AN5213

Configuration and Programming Options for the PCI1xxxx

Authors: *Andrew Rogers, Josh Averyt and Sukanya Palanisami*
Microchip Technology Inc.

1.0 INTRODUCTION

The Microchip PCI1XXXX is a family of highly customizable PCI Express switches with integrated PCI Express (PCIe) endpoint devices including USB 3.2 10 Gbit/s Host, Ethernet MAC, and multi-function device with UART, I^2C, SPI, and GPIO peripherals. This family of devices includes PCI12000, PCI11010, PCI11101, PCI11400, and PCI11414.

The PCI1XXXX devices can be configured in the following ways:

- External EEPROM Memory
- One Time Programmable (OTP) Memory
- I^2C or SPI Configuration Interface
- Run-time Configuration via PCIe Host

1.1 Sections

This application note covers the following sections:

1.2 References

The following documents should be referenced when using this application note:

- *PCI12000 Data Sheet* (www.microchip.com/DS00003794)
- *PCI11010 Data Sheet* (www.microchip.com/DS00003793)
- *PCI11101 Data Sheet* (www.microchip.com/DS00003791)
- *PCI11400 Data Sheet* (www.microchip.com/DS00003447)
- *PCI11414 Data Sheet* (www.microchip.com/DS00003792)
- *PCI Express Base r3.1a,* (Available from PCI-SIG)
- *I^2C Specification*
- *SPI Specification*
- *System Management Bus Specification*, Version 1.0 (http://smbus.org/specs)

2.0 DEVICE PROGRAMMING AND CONFIGURATION SUMMARY

2.1 Sections

- Section 2.2, "Configuration Options"
- Section 2.3, "Required Configuration Selections Summary"

The PCI1xxxx family of devices has a wide array available features and functions as shown in Table 1.

TABLE 1: DEVICE PROPERTIES AND PROGRAMMING OPTIONS

Part Number	Package	PCIe Up stream	PCIe Down stream	USB2 PHYs	USB3 PHYs	Ethernet MAC	UART Interfaces (Note 2)	I2C Hosts	SPI Hosts	GPIOs (max)
PCI12000	72 QFN	x2	Two x1	0	0	0	0	1	2	18
PCI11010	100 QFN	x2	One x1	0	0	1	0	1	2	46
PCI11101	132 DQFN	x4	One x2	1	1	0	1	1	2	48
PCI11400	132 DQFN	x4	One x1	4	2	0	0	1	2	38
PCI11414	164 DQFN	x4	One x1	4	2	1	4	1	2	64

Note 1: To achieve the maximum supported number of GPIOs listed, many of the primary device interfaces must be disabled in order to reconfigure the associated PROGx pins as GPIOs.

2: Default configuration. Each SKU can be configured with up to 4 UARTs.

2.2 Configuration Options

The PCI1xxxx family of devices has a wide array of available features and functions, many of which need to be configured before use. The bulk of the configuration must be done via either EEPROM, OTP memory, or SPI/I^2C. While the device supports configuration from all three pathways simultaneously, it is recommended to select just one of these methods to simplify and centralize the configuration data, see Table 2.

TABLE 2: CONFIGURATION METHODS

Configuration Method	Notes
Hardware Strap Options	Hardware straps are limited to a few key items, which all designs must select and handle in the schematic design.
EEPROM	EEPROM devices may be reprogrammed across lifespan of product, but increase total system BOM cost.
OTP Memory	OTP memory incurs no additional cost, but is limited in future upgradability.
SPI/I^2C Configuration	If an MCU/SoC is located on the system, SPI/I^2C configuration may be a preferred approach. This method also allows for dynamic reconfiguration of certain features, which EEPROM and OTP methods cannot achieve.
Run-time Reconfiguration via I^2C/SPI/PCIe	Certain configuration items can be modified during run-time. This requires application level software and many configuration items are unable to be controlled in this way.

2.3 Required Configuration Selections Summary

Many functions required some basic level of configuration to become operable. These include:

- **Programmable Function Pins:** The majority of programmable function pins operate as GPIOs by default. However, most USB/Ethernet/Peripheral functions require support I/O which must be enabled and muxed to the most appropriate PROGx pin.
- **PCIe Switch:** Default configuration may be sufficient, but if any PCIe ports are unused, they should be disabled via configuration.
- **USB Ports**: USB ports should be configured for speed and connection type (i.e.,: Type-C, Type-A, or embedded device).
- **Ethernet MAC:** Ethernet MAC must be configured according to the selected PHY, system MAC address, and any special speed or mode of operation requirements.
- **GPIO, UART, I^2C, and SPI Controllers:** Bridge controllers typically require basic configuration to match intended operation. Configuration options include pin multiplexing selection, choosing the desired number of ports, pin pad options, and so on.

3.0 QUESTION AND ANSWER

1. Is EEPROM/OTP programming required?

No. EEPROM/OTP programming is not required if configuration is handled via an MCU/SOC via either I^2C or SPI.

The PCI1xxxx is not designed to provide any functional modes of operation without some form of configuration. Default settings will not yield a usable mode of operation.

2. Is a preprogrammed EEPROM device available?

No. A preprogrammed EEPROM device is not available from Microchip.

3. Does PCI1xxx execute firmware?

No. The PCI1xxxx does not contain a processor and does not execute firmware.

4. Unused Peripherals - Can and should they be disabled?

Unused peripherals can be disabled in the configuration data. If there are any unused peripherals in the final hardware design, it is recommended to disable these functions for the following reasons:

- To mask the peripheral from appearing on the host computer
- To minimize power consumption
- To prevent unexpected behavior stemming from “floating” peripheral hardware pins

Note:	The GPIO subsystem should not be disabled as this interface is also used for programming OTP or EEPROM configuration.

5. Can the USB host port speeds be modified or disabled all together?

To disable the USB 3.1 interface of a port, modify both of the following configuration parameters:

- USB_PORT_ENABLE_REG.HOST_NUM_U3_PORT[3:0]
- USB_PORT_ENABLE_REG.USB_U3_PORT_ENABLE[3:0]

To disable the USB 2.0 interface of a given port, modify both of the following configuration parameters:

- USB_PORT_ENABLE_REG.HOST_NUM_U2_PORT[3:0]
- USB_PORT_ENABLE_REG.USB_U2_PORT_ENABLE[3:0]

6. Can the Ethernet speeds be fixed?

Yes. The Ethernet speeds can be fixed to specific speeds if desired. This will cause connection failure if connected to a network which operates at any other speed(s) than the selected speed, so this is only recommended under specific scenarios or for test purposes.

7. Is the MAC address preprogrammed?

No. By default, the MAC address is not preprogrammed and the end system integrator is responsible for provisioning the MAC address through proper channels and programming it. Microchip provides programming tools to facilitate and automate the programming process.

To set a MAC address, program the following registers:

- MAC_RX_ADDRH
- MAC_RX_ADDRL

8. Can the Ethernet MAC address be overridden by the host computer?

Yes. The host computer can override the MAC address through the run-time reconfiguration process. The MAC address registers are simply overwritten with the new value.

9. How do I obtain a block of MAC addresses for my product?

Generally, blocks of MAC addresses are purchased from the IEEE. Visit the IEEE Registration Authority for more details (https://standards.ieee.org/products-programs/regauth/).

4.0 BOOT/CONFIGURATION SEQUENCE

During the configuration phase, the pin muxing will be configured to align to the system's PCB design and features are configured to align with system requirements. The initialization is expected to optionally be provided by external configuration sources, either SMBus/I^2C or SPI. Due to IC packaging limitations, PCI1xxxx contains more pin functions than can be supported all at once, this places explicit requirements upon pin muxing and careful selection of concurrent functions which will be implemented within a system.

The intended external configuration sources must be indicated via pin straps with subsequent pin muxing actions, in order for the correct configuration interface to become exposed.

4.1 Sections

- Section 4.2, "Configuration Sequence"
- Section 4.3, "Boot Stage 1: Load OTP Memory"
- Section 4.4, "Boot Stage 2: Load EEPROM Memory"
- Section 4.5, "Boot Stage 3: Serial Configuration"
- Section 4.6, "Run-time"

4.2 Configuration Sequence

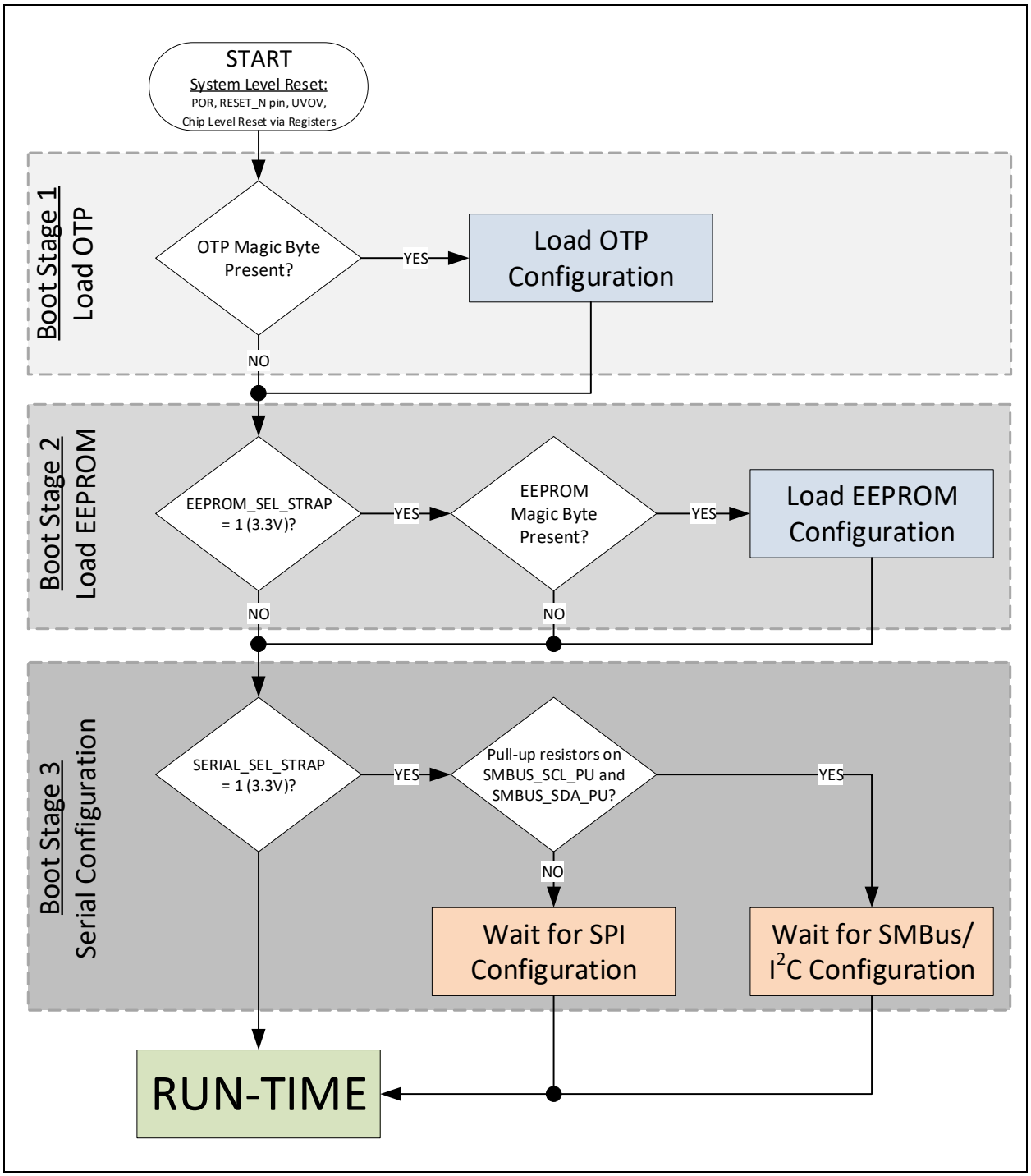
This configuration must be loaded sequentially such that either of these initialization configuration flows are possible, where the register contents become superseded (if overridden) by each configuration source in the sequence. See Figure 1 for the configuration flow.

The supported configuration sequences are:

1. HW Default Configuration > OTP Configuration
2. HW Default Configuration > OTP Configuration > EEPROM Configuration
3. HW Default Configuration > OTP Configuration > EEPROM Configuration > SPI/I^2C Configuration
4. HW Default Configuration > OTP Configuration > SPI/I^2C Configuration
5. HW Default Configuration > EEPROM Configuration
6. HW Default Configuration > EEPROM Configuration > SPI/I^2C Configuration
7. HW Default Configuration > SPI/I^2C Configuration

Upon completion of initialization configuration, the PCIe switch shall be able to be enumerated upon the PCIe bus. After PCIe connection occurs, configuration is complete and PCI1xxxx is considered to be in a Run-time mode of operation.

FIGURE 1: BOOT CONFIGURATION FLOW



4.3 Boot Stage 1: Load OTP Memory

Upon boot, the PCI1xxxx checks for the “Magic Byte”, a value of A5h, in OTP memory location 0000h. If any value other than A5h is read, OTP memory is considered to be invalid and the PCI1xxxx moves onto Boot Stage 2.

If a value of A5h, the OTP memory is loaded sequentially from the beginning of the OTP memory to the end. The OTP memory loading process is documented in Section 14.2, "Loading OTP Memory at Boot".

For additional detail on EEPROM confirmation, see Section 13.0, "Configuration File Formatting".

4.4 Boot Stage 2: Load EEPROM Memory

In Boot Stage 2, the EEPROM_SEL_STRAP hardware resistor strap option is checked. This is a simple digital input pin. If the voltage is sensed as low (below VIL specification for the pin), then the PCI1xxxx moves onto Boot Stage 3.

If the voltage on EEPROM_SEL_STRAP is sensed as high (above VIH specification for the pin), then the PCI1xxxx attempts to read from the attached EEPROM device as an SMBus/I^{2}C Controller. The EEPROM memory is first checked for the “Magic Byte”, a value of A5h, in EEPROM memory offset 0000h. If a value of A5h is read back from the “Magic Byte” memory location, the EEPROM memory is loaded sequentially from the beginning of the EEPROM memory to the end. The EEPROM memory loading process is documented in Section 15.2, "EEPROM Memory Load at Boot".

The configuration data located within an EEPROM memory device follows the same command formatting as the OTP memory configuration data.

Note:	If any register is configured both within OTP and EEPROM memory, the value as set from EEPROM configuration data will prevail.

For additional detail on EEPROM confirmation, see Section 15.0, "EEPROM Memory Programming (Read and Write)".

4.5 Boot Stage 3: Serial Configuration

In Boot Stage 3, the SERIAL_SEL_STRAP hardware resistor strap option is checked. This is a simple digital input pin. If the voltage is sensed as low (below VIL specification for the pin), then the PCI1xxxx moves directly into Run-time and will be available to connect its upstream PCIe interface to a PCIe host.

If the voltage on SERIAL_SEL_STRAP is sensed as high (above VIH specification for the pin), then the PCI1xxxx checks the SMBUS_SCL_PU and SMBUS_SDA_PU hardware pin straps.

If the voltage on both SMBUS_SCL_PU/SMBUS_SDA_PU pins are sensed as high (above VIH specification for the pin), then the SMBus/I^{2}C configuration interface is enabled. Otherwise, the SPI configuration interface is enabled. The PIC1xxxx will wait within the configuration stage indefinitely until the configuration done bits with EXT_SYS_CONFIG_-DONE_REG are set.

Note:	If any register is configured both within OTP and/or EEPROM as well as via Serial Configuration, the value as set during Serial Configuration will prevail.

For additional detail on EEPROM confirmation, see Section 16.0, "SMBus/I2C Target Configuration" and Section 17.0, "SPI Configuration".

4.6 Run-time

During run-time, any register may be modified through general PCI memory writes using a tool preferred by the designer. However, to ensure stable PCI1XXXX operation, it is recommended to utilize the provided drivers and APIs for all normal use cases. Direct PCI register manipulation should be limited to test and debug.

Users can refer to the register map in *AN4255 PCI12000/PCI11XXXX Register Map*.

5.0 PROGRAMMABLE PIN MUXING CONFIGURATION

5.1 Programmable Pin Muxing

Each programmable pin has several selectable options that can be configured based upon end application use cases. The product data sheet contains a table of the available programmable pins and the default selections.

If a change to the default configuration is required, then the respective programmable function register must be modified. Address offsets for each configuration register are documented in Table 3.

TABLE 3: PIN MUXING CONFIGURATION

Address	Register	Address	Register	Address	Register
0x0024_0400	PF0_CTL_REG	0x0024_0480	PF32_CTL_REG	0x0024_0500	PF64_CTL_REG
0x0024_0404	PF1_CTL_REG	0x0024_0484	PF33_CTL_REG	0x0024_0504	PF65_CTL_REG
0x0024_0408	PF2_CTL_REG	0x0024_0488	PF34_CTL_REG	0x0024_0508	PF66_CTL_REG
0x0024_040C	PF3_CTL_REG	0x0024_048C	PF35_CTL_REG	0x0024_050C	PF67_CTL_REG
0x0024_0410	PF4_CTL_REG	0x0024_0490	PF36_CTL_REG	0x0024_0510	PF68_CTL_REG
0x0024_0414	PF5_CTL_REG	0x0024_0494	PF37_CTL_REG	0x0024_0514	PF69_CTL_REG
0x0024_0418	PF6_CTL_REG	0x0024_0498	PF38_CTL_REG	0x0024_0518	PF70_CTL_REG
0x0024_041C	PF7_CTL_REG	0x0024_049C	PF39_CTL_REG	0x0024_051C	PF71_CTL_REG
0x0024_0420	PF8_CTL_REG	0x0024_04A0	PF40_CTL_REG	0x0024_0520	PF72_CTL_REG
0x0024_0424	PF9_CTL_REG	0x0024_04A4	PF41_CTL_REG	0x0024_0524	PF73_CTL_REG
0x0024_0428	PF10_CTL_REG	0x0024_04A8	PF42_CTL_REG	0x0024_0528	PF74_CTL_REG
0x0024_042C	PF11_CTL_REG	0x0024_04AC	PF43_CTL_REG	0x0024_052C	PF75_CTL_REG
0x0024_0430	PF12_CTL_REG	0x0024_04B0	PF44_CTL_REG	0x0024_0530	PF76_CTL_REG
0x0024_0434	PF13_CTL_REG	0x0024_04B4	PF45_CTL_REG	0x0024_0534	PF77_CTL_REG
0x0024_0438	PF14_CTL_REG	0x0024_04B8	PF46_CTL_REG	0x0024_0538	PF78_CTL_REG
0x0024_043C	PF15_CTL_REG	0x0024_04BC	PF47_CTL_REG	0x0024_053C	PF79_CTL_REG
0x0024_0440	PF16_CTL_REG	0x0024_04C0	PF48_CTL_REG	0x0024_0540	PF80_CTL_REG
0x0024_0444	PF17_CTL_REG	0x0024_04C4	PF49_CTL_REG	0x0024_0544	PF81_CTL_REG
0x0024_0448	PF18_CTL_REG	0x0024_04C8	PF50_CTL_REG	0x0024_0548	PF82_CTL_REG
0x0024_044C	PF19_CTL_REG	0x0024_04CC	PF51_CTL_REG	0x0024_054C	PF83_CTL_REG
0x0024_0450	PF20_CTL_REG	0x0024_04D0	PF52_CTL_REG	0x0024_0550	PF84_CTL_REG
0x0024_0454	PF21_CTL_REG	0x0024_04D4	PF53_CTL_REG	0x0024_0554	PF85_CTL_REG
0x0024_0458	PF22_CTL_REG	0x0024_04D8	PF54_CTL_REG	0x0024_0558	PF86_CTL_REG
0x0024_045C	PF23_CTL_REG	0x0024_04DC	PF55_CTL_REG	0x0024_055C	PF87_CTL_REG
0x0024_0460	PF24_CTL_REG	0x0024_04E0	PF56_CTL_REG	0x0024_0560	PF88_CTL_REG
0x0024_0464	PF25_CTL_REG	0x0024_04E4	PF57_CTL_REG	0x0024_0564	PF89_CTL_REG
0x0024_0468	PF26_CTL_REG	0x0024_04E8	PF58_CTL_REG	0x0024_0568	PF90_CTL_REG
0x0024_046C	PF27_CTL_REG	0x0024_04EC	PF59_CTL_REG	0x0024_056C	PF91_CTL_REG
0x0024_0470	PF28_CTL_REG	0x0024_04F0	PF60_CTL_REG	0x0024_0570	PF92_CTL_REG
0x0024_0474	PF29_CTL_REG	0x0024_04F4	PF61_CTL_REG	0x0024_057C	Reserved
0x0024_0478	PF30_CTL_REG	0x0024_04F8	PF62_CTL_REG	to	
0x0024_047C	PF31_CTL_REG	0x0024_04FC	PF63_CTL_REG	0x0024_05FF	

Each programmable pin register has the standard bit mapping as shown in Table 4.

TABLE 4: PROGRAM FUNCTION REGISTER FORMAT

PFX_CTL_REG		
Bit(s)	**R/W**	**Description**
31:17	R	Reserved
16	R/W	Selects the RGMII/MII Pad Mode: 0b: GMII 1b: RGMII **Note:** This register takes effect only if PAD_TYPE = 1b. When PAD_TYPE = 0b, this field is 00b and read-only/reserved. When SELECT[3:0] is set to FUNC1 and PAD_TYPE = 1b, this field shall be read-only and controlled by internal hardware.
15:14	R	Reserved
12:13	R/W	Selects the drive strength of the pin when operating as a general purpose I/O. 00b: 2 mA 01b: 4 mA 10b: 8 mA 11b:10 mA **Note:** This register takes effect only if PAD_TYPE = 0b. When PAD_TYPE = 1b, this field is 00b and read-only/reserved.
11:9	R	Reserved
8	R	Indicates the type of pin pad. 0b - General purpose 3.3V I/O 1b - GMII/RGMII
7:4	R	Reserved
3:0	R/W	0000b - FUNC0 - GPIO 0001b - FUNC1 HW Defined Function 0010b - FUNC2 HW Defined Function 0011b - FUNC3 HW Defined Function 0100b - FUNC4 HW Defined Function 0101b - FUNC5 HW Defined Function 0110b - FUNC6 HW Defined Function 0111b - FUNC7 HW Defined Function 1000b - FUNC8 HW Defined Function 1001b - FUNC9 HW Defined Function 1010b - FUNC10 HW Defined Function 1011b - FUNC11 HW Defined Function 1100b - FUNC12 HW Defined Function 1101b - FUNC13 HW Defined Function 1110b - FUNC14 HW Defined Function 1111b - FUNC15 HW Defined Function **Note:** For the Hardware Defined Functions, refer to the Programmable Function Pin Map table of the respective product data sheet.

6.0 PCIE SWITCH IMPLEMENTATION

The integrated PCIe switch is configured as shown in Table 5.

TABLE 5: PCIE SWITCH CONFIGURATION

Port #	Direction	Lane Count	Bit Rate	Description and Connection	Device Support
0	Upstream	2/4	8GT/s	Upstream Port	2 for PCI12000 and PCI11010, 4 for the rest
1	Downstream	1/2	8GT/s	Downstream PCIe port for External Device Connection	2 for PCI11101, 1 for the rest
2	Downstream	4	8GT/s	Downstream PCIe port for External Device Connection or USB 3.2 Gen 2 Host Controller	PCI11101, PCI11400, PCI11414
3	Downstream	1	5GT/s	Ethernet Controller	PCI11010, PCI11414
4	Downstream	1	2.5GT/s	Multi-Function Device Endpoint (GPIO/SMBus/SPI/UART)	All

The PCI Switch IDs are:

- Vendor ID: 0x1055 (Microchip Technology, Inc/SMSC)
- Device ID:
  - PCI12000: 0xA008 (Port 0), 0xA009 (Port 1), 0xA00A (Port 2), 0xA00B (Port 2), 0xA00C (Port 4)
  - PCI11010: 0xA018 (Port 0), 0xA019 (Port 1), 0xA01A (Port 2), 0xA01B (Port 2), 0xA01C (Port 4)
  - PCI11101: 0xA028 (Port 0), 0xA029 (Port 1), 0xA02A (Port 2), 0xA02B (Port 2), 0xA02C (Port 4)
  - PCI11400: 0xA038 (Port 0), 0xA039 (Port 1), 0xA03A (Port 2), 0xA03B (Port 2), 0xA03C (Port 4)
  - PCI11414: 0xA048 (Port 0), 0xA049 (Port 1), 0xA04A (Port 2), 0xA04B (Port 2), 0xA04C (Port 4)

6.1 PCIe Switch Subsystem Registers

The PCIE_SWITCH_ADDR_BASE = 0x001C_0000 available registers are listed in Table 6.

