match the EtherType.

- On egress, if the pattern matches an EtherType = 0800 (IPv4), the packet will be encrypted.
- On ingress, if the pattern matches an EtherType = 88e5 (802.1ae), the packet will be decrypted.

Generate IPv4 Traffic (EtherType = 0800) and transmit into Port 0 of the PHY. Verify that the Port 3 data output matches that input into Port 0.

Generate IPv4 Traffic (EtherType = 0800) and transmit into Port 3 of the PHY. Verify that the Port 0 data output matches that input into Port 0.

Using the monitor, examine the Ethernet packet. Check the following:

- Ingress EtherType = 0800 (Port 0)
- Egress EtherType = 0800 (Port 3)
- Ingress Packet size = "x" (example, 128 bytes)
- Egress Packet size (Port 3) = "x" (example, 128 bytes)

To disable the MACsec operation for the ports, enter:

```
>macsec_off 1
>macsec_off 2
```

Data flow will now return to normal operation.

#### 6.4.3 MACSEC OVERVIEW AND CAPABILITIES WITHIN MESA API

While the Sample Application provides a simple MACsec implementation, there are many more MACsec API options available. The Sample Application may not have provided a way to test all of the MACsec APIs listed:

- **MACSEC Block Initialization**
  - vtss\_macsec\_init\_get
  - vtss\_macsec\_init\_set
- **MACSEC Default Action Get/Set**
  - vtss\_macsec\_default\_action\_get
  - vtss\_macsec\_default\_action\_set
- **MACSEC SECY Management**
  - vtss\_macsec\_secy\_conf\_add
  - vtss\_macsec\_secy\_conf\_get
  - vtss\_macsec\_secy\_conf\_update
  - vtss\_macsec\_secy\_cap\_get
- **MACSEC Controlled Port Get/Set**
  - vtss\_macsec\_secy\_controlled\_get
  - vtss\_macsec\_secy\_controlled\_set
- **MACSEC SECY Port Status**
  - vtss\_macsec\_secy\_port\_status\_get
- **MACSEC Rx SC Creation**
  - vtss\_macsec\_rx\_sc\_add
- **MACSEC Rx SC Control**
  - vtss\_macsec\_rx\_sc\_update
  - vtss\_macsec\_rx\_sc\_get\_conf
  - vtss\_macsec\_rx\_sc\_get\_next
  - vtss\_macsec\_rx\_sc\_del
- **MACSEC Rx SC Status: Receiving/createdTime/startedTime/stoppedTime**
  - vtss\_macsec\_rx\_sc\_status\_get
- **MACSEC Tx SC Creation**
  - vtss\_macsec\_tx\_sc\_set

- **MACSEC Tx SC Control**
  - vtss\_macsec\_tx\_sc\_update
  - vtss\_macsec\_tx\_sc\_get\_conf
  - vtss\_macsec\_tx\_sc\_del
- **MACSEC Rx SC Status: Transmitting/createdTime/startedTime/stoppedTime**
  - vtss\_macsec\_tx\_sc\_status\_get
- **MACSEC Rx SA Creation**
  - vtss\_macsec\_rx\_sa\_set
  - vtss\_macsec\_rx\_sa\_get
- **MACSEC Rx SA Control**
  - vtss\_macsec\_rx\_sa\_activate
  - vtss\_macsec\_rx\_sa\_lowest\_pn\_update
  - vtss\_macsec\_rx\_sa\_disable
  - vtss\_macsec\_rx\_sa\_del
- **MACSEC Rx SA Status**
  - vtss\_macsec\_rx\_sa\_status\_get
- **MACSEC Tx SA Creation**
  - vtss\_macsec\_tx\_sa\_set
  - vtss\_macsec\_tx\_sa\_get
- **MACSEC Tx SA Control**
  - vtss\_macsec\_tx\_sa\_activate
  - vtss\_macsec\_tx\_sa\_disable
  - vtss\_macsec\_tx\_sa\_del
- **MACSEC Tx SA Status**
  - vtss\_macsec\_tx\_sa\_status\_get
- **MACSEC Control Frame Match Config**
  - vtss\_macsec\_control\_frame\_match\_conf\_set
  - vtss\_macsec\_control\_frame\_match\_conf\_get
  - vtss\_macsec\_control\_frame\_match\_conf\_del
- **MACSEC Pattern Config**
  - vtss\_macsec\_pattern\_set
  - vtss\_macsec\_pattern\_get
  - vtss\_macsec\_pattern\_del
- **MACSEC VLAN Tag and Tag Bypass Config**
  - vtss\_macsec\_bypass\_mode\_set
  - vtss\_macsec\_bypass\_mode\_get
  - vtss\_macsec\_bypass\_tag\_set
  - vtss\_macsec\_bypass\_tag\_get
- **MACSEC Event Config**
  - vtss\_macsec\_event\_enable\_set
  - vtss\_macsec\_event\_enable\_get
  - vtss\_macsec\_event\_poll
  - vtss\_macsec\_event\_seq\_threshold\_set
  - vtss\_macsec\_event\_seq\_threshold\_get
- **MACsec Common Counters**
  - vtss\_macsec\_common\_counters\_get
  - vtss\_macsec\_counters\_update
  - vtss\_macsec\_counters\_clear