Caution: PCI1xxxx devices have thousands of registers which are either reserved for future use or whose contents are masked from public documentation because they have no end system integrator use case (i.e.: they are used by hardware to perform low level run-time tasks or are used/controlled directly by a driver during run-time). **Do not modify any contents of undocumented registers.**

TABLE 6: USB SUBSYSTEM REGISTER OFFSETS FROM PCIE_SWITCH_ADDR_BASE

Register Name	Port 0	Port 1	Port 2	Port 3	Port 4
PCIE_SW_CONFIG_REG	0x0_0000				
PCIE_SW_CTL_REG	0x0_0004				
PCIE_SW_TESTIN_REG	0x0_0020				
PCIE_SW_PORT_HP_CAP_REG	n/a	0x1_1020	n/a	n/a	n/a
PCIE_SW_PORT_CTL_REG	0x1_0024	0x1_1024	0x1_2024	0x1_3024	0x1_4024
PCIE_SW_PORT_CFG_VID_REG	0x1_0040	0x1_1040	0x1_2040	0x1_3040	0x1_4040
PCIE_SW_PORT_CFG_-DID_REG	0x1_0044	0x1_1044	0x1_2044	0x1_3044	0x1_4044
PCIE_SW_PORT_DBG_REG	0x1_0150	0x1_1150	0x1_2150	0x1_3150	0x1_4150
PCIE_SW_PORT_PCI_CON-FIG_SPACE	0x1_0800	0x1_1800	0x1_2800	0x1_3800	0x1_4800

The PCIE Switch Configuration Register contents are shown in Table 7.

TABLE 7: PCIE_SW_CONFIG_REG

PCIE SWITCH CONFIGURATION REGISTER			**Default:** PCI12000: 0x0000_0000 PCI11414/PCI11400/PCI11101/PCI11010: 0x0000_0004
Bits	**Name**	**R/W**	**Description**
31:6	Reserved	R	Always 0's
5	PCIE_SW_ EXT_OBFF_WIRE_OUT_EN	R/W	Connects the WAKE# signal output from the pl1_wake_oen Wake output on port 1 to the PCIE_WAKE_N pin to enable the switch to signal OBFF to downstream PCI Devices via wire: 0b: External OBFF output disabled. 1b: External OBFF output enabled.
4	PCIE_SW_EXT_WAKE_EN	R/W	Connects the WAKE# signal input from the pl1_wake_in Wake input on port 1 of the switch to the PCIE_WAKE_N pin to enable downstream PCI Devices to signal WAKE# via wire: 0b: External WAKE# input disabled. 1b: External WAKE# input enabled.
3	Reserved	R	Always 0's
2	PCIE_SW_P2_XHCI_CONN	R/W	Connects port 2 of the switch to the xHCI PCIe Endpoint: 0b: Port 2 connected to external PCIe Port. 1b: Port 2 connected to xHCI PCIe Endpoint. **Note:** Do not modify this register.
1	PCIE_SW_OBFF_WIRE_IN_EN	R/W	Connects the wake signal output from the PCIE_WAKE_N pin to the Wake(in) inputs of the PCIe Endpoints to enable OBFF signaling downstream: 0b: OBFF input disabled. 1b: OBFF input enabled.
0	PCIE_SW_BEACON_EN	R/W	Connects the wake signal outputs from the PCIe Endpoints to the plx_wake_oen inputs of the downstream ports of the switch to enable upstream Beaconing of wake signals: 0b: Beaconing disabled. 1b: Beaconing enabled.

The PCIE Switch CTL Register contents are shown in Table 8.

TABLE 8: PCIE_SW_CTL_REG

PCIE SWITCH CONFIGURATION REGISTER			Default: Depends on device
Bits	**Name**	**R/W**	**Description**
31:5	Reserved	R	Always 0's
4:0	PCIE_SW_PORT_ENABLE[3:0]	R/W	Defaults set by the hardware to indicate whether a port has been disabled for the PCIe Switch where bit 0 corresponds to port 1, bit 1 corresponds to Port 2 (Port 0 cannot be disabled), etc.: 0b: Port is disabled 1b: Port is enabled **Note:** The PCIE_SW_PORT_ENABLE[3:0] default values are set automatically by hardware depending on the device part number, but can be reconfigured from OTP/EEPROM/SMBus/SPI before system startup to override these defaults.

The PCIE Switch Test Register contents are shown in Table 9.

TABLE 9: PCIE_SW_TESTIN_REG

PCIE SWITCH TEST REGISTER			Default: 0x0000_0018
Bits	**Name**	**R/W**	**Description**
31:22	Reserved	R	Always read '0'
21	PCIE_SW_TESTIN_DIS_ PARITY_CHK R/	R/W	Disable 128b/130b SKP OS parity checking and reporting.
20	PCIE_SW_TESTIN_EN_ DC_BAL_WARN	R/W	Enable warnings when incorrect DC balance symbols are received in training sets.
19	Masked	R	Masked. Internal use only.
18:15	Reserved	R	Always read '0'
14	PCIE_SW_TESTIN_FILTER_ COMP_EXIT	R/W	Filter out exits from electrical in Polling.Compliance during rate change.
13	PCIE_SW_- TESTIN_DIS_EQ_REP	R/W	Disable reporting of equalization problems.
12	Reserved	R	Always read '0'
11	PCIE_SW_TESTIN_FORCE_ RX_DET	R/W	Force entry in Polling.Compliance from Polling.active.
10	PCIE_SW_TESTIN_FORCE_ COMPLIANCE	R/W	Disable entry in Polling.Compliance from Polling.active (does not apply if PCIE_SW_TESTIN_SET_COMP_ RX_BIT is set).
9	PCIE_SW_TESTIN_DIS_ COMPLIANCE	R/W	Disable entry in Polling.Compliance from Polling.active (does not apply if PCIE_SW_TESTIN_SET_COMP_ RX_BIT is set).
8	Reserved	R	Always read '0'
7	PCIE_SW_TESTIN_SET_ COMP_RX_BIT	R/W	Set compliance receive bit in transmitted TS1 ordered set.
6	PCIE_SW_ TESTIN_DIS_SCRAMBLING	R/W	Disable scrambling (Gen 1/Gen 2 modes only).

TABLE 9: PCIE_SW_TESTIN_REG (CONTINUED)

PCIE SWITCH TEST REGISTER			Default: 0x0000_0018
5	PCIE_SW_TESTIN_EN_ RXTX_ASSRT	R/W	Enable RX/TX performance throughput assertions.
4	PCIE_SW_TESTIN_EN_ INFO_ASSRT	R/W	Enable information assertions (Enabled by default).
3	PCIE_SW_ TESTIN_EN_WARN_ASSRT	R/W	Enable warning assertions (Enabled by default).
2	PCIE_SW_TESTIN_ LOOPBACK_MST	R/W	Loopback Lead: This signal must be set to 1 to direct the Link to loopback (in Loopback Lead mode). When the Core is in Loopback Lead mode, then it transmits a Modified Compliance Pattern in Loopback.Active.
1	PCIE_SW_ TESTIN_DIS_LPS_NEG	R/W	Disable Low-Power State Negotiation: When asserted, this signal disables all low-power state negotiation.
0	Masked	R	Masked. Internal use only.

The PCIE Switch Port HP Capability Register contents are shown in Table 10.

TABLE 10: PCIE_SW_PORT_HP_CAP_REG

PCIE SWITCH PORT HP CAPABILITY REGISTER			Default: 0x0008_0000
Bits	**Name**	**R/W**	**Description**
31:19	HP_CAP_PHYS_SLOT_ NUM[12:0]	R/W	Physical slot number. A physical slot is only supported on Port 1 of PCI1xxxx devices.
18	HP_CAP_NO_COMMAND_ COMPLETE	R/W	No Command Complete Support
17	HP_CAP_ELECTROMAG_ INTERLOCK	R/W	Electromechanical interlock present
16:15	HP_CAP_SLOT_ PWR_LIM_SCL[1:0]	R/W	Slot power limit scale
14:7	HP_CAP_SLOT_ PWR_LIM_VAL[7:0]	R/W	Slot power limit value
6	HP_CAP_CAPABLE	R/W	Hot-Plug capable: 0b - not capable 1b - capable
5	HP_CAP_SURPRISE	R/W	Hot-Plug surprise: 0b - not capable 1b - capable
4	HP_CAP_PWR_IND	R/W	Power indicator present 0b - not capable 1b - capable
3	HP_CAP_ATT_IND	R/W	Attention indicator present 0b - not capable 1b - capable
2	HP_CAP_MRL	R/W	MRL sensor present 0b - not capable 1b - capable
1	HP_CAP_PWR_CTL	R/W	Power controller present 0b - not capable 1b - capable

TABLE 10: PCIE_SW_PORT_HP_CAP_REG (CONTINUED)

PCIE SWITCH PORT HP CAPABILITY REGISTER			Default: 0x0008_0000
Bits	**Name**	**R/W**	**Description**
0	HP_CAP_ATT_BUTN	R/W	Attention button present 0b - not capable 1b - capable

The PCIE Switch Port Control Register contents are shown in Table 11.

TABLE 11: PCIE_SW_PORT_CTL_REG

PCIE SWITCH PORT CONTROL REGISTER			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:1	Reserved	R	Always read '0'
0	PCIE_SW_PORT_RESET	R/W	Per port Reset override for PCIe Ports This register is used to force the port into Reset. 0b: Port resets follow normal operation. 1b: Port Reset signals are asserted and port is forced into a Reset state. **Note:** Reset can be used to configure registers in the memory space PCIE_SWITCH_PORTx_ADDR_BASE where 'x' is the port number. This can be utilized for registers which may only be configured while the switch port is held in a Reset state. Usage is to set this bit by default from OTP or EEPROM, and then clear the bit once the relevant registers have been configured.

The PCIE Switch Port Vendor ID Register contents are shown in Table 12.

TABLE 12: PCIE_SW_PORT_CFG_VID_REG

PCIE SWITCH PORT VENDOR ID REGISTER			Default: 0x0000_1055
Bits	**Name**	**R/W**	**Description**
31:16	Reserved	R	Always read '0'.
15:0	PORT_PCI_VID[15:0]	R/W	PCI Vendor ID.

The PCIE Switch Port Device ID Register contents are shown in Table 13.

TABLE 13: PCIE_SW_PORT_CFG_DID_REG

PCIE SWITCH PORT DEVICE ID REGISTER			Default: Per Port
Bits	**Name**	**R/W**	**Description**
31:24	Reserved	R	Always read '0'.
23:16	PORT_PCI_RID[7:0]	R/W	PCI Revision ID
15:0	PORT_PCI_DID[15:0]	R/W	PCI Vendor ID

The PCIE Switch Port Debug Register contents are shown in Table 14.

TABLE 14: PCIE_SW_PORT_DBG_REG

PCIE SWITCH PORT DEBUG REGISTER			Default: 0xXXXX_XXXX
Bits	**Name**	**R/W**	**Description**
31:18	Reserved	R	Always read ‘0’.
17:16	DBG_EQU_PHASE[1:0]	R	Indicates the equalization phase when LTSSM is in the Recovery. • 00b: Phase 0 • 01b: Phase 1 • 10b: Phase 2 • 11b: Phase 3
15:13	Reserved	R	Always read ‘0’.
12:8	DBG_LINK_SPEED[4:0]	R	Reports PCIE link speed: • 0_0001b: 2.5 GT/s • 0_0010b: 5.0 GT/s • 0_0011b: 8.0 GT/s **Note:** This signal does not indicate that the link is up.
7:5	Reserved	R	Always read ‘0’.
4:0	DBG_LTSSM_STATE[4:0]	R	Reports the current LTSSM state: • 00: detect.quiet • 01: detect.active • 02: polling.active • 03: polling.compliance • 04: polling.configuration • 05h: config.linkwidthstart • 06h: config.linkwidthaccept • 07h: config.lanenumwait • 08h: config.lanenumaccept • 09h: config.complete • 0Ah: config.idle • 0Bh: recovery.receiverlock • 0Ch: recovery.equalization • 0Dh: recovery.speed • 0Eh: recovery.receiverconfig • 0Fh: recovery.idle • 10h: L0 • 11h: L0s • 12h: L1.entry • 13h: L1.idle • 14h: L2.idle • 15h: L2.transmitwake • 16h: disable • 17h: loopback.entry • 18h: loopback.active • 19h: loopback.exit • 1Ah: hotreset

The PCIE Switch Port PCI Configuration Space Register contents are shown in Table 15.

TABLE 15: PCIE_SW_PORT_PCI_CONFIG_SPACE

PCIE SWITCH PORT PCI CONFIGURATION SPACE			Default: Depends on Function
Bits	**Name**	**R/W**	**Description**
4095:0	PCIE_SW_PORT_PCI_CONFIG_SPACE	R/W	See Section Appendix B:, "PCI Configuration Space"

7.0 USB HOST IMPLEMENTATION

The USB Host Controller is available in the following devices:

- PCI11400
- PCI11414
- PCI11101

The USB Host Controller IDs are:

- Vendor ID: 0x1055 (Microchip Technology, Inc/SMSC)
- Device ID:
  - PCI12000: N/A (no USB)
  - PCI11010: N/A (no USB)
  - PCI11101: 0xA020
  - PCI11400: 0xA030
  - PCI11414: 0xA040

Generally the following items shall be configured by the end-system integrator:

- Max Port Speed: 10 Gb, 5 Gb, 480 Mb
- Port 1-4 Disable
- Port Type: Type-A or USB Type-C®
- USB Type-C Rp vSafe5V current capability
- Port Power Control Signals

7.1 Sections

7.2 Supported Port Combinations

The PCI11400 and PCI11414 are capable of implementing up to two (2) USB Type-C ports, with the remaining ports being USB Type-A.

When two USB Type-C ports at USB 3.2 Gen 2 speed are desired, then the PCI11400/PCI11414 transceivers must be remapped within their configuration registers, to associate the USB 3.2 transceivers with the correct USB Port index. See Table 16, Table 17, and Table 18 for the supported USB port combinations.

TABLE 16: PCI11400 & PCI11414 SUPPORTED USB PORT COMBINATIONS

Mode #	Physical Port 1	Physical Port 2	Physical Port 3	Physical Port 4
1	**Max Speed:** USB 3.2 Gen 2 **Connector Type:** Type-A **USB 2.0 PHY #:** 1 **USB 2.0 Logical Port Mapping:** Port 1 **USB 3.2 PHY #:** 1 **USB 3.2 Logical Port Mapping:** Port 1	**Max Speed:** USB 3.2 Gen 2 **Connector Type:** Type-A **USB 2.0 PHY #:** 2 **USB 2.0 Logical Port Mapping:** Port 2 **USB 3.2 PHY #:** 2 **USB 3.2 Logical Port Mapping:** Port 2	**Max Speed:** USB 2.0 **Connector Type:** Type-A **USB 2.0 PHY #:** 3 **USB 2.0 Logical Port Mapping:** Port 3	**Max Speed:** USB 2.0 **Connector Type:** Type-A **USB 2.0 PHY #:** 4 **USB 2.0 Logical Port Mapping:** Port 4
2	**Max Speed:** USB 2.0 **Connector Type:** Type-A **USB 2.0 PHY #:** 1 **USB 2.0 Logical Port Mapping:** Port 1	**Max Speed:** USB 2.0 **Connector Type:** Type-A **USB 2.0 PHY #:** 2 **USB 2.0 Logical Port Mapping:** Port 2	**Max Speed:** USB 2.0 **Connector Type:** Type-A **USB 2.0 PHY #:** 3 **USB 2.0 Logical Port Mapping:** Port 3	**Max Speed:** USB 2.0 **Connector Type:** Type-A **USB 2.0 PHY #:** 4 **USB 2.0 Logical Port Mapping:** Port 4
3	**Max Speed:** USB 3.2 Gen 2 **Connector Type:** Type-C **USB 2.0 PHY #:** 1 **USB 3.2 Logical Port Mapping:** Port 1 **USB 3.2 PHY #:** 1 **USB 3.2 Logical Port Mapping:** Port 1	**Max Speed:** USB 3.2 Gen 2 **Connector Type:** Type-C **USB 2.0 PHY #:** 2 **USB 2.0 Logical Port Mapping:** Port 2 **USB 3.2 PHY #:** 2 **USB 3.2 Logical Port Mapping:** Port 2	**Max Speed:** USB 2.0 **Connector Type:** Type-A **USB 2.0 PHY #:** 3 **USB 2.0 Logical Port Mapping:** Port 3	**Max Speed:** USB 2.0 **Connector Type:** Type-A **USB 2.0 PHY #:** 4 **USB 2.0 Logical Port Mapping:** Port 4

TABLE 16: PCI11400 & PCI11414 SUPPORTED USB PORT COMBINATIONS (CONTINUED)

Mode #	Physical Port 1	Physical Port 2	Physical Port 3	Physical Port 4
4	**Max Speed:** USB 2.0 **Connector Type:** Type-C **USB 2.0 PHY #:** 1 **USB 2.0 Logical Port Mapping:** Port 1	**Max Speed:** USB 2.0 **Connector Type:** Type-C **USB 2.0 PHY #:** 2 **USB 2.0 Logical Port Mapping:** Port 2	**Max Speed:** USB 2.0 **Connector Type:** Type-A **USB 2.0 PHY #:** 3 **USB 2.0 Logical Port Mapping:** Port 3	**Max Speed:** USB 2.0 **Connector Type:** Type-A **USB 2.0 PHY #:** 4 **USB 2.0 Logical Port Mapping:** Port 4
5	**Max Speed:** USB 3.2 Gen 2 **Connector Type:** Type-C **USB 2.0 PHY #:** 1 **USB 2.0 Logical Port Mapping:** Port 1 **USB 3.2 PHY #:** 1 + 2 **USB 3.2 Logical Port Mapping:** Port 1	**Max Speed:** USB 2.0 **Connector Type:** Type-A **USB 2.0 PHY #:** 2 **USB 2.0 Logical Port Mapping:** Port 2	**Max Speed:** USB 2.0 **Connector Type:** Type-A **USB 2.0 PHY #:** 3 **USB 2.0 Logical Port Mapping:** Port 3	**Max Speed:** USB 2.0 **Connector Type:** Type-A **USB 2.0 PHY #:** 4 **USB 2.0 Logical Port Mapping:** Port 4

TABLE 17: PCI11101 SUPPORTED USB PORT COMBINATIONS

Mode #	Physical Port 1
1	**Max Speed:** USB 3.2 Gen 2 **Connector Type:** Type-A **USB 2.0 PHY #:** 1 **USB 2.0 Logical Port Mapping:** Port 1 **USB 3.2 PHY #:** 1 **USB 3.2 Logical Port Mapping:** Port 2
2	**Max Speed:** USB 2.0 **Connector Type:** Type-A **USB 2.0 PHY #:** 1 **USB 2.0 Logical Port Mapping:** Port 1

TABLE 18: SUPPORTED PORT COMBINATIONS

Mode	Port 1	Port 2	Port 3	Port 4
1	A			
	A	A		
	A	A		
	A	A	A	
	A	A	A	A
2	A			
	A	A		
	A	A		
	A	A		
3	C			
	C	C		
	C	A		
	C	A	A	
	C	A	A	A

Legend:

Green = USB 2 Blue = USB 3 Yellow = USB 3 with External Mux

TABLE 18: SUPPORTED PORT COMBINATIONS (CONTINUED)

Mode	Port 1	Port 2	Port 3	Port 4
4	C			
	C	C		
	C	A		
	C	A	A	
	C	A	A	A
5	C			
	C	A		
	C	A	A	
	C	A	A	A
	C	A		
	C	A	A	
	C	A	A	A
	C	C		
	C	C	A	
	C	C	A	A

Legend:

Green = USB 2 Blue = USB 3 Yellow = USB 3 with External Mux

7.3 Required Sideband Signals for USB Ports

7.3.1 USB TYPE-A PORTS

The ports which will be implemented as USB Type-A require that these corresponding signals are muxed out of the available fixed function and programmable (PROG) function pins for connection to external USB system circuitry. See Table 19 for the list of USB Type-A Sideband Signals.

TABLE 19: USB TYPE-A SIDEBAND SIGNALS

Function	Pin Availability	Description	External Connection
VB_PRT_CTL_P1/ VB_OCS_P1_N	**PROG70 - Function 8 (PCI11400 default)** **PROG81 - Function 8 (PCI11414 default)**	Combined port power controller/load switch and overcurrent fault sense pin for USB Port X. Internal pull-up resistor is enabled. This is a PROG function signal.	Connects to external 5V USB port power switch's Enable pin, and wire-ORed with external 5V USB port power switch's active-low FAULT# or equivalent overcurrent flag pin.
VB_PRT_CTL_P2/ VB_OCS_P2_N	**PROG71 - Function 9 (PCI11400 default)** **PROG82 - Function 9 (PCI11414 default)**		
VB_PRT_CTL_P3/ VB_OCS_P3_N	**PROG80 - Function 5 (PCI11400 default)** **PROG83 - Function 8 (PCI11414 default)**		
VB_PRT_CTL_P4/ VB_OCS_P4_N	**PROG81 - Function 9 (PCI11400 default)** **PROG84 - Function 1 (PCI11414 default)**		

Note: By default, PCI11101 does not configure a VB_PRT_CTL_P1/VB_OCS_P1_N pin for it's signal available USB 2.0 port. If a port control pin is needed within the PCI11101 application, either PROG70 or PROG81 must be selected via configuration (EEPROM or OTP).

7.3.2 PORT CONTROL I/O MODES

The VB_PRT_CTL_Px signals operate differently as shown in Table 20.

TABLE 20: USB TYPE-C SIDEBAND SIGNALS

Function	Port Disabled	Port Enabled
Pin Direction	Output	Input
Internal Pull-Up Enable	Disabled	Enabled
Push/Pull Assertion	Drive Low	No Assertion
System Objective	Force VBUS supply to shut off.	Enable VBUS supply via internal pull-up resistor and monitor input status for overcurrent (OCS) events from an open-drain Fault indicator output originating from the VBUS supply circuitry.

7.3.3 OVERCURRENT DETECTION

The following describes the operation of the filter block. See also Figure 2 and Figure 3 for the connection diagrams.

- OCS events are ignored when port power is not enabled.
- OCS event are ignored for the first time based on USB_OCS_REG.START_LOCKOUT_TIMER[7:0] value after
- Port power is turned on. Do not set START_LOCKOUT_TIMER[7:0] to zero.
- If an OCS event occurs which is less than the USB_OCS_REG.OC_TIMER[1:0] value, it will be ignored.
- Valid OCS conditions are:
  - USB_OCS_REG.OCS_INACT[7:0] = 0. It is registered as OCS condition (single pulse OCS after USB_OCS_REG.OC_TIMER[1:0] expires).
  - USB_OCS_REG.OCS_MIN_WIDTH[6:0] = 0. It is registered as OCS condition (single pulse OCS after USB_OCS_REG.OC_TIMER[1:0] expires).
  - USB_OCS_REG.OCS_INACT[7:0] != 0. Two or more OCS event occur in same inactive period. If a second OCS does not occur in the same ocs_inact period, the first event is ignored.
  - If the OCS pin is low for more than USB_OCS_REG.OCS_MIN_WIDTH[6:0] value, it is registered as an OCS condition (single long width OCS event).

FIGURE 2: CONNECTION DIAGRAM TYPE-A USB 2.0

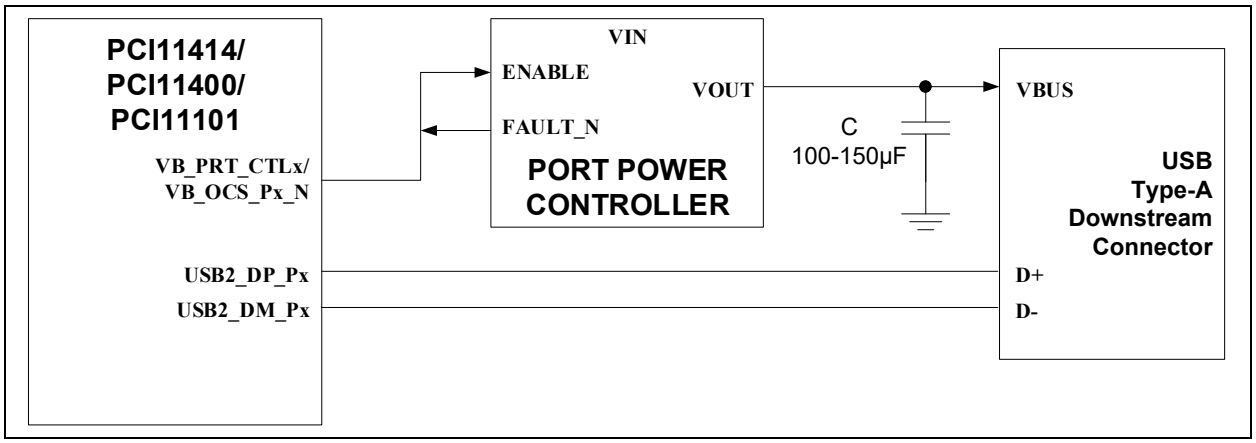
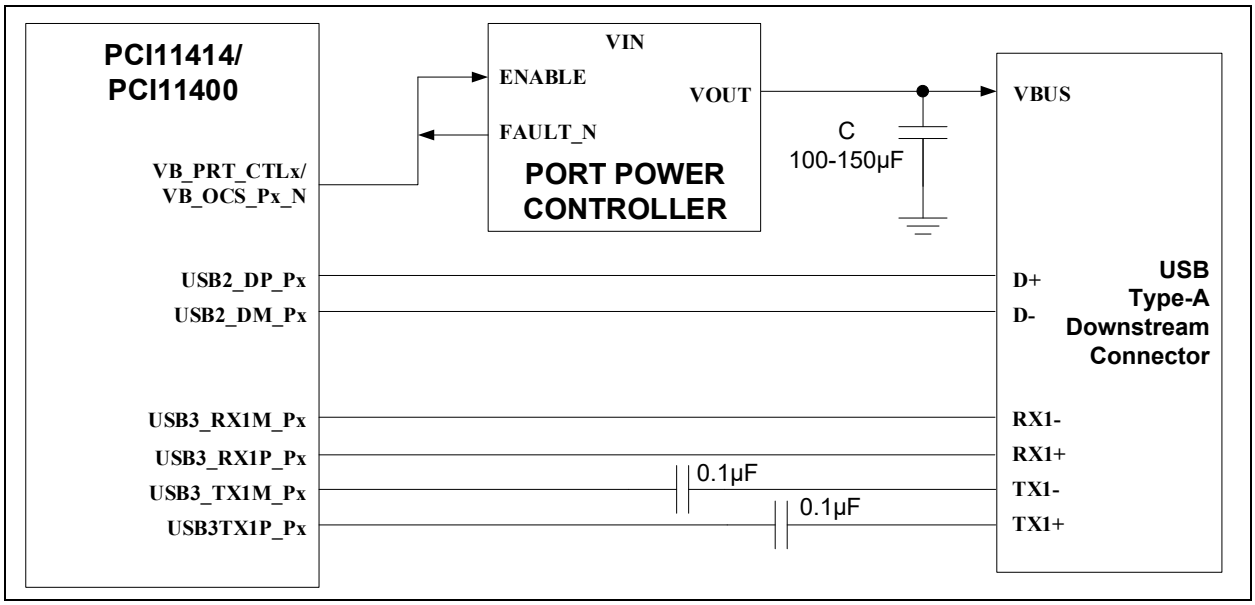


FIGURE 3: CONNECTION DIAGRAM TYPE-A USB 3.2



7.3.4 ADDITIONAL SIGNALS REQUIRED FOR USB TYPE-C® PORTS

PCI1xxxx supports up to two USB Type-C Ports. Any ports which will be implemented as USB Type-C also require that these corresponding signals are muxed out of one of the available programmable (PROG) function pins for connection to external USB system circuitry. See Table 21 and Table 22 for the details of sideband signals and Figure 4, Figure 5, and Figure 6 for the connection diagrams.

USB Type-C ports must also include VB_PRT_CTL_Px/VB_OCS_Px_N mentioned in Section 7.3.1.

Note:	PCI11101 does not support USB Type-C ports. If a Type-C port is required in a PCI11101 application, an external USB Type-C port controller is required.

TABLE 21: USB TYPE-C® FIXED FUNCTION SIDEBAND SIGNALS

Function	Pin Mapping	Description	External Connection
CC1_P1	PCI11414 - Pin 114 PCI11400 - Pin 90	Configuration Channel USB Type-C® signal used to detect the presence of a valid Type-C port end-to-end attach, and also communicate to the attached device the supported maximum amperage of VBUS on the port. An attach on CC1 indicates an 'unflipped' Type-C orientation and that the active USB 3.2 communication pins on the Type-C connector are TX1+/- and RX1+/-.	Connect directly to the CC1 pin of the USB Type-C connector.
CC1_P2	PCI11414 - Pin 103 PCI11400 - Pin 79		

TABLE 21: USB TYPE-C® FIXED FUNCTION SIDEBAND SIGNALS (CONTINUED)

Function	Pin Mapping	Description	External Connection
CC2_P1	PCI11414 - Pin 115 PCI11400 - Pin 91	Configuration Channel Type-C signal used to detect the presence of a valid Type-C port end-to-end attach, and also communicate to the attached device the supported maximum amperage of VBUS on the port. An attach on CC2 indicates an 'flipped' Type-C orientation and that the active USB 3.2 communication pins on the Type-C connector are TX2+/- and RX2+/-.	Connect directly to the CC2 pin of the USB Type-C connector.
CC2_P2	PCI11414 - Pin 104 PCI11400 - Pin 80		
VBUS_MON_P1	PCI11414 - Pin 129 PCI11414 - Pin 105	Analog input pin for monitoring VBUS, for compliant USB Type-C behavior. Ensures external 5V USB port power switch will not become enabled when VBUS is already energized by another source. This is a fixed function signal.	Connects to external VBUS voltage divider circuit, at the output of the external 5V USB port power switch.
VBUS_MON_P2	PCI11414 - Pin 132 PCI11400 - Pin 108		

TABLE 22: USB TYPE-C® PROGRAMMABLE SIDEBAND SIGNALS

Function	Function Pin Availability	Description	External Connection
VBUS_DIS_P1	**PROG76 - Function 7 (PCI11414 & PCI11400 default)**	Digital output pin for enabling external "VBUS discharge" circuit in sync with USB Type-C® port state and ensure the receptacle power is "cold" (<800 mV) prior to next connection. This is a PROG function signal.	Connects to external discrete VBUS discharge circuit OR no-connect if external 5V USB port power switch is USB Type-C compatible (auto-discharge on deassertion of Enable).
VBUS_DIS_P2	PROG51 - Function 4 PROG75 - Function 7 PROG86 - Function 2		
VCONN1_EN_P1	**PROG77 - Function 7 (PCI11414 default)**	Digital output pin for enabling an external VCONN source onto CC1 of USB Type-C receptacle, when a USB Type-C "active cable" is connected to the port. This is a PROG function signal.	Connects to external VCONN source circuit which is capable of providing a minimum of 1 Watt to a USB Type-C "active cable", enabled exclusively onto USB Type-C CC1 pin.
VCONN1_EN_P2	PROG23 - Function 3 PROG48 - Function 6		
VCONN2_EN_P1	**PROG78 - Function 7 (PCI11414 default)**	Digital output pin for enabling external VCONN source onto CC2 of USB Type-C receptacle, when a USB Type-C "active cable" is connected to the port. This is a PROG function signal.	Connects to external VCONN source circuit which is capable of providing a minimum of 1 Watt to a USB Type-C "active cable", enabled exclusively onto USB Type-C CC2 pin.
VCONN2_EN_P2	PROG24 - Function 3 PROG49 - Function 6		
CC_ORIENT_P1	PROG6 - Function 3 PROG20 - Function 8 PROG68 - Function 7 PROG71 - Function 12	Optional Signal for signaling detected Type-C orientation when implementing USB Type-C port with an external mux. This is a PROG function signal.	Connects to the orientation selection input of an external muxing device.
CC_ORIENT_P2	PROG47 - Function 6 PROG86 - Function 1 PROG87 - Function 8		

TABLE 22: USB TYPE-C® PROGRAMMABLE SIDEBAND SIGNALS (CONTINUED)

Function	Function Pin Availability	Description	External Connection
CC_ATTACH_P1	PROG5 - Function 3 PROG19 - Function 8 PROG69 - Function 11 PROG70 - Function 11	Optional Signal for signaling detected Type-C end-to-end attach when implementing USB Type-C port with an external mux. This is a PROG function signal.	Connects to the enable pin input of an external muxing device.
CC_ATTACH_P2	PROG46 - Function 6 PROG85 - Function 1 PROG86 - Function 8		

FIGURE 4: CONNECTION DIAGRAM TYPE-C USB 2.0

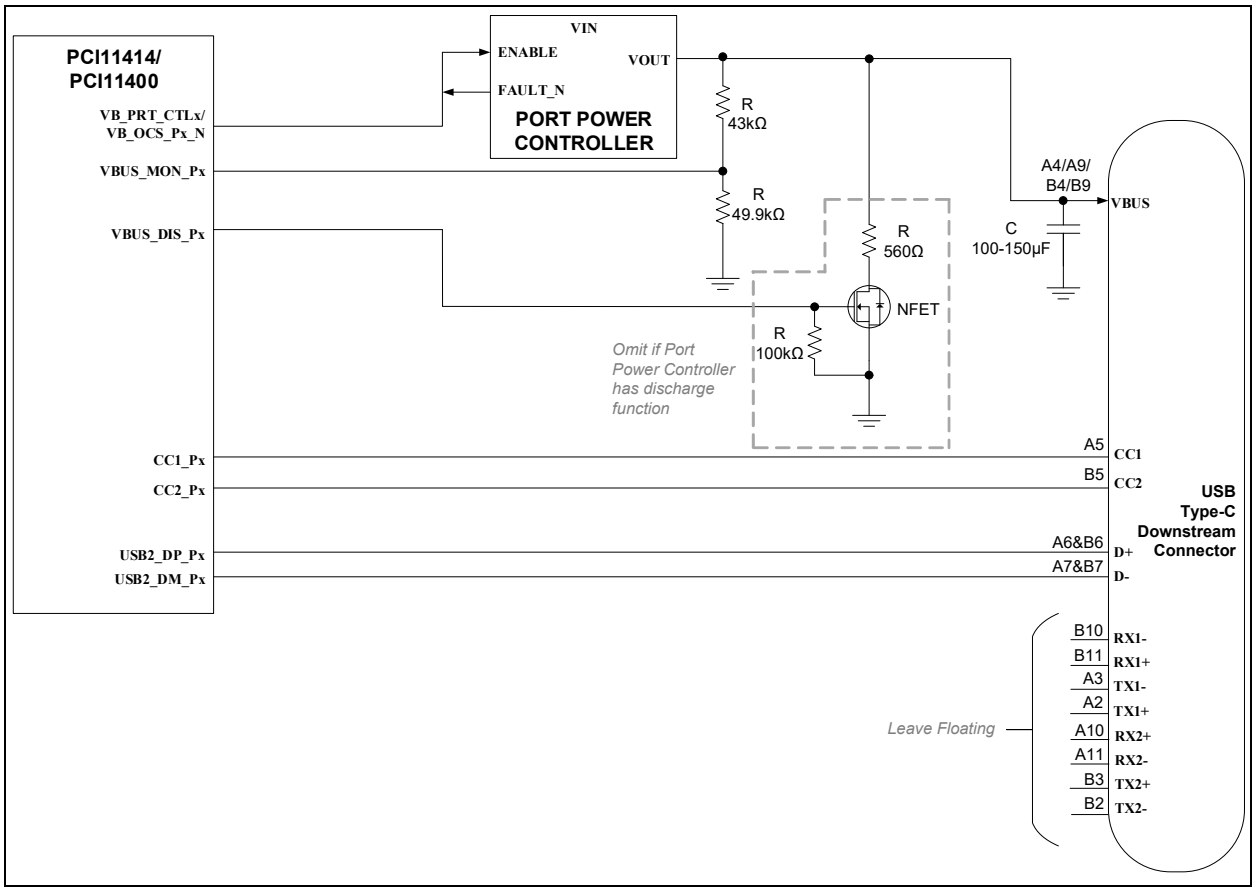


FIGURE 5: CONNECTION DIAGRAM TYPE-C USB 3.2, INTERNAL MUX

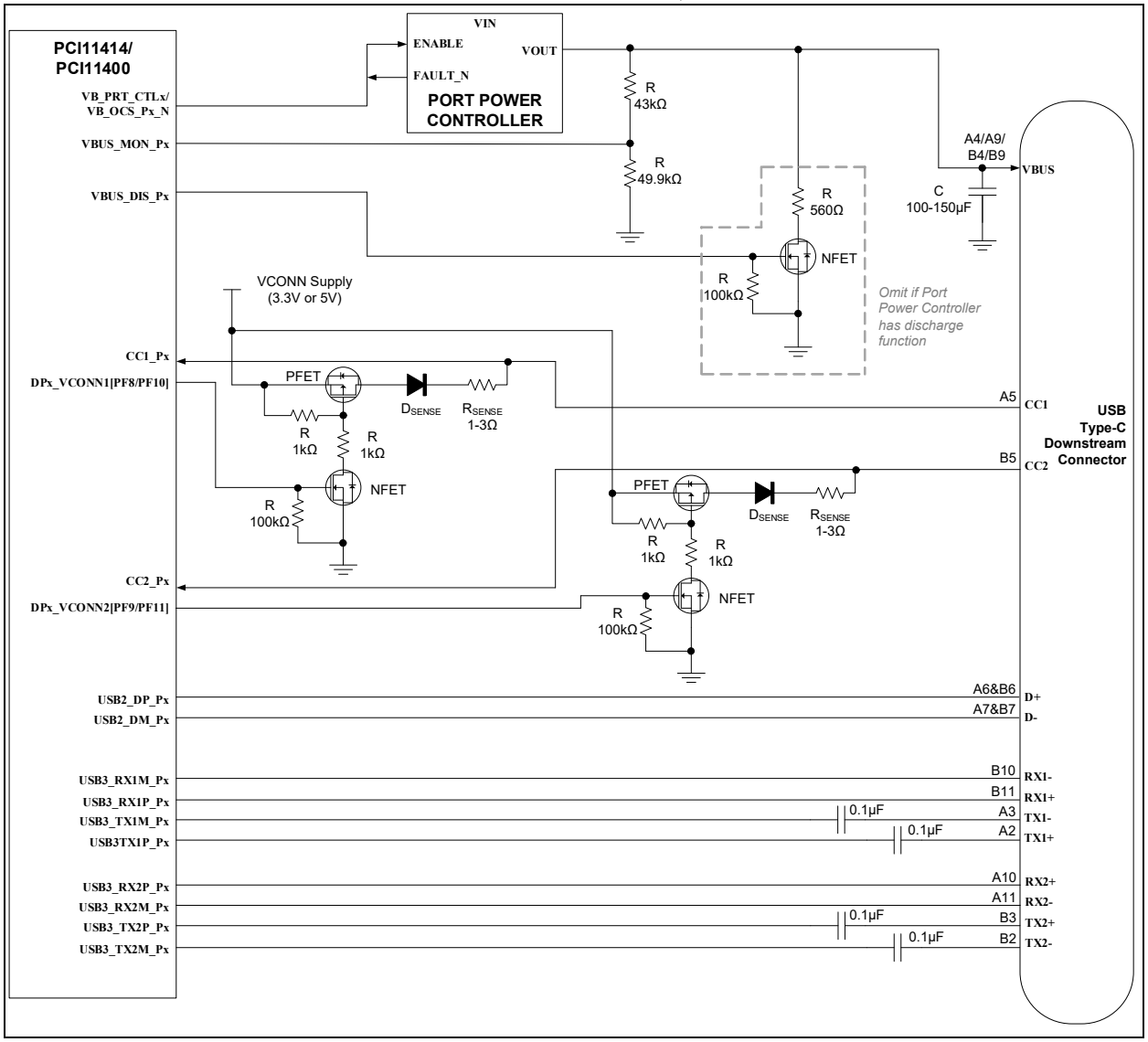
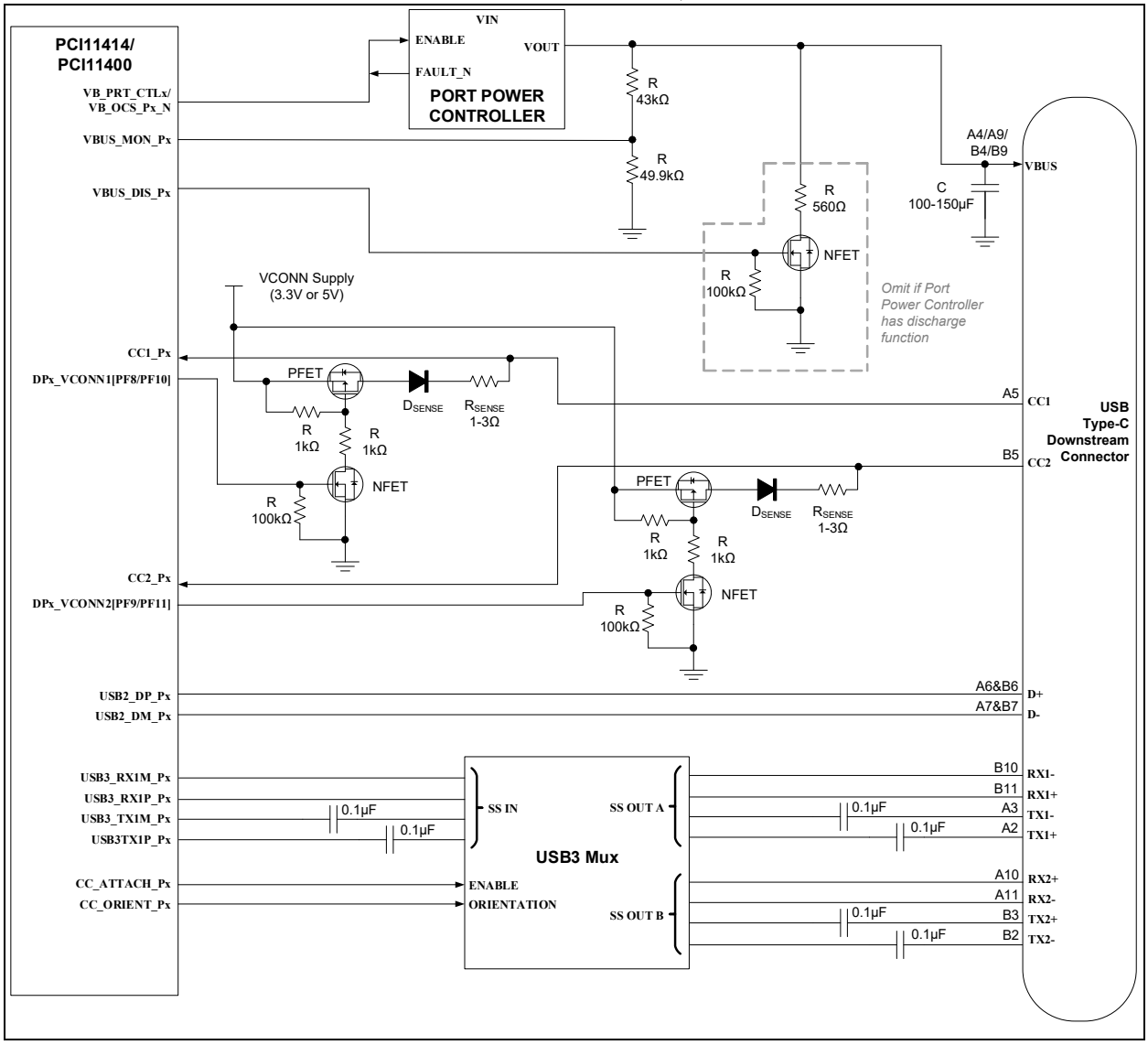


FIGURE 6: CONNECTION DIAGRAM TYPE-C USB 3.2, EXTERNAL MUX



7.4 USB Subsystem Registers

The USB_SUBSYSTEM_ADDR_BASE = 0x0004_0000 registers are listed in Table 23.

Note: PCI1xxxx devices have thousands of registers which are either reserved for future use or the contents are masked from public documentation because they have no end system integrator use-case (i.e. they are used by hardware to perform low level run-time tasks or are used/controlled directly by a driver during run-time). **Do not modify any contents of undocumented registers**.

TABLE 23: USB SUBSYSTEM REGISTER OFFSETS FROM USB_SUBSYSTEM_ADDR_BASE

Register Name	Port 1	Port 2	Port 3	Port 4
USB2_TEST_MODE	0x1_6810			
USB2_TEST_PORT_SEL	0x1_6812			
USB2_AFE_CTRL	0x1_80C9	0x1_84C9	0x1_88C9	0x1_8CC9
USB2_HS_CTRL	0x1_80CA	0x1_84CA	0x1_88CA	0x1_8CCA
USB2_TEST_EN	0x1_80CB	0x1_84CB	0x1_88CB	0x1_8CCB
PHY_UP_REG	0x1_80CC	0x1_84CC	0x1_88CC	0x1_8CCC
LTSSM_STATE	0x1_81C0	0x1_85C0	**n/a**	**n/a**
PORT_CFG_SEL_x	0x1_6004	0x1_6008	0x1_600C	0x1_6010
USB_OCS_SEL_x	0x1_6020	0x1_6024	0x1_6028	0x1_602C
PORT_STAT_REG_x	0x1_6054	0x1_6058	0x1_605C	0x1_6060
USB_OCS_REG_x	0x1_6070			
CC_CONFIG_Px	0x1_A220	0x1_A620	n/a	n/a
VCONN1_PAD_CTL_Px	0x1_A228	0x1_A628	n/a	n/a
VCONN2_PAD_CTL_Px	0x1_A229	0x1_A629	n/a	n/a
DISCHARGE_PAD_CTL_Px	0x1_A22A	0x1_A62A	n/a	n/a
ATTACH_PAD_CTL_Px	0x1_A22B	0x1_A62B	n/a	n/a
ORIENT_PAD_CTL_Px	0x1_A22C	0x1_A62C	n/a	n/a
USB_RESET_REG	0x1_B000			
USB_PORT_ENABLE_REG	0x1_B004			

The USB Test Mode Register details are shown in Table 24.

TABLE 24: USB2_TEST_MODE

USB 2 Test Mode Register			**Default:** 0x00
Bits	**Name**	**R/W**	**Description**
7:4	Reserved	R	Always 0's
3:0	TEST_MODE[3:0]	R/W	0000b: Normal operation mode 0001b: USB 2 PHY Characterization mode 0010b: UTMI BIST mode 0011b - 0111b: Reserved

The USB Test Mode Port Selection Register details are shown in Table 25.

TABLE 25: USB2_TEST_PORT_SEL

USB Test Mode Port Selection Register			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:3	Reserved	R	Always 0's
2:0	DFP_SEL[2:0]	R/W	000b: Reserved 001b: Test mode on Port 1 010b: Test mode on Port 2 011b: Test mode on Port 3 100b: Test mode on Port 4 101b - 111b: Reserved

The USB 2 Analog Front End Control details are shown in Table 26.

TABLE 26: USB2_AFE_CTRL

USB 2 Analog Front End Control			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7	Reserved	R	Always 0
6	USB2_FS_OEB	R/W	FS/LS Driver Output Enable Active-low. 1b0: Driver enabled 1b1: Driver tri-stated
5	USB2_FS_VPO	R/W	FS/LS Positive Output Data Drives data to the DP output
4	USB2_FS_VMO	R/W	FS/LS Negative Output Data Drives data to the DM output
3	USB2_FS_DATA	R	FS/LS Differential Receiver Input State
2:0	Reserved	R	Always 0's

The USB 2 High-Speed AFE Control Register details are shown in Table 27.

TABLE 27: USB2_HS_CTRL

USB 2 High-Speed AFE Control Register			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:6	USB2_HS_TXVALID[1:0]	R/W	HS Transmit Valid Mask. Indicates which bits of the AFE_HS_TXVALID[1:0] bus is valid. • 00b: No bits valid • 01b: LSB valid • 10b: Invalid combination • 11b: Both bits valid
5:4	USB2_HS_TXDATA[1:0]	R/W	HS Transmit Data. Driver data is transmitted LSB first.
3	USB2_HS_CS_EN	R/W	HS Current Source Enable. • 0b: Driver powered-down • 1b: Driver powered-up This signal will be asserted active whenever the port is in High-Speed mode.

TABLE 27: USB2_HS_CTRL (CONTINUED)

USB 2 High-Speed AFE Control Register			Default: 0x00
Bits	**Name**	**R/W**	**Description**
2:0	USB2_HS_DRIVE[2:0]	R/W	HS Output Current. (PHY Boost) • 000b: Nominal 17.78mA • 001b: Decrease by 5% • 010b: Increase by 10% • 011b: Increase by 5% • 100b: Increase by 20% • 101b: Increase by 15% • 110b: Increase by 30% • 111b: Increase by 25%

The USB2 Test Register details are shown in Table 28.

TABLE 28: USB2_TEST_EN

USB 2 Test Register			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7	USB2_RPU_DP_EN	R/W	RPU DP Termination Control. Enable the 1.5 kohm termination on DP when active.
6	USB2_RPU_DM_EN	R/W	RPU DM Termination Control. Enable the 1.5 kohm termination on DM when active.
5	USB2_RPD_DP_EN	R/W	RPD DP Termination Control. Enable the 15 kohm termination on DP when active.
4	USB2_RPD_DM_EN	R/W	RPD DM Termination Control. Enable the 15 kohm termination on DM when active.
3:2	Reserved	R	Always 0's
1	USB2_HS_TXACTIVE	R/W	HS Driver Active. Places the HS driver in Low-power mode when disabled (during IDLE). afe_hs_cs_en must be enabled. 0b: Driver in Low-power mode 1b: Driver in Active Transmit mode (17.78 mA source active)
0	Reserved	R	Always 0

The USB 2 PHY Adjust Register details are shown in Table 29.