- SecY Statistics
  - vtss\_macsec\_secy\_counters\_get
  - vtss\_macsec\_controlled\_counters\_get
  - vtss\_macsec\_uncontrolled\_counters\_get
- Per-SC statistics
  - vtss\_macsec\_rx\_sc\_counters\_get
  - vtss\_macsec\_tx\_sc\_counters\_get
- Per-SA statistics
  - vtss\_macsec\_tx\_sa\_counters\_get
  - vtss\_macsec\_rx\_sa\_counters\_get
- MAC Counters
  - vtss\_macsec\_hmac\_counters\_get
  - vtss\_macsec\_lmac\_counters\_get

#### 6.4.4 SAMPLE DATA PATTERNS

Once use case 1 has been enabled, Port 1 and Port 2 are configured for MACsec operation. The PHY encrypts data upon egress of the PHY and decrypts data upon ingress. Ingress traffic must be of IPv4 traffic (EtherType = 0x0800) to be encrypted. The encrypted traffic that flows between Port 1 and Port 2 will have the designation for EtherType corresponding to IEEE 802.1AE (EtherType = 0x88e5), and the data is encrypted.

The following samples of IPv4 and IEEE 802.1AE packet descriptions aid the user in identifying packet elements.

Sample of Unencrypted IPv4 packet (etherType=0x0800):

Ethernet Packet Format: **DA**|**SA**|**Etype**|Payload|**CRC**

Ethernet Type = **Etype**

```

D6 09 B1 F0 56 63 7A 0D 46 DF 99 8B 08 00 0F 10
11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20
21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30
31 32 33 34 35 36 37 38 39 3A 00 02 9C 97 2B 11

```

Sample of Encrypted packet (etherType=0x88e5):

Encrypted Ethernet Packet Format: **DA**|**SA**|**SecTag**|Encrypted Payload|**ICV**|**CRC**

Ethernet Type = **Etype**

Part of SecTag is the Ethernet Type, which has been changed from IPv4 (0800) to IEEE 802.1AE (88e5). The SecTag can be 8 or 16 octets in size and is illustrated below in the 16 octet size:

| 2 Octets                     | 1 Octet          | 1 Octet      | 4 Octets           | 8 Octets                               |
|------------------------------|------------------|--------------|--------------------|----------------------------------------|
| <b>MACsec EtherType 88e5</b> | <b>TCI/AN 2E</b> | <b>SL 00</b> | <b>PN B2C28465</b> | <b>SCI (optional) 12153524C0895E81</b> |

The ICV is cipher-suite-dependent and must be at least 8 octets in length and no more than 16 octets in length. This example shows a packet with a SecTag and an ICV of 16 octets.

```

D6 09 B1 F0 56 63 7A 0D 46 DF 99 8B 88 E5 2E 00
B2 C2 84 65 12 15 35 24 C0 89 5E 81 70 1A FA 1C
C0 39 C0 D7 65 12 8A 66 5D AB 69 24 38 99 BF 73
18 CC DC 81 C9 93 1D A1 7F BE 8E DD 7D 17 CB 8B
4C 26 FC 81 E3 28 4F 2B 7F BA 71 3D 4F 8D 55 E7
D3 F0 6F D5 A1 3C 0C 29 B9 D5 B8 80 04 66 DC A1