TABLE 29: PHY_UP_REG

USB 2 PHY Adjust Register			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:6	USB2_HS_DISC_TUNE[1:0]	R/W	USB 2.0 AFE HS Disconnect Tune. Connects to USB 2.0 PCS (UTMI). • 00b: Nominal 575 mV Trip Point • 01b: Increase by 50 mV • 10b: Increase by 100 mV • 11b: Increase by 125 mV

TABLE 29: PHY_UP_REG (CONTINUED)

USB 2 PHY Adjust Register			Default: 0x00
Bits	**Name**	**R/W**	**Description**
5	USB2_HS_TERM_EN	R/W	HS Termination Control. Enable the 45 kohm termination on DP and DM when active.
4	USB2_HS_SQUELCH_B	R	AFE High-Speed Squelch Indicates when the HS oversampled data is valid. Active-low. • 0b: Data valid • 1b: Data is invalid
3	USB2_HS_DISC	R	AFE High-Speed Disconnect Indicates when the line is disconnected in HS mode. This signal should only be strobed during HS EOP on the 32nd bit time. • 0b: Normal condition • 1b: Disconnect condition
2:0	USB2_SQ_TUNE[2:0]	R/W	Squelch Tune (Varisense) • 000b: Nominal 100 mV Trip Point • 001b: Decrease by 12.5 mV • 010b: Decrease by 25 mV • 011b: Decrease by 37.5 mV • 100b: Decrease by 50 mV • 101b: Decrease by 62.5 mV • 110b: Increase by 25 mV • 111b: Increase by 12.5 mV

The USB 3 LTSSM State Register details are shown in Table 30.

TABLE 30: LTSSM_STATE

USB 3 LTSSM State Register			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:4	LTSSM_STATE[3:0]	R	LTSSM state 0000b = U0 (no substates) 0001b = U1 (no substates) 0010b = U2 (no substates) 0011b = U3 (no substates) 0100b = SS.Disabled (see substate below) 0101b = Rx.Detect (see substate below) 0110b = SS.Inactive (see substate below) 0111b = Polling (see substate below) 1000b = Recovery (see substate below) 1001b = HotReset (see substate below) 1010b = Compliance (no substates) 1011b = Loopback (no substates)

TABLE 30: LTSSM_STATE (CONTINUED)

USB 3 LTSSM State Register			Default: 0x00
Bits	**Name**	**R/W**	**Description**
3:0	LTSSM_SUB_STATE[3:0]	R	**U0 substates:** none **U1 substates:** 0x0 U1 POWER = PHY Power state P0 -> P1 request 0x1 U1_POWER_A = Wait for PHY power change done 0x2 U1_ACTIVE = Wait for remote/local U1 exit 0x3 U1_EXIT_LOC_RESP = Start U1 exit and wait for min. response from remote partner 0x4 U1_EXIT_LOC_FIN = Locally initiated U1 exit; send min. required U1 exit 0x5 U1_EXIT_REM = Remote initiated U1 exit; send min. required U1 exit 0x6 U1_EXIT_P0 = PHY Power state P1 -> P0 request 0x7 U1_EXIT_P0_A = Wait for PHY power change P1 -> P0 done 0x8 U1_EXIT_DONE = U1 exit successful; U1 -> Recovery **U2 substates:** 0x0 U2_POWER = PHY Power state P0 -> P2 request 0x1 U2_POWER2 = Wait for PHY power change done 0x2 U2_ACTIVE = Wait for remote/local U2 exit 0x3 U2_ACTIVE_0 = Request to start receiver detection 0x4 U2_ACTIVE_1 = Wait for Receiver detection response 0x5 U2_EXIT_LOC_RESP = Start U2 exit and wait for min. response from remote partner 0x6 U2_EXIT_LOC_FIN = Locally initiated U2 exit; send min. required U2 exit 0x7 U2_EXIT_REM = Remote initiated U2 exit; send min. required U2 exit 0x8 U2_EXIT_P0 = PHY Power state P2 -> P0 request 0x9 U2_EXIT_P0_A = Wait for PHY power change P1 -> P0 done 0xA U2_EXIT_DONE = U2 exit successful; U2 -> Recovery **U3 substates:** 0x0 U3_POWER = PHY Power state P0 -> P2/P3 request 0x1 U3_POWER3 = Wait for PHY power change done 0x2 U3_ACTIVE = Wait for remote/local U3 exit 0x3 U3_ACTIVE_0 = Request to start receiver detection 0x4 U3_ACTIVE_1 = Wait for receiver detection response 0x5 U3_EXIT_LOC_POW = PHY Power state P3 -> P0 -> P2 request 0x6 U3_EXIT_LOC_POW_A = Wait for PHY power change done 0x7 U3_EXIT_LOC_RESP = Start U3 exit and wait for min. response from remote partner 0x8 U3_EXIT_LOC_FIN = Locally initiated U3 exit; send min. required U3 exit 0x9 U3_EXIT_REM_POW = PHY Power state P3 -> P0 -> P2 request 0xA U3_EXIT_REM_POW_A = Wait for PHY power change done 0xB U3_EXIT_REM = Remote initiated U3 exit; send min. required U3 exit

TABLE 30: LTSSM_STATE (CONTINUED)

USB 3 LTSSM State Register			Default: 0x00
Bits	**Name**	**R/W**	**Description**
			P2/P3 -> P0 request 0xD U3_EXIT_P0_A = Wait for PHY power change P2/P3 -> P0 done 0xE U3_EXIT = Received remote/local U3 exit 0xF U3_EXIT_DONE U3 = Exit successful; U3 -> Recovery

The Port Config Select Register details are shown in Table 31.

TABLE 31: PORT_CFG_SEL_x

PORT CONFIG SELECT REGISTER			Default: 0x0000_0803
Bits	**Name**	**R/W**	**Description**
31:23	Reserved	R	Always 0's
22	ORIENTATION_POLARITY	R/W	Controls the polarity of the CC_ORIENT_Px output signal pin for external mux control. 0b: CC_ORIENT_Px asserts when CC wire is detected upon CC1 1b: CC_ORIENT_Px asserts when CC wire is detected upon CC2
21	CC_ENABLE	R/W	Enables USB Type-C detection logic. 0b: USB Type-C detection logic disabled 1b: USB Type-C detection logic enabled **Note:** Only applicable to Port 1 and/or Port 2.
20:18	Reserved	R	Always 0's
17:16	VBUS_INPUT_SEL[1:0]	R/W	Selects the VBUS detection logic for USB Type-C ports. 00b: Reserved 01b: Enable VBUS_MON_Px input 10b: Reserved 11b: Reserved **Note:** Only valid when PORT_TYPE[1:0] =2b'01 (configured as USB Type-C). Only applicable to Port 1 and/or Port 2.
15	MUX_EN	R/W	Enables the internal USB 3.2 Gen 2 mux. 0b: Internal USB 3.2 Gen 2 mux disabled 1b: Internal USB 3.2 Gen 2 mux enabled **Note:** Only valid when PORT_TYPE[1:0] =2b'01 (configured as USB Type-C). Only applicable to Port 1 and/or Port 2.
14	Reserved	R	Always 0

TABLE 31: PORT_CFG_SEL_x (CONTINUED)

PORT CONFIG SELECT REGISTER			Default: 0x0000_0803
Bits	**Name**	**R/W**	**Description**
13:12	C_MUX_SEL[1:0]	R/W	Selects how internal USB 3.2 Gen 2 mux is controlled by internal USB Type-C detection logic. 00b: Port type is USB Type-A (No internal muxing) 01b: Reserved 10b: Port is USB 2.0-only on USB Type-C (No internal muxing) 11b: Port is USB 3.2 Gen 2 on USB Type-C (Internal muxing to be controlled by USB Type-C attach and orientation states) **Note:** Only valid when MUX_EN=1 & PORT_TYPE [1:0]=2b'01 (configured as USB Type-C). Only applicable to Port 1 and/or Port 2.
11	C_SEL_BnA	R/W	Indicates the internal mux selection state of USB 3.2 Gen 2 PHY 0/1 0b: USB 3.2 Gen 2 PHY 0 is selected 1b: USB 3.2 Gen 2 PHY 1 is selected **Note:** This bit is set by the internal USB Type-C detection logic, when C_MUX_SEL[1:0]=2b'01 or 2b'11, otherwise returns 0 when read. Only applicable to Port 1 and/or Port 2.
10	C_ATTACH	R/W	Indicates the Attach state from USB Type-C detection logic 0b: Nothing attached to the USB Type-C port 1b: Port is in Attached state **Note:** Only valid when MUX_EN=1 & PORT_ TYPE[1:0]=2b'01 (configured as USB Type-C). Only applicable to Port 1 and/or Port 2
9:8	PORT_TYPE[1:0]	R/W	Selects the USB port type (USB Type-A or USB Type-C) 00b: USB Type-A 01b: USB Type-C 10b: Reserved 11b: Reserved
7:4	Reserved	R	Always 0’s
3:0	PRT_SEL[1:0]	R/W	Configures the internal USB port power control logic 0000b: USB port is disabled, no port power control. 0001b: Reserved 0010b: Reserved 0011b: USB port is USB 2.0 or USB 3.2 Gen 2, port power control logic enabled. 0100b - 1111b: Reserved

The OCS Config Select Register details are shown in Table 32.

TABLE 32: USB_OCS_SEL_x

OVER-CURRENT SIGNAL (OCS) CONFIG SELECT REGISTER			**Default:** 0x0000_0001
Bits	**Name**	**R/W**	**Description**
31:4	Reserved	R	Always 0's
3:0	PRT_OCS_SEL[1:0]	R/W	Selects the signal source for the USB port's overcurrent (OCS) state. 0000b: The USB port is disabled 0001b: OCS is signaled by the port's VB_PRT_CTL_Px/ VB_OCS_Px_N pin 0010b - 1111b: Reserved

The Port Status Register details are shown in Table 33.

TABLE 33: PORT_STAT_REG_x

PORT STATUS REGISTER			**Default:** 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:2	Reserved	R	Always read '0'
1	A_ATTACH_STATUS	R	The current status of the Type-A port attach status
0	C_ATTACH_STATUS	R	The current status of the Type-C port attach status

The Port Overcurrent Status Register details are shown in Table 34.

TABLE 34: USB_OCS_REG_x

PORT OVERCURRENT STATUS REGISTER			**Default:** 0x0014_050A
Bits	**Name**	**R/W**	**Description**
31:26	Reserved	R	Always read '0'
25:24	OC_TIMER[1:0]	R/W	Overcurrent Timer delay. This measures the minimum pulse width for which a pulse is considered valid. 00b = 3 Clock samples 01b = 6 Clock samples 10b = 12 Clock Samples 11b = 24 Clock Samples
23:16	OCS_INACT[7:0]	R/W	This timer counts the inactivity time of second OCS event detection time within which second OCS event will be detected. Configurable Values from 0 - 255 (0x00 - 0xFF) Default value is 20 ms.
15:8	OCS_MIN_WIDTH[6:0]	R/W	OCS Pulse window provides a range from 0 ms to 5 ms.
7:0	START_LOCKOUT_TIMER[7:0]	R/W	This timer blocks the OCS event being detected after port power is on. Any OCS event within this timer value is ignored. Current default value is 10 ms.

The Type-C CC Configuration Register details are shown in Table 35.

TABLE 35: CC_CONFIG_Px

TYPE-C CC CONFIGURATION REGISTER			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:8	Reserved	R	Always read '0'
7	CONFIG_DONE	R	Set by HW after all the internal registers of the CC logic have been initialized.
6	Reserved	R/W	Always 0
5	ERR_RECOVER	R/W	This bit controls whether the Type-C FSM shall attempt to auto recover from an OCS condition sources from either the Port Power Switch or VCONN FET. Setting this bit is not recommended, as ports should be controlled directly via USB Host drivers.
4:2	Reserved	R/W	Always 0's
1:0	PWR_CAP[1:0]	R/W	When operating as a standalone source signals define the charging current supported by the device. 00b: USB 2.0 Default Current 01b: USB 3.0 Default Current 10b: 1.5A 11b: 3.0A

The VCONN1 Pad Control Register details are shown in Table 36.

TABLE 36: VCONN1_PAD_CTL_Px

VCONN1 PAD CONTROL REGISTER			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:4	Reserved	R	Always read '0'
3	CC1_VCONN_PF_POL	R/W	0b: Pin is set high when VCONN1 is enabled and set low when VCONN1 is disabled. 1b: Pin is set low when VCONN1 is enabled and set high when VCONN1 is disabled.
2	CC1_VCONN_PF_OD	R/W	0b: Open-drain operation disabled 1b: When the corresponding output is a one, the output is disabled, and the pull-up is enabled. When the output is a zero, the output is enabled and is driven low. **Note:** that the corresponding VCONN1_EN_Px pin must be configured as open-drain.
1	CC1_VCONN_PF_PD	R/W	0b: Internal pull-down resistor disabled on VCONN1_EN_Px pin 1b: Internal pull-down resistor enabled on VCONN1_EN_Px pin
0	CC1_VCONN_PF_PU	R/W	0b: Internal pull-up resistor disabled on VCONN1_EN_Px pin 1b: Internal pull-up resistor enabled on VCONN1_EN_Px pin

The VCONN2 Pad Control Register details are shown in Table 37.

TABLE 37: VCONN2_PAD_CTL_PX

VCONN2 PAD CONTROL REGISTER			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:4	Reserved	R	Always read '0'

TABLE 37: VCONN2_PAD_CTL_PX (CONTINUED)

VCONN2 PAD CONTROL REGISTER			Default: 0x00
Bits	**Name**	**R/W**	**Description**
3	CC2_VCONN_PF_POL	R/W	0b: Pin is set high when VCONN2 is enabled and set low when VCONN2 is disabled. 1b: Pin is set low when VCONN2 is enabled and set high when VCONN2 is disabled.
2	CC2_VCONN_PF_OD	R/W	0b: Open-drain operation disabled 1b: When the corresponding output is a one, the output is disabled, and the pull-up is enabled. When the output is a zero, the output is enabled and is driven low. **Note:** that the corresponding VCONN2_EN_Px pin must be configured as open-drain.
1	CC2_VCONN_PF_PD	R/W	0b: Internal pull-down resistor disabled on VCONN2_EN_Px pin 1b: Internal pull-down resistor enabled on VCONN2_EN_Px pin
0	CC2_VCONN_PF_PU	R/W	0b: Internal pull-up resistor disabled on VCONN2_EN_Px pin 1b: Internal pull-up resistor enabled on VCONN2_EN_Px pin

The VBus Discharge Pad Control Register details are shown in Table 38.

TABLE 38: DISCHARGE_PAD_CTL_PX

VBUS DISCHARGE PAD CONTROL REGISTER			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:4	Reserved	R	Always read ‘0’
3	DISCHARGE_POL	R/W	0b: Pin is set high when VBUS Discharge is enabled and set low when VBUS Discharge is disabled. 1b: Pin is set low when VBUS Discharge is enabled and set high when VBUS Discharge is disabled.
2	DISCHARGE_OD	R/W	0b: Open-drain operation disabled 1b: When the corresponding output is a one, the output is disabled, and the pull-up is enabled. When the output is a zero, the output is enabled and is driven low. **Note:** that the corresponding VBUS_DIS_Px pin must be configured as open-drain.
1	DISCHARGE_PD	R/W	0b: Internal pull-down resistor disabled on VBUS_DIS_Px pin 1b: Internal pull-down resistor enabled on VBUS_DIS_Px pin
0	DISCHARGE_PU	R/W	0b: Internal pull-up resistor disabled on VBUS_DIS_Px pin 1b: Internal pull-up resistor enabled on VBUS_DIS_Px pin

The USB-C Attach Pad Control Register details are shown in Table 39.

TABLE 39: ATTACH_PAD_CTL_PX

USB-C ATTACH PAD CONTROL REGISTER			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:4	Reserved	R	Always read ‘0’

TABLE 39: ATTACH_PAD_CTL_PX (CONTINUED)

USB-C ATTACH PAD CONTROL REGISTER			Default: 0x00
Bits	**Name**	**R/W**	**Description**
3	CC_ATTACH_POL	R/W	0b: Pin is set high when a valid end-to-end USB-C attach is detected and set low when no valid USB-C attach is detected. 1b: Pin is set low when a valid end-to-end USB-C attach is detected and set high when no valid USB-C attach is detected.
2	CC_ATTACH_OD	R/W	0b: Open-drain operation disabled 1b: When the corresponding output is a one, the output is disabled, and the pull-up is enabled. When the output is a zero, the output is enabled and is driven low. **Note:** that the corresponding CC_ATTACH_Px pin must be configured as open-drain.
1	CC_ATTACH_PD	R/W	0b: Internal pull-down resistor disabled on CC_ATTACH_Px pin 1b: Internal pull-down resistor enabled on CC_ATTACH_Px pin
0	CC_ATTACH_PU	R/W	0b: Internal pull-up resistor disabled on CC_ATTACH_Px pin 1b: Internal pull-up resistor enabled on CC_ATTACH_Px pin

The USB-C Orient Pad Control Register details are shown in Table 40.

TABLE 40: ORIENT_PAD_CTL_PX

USB-C ATTACH PAD CONTROL REGISTER			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:4	Reserved	R	Always read ‘0’
3	CC_ORIENT_POL	R/W	0b: Pin is set high when a valid end-to-end USB-C attach is detected on CC2 and set low when a valid end-to-end USB-C attach is detected on CC1 or no valid USB-C attach is detected. 1b: Pin is set low when a valid end-to-end USB-C attach is detected on CC2 and set high when a valid end-to-end USB-C attach is detected on CC1 or no valid USB-C attach is detected.
2	CC_ORIENT_OD	R/W	0b: Open-drain operation disabled 1b: When the corresponding output is a one, the output is disabled, and the pull-up is enabled. When the output is a zero, the output is enabled and is driven low. **Note:** that the corresponding CC_ORIENT_Px pin must be configured as open-drain.
1	CC_ORIENT_PD	R/W	0b: Internal pull-down resistor disabled on CC_ORIENT_Px pin 1b: Internal pull-down resistor enabled on CC_ORIENT_Px pin
0	CC_ORIENT_PU	R/W	0b: Internal pull-up resistor disabled on CC_ORIENT_Px pin 1b: Internal pull-up resistor enabled on CC_ORIENT_Px pin

The USB Reset Register details are shown in Table 41.

TABLE 41: USB_RESET_REG

USB RESET REGISTER			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:10	Reserved	R	Always 0's
9	USB_RELOAD_PCI	R/W	Defines the configuration to be carried out on the USB PCIe Endpoint when USB_RELOAD is set; this corresponds to the USB_PCIE_EP_ADDR_BASE. 0b: USB PCIe Endpoint configuration will not be reloaded to USB_PCIE_EP_ADDR_BASE. The System Config HW will attempt to write all configuration registers associated with a particular configuration; the Subsystem will block any writes corresponding to the PCIe Endpoint at USB_PCIE_EP_ADDR_BASE. 1b: the USB PCIe Endpoint configuration will be reloaded to USB_PCIE_EP_ADDR_BASE.
8	USB_RELOAD_EN	R/W	Defines the configuration to be carried out on the USB function IP when USB_RELOAD is set; USB function IP corresponds to all Subsystem addresses in USB_SUBSYSTEM_ADDR_BASE, except for those corresponding USB_PCIE_EP_ADDR_BASE. which are controlled by the USB_RELOAD_PCI bit. 0b: the USB function IP configuration will not be reloaded. The System Config HW will attempt to write all configuration registers associated with a particular configuration; the Subsystem will block any writes corresponding to the function IP. 1b: the USB function IP configuration will be reloaded.
7:4	Reserved	R	Always 0's
3	USB_HOST_RESET	R/W	Holds USB host controller in Reset. 0b: USB Host controller not held in Reset 1b: USB Host controller held in Reset
2	USB_RELOAD	R/W/SC	Initiate USB reload.
1	USB_LRST	R/W/SC	Initiate light function Reset of the entire USB subsystem except for the PCIe endpoint. All registers are Reset.
0	USB_SRST	R/W/SC	Initiate Soft Reset to reset the entire USB subsystem, including the PCIe endpoint. All registers are Reset.

The USB Port Enable Register details are shown in Table 42.

TABLE 42: USB_PORT_ENABLE_REG

USB Port Enable Register			Default: PCI11400/PCI11414: 0x0204_030F PCI11101: 0x0001_0001 Others: N/A
Bits	**Name**	**R/W**	**Description**
31:28	Reserved	R	Always 0's

TABLE 42: USB_PORT_ENABLE_REG (CONTINUED)

USB Port Enable Register			Default: PCI11400/PCI11414: 0x0204_030F PCI11101: 0x0001_0001 Others: N/A
Bits	**Name**	**R/W**	**Description**
27:24	HOST_NUM_U3_PORT[3:0]	R/W	Number of USB 3.2 ports For PCI11400 & PCI11414, the valid value of these bits are from 0000b (no ports) to 0010b (2 ports). For PCI11101, either 0000b (no ports) or 0001b (1 port) is valid. **Note:** The USB Host Controller needs to be Reset after any change to the HOST_NUM_U3_PORT[3:0] field in order to apply the change.
23:20	Reserved	R	Always 0's
19:16	HOST_NUM_U2_PORT[3:0]	R/W	Number of USB 2.0 ports For PCI11400 & PCI11414, the valid value of these bits are from 0001b (1 port) to 0100b (4 ports). For PCI11101, only 0001b is valid. These bits shall never be set to 0000b. At least one USB 2.0 port is required. **Note:** The USB Host Controller needs to be Reset after any change to the HOST_NUM_U2_PORT[3:0] field in order to apply the change.
15:10	Reserved	R	Always 0's
9:8	USB_U3_PORT_ENABLE[1:0]	R/W	Set by the hardware to indicate whether a USB 3 port has been enabled.Changes to these bits must be made in configuration and cannot be made live during run-time. Bit[0] corresponds to Port 1, Bit[1] corresponds to Port 2, Bit[2] corresponds to Port 3, Bit[3] corresponds to Port 4 0b: Port is disabled 1b: Port is enabled This signal, when '1', stops reporting connect/disconnect events on the port. This could be used for security reasons where hardware can disable a port irrespective of whether xHCI driver enables a port or not. **Note:** Port enabling is configured based on the device part number, but can be over-ridden by Configuration settings. The host_u2_port_disable is then used to disable the relevant port on the xHCI controller.
7:4	Reserved	R	Always 0's

TABLE 42: USB_PORT_ENABLE_REG (CONTINUED)

USB Port Enable Register			Default: PCI11400/PCI11414: 0x0204_030F PCI11101: 0x0001_0001 Others: N/A
Bits	**Name**	**R/W**	**Description**
3:0	USB_U2_PORT_ENABLE[3:0]	R/W	Set by the hardware to indicate whether a USB 2 port has been enabled.Changes to these bits must be made in configuration and cannot be made live during run-time. Bit[0] corresponds to Port 1, Bit[1] corresponds to Port 2, Bit[2] corresponds to Port 3, Bit[3] corresponds to Port 4 0b: Port is disabled 1b: Port is enabled This signal, when '1', stops reporting connect/disconnect events on the port. This could be used for security reasons where hardware can disable a port irrespective of whether xHCI driver enables a port or not. **Note:** Port enabling is configured based on the device part number, but can be over-ridden by Configuration settings. The host_u2_port_disable is then used to disable the relevant port on the xHCI controller.

7.5 USB Port Configuration Examples

7.5.1 DISABLE UNUSED PORTS

The following examples in Table 43-Table 46 shows the configuration details of PCI11101 to PCI11414 disable USB ports.

TABLE 43: PCI11414 DISABLE USB PORT 4

#	Description	Notes	Register Address	Value
1	Disable Port 4 in USB_PORT_ENABLE_REG	HOST_NUM_U3_PORT[3:0] = 10b HOST_NUM_U2_PORT[3:0] = 11b USB_U3_PORT_ENABLE[3:0] = 0011b USB_U2_PORT_ENABLE[3:0] = 0111b	BASE ADDRESS + 0x1_B004	0x0203_0307

TABLE 44: PCI11414 DISABLE USB PORT 3 & 4

#	Description	Notes	Register Address	Value
1	Disable Port 3 & 4 in USB_PORT_ENABLE_REG	HOST_NUM_U3_PORT[3:0] = 10b HOST_NUM_U2_PORT[3:0] = 10b USB_U3_PORT_ENABLE[3:0] = 0011b USB_U2_PORT_ENABLE[3:0] = 0011b	BASE ADDRESS + 0x1_B004	0x0202_0303

TABLE 45: PCI11414 DISABLE USB PORT 2, 3, & 4

#	Description	Notes	Register Address	Value
1	Disable Port 2, 3 & 4 in USB_PORT_ENABLE_REG	HOST_NUM_U3_PORT[3:0] = 01b HOST_NUM_U2_PORT[3:0] = 01b USB_U3_PORT_ENABLE[3:0] = 0001b USB_U2_PORT_ENABLE[3:0] = 0001b	BASE ADDRESS + 0x1_B004	0x0101_0101

TABLE 46: PCI11414 DISABLE ALL PORTS (DISABLE USB HOST CONTROLLER)

#	Description	Notes	Register Address	Value
1	Disable USB Host controller by disabling the PCIe port in PCIe Switch configuration	USB_Enable = 1'b0	BASE ADDRESS + 0x00E0	Set Per CPN Default Register Configuration

7.5.2 SUPPORTED PORT MODE COMBINATION

1. PCI11414/PCI11400 Set to Port Configuration Mode 1 Example

The various supported combinations of USB port configurations are described in full in Section 7.2, "Supported Port Combinations". In summary, Port Configuration Mode 1 is:

- Port 1: USB 3.2 Gen 2 Type-C Port with internal USB 3 data mux
- Port 2: USB 2.0 Type-A Port
- Port 3: USB 2.0 Type-A Port
- Port 4: USB 2.0 Type-A Port

TABLE 47: PCI11414/PCI11400 SET TO PORT CONFIGURATION 1

#	Description	Notes	Register Address	Value
1	Set Port 1 as a Type-C USB 3.2 Gen 2 Port with internal USB-C data mux in PORT_CFG_SEL_x	CC_ENABLE = 1b to enable Type-C logic VBUS_INPUT_SEL[1:0] = 01b to enable VBUS voltage monitoring pin MUX_EN = 1b to enable internal Type-C mux C_MUX_SEL[1:0] = 11b to select Type-C internal mux PORT_TYPE[1:0] = 01b to select USB Type-C Port Mode	BASE ADDRESS + 0x5_6004	0x0000_0003
2	Multiplex Port 1 PRT_CTL to a pin	GPIO70 Drive_Strength[1:0]=2b00 to drive at 2mA SELECT[3:0]= 4b1000	Base Address + 0x0118	0x0000_0008

TABLE 47: PCI11414/PCI11400 SET TO PORT CONFIGURATION 1 (CONTINUED)

#	Description	Notes	Register Address	Value
3	Multiplex Port 1 CC1 to a pin	GPIO5 Drive_Strength[1:0]=2b00 to drive at 2mA SELECT[3:0]= 4b0011	Base Address + 0x0014	0x0000_0003
4	Multiplex Port 1 CC2 to a pin	GPIO46 Drive_Strength[1:0]=2b00 to drive at 2mA SELECT[3:0]= 4b0110	Base Address + 0x00B8	0x0000_0006
5	Multiplex Port 1 VCONN1 to a pin	GPIO48 Drive_Strength[1:0]=2b00 to drive at 2mA SELECT[3:0]= 4b0110	Base Address + 0x00C0	0x0000_0006
6	Multiplex Port 1 VCONN2 to a pin	GPIO49 Drive_Strength[1:0]=2b00 to drive at 2mA SELECT[3:0]= 4b0110	Base Address + 0x00C4	0x0000_0006
7	Multiplex Port 1 VBUS_ DISCHARGE to a pin	GPIO76 Drive_Strength[1:0]=2b00 to drive at 2mA SELECT[3:0]= 4b0111	Base Address + 0x0130	0x0000_0007
8	Multiplex Port 1 VBUS_MON to a pin	CC Attach 1 must be enabled. If CPN is 11400 or 11414 this pin will be routed to CC1[2]	N/A	N/A
9	Set Port 2 as a Type-A USB 2.0 Port		PCIE_SWITCH _ADDR_BASE + 0x0000_0008	0x0803
10	Multiplex Port 2 PRT_CTL to a pin	GPIO71 Drive_Strength[1:0]=2b00 to drive at 2mA SELECT[3:0]= 4b1001	Base Address + 0x011C	0x0000_0009
11	Set Port 3 as a Type-A USB 2.0 Port		PCIE_SWITCH _ADDR_BASE + 0x0000_000C	0x0803
12	Multiplex Port 3 PRT_CTL to a pin	GPIO83 Drive_Strength[1:0]=2b00 to drive at 2mA SELECT[3:0]= 4b1000	Base Address + 0x014C	0x0000_0008
13	Set Port 4 as a Type-A USB 2.0 Port		PCIE_SWITCH _ADDR_BASE + 0x0000_0010	0x0803
14	Multiplex Port 4 PRT_CTL to a pin	GPIO81 Drive_Strength[1:0]=2b00 to drive at 2mA SELECT[3:0]= 4b1001	Base Address + 0x0144	0x0000_0009

2. PCI11414/PCI11400 Set to Port Configuration Mode 2

The various supported combinations of USB port configurations are described in full in Section 7.2, "Supported Port Combinations". In summary, Port Configuration Mode 2 is:

- Port 1: USB 3.2 Gen 2 Type-C Port with external USB 3 data mux
- Port 2: USB 3.2 Gen 2 Type-C Port with external USB 3 data mux
- Port 3: USB 2.0 Type-A Port
- Port 4: USB 2.0 Type-A Port

3. PCI11414/PCI11400 Set to Port Configuration Mode 3

The various supported combinations of USB port configurations are described in full in Section 7.2, "Supported Port Combinations". In summary, Port Configuration Mode 3 is:

- Port 1: USB 3.2 Gen 2 Type-A Port
- Port 2: USB 3.2 Gen 2 Type-A Port
- Port 3: USB 2.0 Type-A Port
- Port 4: USB 2.0 Type-A Port

4. PCI11414/PCI11400 Set to Port Configuration Mode 4

The various supported combinations of USB port configurations are described in full in Section 7.2, "Supported Port Combinations". In summary, Port Configuration Mode 4 is:

- Port 1: USB 2.0 Type-C Port
- Port 2: USB 2.0 Type-C Port
- Port 3: USB 2.0 Type-A Port
- Port 4: USB 2.0 Type-A Port

5. PCI11414/PCI11400 Set to Port Mode 5

The various supported combinations of USB port configurations are described in full in Section 7.2, "Supported Port Combinations". In summary, Port Configuration Mode 5 is:

- Port 1: USB 2.0 Type-A Port
- Port 2: USB 2.0 Type-A Port
- Port 3: USB 2.0 Type-A Port
- Port 4: USB 2.0 Type-A Port

7.5.3 SETTING TYPE-C RP VALUE

1. Set Port 1 Type-C port to 3A

- Modify PWR_CAP[1:0] of Port 1CC_CONFIG_Px register

BASE ADDRESS + 0x1_A220 = 0x0000_0003

2. Set Port 2 Type-C port to 3A

- Modify PWR_CAP[1:0] of Port 2 CC_CONFIG_Px register

BASE ADDRESS + 0x1_A620 = 0x0000_0003

3. Set Port 1 Type-C port to 1.5A

- Modify PWR_CAP[1:0] of Port 1 CC_CONFIG_Px register

BASE ADDRESS + 0x1_A220 = 0x0000_0002

4. Set Port 2 Type-C port to 1.5A

- Modify PWR_CAP[1:0] of Port 2 CC_CONFIG_Px register

BASE ADDRESS + 0x1_A620 = 0x0000_0002

8.0 ETHERNET IMPLEMENTATION

The PCI1xxxx provides a PCIe to Ethernet MAC which requires a connection to an external Ethernet PHY. PCI1xxxx also supports:

- PCIe 3.1a Gigabit Ethernet Controller at 1 Lane at 5GT/s
- PCIe L1.1 and L1.2 substates and D3 hot and cold
- RGMII v1.3 and v2.0
- SGMII
- 1000/2500BASE-X connect to external Ethernet PHY. SGMII/1000/2500BASE-X pins are shared with RGMII RX pins
- Filtering and Wake on LAN
- Multiple TX and RX channels
- Microsoft RSS
- 1588v2 and 1588v2.1

The PCI1xxxx Ethernet IDs are:

- Vendor ID: 0x1055 (Microchip Technology, Inc/SMSC)
- Device ID:
  - PCI11010: 0xA011
  - PCI11414: 0xA041
- Class ID: 0x020000 (“Ethernet Controller”)

8.1 Sections

- Section 8.2, "Ethernet MAC Address"
- Section 8.3, "EXTERNAL PHY RESET"
- Section 8.4, "Receive Filtering Engine (RFE)"
- Section 8.5, "DMA Controller (DMAC)"
- Section 8.6, "Ethernet Interface Registers"

8.2 Ethernet MAC Address

All devices ship with a blank MAC Address. In order to operate within a network, each PCI1xxxx needs to be configured with a unique MAC Address.

Generally, blocks of MAC addresses are purchased from the IEEE. This ensures that the MAC address is entirely unique. Visit the IEEE Registration Authority for more details (https://standards.ieee.org/products-programs/regauth/)

Another option is to pass a MAC address from a host PC to the PCI1xxxx via the driver.

To configure the MAC address, the following registers must be set:

- MAC_RX_ADDRH
- MAC_RX_ADDRL

8.3 EXTERNAL PHY RESET

A dedicated ENET_PHY_RESET_N pin automatically resets the external Ethernet PHY. This pin automatically asserts as part of the Reset sequence. The PHY_RESET_N may also be asserted by an Ethernet PHY Software Reset.

8.4 Receive Filtering Engine (RFE)

The RFE receives Ethernet frames from the Ethernet MAC, processes them, and passes them to the RX FCT. The RFE is responsible for filtering the received Ethernet frames, verifying the TCP/UDP/ICMP/IGMP and IP checksum, and removing the VLAN tag.

When receiving a frame from the MAC, the RFE will obtain the frame data and status information. Upon completion of frame processing, the RFE encapsulates its status with the status information obtained from the MAC and passes this information (along with the frame data) on to the FCT in the form of RX Command A, RX Command B, RX Command C, and RX Command D. The RFE also passes the receive timestamp from the 1588 module to the FCT.

The RFE, if enabled, can remove a VLAN tag from the frame. VLAN tag stripping is controlled by the Enable VLAN Tag Stripping bit of the Receive Filtering Engine Control Register (RFE_CTL). If this bit is set, the tag will be stripped. If clear, the RFE will not modify the frame in any way.

Note: If multiple VLAN tags are present in a frame, the RFE only removes the first tag (adjacent to the MAC source address).

The RFE provides the Layer 3 Checksum (if enabled) and VLAN ID via RX Command B, while RX Command A, RX Command C and RX Command D contain the frame's status. When the RFE determines a frame has a checksum error, it sets the appropriate error bits in RX Command A to identify the error condition.

Note: The FCT does not rewind frames that failed checksum validation from the FCT RX FIFO.

The RFE also determines the correct RX FCT channel in which to place the frame, based on various priorities methods or MAC source or destination address or based on the Microsoft Receive Side Scaling specification. In order to minimize delays through the RFE, data is processed on the fly and stored into all FIFOs in parallel. Once a determination is made as to the correct destination FIFOs, the other FIFOs are instructed to drop the packet.

8.5 DMA Controller (DMAC)

The DMA Controller (DMAC) consists of independent receive (RX) and transmit (TX) DMACs, a series of arbiters, and a control and status register space (CSRs). The TX DMAC transfers Ethernet frames from host memory to the FIFO Controller (FCT), while the RX DMAC transfers Ethernet frames from the FCT to host memory.

Both the RX and TX DMACs have independent channels allocated to them (4 RX, 4 TX), through which the data transfer occurs. Both the RX and TX DMACs utilize descriptors to efficiently move data from source to destination with minimal CPU intervention. Descriptors are data structures in host memory that inform the DMAC of the location of data buffers in host memory. In the case of the RX DMAC, it also provides a mechanism for communicating status to the CPU on completion of DMA transactions. The host is responsible for setting up the descriptor rings and allocating RX descriptor buffers. TX descriptor buffers are allocated and placed into the ring as needed. Each channel has its own descriptor ring and data buffers. Descriptors are cached on chip to help absorb host bus latency.

The DMAC can be programmed to assert an interrupt for situations such as frame transmit or receive transfer completed, and other conditions.

8.6 Ethernet Interface Registers

The ENET_SUBSYSTEM_ADDR_BASE = 0x000C_0000 registers are listed in Table 48.

Note: PCI1xxxx devices have thousands of registers which are either reserved for future use or the contents are masked from public documentation because they have no end system integrator use-case (i.e.: they are used by hardware to perform low level run-time tasks or are used/controlled directly by a driver during run-time). **Do not modify any contents of undocumented registers**.

TABLE 48: ETHERNET REGISTER OFFSETS FROM ENET_SUBSYSTEM_ADDR_BASE

Register Name	Address
ID_REV	0x0000
VLAN_TYPE	0x0008
HW_CFG	0x0010
PMT_CTL	0x0014
DP_SEL	0x0024
DP_CMD	0x0028
DP_ADDR	0x002C
DP_DATA_0	0x0030
DP_DATA_1	0x0034
DP_DATA_2	0x0038

TABLE 48: ETHERNET REGISTER OFFSETS FROM ENET_SUBSYSTEM_ADDR_BASE (CONTINUED)

Register Name	Address
DP_DATA_3	0x003C
PCIE_CFG_ADDR	0x0048
PCIE_CFG_DATA	0x004C
MAC_CR	0x0100
MAC_RX	0x0104
MAC_TX	0x0108
MAC_FLOW	0x010C
MAC_RAND_SEED	0x0110
MAC_ERR_STS	0x0114
MAC_RX_ADDRH	0x0118
MAC_RX_ADDRL	0x011C
MII_ACCESS	0x0120
MII_DATA	0x0124
MII_RGMII_ID	0x0128
EEE_TX_LPI_REQUEST_DELAY_CNT	0x0130
WUCSR2	0x0600
RFE_CTL	0x0508
DMAC_CFG	0x0C00
DMAC_CMD	0x0C0C
DMAC_INT_SYS	0x0C10
SGMII_CTL	0x0728
INT_SYS	0x0780
INT_SET	0x0784
INT_EN_SET	0x0788
INT_EN_CLR	0x078C

The Device ID and Revision Register details are shown in Table 49.

TABLE 49: ID_REV

Device ID and Revision Register			**Default:** Depends on device/revision
Bits	**Name**	**R/W**	**Description**
31:16	CHIP_ID	R	Device ID. The following are the only valid values for this register: PCI11010 = 0xA011 PCI11414 = 0xA041
15:0	CHIP_REV	R	Device Revision 0x00A0 for Revision A0 silicon, and so on.

The VLAN Type Register details are shown in Table 50.

TABLE 50: VLAN_TYPE

VLAN Type Register			Default: 0x0000_8100
Bits	**Name**	**R/W**	**Description**
31:16	Reserved	R	Always 0's
15:0	VLAN_ETHERNET_TYPE	R/W	Contains the value of the Ethernet type when VLAN tag insertion is enabled. 0x8100 is the expected value. Proprietary VLAN tagging may be supported by changing the value of this parameter.

The Hardware Configuration Register details are shown in Table 51.

TABLE 51: HW_CFG

Hardware Configuration Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:20	Reserved	R	Always 0's
19	RST_PROTECT_PCIe	R/W	Reset Protection PCIe Configuration Setting this bit protects select fields of certain PCIe configuration space registers from being affected by resets other than POR_N. Reset protection also prohibits loading of the default values from the device level configuration loader. This field is only Reset by POR_N.
18	FLR_DIS	R/W	Function Level Reset Disable This bit, along with D3 VAUX Override (D3_VAUX_OVR), modifies the subsystem Reset operation when a Function Level Reset is received. When this bit is set and either the VAUX_DET pin is a high or D3_VAUX_OVR is set, only the function-specific portion of the PCIe configuration interface is Reset. Otherwise, the entire function-specific portion of the subsystem is Reset. The non-function-specific portion of the PCIe interface (endpoint controller, registers, link, etc.) is never Reset on an FLR.
17	EEE_ PHY_LINK_CHANGE_ SPEED_UP	R/W	EEE PHY Link Up Speed Up When set, the PHY link status needs to be "up" for 200 uS in order to request and decode LPI. When cleared, the PHY link status needs to be up for one second. **Note:** Software should only change this field when the Energy Efficient Ethernet Enable (EEEEN) field in the MAC Control Register (MAC_CR) is cleared. This field is protected by Reset Protection (RST_PROTECT).

TABLE 51: HW_CFG (CONTINUED)

Hardware Configuration Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
16	EEE_TIMER_SPEED_UP	R/W	EEE Timer Speed Up When set, the EEE timers and counters that normally are measured in µs will run at the speed of their clock tree as follows: EEE_TX_LPI_REQUEST_DELAY_CNT - MAC TX clock period (3.2 ns/8 ns/40 ns) EEE_TX_LPI_AUTO_REMOVAL_DELAY_CNT - MAC TX clock period (3.2 ns/8 ns/40 ns) EEE RX LPI Time - crystal clock period EEE TX LPI Time - crystal clock period Software should only change this field when the Energy Efficient Ethernet Enable (EEEEN) field in the MAC Control Register (MAC_CR) is cleared.
15	HOT_RESET_DIS	R/W	Hot Reset Disable This bit, along with D3 VAUX Override (D3_VAUX_OVR), modifies the subsystem Reset operation when a Hot Reset is received. When this bit is set and either the VAUX_DET pin is a high or D3_VAUX-_OVR is set, only the PCIe interface and related logic are Reset. Otherwise, the full subsystem is Reset.
14	D3_VAUX_OVR	R/W	D3 VAUX Override This bit along with Function Level Reset Disable (FLR_DIS), modifies the subsystem Reset operation when a Function Level Reset is received. This bit along with Hot Reset Disable (HOT_RESET_DIS), modifies the subsystem Reset operation when a Hot Reset is received. When FLR_DIS or HOST_RESET_DIS is set, this bit determines if VAUX_DET is considered in the Reset operation. When FLR_DIS and HOST_RESET_DIS are clear, this bit has no effect.
13	D3_RESET_DIS	R/W	D3Cold Reset Disable This bit modifies the subsystem Reset operation when exiting D3Cold (PERST# deasserts). When this bit is set and the VAUX_DET pin is a high only, the PCIe interface, and related logic are Reset. Otherwise, the full subsystem is Reset.

TABLE 51: HW_CFG (CONTINUED)

Hardware Configuration Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
12	RST_PROTECT	R/W	Reset Protection Setting this bit protects select fields of certain registers from being affected by resets other than POR_N. Reset protection also prohibits loading or restoring of the default values from the device level configuration loader. **Note:** This field is only Reset by POR_N.
11:6	RELOAD_TYPE	R/W	Reload Type These bits specify which sections of the configuration data are utilized upon a CONFIG_RELOAD command. All combinations are valid. bit 0 - MAC Address bit 1 - PCIe Configuration Registers and Shadow Registers bit 2 - Function Register Space (minus the MAC Address and System (Device level) Registers bit 3 - System (Device level) Registers bit 4 - Strap Register bit 5 - Reserved
5	CONFIG_DL	R/W1C	Configuration Data Loaded When set, this bit indicates that a valid configuration was found and that register programming has completed normally. This bit is set after a successful load of the data after power-up, various resets or after a CONFIG_RELOAD command has completed. **Note:** This signal is set during the configuration process upon the first register write that passes the APB target's configuration filter. For the case of a CONFIG_RELOAD command, it may go active before the CONFIG_RELOAD field. This signal is set based on the APB writes as filtered by the APB target's configuration filter. It will go active even if subsequent Reset Protection prevents the actual register bits from being written.

TABLE 51: HW_CFG (CONTINUED)

Hardware Configuration Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
4	CONFIG_RELOAD	R/W1S/SC	Configuration Reload Setting this bit Instructs the device configuration loader (EEPROM/OTP, etc.) to reload the configuration. The modules targeted by the reload operation are specified by the Reload Type field. The Configuration Data Loaded (CONFIG_DL) bit indicates a successful reload (at least one register of the reload type requested was programmed). This bit will self clear once the reload command is complete. **Note:** Reset Protection, if set, blocks the Reload of Reset protected bits. Any attempted read from the CSRs or the Device level registers during a Reload will return a value of all F's. This may be used as an indication that the reload is in progress. Writes to the CSRs or the Device level registers may result in an untoward operation and should not be attempted.
3:2	Reserved	R	Always 0's
1	LRST	R/W1S/SC	Light Reset Writing 1 generates the light software Reset of the subsystem. A light Reset does not affect the PCIe interface or controller. The contents from the OTP/EEPROM are not reloaded. This Reset does not reset the Ethernet PHY, re-tune the IO pads or re-latch the configuration straps.
0	SRST	R/W1S/SC	Soft Reset Writing 1 generates a software initiated Reset of the subsytem. If an external Ethernet PHY is used, it will be Reset as well. A software Reset will result in the contents of the EEPROM being reloaded. While the Reset sequence is in progress, the PCIe interface will be disconnected.

The Power Management Control Register details are shown in Table 52.

TABLE 52: PMT_CTL

Power Management Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31	D3_PLL_OVR	R/W	D3 PLL Request Override When set, the PLL clock request (PLL_REQ) will be kept active during D3. **Note:** This field is protected by Reset Protection (RST_PROTECT).
30	D3_XTAL_OVR	R/W	D3 Crystal Request Override When set, the crystal clock request (XTAL_REQ) will be kept active during D3. **Note:** This field is protected by Reset Protection (RST_PROTECT).
29:28	Reserved	R	Always 0's
27	SGMII_ETH_PHY_D3_COLD_OVR	R/W	SGMII and Ethernet PHY Override Cold When set, the SGMII/1000/2500BASE-X interface and the External Ethernet PHY is kept active while the subsystem is in D3cold. **Note:** This field is protected by Reset Protection (RST_PROTECT).
26	MAC_D3_TX_CLK_OVR	R/W	MAC TX Clock Override When set, the transmit clock to the Ethernet MAC and 1588 module is kept running while the subsystem is in D3. The crystal clock request (XTAL_REQ) is also kept active during D3. **Note:** This field is protected by Reset Protection (RST_PROTECT).
25	MAC_D3_RX_CLK_OVR	R/W	MAC RX Clock Override When set, the receive clock to the Ethernet MAC and 1588 module is kept running while the subsystem is in D3. The crystal clock request (XTAL_REQ) is also kept active during D3. **Note:** This field is protected by Reset Protection (RST_PROTECT).
24	Reserved	R	Always 0

TABLE 52: PMT_CTL (CONTINUED)

Power Management Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
23	SGMII_ETH_PHY_D3_OVR	R/W	SGMII and Ethernet PHY Override When set, the SGMII/1000/2500BASE-X interface and the External Ethernet PHY is kept active while the subsystem is in D3. The crystal clock request (XTAL_REQ) is also kept active during D3. **Note:** This field is protected by Reset Protection (RST_PROTECT).
22	INTC_D3_CLK_OVR	R/W	Interrupt Controller Clock Override When set, the system clock to the interrupt controller is kept running while the subsystem is in D3. The PLL and crystal clock requests (PLL_REQ and XTAL_REQ) are also kept active during D3. **Note:** This field is protected by Reset Protection (RST_PROTECT).
21	DMAC_D3_CLK_OVR	R/W	DMA Controller Clock Override When set, the system clock to the DMA controller is kept running while the subsystem is in D3. The PLL and crystal clock requests (PLL_REQ and XTAL_REQ) are also kept active during D3. **Note:** This field is protected by Reset Protection (RST_PROTECT).
20	1588_D3_CLK_OVR	R/W	1588 Clock Override When set, the system clock to the 1588 module is kept running while the subsystem is in D3. The RX and TX network clocks are controlled separately. The PLL and crystal clock requests (PLL_REQ and XTAL_REQ) are also kept active during D3. **Note:** This field is protected by Reset Protection (RST_PROTECT).
19	MAC_D3_CLK_OVR	R/W	MAC Clock Override When set, the system clock to the Ethernet MAC is kept running while the subsystem is in D3. The RX and TX network clocks are controlled separately. The PLL and crystal clock requests (PLL_REQ and XTAL_REQ) are also kept active during D3. **Note:** This field is protected by Reset Protection (RST_PROTECT).