```

## APPENDIX A: FUNCTION CALL SEQUENCE IN VTSS PHY API TO INITIALIZE PHY FOR MACSEC OPERATION

An unencrypted IPv4 Ethernet packet has the following format:

|DA|SA|Payload|CRC|

The "Std" calls initialize the MACsec software which creates the following encrypted Ethernet packet format:

|DA|SA|SECTag|Encrypted Payload|ICV|CRC|

The "Optional VLAN Tag Bypass" adds additional initialization for the MACsec software, which creates an encrypted Ethernet packet of the following format:

|DA|SA|VLAN|SECTag|Encrypted Payload|ICV|CRC|

Std:Initialize macsec block for given port.

```
vtss_macsec_init_set(inst, macsec_port, init_data);
```

Std:Initialize macsec default action.

```
vtss_macsec_default_action_set(inst, macsec_port, default_action_policy);
```

Option #1 (Part A) Adding VLAN Tag Bypass (Optional - Only used If adding VLAN Tag Bypass)

```
vtss_macsec_bypass_mode_set(inst, macsec_physical_port, bypass_ctrl);
```

Std:Create a SecY

```
vtss_macsec_secy_conf_add(inst, macsec_port, macsec_port_conf);
```

Option #1 (Part B): Adding VLAN Tag Bypass (Optional - Only used If adding VLAN Tag Bypass)

```
vtss_macsec_bypass_tag_set(inst, macsec_physical_port, bypass_tag_type);
```

Std:Match a frame to be encrypted. (SA Classification). Associate the pattern with the controlled MACsec port - Ingress Direction

```
vtss_macsec_pattern_set(inst, macsec_port, DIRECTION_INGRESS,
 MATCH_ACTION_CONTROLLED_PORT, pattern_ctrl_ing);
```

Std:Match a frame to be encrypted. (SA Classification). Associate the pattern with the controlled MACsec port - Egress Direction

```
vtss_macsec_pattern_set(inst, macsec_port, DIRECTION_EGRESS,
 MATCH_ACTION_CONTROLLED_PORT, pattern_ctrl_egr);
```

Std:Match a frame to be encrypted. (SA Classification). Associate the pattern with the uncontrolled MACsec port - Ingress Direction

```
vtss_macsec_pattern_set(inst, macsec_port, DIRECTION_INGRESS,
 MATCH_ACTION_UNCONTROLLED_PORT, pattern_unctrl_ing);
```

Std:Match a frame to be encrypted. (SA Classification). Associate the pattern with the uncontrolled MACsec port - Egress Direction

```
vtss_macsec_pattern_set(inst, macsec_port, DIRECTION_EGRESS,
 MATCH_ACTION_UNCONTROLLED_PORT, pattern_unctrl_egr);
```

Std:Enable the SecY

```
vtss_macsec_secy_controlled_set(inst, macsec_port, enable);
```

Std:Add a new TX Secure channel

```
vtss_macsec_tx_sc_set(inst, macsec_port);
```

Std:Install the TX key in the HW on the Egress side

```
vtss_macsec_tx_sa_set(inst, macsec_port, assoc_no, next_pn, confidentiality, sak_tx_sa_0);
```

Std:Activate the key in the HW on the Egress side

```
vtss_macsec_tx_sa_activate(inst, macsec_port, assoc_no);
```

Std:Add a MACsec peer

Std:Add a new RX secure channel

```
vtss_macsec_rx_sc_add(inst, macsec_port, sci_rx);
```

Std:Update the Hash key in the SAK, then Install the key in HW for decryption

```
vtss_macsec_rx_sa_set(inst, macsec_port, sci_rx, assoc_no, lowest_pn, sak_rx_sa_0);
```

Std:Activate, Identifying the SA and the corresponding SC

```
vtss_macsec_rx_sa_activate(inst, macsec_port, sci_rx, assoc_no);
```

## APPENDIX B: APPLICATION NOTE REVISION HISTORY

TABLE B-1: REVISION HISTORY

| Revision Level & Date     | Section/Figure/Entry | Correction |
|---------------------------|----------------------|------------|
| DS00004028A<br>(06-16-21) | Initial release      |            |

## THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at [www.microchip.com](http://www.microchip.com). This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

## CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at [www.microchip.com](http://www.microchip.com). Under "Support", click on "Customer Change Notification" and follow the registration instructions.

## CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

**Technical support is available through the web site at: <http://microchip.com/support>**



---

**Note the following details of the code protection feature on Microchip devices:**

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods being used in attempts to breach the code protection features of the Microchip devices. We believe that these methods require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Attempts to breach these code protection features, most likely, cannot be accomplished without violating Microchip's intellectual property rights.
- Microchip is willing to work with any customer who is concerned about the integrity of its code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is "unbreakable." Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

---

Information contained in this publication is provided for the sole purpose of designing with and using Microchip products. Information regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications.

THIS INFORMATION IS PROVIDED BY MICROCHIP "AS IS". MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL OR CONSEQUENTIAL LOSS, DAMAGE, COST OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP'S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

**Trademarks**

The Microchip name and logo, the Microchip logo, Adaptec, AnyRate, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, chipKIT, chipKIT logo, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, HELDO, IGLOO, JukeBlox, KeeLoq, Kleer, LANCheck, LinkMD, maXStylus, maXTouch, MediaLB, megaAVR, Microsemi, Microsemi logo, MOST, MOST logo, MPLAB, OptoLyzer, PackTime, PIC, picoPower, PICSTART, PIC32 logo, PolarFire, Prochip Designer, QTouch, SAM-BA, SenGenuity, SpyNIC, SST, SST Logo, SuperFlash, Symmetricom, SyncServer, Tachyon, TimeSource, tinyAVR, UNI/O, Vectron, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AgileSwitch, APT, ClockWorks, The Embedded Control Solutions Company, EtherSynch, FlashTec, Hyper Speed Control, HyperLight Load, IntelliMOS, Libero, motorBench, mTouch, Powermite 3, Precision Edge, ProASIC, ProASIC Plus, ProASIC Plus logo, Quiet-Wire, SmartFusion, SyncWorld, Temux, TimeCesium, TimeHub, TimePictra, TimeProvider, WinPath, and ZL are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, Augmented Switching, BlueSky, BodyCom, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, Espresso T1S, EtherGREEN, IdealBridge, In-Circuit Serial Programming, ICSP, INICnet, Intelligent Paralleling, Inter-Chip Connectivity, JitterBlocker, maxCrypto, maxView, memBrain, Mindi, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PowerSmart, PureSilicon, QMatrix, REAL ICE, Ripple Blocker, RTAX, RTG4, SAM-ICE, Serial Quad I/O, simpleMAP, SimpliPHY, SmartBuffer, SMART-I.S., storClad, SQL, SuperSwitcher, SuperSwitcher II, Switchtec, SynchroPHY, Total Endurance, TSHARC, USBCheck, VariSense, VectorBlox, VeriPHY, ViewSpan, WiperLock, XpressConnect, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

The Adaptec logo, Frequency on Demand, Silicon Storage Technology, and Symmcom are registered trademarks of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

© 2021, Microchip Technology Incorporated, All Rights Reserved.

ISBN: 978-1-5224-8376-2

For information regarding Microchip's Quality Management Systems, please visit [www.microchip.com/quality](http://www.microchip.com/quality).

## Worldwide Sales and Service

### AMERICAS

**Corporate Office**  
2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-792-7200  
Fax: 480-792-7277  
Technical Support:  
<http://www.microchip.com/support>  
Web Address:  
[www.microchip.com](http://www.microchip.com)

**Atlanta**  
Duluth, GA  
Tel: 678-957-9614  
Fax: 678-957-1455

**Austin, TX**  
Tel: 512-257-3370

**Boston**  
Westborough, MA  
Tel: 774-760-0087  
Fax: 774-760-0088

**Chicago**  
Itasca, IL  
Tel: 630-285-0071  
Fax: 630-285-0075

**Dallas**  
Addison, TX  
Tel: 972-818-7423  
Fax: 972-818-2924

**Detroit**  
Novi, MI  
Tel: 248-848-4000

**Houston, TX**  
Tel: 281-894-5983

**Indianapolis**  
Noblesville, IN  
Tel: 317-773-8323  
Fax: 317-773-5453  
Tel: 317-536-2380

**Los Angeles**  
Mission Viejo, CA  
Tel: 949-462-9523  
Fax: 949-462-9608  
Tel: 951-273-7800

**Raleigh, NC**  
Tel: 919-844-7510

**New York, NY**  
Tel: 631-435-6000

**San Jose, CA**  
Tel: 408-735-9110  
Tel: 408-436-4270

**Canada - Toronto**  
Tel: 905-695-1980  
Fax: 905-695-2078

### ASIA/PACIFIC

**Australia - Sydney**  
Tel: 61-2-9868-6733

**China - Beijing**  
Tel: 86-10-8569-7000

**China - Chengdu**  
Tel: 86-28-8665-5511

**China - Chongqing**  
Tel: 86-23-8980-9588

**China - Dongguan**  
Tel: 86-769-8702-9880

**China - Guangzhou**  
Tel: 86-20-8755-8029

**China - Hangzhou**  
Tel: 86-571-8792-8115

**China - Hong Kong SAR**  
Tel: 852-2943-5100

**China - Nanjing**  
Tel: 86-25-8473-2460

**China - Qingdao**  
Tel: 86-532-8502-7355

**China - Shanghai**  
Tel: 86-21-3326-8000

**China - Shenyang**  
Tel: 86-24-2334-2829

**China - Shenzhen**  
Tel: 86-755-8864-2200

**China - Suzhou**  
Tel: 86-186-6233-1526

**China - Wuhan**  
Tel: 86-27-5980-5300

**China - Xian**  
Tel: 86-29-8833-7252

**China - Xiamen**  
Tel: 86-592-2388138

**China - Zhuhai**  
Tel: 86-756-3210040

### ASIA/PACIFIC

**India - Bangalore**  
Tel: 91-80-3090-4444

**India - New Delhi**  
Tel: 91-11-4160-8631

**India - Pune**  
Tel: 91-20-4121-0141

**Japan - Osaka**  
Tel: 81-6-6152-7160

**Japan - Tokyo**  
Tel: 81-3-6880-3770

**Korea - Daegu**  
Tel: 82-53-744-4301

**Korea - Seoul**  
Tel: 82-2-554-7200

**Malaysia - Kuala Lumpur**  
Tel: 60-3-7651-7906

**Malaysia - Penang**  
Tel: 60-4-227-8870

**Philippines - Manila**  
Tel: 63-2-634-9065

**Singapore**  
Tel: 65-6334-8870

**Taiwan - Hsin Chu**  
Tel: 886-3-577-8366

**Taiwan - Kaohsiung**  
Tel: 886-7-213-7830

**Taiwan - Taipei**  
Tel: 886-2-2508-8600

**Thailand - Bangkok**  
Tel: 66-2-694-1351

**Vietnam - Ho Chi Minh**  
Tel: 84-28-5448-2100

### EUROPE

**Austria - Wels**  
Tel: 43-7242-2244-39  
Fax: 43-7242-2244-393

**Denmark - Copenhagen**  
Tel: 45-4485-5910  
Fax: 45-4485-2829

**Finland - Espoo**  
Tel: 358-9-4520-820

**France - Paris**  
Tel: 33-1-69-53-63-20  
Fax: 33-1-69-30-90-79

**Germany - Garching**  
Tel: 49-8931-9700

**Germany - Haan**  
Tel: 49-2129-3766400

**Germany - Heilbronn**  
Tel: 49-7131-72400

**Germany - Karlsruhe**  
Tel: 49-721-625370

**Germany - Munich**  
Tel: 49-89-627-144-0  
Fax: 49-89-627-144-44

**Germany - Rosenheim**  
Tel: 49-8031-354-560

**Israel - Ra'anana**  
Tel: 972-9-744-7705

**Italy - Milan**  
Tel: 39-0331-742611  
Fax: 39-0331-466781

**Italy - Padova**  
Tel: 39-049-7625286

**Netherlands - Drunen**  
Tel: 31-416-690399  
Fax: 31-416-690340

**Norway - Trondheim**  
Tel: 47-7288-4388

**Poland - Warsaw**  
Tel: 48-22-3325737

**Romania - Bucharest**  
Tel: 40-21-407-87-50

**Spain - Madrid**  
Tel: 34-91-708-08-90  
Fax: 34-91-708-08-91

**Sweden - Gothenberg**  
Tel: 46-31-704-60-40

**Sweden - Stockholm**  
Tel: 46-8-5090-4654

**UK - Wokingham**  
Tel: 44-118-921-5800  
Fax: 44-118-921-5820