TABLE 52: PMT_CTL (CONTINUED)

Power Management Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
18	RX_FCT_RFE_D3_CLK_OVR	R/W	RX FCT and RFE D3 Clock Override When set, the system clock to the RX FCT and RFE is kept running while the subsystem is in D3. The PLL and crystal clock requests (PLL_REQ and XTAL_REQ) are also kept active during D3. **Note:** This field is protected by Reset Protection (RST_PROTECT).
17	TX_FCT_LSO_D3_CLK_OVR	R/W	TX FCT and LSO D3 Clock Override When set, the system clock to the TX FCT and LSO is kept running while the subsystem is in D3. The PLL and crystal clock requests (PLL_REQ and XTAL_REQ) are also kept active during D3. **Note:** This field is protected by Reset Protection (RST_PROTECT)
16:14	Reserved	R	Always 0's
13	EEE_WAKEUP_EN	R/W	EEE WAKEUP Enable Enables EEE WAKEUP Status (EEE_WUPS) as a wake-up event. This bit is automatically cleared when the PM_PME message is sent due to a wake-up event if Resume Clears Remote Wake-up Enables (RES_CLR_WKP_EN) is set. **Note:** This field is protected by Reset Protection (RST_PROTECT).
12	EEE_WUPS	R/W1C	EEE WAKEUP Status This field indicates if the cause of the current wake-up event is due to EEE. It is used along with WAKEUP Status (WUPS). See the WUPS field for the encoding. This bit is set when Energy Efficient Ethernet TX Wake (EEE_TX_WAKE) or Energy Efficient Ethernet RX Wake (EEE_RX_WAKE) are set. This bit will set regardless of the value in EEE WAKEUP Enable (EEE_WAKEUP_EN). If the Resume Clears Remote Wake-up Status (RES_CLR_WKP_STS) bit is set, this bit will clear when the PM_PME message is sent due to a wake-up event. See the RES_CLR_WKP_STS bit for further details.
11:10	Reserved	R	Always 0's

TABLE 52: PMT_CTL (CONTINUED)

Power Management Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
9	RES_CLR_WKP_STS	R/W	Resume Clears Remote Wake-up Status When set, the following status signals will clear when the PM_PME message is sent due to a wake-up event: Wake-up Control and Status Register 1 (WUCSR1): • RFE Wake-up Frame Received (RFE_WAKE_FR) • Perfect DA Frame Received (PFDA_FR) • Remote Wake-up Frame Received (WUFR) • Magic Packet Received (MPR) • Broadcast Frame Received (BCAST_FR) • Energy Efficient Ethernet TX Wake (EEE_TX_WAKE) • Energy Efficient Ethernet RX Wake (EEE_RX_WAKE) Wake-up Control and Status Register 2 (WUCSR2): • IPv6 TCP SYN Packet Received (IPV6_TCPSYN_RCD) • IPv4 TCP SYN Packet Received (IPV4_TCPSYN_RCD) Power Management Control Register (PMT_CTL): • WAKE-UP Status (WUPS) • EEE WAKE-UP Status (EEE_WUPS) Wake-ups initiated by the host do not clear the wake-up statuses. When cleared, the wake-up status signals are not cleared when the host clears the PME_Status bit. **Note:** This field is protected by Reset Protection (RST_PROTECT).

TABLE 52: PMT_CTL (CONTINUED)

Power Management Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
8	RES_CLR_WKP_EN	R/W	Resume Clears Remote Wake-up Enables When set, the following wake-up enables will clear when the PM_PME message is sent due to a wake-up event: Wake-up Control and Status Register 1 (WUCSR1): • RFE Wake Enable (RFE_WAKE_EN) • Perfect DA Wake-up Enable (PFDA_EN) • Wake-up Frame Enable (WUEN) • Magic Packet Enable (MPEN) • Broadcast Wake-up Enable (BCAST_EN) • Energy Efficient Ethernet TX Wake Enable (EEE_ TX_WAKE_EN) • Energy Efficient Ethernet RX Wake Enable (EEE_ RX_WAKE_EN) Wake-up Control and Status Register 2 (WUCSR2): • IPv6 TCP SYN Wake Enable (IPV6_TCPSYN _WAKE_EN) • IPv4 TCP SYN Wake Enable (IPV4_TCPSYN _WAKE_EN) Power Management Control Register (PMT_CTL) • Wake-On-LAN Enable (WOL_EN) • Ethernet PHY Interrupt Enable (ETH_PHY_WAKE_EN) • EEE WAKEUP Enable (EEE_WAKEUP_EN) • GPIO WAKEUP Enable (GPIO_WAKEUP_EN) Wake-ups initiated by the host do not clear the wake-up enables. When cleared, the wake-up enables are not cleared when the host clears the PME_Status bit. **Note:** This field is protected by Reset Protection (RST_ PROTECT).
7	READY	R	Device Ready The READY bit is used to indicate when the subsystem is ready for normal operation. The READY bit remains deasserted for the following reasons. • Waiting for the Gigabit Ethernet PHY Reset sequence to complete and the PHY to become operational. This could take over 125 ms for the external PHY SKUs depending on how the subsystem is configured. • Waiting for content to be loaded from the device level configuration loader.

TABLE 52: PMT_CTL (CONTINUED)

Power Management Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
6	RX_FIFO_L1_EXIT_EN	R/W	RX FIFO L1-ASPM Exit Enable When this bit is cleared, the L1-ASPM Link state is exited based PCIe activity. When this bit is set, the L1-ASPM Link state is also exited if there is data stored in the RX FIFOs. **Note:** This field is protected by Reset Protection (RST_PROTECT).
5	EXT_PHY_RDY_EN	R/W	External PHY Ready Delay Enable This bit enables waiting an additional 125 ms to allow the external Ethernet PHY to become ready before asserting the Device Ready (READY) bit. **Note:** This field is protected by Reset Protection (RST_PROTECT).
4	ETH_PHY_RST	R/W1S/SC	Ethernet PHY Reset Writing a '1' to this bit resets the Ethernet PHY. The internal logic automatically holds the PHY Reset for a minimum of 2 ms. When the PHY is released from Reset, this bit is automatically cleared. All writes to this bit are ignored while this bit is high. The Device Ready (READY) bit shall clear upon setting this bit. When the PHY is operational Device Ready (READY) bit shall assert.
3	WOL_EN	R/W	Wake-On-LAN Enable Enables WOL as a wake-up event which includes the following: • Wake-up Frame • Magic Packet • Perfect DA Match • Broadcast frame • IPv4 TCP SYN Packet • IPv6 TCP SYN Packet This bit is automatically cleared when the PM_PME message is sent due to a wake-up event if Resume Clears Remote Wake-up Enables (RES_CLR_WKP_EN) is set. **Note:** This field is protected by Reset Protection (RST_PROTECT).

TABLE 52: PMT_CTL (CONTINUED)

Power Management Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
2	ETH_PHY_WAKE_EN	R/W	Ethernet PHY Interrupt Enable Enables the Ethernet PHY Interrupt and EDPD energy detection as a wake-up event. This bit is automatically cleared when the PM_PME message is sent due to a wake-up event if Resume Clears Remote Wake-up Enables (RES_CLR_WKP_EN) is set. **Note:** This field is protected by Reset Protection (RST_PROTECT).

TABLE 52: PMT_CTL (CONTINUED)

Power Management Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
1:0	WUPS	R/W1C	WAKE-UP Status This field, along with EEE WAKE-UP Status (EEE_WUPS), indicates the cause of the current wake-up event. WUPS bits are cleared by writing a '1' to the appropriate bit. The encoding of these bits is as follows: EEE_WUPS, WUPS[1:0]: • 0,00: No Wake-up event Detected • X,X1: Ethernet PHY Wake Event • X,1X: Wake-up Frame, Magic Packet, Perfect DA Match Broadcast frame, IPv4/IPv6 TCP SYN Packet, or RFE frame • 1,XX: EEE Receive Wake (EEE_RX_WAKE) / EEE Transmit Wake (EEE_TX_WAKE) More than one bit may be set indicating that multiple events occurred. WUPS[1] is set if any of the following are set: • RFE Wake-up Frame Received (RFE_WAKE_FR) • Perfect DA Frame Received (PFDA_FR) • Remote Wake-up Frame Received (WUFR) • Magic Packet Received (MPR) • Broadcast Frame Received (BCAST_FR) • IPv6 TCP SYN Packet Received (IPV6_TCPSYN_RCD_WK) • IPv4 TCP SYN Packet Received (IPV4_TCPSYN_RCD_WK) WUPS[1] will set regardless of the value in Wake-On-LAN Enable (WOL_EN). WUPS[0] will set regardless of the value in Ethernet PHY Interrupt Enable (ETH_PHY_WAKE_EN). If the Resume Clears Remote Wake-up Status (RES_CLR_WKP_STS) bit is set, WUPS[1] and WUPS[0] will clear when the PM_PME message is sent due to a wake-up event. See the RES_CLR_WKP_STS bit for further details.

The Data Port Select Register details are shown in Table 53.

TABLE 53: DP_SEL

Data Port Select Register			Default: 0x8000_0000
Bits	**Name**	**R/W**	**Description**
31	DPRDY	R	Data Port Ready The Data Port Ready bit indicates when the data port RAM access has completed. In the case of a read operation, this bit indicates when the read data has been stored in the Data Port Data Registers. 1b = Data Port is ready. 0b = Data Port is busy processing a transaction.
30	DP_EN	R/W	Data Port Enable When this bit is set, the RAM indicated by the Select (SEL) field is disconnected from its function and made available to the Data Port interface.
29:5	Reserved	R	Always 0's
4:0	SEL	R/W	Select Selects which RAM to access. 11100b - LSO_RAM_3 11011b - LSO_RAM_2 11010b - LSO_RAM_1 11001b - DMAC_REORDER_BUFFER _3 11000b - DMAC_REORDER_BUFFER _2 10111b - DMAC_REORDER_BUFFER _1 10110b - PCIE_DRCV_RAM 10101b - PCIE_HRCV_RAM 10100b - PCIE_SOT_RAM 10011b - PCIE_RETRY_RAM 10010b - DMAC_TX_RAM_3 10001b - DMAC_TX_RAM_2 10000b - DMAC_TX_RAM_1 01111b - DMAC_TX_RAM_0 01110b - DMAC_RX_RAM_3 01101b - DMAC_RX_RAM_2 01100b - DMAC_RX_RAM_1 01011b - DMAC_RX_RAM_0 01010b - DMAC_REORDER_BUFFER_0 01001b - FCT_TX_RAM_3 01000b - FCT_TX_RAM_2 00111b - FCT_TX_RAM_1 00110b - FCT_TX_RAM_0 00101b - FCT_RX_RAM_3 00100b - FCT_RX_RAM_2 00011b - FCT_RX_RAM_1 00010b - FCT_RX_RAM_0 00001b - RFE_RAM (VLAN and DA Hash Table (VHF RAM) 00000b - LSO_RAM_0 (LSO Header RAM channel 0) Data port access to the select RAM interferes with the functional use of that RAM. For the PCIe RAMs, data port access will result in improper PCIe functionality.

The Data Port Command Register details are shown in Table 54.

TABLE 54: DP_CMD

Data Port Command Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:1	Reserved	R	Always 0's
0	DATA_PORT_WRITE	R/W	Writing to this bit initiates the data-port access. 1b - Write operation 0b - Read operation

The Data Port Address Register details are shown in Table 55.

TABLE 55: DP_ADDR

Data Port Address Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:14	Reserved	R	Always 0's
13:0	DATA_PORT_ADDR	R/W	Data Port Address

The Data Port Data 0 Register details are shown in Table 56.

TABLE 56: DP_DATA_0

Data Port Data 0 Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:0	DATA_PORT_DATA[31:0]	R/W	Data Port Data[31:0]

The Data Port Data 1 Register details are shown in Table 57.

TABLE 57: DP_DATA_1

Data Port Data 1 Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:0	DATA_PORT_DATA[63:32]	R/W	Data Port Data[63:32]

The Data Port Data 2 Register details are shown in Table 58.

TABLE 58: DP_DATA_2

Data Port Data 2 Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:0	DATA_PORT_DATA[95:64]	R/W	Data Port Data[95:64]

The Data Port Data 3 Register details are shown in Table 59.

TABLE 59: DP_DATA_3

Data Port Data 3 Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:0	DATA_PORT_DATA[127:96]	R/W	Data Port Data[127:96]

The PCIe Configuration Address Register details are shown in Table 60.

TABLE 60: PCIE_CFG_ADDR

PCIe Configuration Address Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31	PCIE_CONFIG_BUSY	R	PCIe Configuration Busy Indicates that a PCIe Configuration space write is in process. This bit sets when the PCIe Configuration Data Register (PCIE_CFG_DATA) is written and clears once the write into the PCIe Configuration space is completed.
30:16	Reserved	R	Always 0's
15	PCIE_CONFIG_SHADOW_SEL	R/W	PCIe Configuration Shadow Select Selects the special shadow registers in the PCIe Configuration module.
14:12	Reserved	R	Always 0's
11:0	PCIE_CONFIG_ADDR	R/W	PCIe Configuration Address[11:0] Specifies the PCIe Configuration register BYTE address.

The PCIe Configuration Data Register details are shown in Table 61.

TABLE 61: PCIE_CFG_DATA

PCIe Configuration Data Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:0	PCIE_CFG_DATA	W	PCIe Configuration Data

The MAC Control Register details are shown in Table 62.

TABLE 62: MAC_CR

MAC Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:24	Reserved	R	Always 0's

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

TABLE 62: MAC_CR (CONTINUED)

MAC Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
20	AUTO_CLR_EEEEN	R/W NASR	Clear Energy Efficient Ethernet Enable on Link Loss When this bit is set, the EEEEN bit is automatically cleared upon link loss. **Note:** This field is protected by Reset Protection (RST_PROTECT).
19	ASD_LS	R/W NASR	Automatic Speed, Duplex, Link Source This bit selects the source of the Automatic Speed, Duplex and Link 1b - PHY interface 0b - Speed calculated from the receive clock, duplex and link from pins. **Note:** This field is protected by Reset Protection (RST_PROTECT).
18	EEE_TX_CLK_STOP_EN	R/W NASR	Energy Efficient Ethernet TX Clock Stop Enable When set, the MAC will halt the RGMII_TXC to the PHY during TX LPI. **Note:** Software or OTP/EEPROM contents should only set this bit if the Clock stop capable bit in PHY MMD register 3.1 indicates that the PHY is capable of allowing a stopped TX clock. This field is protected by Reset Protection (RST_PROTECT).
17	EEEEN	R/W NASR	Energy Efficient Ethernet Enable When set, enables Energy Efficient Ethernet operation in the MAC. When cleared, Energy Efficient Ethernet operation is disabled. The MAC will generate LPI requests even if Transmitter Enable (TXEN) is cleared and will decode the LPI indication even if Receiver Enable (RXEN) is cleared. Although the Energy Efficient Ethernet Enable (EEEEN) bit controls the generation and decoding of Low-Power Idle signaling to and from the PHY, it does not control the Auto-Negotiation EEE next page messaging within the PHY. This bit does not immediately initiate LPI transmission. LPI is delayed by the amount specified in EEE PHY Link Up Speed Up (EEE_PHY_LINK_CHANGE_SPEED_UP). **Note:** This field is protected by Reset Protection (RST_PROTECT).

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

TABLE 62: MAC_CR (CONTINUED)

MAC Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
16	EEE_TX_LPI_AUTO_ REMOVAL_EN	R/W NASR	Energy Efficient Ethernet TX LPI Automatic Removal Enable When set, enables the automatic deassertion of LPI in anticipation of a periodic transmission event. The time to wait is specified in the EEE TX LPI Automatic Removal Delay Register (EEE_TX_LPI_AUTO_REMOVAL_DELAY). The interval is timed from the point where the MAC initiates LPI signaling. **Note:** Software should only change this field when Energy Efficient Ethernet Enable (EEEEN) field is cleared. This field is protected by Reset Protection (RST_PROTECT).
15	FLS	R/W NASR	Force Link Status This bit is used to force the link status to indicate that the PHY has a valid link. 0b - Link Status determined by normal automatic methods per Section 15.0, "EEPROM Memory Programming (Read and Write)". 1b - Link Status forced valid.
14	LSP	R/W NASR	Link Status Polarity This bit indicates the polarity of the ENET_LINK pin. 0b - Active-low - ENET_LINK asserted low indicates the PHY has a valid link. 1b - Active-high - ENET_LINK asserted high indicates the PHY has a valid link. **Note:** Software should not modify this field while the MAC's receiver or transmitter is enabled (Receiver Enable (RXEN) or Transmitter Enable (TXEN) bit set). This field is protected by Reset Protection (RST_PROTECT).
13	ADP	R/W NASR	Automatic Duplex Polarity This bit indicates the polarity of the ENET_DUPLEX pin. 0b - Active-low - ENET_DUPLEX asserted low indicates the PHY is in Full-duplex mode. 1b - Active-high - ENET_DUPLEX asserted high indicates the PHY is in Full-duplex mode. **Note:** Software should not modify this field while the MAC's receiver or transmitter is enabled (Receiver Enable (RXEN) or Transmitter Enable (TXEN) bit set). This field is protected by Reset Protection (RST_PROTECT).

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

TABLE 62: MAC_CR (CONTINUED)

MAC Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
12	ADD	R/W NASR	Automatic Duplex Detection When set, the MAC ignores the setting of the Duplex mode (DPX) bit and automatically determines the Duplex Operational mode. The MAC uses the ENET_DUPLEX pin to accomplish mode detection and reports the last determined status via the Duplex mode (DPX) bit. When Reset, the setting of the Duplex Mode (DPX) bit determines Duplex operation. **Note:** Software should not modify this field while the MAC's receiver or transmitter is enabled (Receiver Enable (RXEN) or Transmitter Enable (TXEN) bit set). This field is protected by Reset Protection (RST_PROTECT).
11	ASD	R/W NASR	Automatic Speed Detection When set, the MAC ignores the setting of the MAC Configuration (CFG) field and automatically determines the speed of operation. The MAC samples the receive clock to accomplish speed detection and reports the last determined speed via the MAC Configuration (CFG) field. When Reset, the setting of the MAC Configuration (CFG) field determines operational speed. **Note:** Software should not modify this field while the MAC's receiver or transmitter is enabled (Receiver Enable (RXEN) or Transmitter Enable (TXEN) bit set). This field is protected by Reset Protection (RST_PROTECT).
10	INT_LOOP	R/W NASR	Internal Loopback Operation Mode Loops back data between the TX datapath and RX datapath interfaces. This is only for Full-duplex mode. In internal Loopback mode, the TX frame is received by the Internal GMII interface, and sent back to the MAC without being sent to the PHY. 0b - Normal mode 1b - Internal loopback enabled **Note:** Software should not modify this field while the MAC's receiver or transmitter is enabled (Receiver Enable (RXEN) or Transmitter Enable (TXEN) bit set).
9:8	Reserved	R	Always 0’s

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

TABLE 62: MAC_CR (CONTINUED)

MAC Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
7:6	BOLMT	R/W NASR	Back Off Limit The BOLMT bits allow the user to set its back off limit in a relaxed or aggressive mode. According to IEEE 802.3, the MAC has to wait for a random number [r] of slot-times after it detects a collision, where: (eq.1) 0 < r < 2K The exponent K is dependent on how many times the current frame to be transmitted has been retried, as follows: (eq.2) K = min. (n, 10) where n is the current number of retries. If a frame has been retried three times, then K = 3 and r = 8 slot-times maximum. If it has been retried 12 times, then K = 10, and r = 2(10) = 20 slot-times maximum. An LFSR (Linear Feedback Shift Register) counter emulates a random number generator, from which r is obtained. Once a collision is detected, the number of the current retry of the current frame is used to obtain K (eq.2). This value of K translates into the number of bits to use from the LFSR counter. If the value of K is 3, the MAC takes the value in the first three bits of the LFSR counter and uses it to count down to zero on every slot-time. This effectively causes the MAC to wait eight slot-times. To give the user more flexibility, the BOLMT value forces the number of bits to be used from the LFSR counter to a predetermined value as in the table below. Thus, if the value of K = 10, the MAC will look at the BOLMT if it is 00, then use the lower ten bits of the LFSR counter for the wait countdown. If the BOLMT is 10, then it will only use the value in the first four bits for the wait countdown, etc. Slot-time = 512 bit times **Note:** Software should not modify this field while the MAC's receiver or transmitter is enabled (Receiver Enable (RXEN) or Transmitter Enable (TXEN) bit set). This field is protected by Reset Protection (RST_PROTECT).
5	CNTR_RST	R/ W1S/ SC	Counter Reset When this bit is set, all of the statistics counters and rollover statuses are Reset. This bit self-clears.
4	CNTR_WEN	R/W NASR	Counter Write En When this bit is set, all of the statistics counters are writable for test purposes. When clear, the statistics counters are read-only.

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

TABLE 62: MAC_CR (CONTINUED)

MAC Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
3	DPX	NASR	Duplex Mode This bit determines the Duplex Operational mode of the MAC when the Automatic Duplex Detection (ADD) bit is Reset. When the Automatic Duplex Detection (ADD) bit is set, this bit is read-only and reports the last determined Duplex Operational mode. 0b - MAC is in Half-duplex mode 1b - MAC is in Full-duplex mode **Note:** Software should not modify this field while the MAC's receiver or transmitter is enabled (Receiver Enable (RXEN) or Transmitter Enable (TXEN) bit set). Half-duplex mode is disabled if the detected or manually set speed is 1000/2500 Mbps, regardless of the setting of this bit. This field is protected by Reset Protection (RST_PROTECT).
2:1	CFG	NASR	MAC Configuration This field determines the operational speed of the MAC when the Automatic Speed Detection (ASD) bit is Reset. When the Automatic Speed Detection (ASD) bit is set, this field is read-only and reports the last determined operational speed. 00b - 10 Mbps 01b - 100 Mbps 10b/11b - 1000 Mbps/2500 Mbps **Note:** When Automatic Speed Detection (ASD) bit is Reset, values of 2 and 3 are equivalent and the actual operating speed is determined by the clock speed provided by the SGMII/1000/2500BASE-X interface. Software may use the unique values as flags. When Automatic Speed Detection (ASD) bit is set, the reported speed value will be 2 for both 1000 Mbps and 2500 Mbps rates. Software should not modified this field while the MAC's receiver or transmitter is enabled (Receiver Enable (RXEN) or Transmitter Enable (TXEN) bit set). This field is protected by Reset Protection (RST_PROTECT).
0	MRST	R/W1S/SC	MAC Reset 0b - MAC is enabled 1b - MAC is Reset Register bits marked as NASR are not Reset.

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

The MAC Receiver Register details are shown in Table 63.

TABLE 63: MAC_RX

MAC Receiver Register			Default: 0x05EE_0068
Bits	**Name**	**R/W**	**Description**
31:30	Reserved	R	Always 0's
29:16	MAX_SIZE	R/W NASR	Maximum Frame Size Defines the maximum size for a received frame. Frames exceeding this size are aborted. **Note:** The maximum guaranteed supported frame size is 9,220 bytes. Values for this field larger than 15360 are ineffective, since the watchdog timer would take affect and truncate the frame at 15361. Hardware prevents this field from being modified while the MAC's receiver is enabled (Receiver Enable (RXEN) bit set in MAC Receive Register (MAC_RX).
15:7	Reserved	R	Always 0's
6	LEN_FLD_LT_CHK	R/W NASR	Length Field Less Than Check 0b = Allow frames larger than 64 bytes to have a length/type field value less than the received frame length (minus 18 bytes). 1b = Abort and count frames larger than 64 bytes whose length/type field value is less than the received frame length (minus 18 bytes). **Note:** Hardware prevents this bit from being modified while the MAC's receiver is enabled (Receiver Enable (RXEN) bit set in MAC Receive Register (MAC_RX). This field is protected by Reset Protection (RST_PROTECT).
5	WTL	R/W NASR	Watchdog Truncation Length 0b = The MAC truncates the Rx FRAME at MAC_RX.MAX_SIZE+1 or 15361, whichever is smaller. The RxCmdA of the truncated received frame passed to the FCT will have a length value of MAC_RX.MAX_SIZE+1 or 15361, whichever is smaller. The RWT bit will be set if the MAC_RX.MAX_SIZE is set to 15360 or larger. The LONG bit will be set if MAC_RX.MAX_SIZE is set to 15360 or less. The truncation will most likely result in an FCS error (FCS bit set). 1b = The MAC truncates the Rx FRAME at 15361. The RxCmdA of the truncated received frame passed to the FCT will have the RWT bit set and a length value of 15361. The LONG bit will be set if MAC_RX.MAX_SIZE is set to 15360 or less. The truncation will most likely result in an FCS error (FCS bit set). **Note:** Hardware prevents this bit from being modified while the MAC's receiver is enabled (Receiver Enable (RXEN) bit set in MAC Receive Register (MAC_RX).

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

TABLE 63: MAC_RX (CONTINUED)

MAC Receiver Register			Default: 0x05EE_0068
Bits	**Name**	**R/W**	**Description**
4	FCS_STRIPPING	R/W NASR	FCS Stripping When set, the MAC will strip the FC (last 4 bytes) off of all received frames. **Note:** Hardware prevents this bit from being modified while the MAC's receiver is enabled (Receiver Enable (RXEN) bit set in MAC Receive Register (MAC_RX).
3	LFCD	R/W NASR	Length Field Checking Disable 0b = Abort all frames whose frame length is inconsistent with the length value specified in the length/type field. 1b = Ignore length field. When cleared, frames whose frame length is inconsistent with the length value specified in the length/type field are considered to be errored and are dropped at the RX FIFO (unless the RX FIFO is configured to store bad frames) and counted. Wake-up frames do not cause a wake event and ARP/NS offload requests do not cause a response. When set, frame length errors are still reported and counted but the frame is not dropped. Wake-up frames may cause a wake event and ARP/NS offload requests may cause a response. **Note:** Hardware prevents this bit from being modified while the MAC's receiver is enabled (Receiver Enable (RXEN) bit set in MAC Receive Register (MAC_RX). This field is protected by Reset Protection (RST_PROTECT).
2	FSE	R/W NASR	VLAN Frame Size Enforcement 0b = Abort all frames larger than the maximum frame size. 1b = Abort all non-VLAN frames larger than maximum frame size. Abort all frames with a single VLAN tag that are larger the maximum frame size + 4. Abort all frames with two VLAN tags that are larger than the maximum frame size + 8. **Note:** Hardware prevents this bit from being modified while the MAC's receiver is enabled (Receiver Enable (RXEN) bit set in MAC Receive Register (MAC_RX).
1	RXD	R/W NASR	Receiver Disabled This bit indicates the MAC’s receiver has been successfully disabled via clearing the Receiver Enable (RXEN) bit. It is set when the hardware disabling process invoked by a transition of the Receiver Enable (RXEN) bit from 1 to 0 (enabled to disabled) completes.

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

TABLE 63: MAC_RX (CONTINUED)

MAC Receiver Register			Default: 0x05EE_0068
Bits	**Name**	**R/W**	**Description**
0	RXEN	R/W NASR	Receiver Enable When set, the MAC's receiver is enabled and will receive frames from the PHY. When Reset, the MAC's receiver is disabled and will not receive any frames from the PHY. If this bit is deasserted while a frame is being received, the received frame's allowed to complete. Upon completion, the MAC's receiver is disabled and the Receiver Disabled (RXD) bit is asserted.

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

The MAC Transmit Register details are shown in Table 64.

TABLE 64: MAC_TX

MAC Transmit Register			Default: 0x0000_0040
Bits	**Name**	**R/W**	**Description**
31:9	Reserved	R	Always 0's
8:4	WRR_WTS	R/W NASR	Weighted Round Robin Weight Table Size This field specifies the size of the Weighted Round Robin Weight Table. Valid values are 1 through 16. **Note:** Values outside of this range may cause unexpected results.
3	TX_FIFO_SRV	R/W NASR	TX FIFO Service Mode When set, the MAC services the TX FIFOs in a strict order, fixed priority. When cleared, a Weighted Round Robin (WRR) order is followed.
2	BFCS	R/W NASR	Bad FCS When set, the MAC's transmitter will append a bad FCS on all transmitted frames. This feature is useful for diagnostic purposes. This function may only be used in conjunction with Insert FCS and Pad of TX Command A.
1	TXD	R/W1C NASR	Transmitter Disabled This bit indicates the MAC's transmitter has been successfully disabled via clearing the Transmitter Enable (TXEN) bit. It is set when the hardware disabling process invoked by a transition of the Transmitter Enable (TXEN) bit from 1 to 0 (enabled to disabled) completes.

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

TABLE 64: MAC_TX (CONTINUED)

MAC Transmit Register			Default: 0x0000_0040
Bits	**Name**	**R/W**	**Description**
0	TXEN	R/W NASR	Transmitter Enable When set, the MAC's transmitter is enabled and it will transmit frames from the transmit buffer onto the cable. When Reset, the MAC's transmitter is disabled and will not transmit any frames. If this bit is cleared while a frame is being transmitted, the frame is allowed to complete when in Full-duplex mode. Upon completion, the MAC's transmitter is disabled and the Transmitter Disabled (TXD) bit is asserted. In Half-duplex mode the frame shall be aborted if a collision is encountered while transmitting.

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

The Flow Control Register details are shown in Table 65.

TABLE 65: MAC_FLOW

Flow Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31	FORCE_FC	R/W1S/SC	Force Transmission of TX Flow Control Frame This bit forces the transmission of a TX flow control frames. Writing a "1" initiates the frame transmission. The frame will be generated with the Pause Time value. After the frame is transmitted, the MAC will clear this bit.
30	TX_FCEN	R/W NASR	TX Flow Control Enable When set, enables the transmit MAC flow control function based on high and low watermarks in the RX FIFO, as discussed in this section. **Note:** The threshold values in the FCT Flow Control x Threshold Register (FCT_FLOW_x) must be programmed before this bit is set. Conversely, this bit must be cleared before the threshold values in the FCT Flow Control x Threshold Register (FCT_FLOW_x) are programmed to 0.
29	RX_FCEN	R/W NASR	RX Flow Control Enable When set, enables the receive MAC flow control function. The MAC decodes all incoming frames for control frames. If it receives a valid control frame (PAUSE command), it disables the transmitter for a specified time (Decoded pause time x slot time). When not set, the MAC flow control function is disabled. The MAC does not decode frames for control frames.

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

TABLE 65: MAC_FLOW (CONTINUED)

Flow Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
28	FPF	R/W NASR	Forward Pause Frames Enables passing received pause frames to RX datapath interface. 0b = Sink received pause frames. 1b = Pass received pause frames to the RX datapath interface. **Note:** Flow Control is applicable when the MAC is set in Full-duplex mode.
27:16	Reserved	R	Always 0's
15:0	FCPT	R/W NASR	Pause Time This field indicates the value to be used in the Pause Time field in the control frame.

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

The Random Number Seed Value Register details are shown in Table 66.

TABLE 66: MAC_RAND_SEED

Random Number Seed Value Register			Default: 0x0000_9876
Bits	**Name**	**R/W**	**Description**
31:16	Reserved	R	Always 0's
15:0	RAND_SEED	R/W NASR	Random Number Seed The MAC random number generator seed value. The content of this register is the seed value for the LFSR (Linear Feedback Shift Register) counter used to emulate the random number generator in the MAC TX back-off timer logic.

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

The Error Status Register details are shown in Table 67.

TABLE 67: MAC_ERR_STS

Error Status Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:11	Reserved	R	Always 0's
10	LEN_ERR	R/W1C NASR	Length Field Error Indicates that the frame length was inconsistent with the length value specified in the length/type field.

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

TABLE 67: MAC_ERR_STS (CONTINUED)

Error Status Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
9	RXERR	R/W1C NASR	RX Error Indicates that a receive error (PHY RX error signal asserted) has been detected during frame reception (including during the preamble and SFD). **Note:** This bit will be set regardless of other error conditions (alignment, FCS, lengths, etc.).
8	FERR	R/W1C NASR	FCS Error An FCS errored frame has been received. This bit is set regardless of the frame length. **Note:** This bit will not be set if the ALERR bit is set. Only frames with an integer multiple of 8 bits are considered.
7	LFERR	R/W1C NASR	Large Frame Error A frame larger than the maximum allow frame size has been received.
6	RFERR	R/W1C NASR	Runt/Short Frame Error A runt frame or a short frame has been received. This bit is set regardless of the FCS status.
5	RWTERR	R/W1C NASR	Receive Watchdog Timer Expired When set, this bit indicates the received frame was longer than 15,360 bytes and was truncated by the MAC.
4	ECERR	R/W1C NASR	Excessive Collision Error A transmit frame was aborted due to sixteen collisions occurring.
3	ALERR	R/W1C NASR	Alignment Error An alignment error (non-integer multiple of 8 bits and a bad FCS or an RX symbol error (including during the preamble and SFD)) has been detected on a received frame.
2	URERR	R/W1C NASR	Under Run Error The MAC has been under run by the transmit data-path.
1:0	Reserved	R	Always 0’s

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

The MAC Receiver Address High Register details are shown in Table 68.

TABLE 68: MAC_RX_ADDRH

MAC Receiver Address High Register			Default: 0x0000_FFFF
Bits	**Name**	**R/W**	**Description**
31:16	Reserved	R	Always 0's
15:0	MAC_PHY_ADDR[47:32]	R/W NALR NASR	Physical Address [47:32] This field contains the upper 16 bits [47:32] of the physical address of the MAC. **Note:** This field is protected by Reset Protection (RST_PROTECT).

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

The MAC Receiver Address Low Register details are shown in Table 69.

TABLE 69: MAC_RX_ADDRL

MAC Receiver Address High Register			Default: 0xFFFF_FFFF
Bits	**Name**	**R/W**	**Description**
31:0	MAC_PHY_ADDR[31:0]	R/W NALR NASR	Physical Address [31:0] This field contains the lower 32 bits [31:0] of the physical address of the MAC. **Note:** This field is protected by Reset Protection (RST_PROTECT).

Note: The NASR designation is only applicable when the MAC Reset (MRST) bit of the MAC Control Register (MAC_CR) is set. Register bits designated as NASR are Reset via other chip level software initiated resets.

The MII Access Register details are shown in Table 70.

TABLE 70: MII_ACCESS

MII Access Register			Default: 0x0000_0800
Bits	**Name**	**R/W**	**Description**
31:16	Reserved	R	Always 0's
15:11	PHY_ADDR	R/W NASR	PHY Address This field specifies the PHY address. This field must be set to the address of the external PHY.
10:6	MII_REGISTER_INDEX	R/W NASR	MII Register Index For Clause 22 operation, these bits select the desired MII register in the PHY. For Clause 45 operation, these bits select the desired MMD in the PHY.

TABLE 70: MII_ACCESS (CONTINUED)

MII Access Register			Default: 0x0000_0800
Bits	**Name**	**R/W**	**Description**
5:4	MDC_CYCLE_TIME	R/W NASR	MDC Cycle Time This field selects MDC clock rate 00b = 2.5 MHz (400 ns) - IEEE 802.3 standard 01b = 5 MHz (200 ns) 10b = 12.5 MHz (80 ns) 11b = 25 MHz (40 ns)
3	MIICL45	R/W NASR	MII Clause 45 This bit selects between 802.3 Clause 22 and Clause 45 command formats. 0b = Clause 22 1b = Clause 45
2:1	MIIWnR	R/W NASR	MII Write When in Clause 22 mode, setting the lower bit tells the PHY that this will be a write operation using the MII data register. If the lower bit is not set, this will be a read operation, packing the data in the MII data register. The upper bit is not used. When in Clause 45 mode, these bits determine the command as follows: 00b = Address 01b = Write 10b = Read 11b = Read w/ post address increment **Note:** The values for Read and Read w/ post address increment are swapped as compared with the values given for the frame structure in IEEE 802.3 clause 45.3.4 OP (operation code).
0	MIIBZY	R/W NASR	MII Busy This bit must be polled to determine when the MII register access is complete. This bit must read a logical 0 before writing to this register or to the MII data register. The LAN driver software must set (1b) this bit in order for the Host to read or write any of the MII PHY registers. During a MII register access, this bit will be set, signifying a read or write access is in progress. The MII data register must be kept valid until the MAC clears this bit during a PHY write operation. The MII data register is invalid until the MAC has cleared this bit during a PHY read operation.

The MII Data Register details are shown in Table 71.

TABLE 71: MII_DATA

MII Data Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:16	Reserved	R	Always 0's
15:0	MII_DATA	R/W NASR	MII Data This contains the 16-bit value read from the PHY read operation or the 16-bit data value to be written to the PHY before an MII write operation. For Clause 22 access or for Clause 45 write, read and read with post increment commands, this register contains either the data to be written to the PHY register specified in the MII Access Register, or the read data from the PHY register whose index is specified in the MII Access Register (MII_ ACCESS). For the Clause 45 Address command, this register supplies the address of the register that will be subsequently accessed. **Note:** The MII Busy (MIIBZY) bit in the MII Access Register (MII_ACCESS) must be cleared before writing to this register.

The MII RGMII Internal Delay Register details are shown in Table 72.

TABLE 72: MII_RGMII_ID

MII RGMII Internal Delay Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:2	Reserved	R	Always 0's
1	RGMII_TXC_DELAY_EN	R/W NASR	RGMII TXC Delay Enable Configures the RGMII TXC Delay mode: 0b = RGMII TXC Delay mode disabled 1b = RGMII TXC Delay mode enabled **Note:** This bit only has meaning while in RGMII mode. This field is protected by Reset Protection (RST_PROTECT).
0	RGMII_RXC_DELAY_EN	R/W NASR	RGMII RXC Delay Enable Configures the RGMII RXC Delay mode: 0 = RGMII RXC Delay mode disabled 1 = RGMII RXC Delay mode enabled **Note:** This bit only has meaning while in RGMII mode. This field is protected by Reset Protection (RST_PROTECT).

The Energy Efficient Ethernet TX LPI Request Delay Count Register details are shown in Table 73.

TABLE 73: EEE_TX_LPI_REQUEST_DELAY_CNT

Energy Efficient Ethernet TX LPI Request Delay Count Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:0	EEE_TX_LPI_REQUEST_ DELAY_CNT	R/W NASR	EEE TX LPI Request Delay Count Contains the count corresponding to the amount of time, in microseconds, the MAC must wait after the TX FIFO is empty or after a Pause frame or ARP/NS response is transmitted before invoking the LPI protocol. Whenever the TX FIFO is empty, the MAC checks the Energy Efficient Ethernet Enable (EEEEN) bit of the MAC Control Register (MAC_CR) to determine whether or not the Energy Efficient Ethernet mode of operation is in effect. If the bit is clear, no action is taken, otherwise, the MAC waits the amount of time indicated in this register. After the wait period has expired, the LPI protocol is initiated and the Energy Efficient Ethernet Start TX Low-Power Interrupt (EEE_START_ TX_LPI_INT) bit of the MAC Interrupt Status Register (MAC_INT_STS) will be set. If the TX FIFO becomes non-empty or the MAC autonomously sends a Pause frame or ARP/NS response, the timer is restarted. Host software should only change this field when Energy Efficient Ethernet Enable (EEEEN) is cleared. **Note:** Due to a 1 µs pre-scaler, the actual time can be up to 1 us longer than specified. A value of zero is valid and will cause no delay to occur. However, a value of zero may adversely affect the ability of the TX data path to support Gigabit operation. A reasonable value when the part is operating at Gigabit speeds is 50 µs. This value may be increased pending the results of performance testing with EEE enabled. The motivation for 802.3 az is the scenario where the EEE link is idle most of the time with the occasional full bandwidth transmission bursts. Aggressively optimizing power consumption during pockets of inactivity is not the objective for this mode of operation. The time is in MAC TX Clock period increments when EEE Timer Speed Up (EEE_TIMER_SPEED_UP) is set.

The Wake-up and Control Register details are shown in Table 74.

TABLE 74: WUCSR2

Wake-up and Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31	CSUM_DISABLE	R/W NASR	Checksum Disable When clear, the IP header checksum, TCP checksum, ICMP payload checksum, and FCS are calculated and all must agree with the frame contents, in order for the frame (TCP_SYN, or NS) to be considered for detection analysis. When set, only the FCS is calculated and checked for TCP_SYN, and NS frames. The IP header checksum, ICMP payload checksum, and TCP checksum are not calculated, hence any mismatches are ignored. **Note:** This field is protected by Reset Protection (RST_PROTECT).
30	EN_ROUTE_HDR	R/W NASR	Enable Other Routing Headers This bit allows the usage of IPv6 Routing headers other than type 0 and 2 when validating the TCP checksum for TCP SYN frames. When cleared, IPv6 Routing headers other than type 0 and 2 are not supported and the checksum is not verified. When set, IPv6 Routing headers other than type 0 and 2 are skipped, if the Segments Left field in the header is zero, otherwise, the checksum is not verified. **Note:** This field is protected by Reset Protection (RST_PROTECT).
29:11	Reserved	R	Always 0's
10	FARP_FR	R/W NASR	Forward ARP Frames Enables passing received ARP frames that target this MAC and were processed by the ARP offload logic to the RX datapath interface. 0b = Sink received ARP frames. 1b = Pass received ARP frames to the RX datapath interface.
9	FNS_FR	R/W NASR	Forward NS Frames Enables passing received NS frames that target this MAC and were processed by the NS offload logic to the RX datapath interface. 0b = Sink received NS frames. 1b = Pass received NS frames to the RX datapath interface.

TABLE 74: WUCSR2 (CONTINUED)

Wake-up and Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
8	NA_SA_SEL	R/W NASR	NA SA Select Used to select source for IPv6 SA in NA message. When set, NSx IPv6 Destination Address Register (NSx_IPV6_ADDR_DEST) value is used as the source. When cleared, the Target Address in NS packet is used.
7	NS_RCD	R/W1C NASR	NS Packet Received The MAC sets this bit upon receiving a valid NS packet.
6	ARP_RCD	R/W1C NASR	ARP Packet Received The MAC sets this bit upon receiving a valid ARP packet.
5	IPV6_TCPSYN_RCD	R/W1C NASR	IPv6 TCP SYN Packet Received The MAC sets this bit upon receiving a valid IPv6 TCP SYN packet. This bit will not set if IPv6 TCP SYN Wake Enable (IPV6_TCPSYN_WAKE_EN) is cleared. This bit will not set if either: RFE Wake-up Frame Received (RFE_WAKE_FR) Perfect DA Frame Received (PFDA_FR) Remote Wake-up Frame Received (WUFR) Magic Packet Received (MPR) Broadcast Frame Received (BCAST_FR) IPv4 TCP SYN Packet Received (IPV4_TCPSYN_RCD) are already set. This bit is automatically cleared when the Endpoint sends the PM_PME message due to a wake-up event if Resume Clears Remote Wake-up Status (RES_CLR_WKP_STS) is set.

TABLE 74: WUCSR2 (CONTINUED)

Wake-up and Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
4	IPV4_TCPSYN_RCD	R/W1C NASR	IPv4 TCP SYN Packet Received The MAC sets this bit upon receiving a valid IPv4 TCP SYN packet. This bit will not set if IPv4 TCP SYN Wake Enable (IPV4_TCPSYN_WAKE_EN) is cleared. This bit will not set if either: RFE Wake-up Frame Received (RFE_WAKE_FR) Perfect DA Frame Received (PFDA_FR) Remote Wake-up Frame Received (WUFR) Magic Packet Received (MPR) Broadcast Frame Received (BCAST_FR) IPv6 TCP SYN Packet Received (IPV6_TCPSYN_RCD) are already set. This bit is automatically cleared when the Endpoint sends the PM_PME message due to a wake-up event if Resume Clears Remote Wake-up Status (RES_CLR_WKP_STS) is set.
3	NS_OFFLOAD_EN	R/W NASR	NS Offload Enable When set, enables the response to Neighbor Solicitation packets.
2	ARP_OFFLOAD_EN	R/W NASR	ARP Offload Enable When set, enables the response to ARP packets.
1	IPV6_TCPSYN_WAKE_EN	R/W NASR	IPv6 TCP SYN Wake Enable When set, enables the wake-up on receiving an IPv6 TCP SYN packet. This bit is automatically cleared when the Endpoint sends the PM_PME message due to a wake-up event if Resume Clears Remote Wake-up Enables (RES_CLR_WKP_EN) is set. **Note:** This field is protected by Reset Protection (RST_ PROTECT).
0	IPV4_TCPSYN_WAKE_EN	R/W NASR	IPv4 TCP SYN Wake Enable When set, enables the wake-up on receiving an IPv4 TCP SYN packet. This bit is automatically cleared when the Endpoint sends the PM_PME message due to a wake-up event if Resume Clears Remote Wake-up Enables (RES_CLR_WKP_EN) is set. **Note:** This field is protected by Reset Protection (RST_ PROTECT).

The Receive Filtering Engine Control Register details are shown in Table 75.

TABLE 75: RFE_CTL

Receive Filtering Engine Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:19	Reserved	R	Always 0's
18	EN_OTHR_ROUTE_HDR	R/W NASR	Enable Other Routing Headers This bit allows the usage of IPv6 Routing headers other than type 0 and 2 when validating the UDP, TCP or ICMPv6 checksum for RX checksum offloading. When cleared, IPv6 Routing headers other than type 0 and 2 are not supported and the checksum is not verified. When set, IPv6 Routing headers other than type 0 and 2 are skipped, if the Segments Left field in the header is zero, otherwise, the checksum is not verified. **Note:** The raw checksum is still calculated regardless of this bit.
17:16	DFLT_REC_CHAN	R/W NASR	Default Receive Channel This field specifies the default receive channel.
15	PASS_WKP	R/W NASR	Always Pass Wake-up Frame When set, the RFE shall never discard a received wake-up frame which awakened the device while in D3 and Store Wake-up Frame (STORE_WAKE) is set.
14	EN_IGMP_CHK_VAL	R/W NASR	Enable IGMP Checksum Validation When set, the RFE will check the IGMP checksum. Additionally, the RFE calculates the L3 raw checksum and inserts it into RX Command B. **Note:** If the frame is not IGMP raw checksum is still calculated.
13	EN_ICMP_CHK_VAL	R/W NASR	Enable ICMP Checksum Validation When set, the RFE will check the ICMP checksum. Additionally, the RFE calculates the L3 raw checksum and inserts it into RX Command B. **Note:** If the frame is not ICMP raw checksum is still calculated.
12	EN_TCPUDP_CHK_VAL	R/W NASR	Enable TCP/UDP Checksum Validation When set, the RFE will check the TCP or UDP checksum. Additionally, the RFE calculates the L3 raw checksum and inserts it into RX Command B. **Note:** If the frame is not TCP or UDP the raw checksum is still calculated.

TABLE 75: RFE_CTL (CONTINUED)

Receive Filtering Engine Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
11	EN_IP_CHK_VAL	R/W NASR	Enable IP Checksum Validation When set, the RFE will check the IP checksum. This bit has no effect if the frame is not IPv4 or IPv6.
10	AB	R/W NASR	Accept Broadcast Frames When set, all broadcast frames are accepted. Otherwise broadcast frames are dropped.
9	AM	R/W NASR	Accept Multicast Frames When set, all multicast frames are accepted. Otherwise multicast frames must pass the perfect filtering or hash filtering. **Note:** This bit does not apply to broadcast frames.
8	AU	R/W NASR	Accept Unicast Frames When set, all unicast frames are accepted.
7	EN_VLAN_TAG_STRIPPING	R/W NASR	Enable VLAN Tag Stripping When set, this bit enables stripping of a received frame's VLAN ID.
6	UF	R/W NASR	Untagged Frame Filtering When set, all untagged receive frames are discarded.
5	VF	R/W NASR	Enable VLAN Filtering When set, this bit enables filtering of a received frame's VLAN ID.
4	SPF	R/W NASR	Enable Source Address Perfect Filtering When set, this bit enables perfect filtering of a received frame's Ethernet source address. **Note:** If destination address filtering is enabled (perfect or hash), the frame must pass both source address filtering and destination address filtering to not be discarded.
3	MHF	R/W NASR	Enable Multicast Address Hash Filtering When set, multicast destination addresses will be hashed. **Note:** The broadcast address is never hashed.
2	DHF	R/W NASR	Enable Destination Address Hash Filtering When set, unicast destination addresses will be hashed.
1	DPF	R/W NASR	Enable Destination Address Perfect Filtering When set, this bit enables perfect filtering of a received frame's Ethernet destination address.

TABLE 75: RFE_CTL (CONTINUED)

Receive Filtering Engine Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
0	RST_RECV_FILT_ENG	SC	Reset Receive Filtering Engine When set, this bit resets the RFE. Register bits marked as NASR are not Reset.

The DMA Controller Configuration Register details are shown in Table 76.

TABLE 76: DMAC_CFG

DMA Controller Configuration Register			Default: 0x0000_0360
Bits	**Name**	**R/W**	**Description**
31:19	Reserved	R	Always 0's
18	DMA_INTR_DSCR_RD_EN	R/W NASR	DMA Inter Descriptor Space Read Enable When set, the DMAC will read the reserved space between descriptors. **Note:** Software should not modify this field unless all DMA channels are in the Initial state as indicated by the Start Receive x (STRT_R_x)/Stop Receive x (STP_R_x) and Start Transmit x (STRT_T_x)/Stop Transmit x (STP_T_x) bits. One bit controls all RX and TX channels.
17	DMA_INTR_DSCR_WR_EN	R/W NASR	DMA Inter Descriptor Space Write Enable When set, the DMAC will overwrite the reserved space between descriptors. **Note:** Software should not modify this field unless all RX DMA channels are in the Initial state as indicated by the Start Receive x (STRT_R_x)/Stop Receive x (STP_R_x) bits. One bit controls all RX channels.
16	DMA_COAL_EN	R/W NASR	DMA Coalescing Enable When set, DMA coalescing is enabled. **Note:** One bit controls all RX and TX channels. Each channel can have coalescing independently disabled.
15:13	DMA_CMPL_RETRY_CNT	R/W NASR	DMA Completion Retry Count This field specifies the number of retries the DMA will use if a read request resulted in a timeout or had a poisoned indication. **Note:** Software should not modify this field unless all DMA channels are in the Initial state as indicated by the Start Receive x (STRT_R_x)/Stop Receive x (STP_R_x) and Start Transmit x (STRT_T_x)/Stop Transmit x (STP_T_x) bits. One field controls all RX and TX channels.

TABLE 76: DMAC_CFG (CONTINUED)

DMA Controller Configuration Register			Default: 0x0000_0360
Bits	**Name**	**R/W**	**Description**
12	DMA_CMPL_RETRY_EN	R/W NASR	DMA Completion Retry Enable When set, the DMA will retry read requests that resulted in a timeout or had a poisoned indication. **Note:** Software should not modify this field unless all DMA channels are in the Initial state as indicated by the Start Receive x (STRT_R_x)/Stop Receive x (STP_R_x) and Start Transmit x (STRT_T_x)/Stop Transmit x (STP_T_x) bits. One bit controls all RX and TX channels.
11:10	DMA_CH_ARB_SEL	R/W NASR	DMA Channel Arbitration Select This field controls the DMA channel arbitration scheme. 00 = Fixed priority - RX higher than TX 01 = Fixed priority - Channel number order (RX higher than TX when channel number is equal) 10 = Fixed priority - RX higher than TX. Round robin priority within channels 11 = Round robin. **Note:** Software should not modify this field unless all DMA channels are in the Initial state as indicated by the Start Receive x (STRT_R_x)/Stop Receive x (STP_R_x) and Start Transmit x (STRT_T_x)/Stop Transmit x (STP_T_x) bits. One field controls all RX and TX channels.
9	TX_RLS_ARB_DESC_REQ	R/W NASR	TX Release Arbiter on Descriptor Request For TX DMA descriptor reads, this bit controls when the DMAC PCIe arbiter is released and may service other PCIe requests from the DMAC. 0b = Release arbiter only after Read Completion is returned. This setting blocks ALL further DMAC PCIe requests from being serviced. 1b = Release arbiter after the Read Request is transmitted. This setting allows for multiple outstanding TX descriptor read requests (one per channel) as well as other concurrent PCIe read and write requests.
8	RX_RLS_ARB_DESC_REQ	R/W NASR	RX Release Arbiter on Descriptor Request For RX DMA descriptor reads, this bit controls when the DMAC PCIe arbiter is released and may service other PCIe requests from the DMAC. 0b = Release arbiter only after Read Completion is returned. This setting blocks ALL further DMAC PCIe requests from being serviced. 1b = Release arbiter after the Read Request is transmitted. This setting allows for multiple outstanding RX descriptor read requests (one per channel) as well as other concurrent PCIe read and write requests.
7	Reserved	R	Always 0

TABLE 76: DMAC_CFG (CONTINUED)

DMA Controller Configuration Register			Default: 0x0000_0360
Bits	**Name**	**R/W**	**Description**
6:4	MAX_READ_REQ	R/W NASR	Maximum Outstanding Data Read Requests This field limits the number of maximum outstanding DMA read requests for TX data. Valid values are 1 through 6. **Note:** Software should not modify this field unless all TX DMA channels are in the Initial state as indicated by the Start Transmit x (STRT_T_x)/Stop Transmit x (STP_T_x) bits. One field controls all TX channels.
3:2	Reserved	R	Always 0's
1:0	DSPACE	R/W NASR	Descriptor Spacing The size, in bytes, of the block reserved for each descriptor in memory is specified by this field. The descriptor itself will reside in the first 16 bytes of the reserved block. This field is programmed by the host to match the cache line size of the host CPU. 00b = 16 01b = 32 10b = 64 11b = 128 **Note:** Software should not modify this field unless all DMA channels are in the Initial state as indicated by the Start Receive x (STRT_R_x)/Stop Receive x (STP_R_x) and Start Transmit x (STRT_T_x)/Stop Transmit x (STP_T_x) bits. One field controls all RX and TX channels.

The DMA Controller Command Register details are shown in Table 77.

TABLE 77: DMAC_CMD

DMA Controller Command Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31	DMAC_SWR	R/W1S/SC	DMA Software Reset When this bit is set, the entire DMA Engine is Reset. Register bits for the channel marked as NASR are not Reset. Writing a 0 has no effect. This is a self-clearing bit. **Note:** This is an immediate Reset and will cause any in process PCIe request to be aborted. This may likely result in a corrupt PCIe packet.
30:29	Reserved	R	Always 0's
28	DMA_COAL_EXIT	R/W1S/SC	DMA Coalescing Exit Writing a one to this bit will force the DMA Controller to exit the Coalescing state. **Note:** One bit controls all RX and TX channels.

TABLE 77: DMAC_CMD (CONTINUED)

DMA Controller Command Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
27:24	TX_SWR_x	R/W1S/SC	TX DMA Software Reset x When this bit is set, the Transmit DMA Engine for channel x is Reset, including pending and completed requests for the channel within the reorder buffer. Register bits for the channel marked as NASR are not Reset. Writing a 0 has no effect. This is a self-clearing bit. This is an immediate Reset and will cause any in process PCIe request to be aborted. This may likely result in a corrupt PCIe packet.
23:20	STRT_T_x	R/W1S	Start Transmit x When this bit is written as a 1, the TX DMAC for channel x is either started (if in the Initial state) or resumes operation (if in the Stopped state). The Start Transmit command is effective only when the TX DMAC is not running (Initial or Stopped states). Writing this bit while in the Started or Stop Pending states has no effect. **Note:** Before the Start Transmit Command is initially issued a valid base address for the ring must be programmed into TX Channel x Ring Base Address High Register (TX_BASE_ADDRHx) TX Channel x Ring Base Address Low Register (TX_BASE_ADDRLx); and the Ring Length and other parameters programmed into TX Channel x Configuration A Register (TX_CFG_Ax) and TX Channel x Configuration B Register (TX_CFG_Bx); and if head pointer write-back is used, TX Channel x Head Write-Back Address High Register (TX_HEAD_WRITEBACK_ADDRHx) TX Channel x Head Write-Back Address LOW Register (TX_HEAD_WRITEBACK_ADDRLx) otherwise the TX DMAC's behavior will be undefined. This bit will clear once Stop Transmit request is completed. The reading of this field along with the Stop Transmit x (STP_T_x) field indicates the Current state of the TX DMAC channel. STRT_T_x/STP_T_x: 00b - Initial 10b - Started 11b - Stop Pending 01b - Stopped

TABLE 77: DMAC_CMD (CONTINUED)

DMA Controller Command Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
19:16	STP_T_x	R/W1S	Stop Transmit x When this bit is written as a 1, the TX DMAC for channel x enters the Stopped state. The Stop Transmit command is effective only when the TX DMAC is in the Started state. Writing this bit while in the Initial, Stop Pending or Stopped states has no effect. **Note:** Unless otherwise noted, software should not modify any DMA parameters that affect the Stopped channel. The channel should be set to the Initial state via a Reset command before changing parameters. This bit will clear once Start Transmit request is issued. The reading of this field along with the Start Transmit x (STRT_T_x) indicates the current state of the TX DMAC channel. STRT_T_x/STP_T_x: 00b - Initial 10b - Started 11b - Stop Pending 01b - Stopped
15:12	Reserved	R	Always 0's
11:8	RX_SWR_x	R/W1S/SC	RX DMA Software Reset When this bit is set, the Receive DMA Engine for channel x is Reset. Register bits for the channel marked as NASR are not Reset. Writing a 0 has no effect. This is a self-clearing bit. This is an immediate Reset and will cause any in process PCIe request to be aborted. This may likely result in a corrupt PCIe packet.

TABLE 77: DMAC_CMD (CONTINUED)

DMA Controller Command Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
7:4	STRT_R_x	R/W1S	Start Receive When this bit is written as a 1, the RX DMAC for channel x is either started (if in the Initial state) or resumes operation (if in the Stopped State). The Start Receive command is effective only when the RX DMAC is not running (Initial or Stopped states). Writing this bit while in the Started or Stop Pending states has no effect. **Note:** Before the Start Receive command is initially issued, a valid base address for the ring must be programmed into RX Channel x Ring Base Address High Register (RX_BASE_ADDRHx) RX Channel x Ring Base Address Low Register (RX_BASE_ADDRLx); and the Ring Length and other parameters programmed into RX Channel x Configuration A Register (RX_CFG_Ax) and RX Channel x Configuration B Register (RX_CFG_Bx); and if head pointer write-back is used, RX Channel x Head Write-Back Address High Register (RX_HEAD_WRITEBACK_ADDRHx) RX Channel x Head Write-Back Address LOW Register (RX_HEAD_WRITEBACK_ADDRLx) otherwise the RX DMAC's behavior will be undefined. This bit will clear once Stop Receive request is completed. The reading of this field along with the Stop Receive x (STP_R_x) field indicates the current state of the RX DMAC channel. STRT_R_x / STP_R_x: 00b - Initial 10b - Started 11b - Stop Pending 01b - Stopped

TABLE 77: DMAC_CMD (CONTINUED)

DMA Controller Command Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
3:0	STP_R_x	R/W1S	Stop Receive When this bit is written as a 1, the RX DMAC for channel x enters the Stopped state. The Stop Receive command is effective only when the RX DMAC is in the Started state. Writing this bit while in the Initial, Stop Pending or Stopped states has no effect. **Note:** Unless otherwise noted, software should not modify any DMA parameters that affect the Stopped channel, The channel should be set to the Initial state via a Reset command before changing parameters. This bit will clear once Start Receive request is issued. The reading of this field along with the Start Receive x (STRT_R_x) indicates the current state of the RX DMAC channel. STRT_R_x/STP_R_x: 00b - Initial 10b - Started 11b - Stop Pending 01b - Stopped

The DMA Controller Interrupt Status Register details are shown in Table 78.

TABLE 78: DMAC_INT_SYS

DMA Controller Interrupt Status Register			Default: 0x0000_FF00
Bits	**Name**	**R/W**	**Description**
31:28	Reserved	R	Always 0's
27:24	RXPRIx	R	RX Priority Frame x Status Each separate status bit indicates that the current corresponding interrupt is due to a frame classified as a priority frame. Each bit is typically set when the corresponding interrupt bit is set. A bit could remain unset upon an initial setting of the corresponding interrupt bit and be set if a subsequent priority frame occurs. The bits stay set even if a subsequent non-priority frame occurs. The bit is cleared when the corresponding interrupt bit is cleared (see Section 13.0, "Configuration File Formatting" for clearing conditions). **Note:** A simultaneous setting, due to a new priority frame, and clearing will result in the bit (and corresponding interrupt bit) staying set. There is the potential for a bit to be unset on an initial interrupt and set when a subsequent priority frame is received. If these two events overlap with software, responding to the initial interrupt, reading this register and clearing the corresponding interrupt bit, the priority status may be lost. Software should use the PRI bit from the receive descriptor to determine if a priority frame was received.
23:22	Reserved	R	Always 0's
21	DMA_ERR_INT	R	DMA Error Interrupt This interrupt is set when any status bit in the RX Channel x Error Status Register (RX_ERR_STSx) or TX Channel x Error Status Register (TX_ERR_STSx) is set. RX and TX Tail Pointer errors and TX Sequence errors are reported here only if they are enabled. This bit cascades to the DMA_GEN_INT bit in the INT_STS register. **Note:** This is a level-triggered interrupt event that remains asserted until all the error event bits in the RX Channel x Error Status Register (RX_ERR_STSx) and TX Channel x Error Status Register (TX_ERR_STSx) are cleared.
20	Reserved	R	Always 0

Note: This register contains the current status of the DMA interrupt sources. Unless otherwise noted, writing a '1' to the corresponding bits acknowledges and clears the interrupt. Interrupt status bits in this register reflect the state of the interrupt source regardless of the state of the corresponding enable.

TABLE 78: DMAC_INT_SYS (CONTINUED)

DMA Controller Interrupt Status Register			Default: 0x0000_FF00
Bits	**Name**	**R/W**	**Description**
19:16	RXFRMx_INT	R/W1C NASR	RX Frame x Interrupt Each separate interrupt is set when a frame is transferred to host memory by the corresponding channel. These bits cascade to the DMA_RXx_INT bits in the INT_STS register. **Note:** These are edge-triggered interrupt events.
15:12	RXx_STOP_INT	R/W1C NASR	RX DMA x Stopped Interrupt Each separate interrupt is set whenever RX DMA channel x enters into the Stopped state as a result of processing a stop command. These bit cascades to the DMA_GEN_INT bit in the INT_STS register. **Note:** This is an edge-triggered interrupt.
11:8	TXx_STOP_INT	R/W1C NASR	TX DMA x Stopped Interrupt Each separate interrupt is set whenever TX DMA channel x enters into the Stopped state as a result of processing a stop command. This bit cascades to the DMA_GEN_INT bit in the INT_STS register. **Note:** This is an edge-triggered interrupt event.
7:4	Reserved	R	Always 0's
3:0	TXx_INT	R/W1C NASR	TX x Interrupt Each separate interrupt is set whenever the TX DMA engine for channel x closes a transmit descriptor whose IOC or LS bit is set as determined by the TX Timer and Head Pointer Write-back Select (TX_TMR_HPWB_SEL) field. The interrupt may be immediate or delayed. If the transmit descriptor also has the DTI bit set, this interrupt is set after a delay. This bit cascades to the DMA_TXx_INT bit in the INT_STS register. **Note:** This is an edge-triggered interrupt event.

Note: This register contains the current status of the DMA interrupt sources. Unless otherwise noted, writing a '1' to the corresponding bits acknowledges and clears the interrupt. Interrupt status bits in this register reflect the state of the interrupt source regardless of the state of the corresponding enable.

The SGMII Control Register details are shown in Table 79.

TABLE 79: SGMII_CTL

SGMII Control Register			Default: 0x0000_0080
Bits	**Name**	**R/W**	**Description**
31	SGMII_ENABLE	R/W	When set, the MAC will operate in SGMII/1000/2500BASE-X mode.
30:8	Reserved	R	Always 0's

TABLE 79: SGMII_CTL (CONTINUED)

SGMII Control Register			Default: 0x0000_0080
Bits	**Name**	**R/W**	**Description**
7:5	TX_VBOOST_LVL	R/W	Connects to the TX_VBOOST_LVL[2:0] input of the SGMII/ 1000/2500BASE-X PHY
4	TX_VBOOST_EN	R/W	Connects to the TX0_VBOOST_EN input of the SGMII/ 1000/2500BASE-X PHY
3	RBULK_SHORT	R/W	Reduces bulk to GD resistance when enabled for the SGMII/ RGMII Pad Mux 0b = 10 KOhms 1b = 1 KOhms
2	SGMII_CP_DISABLE	R/W	This bit is the controller's control for the charge pump in the SGMII/RGMII Pad Mux. 0b = Charge pump is controlled by the SGMII/RGMII Pad Mux based on RGMII/SGMII mode and the analog voltage. 1b = Charge pump is disabled. **Note:** In actual usage, since the SGMII/RGMII Pad Mux is powered with 3.3V, the charge pump is never enabled.
1	SGMII_PWR_DN	R/W	SGMII Power-Down When set the SGMII/1000/2500BASE-X interface is disabled and in power-down.
0	SGMII_RESET	R/W	SGMII Reset When set the SGMII/1000/2500BASE-X interface is held in Reset.

The Interrupt Status Register details are shown in Table 80.

TABLE 80: INT_SYS

Interrupt Status Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:28	Reserved	R	Always 0's
27:24	DMA_RXx_INT	R/W1C	DMA RX Channel 3:0 Interrupt Indicates a DMA controller interrupt event from an RX channel. This bit is set whenever the following bits in the DMA Controller Interrupt Status Register (DMAC_INT_STS) are set: • RXFRMx_INT In addition to other clearing methods, the RXFRMx_INT and RXPRIx bits in the DMA Controller Interrupt Status Register (DMAC_INT_STS) are cleared when this bit is written with a one. **Note:** The source of this interrupt is a level. The interrupt persists until the bits in the DMA controller are cleared or disabled.
23:20	Reserved	R	Always 0's

TABLE 80: INT_SYS (CONTINUED)

Interrupt Status Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
19:16	DMA_TXx_INT	R	DMA TX Channel x Interrupt Indicates a DMA controller Interrupt event from a TX channel. This bit is set whenever the following bits in the DMA Controller Interrupt Status Register (DMAC_INT_STS) are set: • TXx_INT In addition to other clearing methods, the TXx_INT bits in the DMA Controller Interrupt Status Register (DMAC_INT_STS) are cleared when this register is written with a one. **Note:** The source of this interrupt is a level. The interrupt persists until the bits in the DMA controller are cleared or disabled.
15:11	Reserved	R	Always 0's
10	DMA_GEN_INT	R	DMA General Interrupt Indicates a general DMA controller Interrupt event. This bit is set whenever the following bits in the DMA Controller Interrupt Status Register (DMAC_INT_STS) are set: • DMA_ERR_INT • RXx_STOP_INT • TXx_STOP_INT **Note:** The source of this interrupt is a level. The interrupt persists until the bits in the DMA controller are cleared or disabled.
9	SW_GP_INT	R/W1C	Software General Purpose This interrupt is issued when the Software General Purpose Interrupt Set (SW_GP_INT_SET) bit in the Interrupt Set Register (INT_SET) is written high. There is no associated hardware event to set this field. **Note:** The source of this interrupt is a pulse.
8	PCIe_INT	R	PCIe Interrupt Indicates a PCIe Interrupt event. This bit is set whenever any enabled bits in the PCIe Interrupt Status Register (PCIe_INT_STS) are set. **Note:** The source of this interrupt is a level. The interrupt persists until the bits in the PCIe Interrupt Status Register (PCIe_INT_STS) are cleared or disabled.

TABLE 80: INT_SYS (CONTINUED)

Interrupt Status Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
7	1588_INT	R	1588 Interrupt Indicates a 1588 PTP Interrupt event. This bit is set whenever any enabled bits in the PTP Interrupt Status Register (PTP_INT_STS) are set. **Note:** The source of this interrupt is a level. The interrupt persists until the bits in the 1588 block are cleared or disabled.
6	SGMII_AN_DONE_INT	R	SGMII/1000/2500BASE-X Auto-Negotiation Done Interrupt Indicates an SGMII/1000/2500BASE-X Interrupt event. This bit is set when the SGMII/1000/2500BASE-X interface logic asserts its interrupt output signal. **Note:** The source of this interrupt is a level. The interrupt persists until the bits in the SGMII/1000/2500BASE-X interface are cleared or disabled.
5	ETH_PHY_INT	R	Ethernet PHY Interrupt Indicates an Ethernet PHY Interrupt event. This bit is set when the ENET_PHY_INT_N pin is active. **Note:** The source of this interrupt is a level. The interrupt persists until the bits in the PHY are cleared or disabled or the ENET_PHY_INT_N pin is inactive.
4	DP_INT	R/W1C	Data Port Interrupt Indicates that a pending data port operation has been completed. **Note:** The source of this interrupt is a pulse.
3	MAC_INT	R	MAC Interrupt Indicates an Ethernet MAC interrupt event. This bit is set whenever any enabled status bits in the MAC Interrupt Status Register (MAC_INT_STS) are set. **Note:** The source of this interrupt is a level. The interrupt persists until the bits in the MAC block are cleared or disabled.
2	FCT_INT	R	FCT Interrupt Indicates a FIFO controller interrupt event. This bit is set whenever any enabled status bits in the FIFO Controller Interrupt Status Register (FCT_INT_STS) are set. **Note:** The source of this interrupt is a level. The interrupt persists until the bits in the FCT block are cleared or disabled.

TABLE 80: INT_SYS (CONTINUED)

Interrupt Status Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
1	GPT_INT	R/W1C	GP Timer Interrupt This interrupt is issued when the General Purpose Timer reaches zero. **Note:** The source of this interrupt is a pulse.
0	MAS_INT	R	Controller Interrupt This bit reflects the "ORing" of all the enabled bits in this register. Disabled interrupts (via the individual enables) do not contribute to this bit.

The Interrupt Set Register details are shown in Table 81.

TABLE 81: INT_SET

Interrupt Set Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:28	Reserved	R	Always 0's
27:24	DMA_RXx_INT_EN_SET	R/W1C	DMA RX Channel 3:0 Interrupt Enable Set
23:20	Reserved	R	Always 0's
19:16	DMA_TXx_INT_EN_SET	R/W1C	DMA TX Channel x Interrupt Enable Set
15:11	Reserved	R	Always 0's
10	DMA_GEN_INT_EN_SET	R/W1C	DMA General Interrupt Enable Set
9	SW_GP_INT_EN_SET	R/W1C	Software General Purpose Interrupt Enable Set
8	PCIe_INT_EN_SET	R/W1C	PCIe Interrupt Enable Set
7	1588_INT_EN_SET	R/W1C	1588 Interrupt Enable Set
6	SGMII_AN_DONE_INT_EN_SET	R/W1C	SGMII/1000/2500BASE-X Auto-Negotiation Done Interrupt Enable Set
5	ETH_PHY_INT_EN_SET	R/W1C	Ethernet PHY Interrupt Enable Set
4	DP_INT_EN_SET	R/W1C	Data Port Interrupt Enable Set
3	MAC_INT_EN_SET	R/W1C	MAC Interrupt Enable Set
2	FCT_INT_EN_SET	R/W1C	FCT Interrupt Enable Set
1	GPT_INT_EN_SET	R/W1C	GP Timer Interrupt Enable Set
0	MAS_INT_EN_SET	R/W1C	Controller Interrupt Enable Set This bit is an additional enable that affects all the main interrupts sources.

Note: This register may be used by software to set any of the non-reserved bits in the Interrupt Status Register (INT_STS).

The Interrupt Enable Set Register details are shown in Table 82.

TABLE 82: INT_EN_SET

Interrupt Enable Set Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:28	Reserved	R	Always 0's
27:24	MA_RXx_INT_EN_SET	R/W1C	DMA RX Channel 3:0 Interrupt Enable Set
23:20	Reserved	R	Always 0's
19:16	DMA_TXx_INT_EN_SET	R/W1C	DMA TX Channel x Interrupt Enable Set
15:11	Reserved	R	Always 0's
10	DMA_GEN_INT_EN_SET	R/W1C	DMA General Interrupt Enable Set
9	SW_GP_INT_EN_SET	R/W1C	Software General Purpose Interrupt Enable Set
8	PCIe_INT_EN_SET	R/W1C	PCIe Interrupt Enable Set
7	1588_INT_EN_SET	R/W1C	1588 Interrupt Enable Set
6	SGMII_AN_DONE_INT_EN_SET	R/W1C	SGMII/1000/2500BASE-X Auto-Negotiation Done Interrupt Enable Set
5	ETH_PHY_INT_EN_SET	R/W1C	Ethernet PHY Interrupt Enable Set
4	DP_INT_EN_SET	R/W1C	Data Port Interrupt Enable Set
3	MAC_INT_EN_SET	R/W1C	MAC Interrupt Enable Set
2	FCT_INT_EN_SET	R/W1C	FCT Interrupt Enable Set
1	GPT_INT_EN_SET	R/W1C	GP Timer Interrupt Enable Set
0	MAS_INT_EN_SET	R/W1C	Controller Interrupt Enable Set This bit is an additional enable that affects all the main interrupts sources.

Note: This register is used to set the interrupt enables for the corresponding bits in the Interrupt Status Register (INT_STS). Writing a '1' to a bit sets the corresponding enable and configures the corresponding interrupt as a source for a system interrupt. Writing a '0' has no effect. A read of this register returns the state of the interrupt enables.

The Interrupt Enable Clear Register details are shown in Table 83.

TABLE 83: INT_EN_CLR

Interrupt Enable Clear Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:28	Reserved	R	Always 0's
27:24	MA_RXx_INT_EN_CLR	R/W1C	DMA RX Channel 3:0 Interrupt Enable Clear
23:20	Reserved	R	Always 0's
19:16	DMA_TXx_INT_EN_CLR	R/W1C	DMA TX Channel x Interrupt Enable Clear
15:11	Reserved	R	Always 0's
10	DMA_GEN_INT_EN_CLR	R/W1C	DMA General Interrupt Enable Clear
9	SW_GP_INT_EN_CLR	R/W1C	Software General Purpose Interrupt Enable Clear
8	PCIe_INT_EN_CLR	R/W1C	PCIe Interrupt Enable Clear
7	1588_INT_EN_CLR	R/W1C	1588 Interrupt Enable Clear

Note: This register is used to clear the interrupt enables for the corresponding bits in the Interrupt Status Register (INT_STS). Writing a '1' to a bit clears the corresponding enable. Writing a '0' has no effect. A read of this register returns the state of the interrupt enables.

TABLE 83: INT_EN_CLR (CONTINUED)

Interrupt Enable Clear Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
6	SGMII_AN_DONE_INT_EN_CLR	R/W1C	SGMII/1000/2500BASE-X Auto-Negotiation Done Interrupt Enable Clear
5	ETH_PHY_INT_EN_CLR	R/W1C	Ethernet PHY Interrupt Enable Clear
4	DP_INT_EN_CLR	R/W1C	Data Port Interrupt Enable Clear
3	MAC_INT_EN_CLR	R/W1C	MAC Interrupt Enable Clear
2	FCT_INT_EN_CLR	R/W1C	FCT Interrupt Enable Clear
1	GPT_INT_EN_CLR	R/W1C	GP Timer Interrupt Enable Clear
0	MAS_INT_EN_CLR	R/W1C	Controller Interrupt Enable Clear This bit is an additional enable that affects all the main interrupts sources

Note: This register is used to clear the interrupt enables for the corresponding bits in the Interrupt Status Register (INT_STS). Writing a '1' to a bit clears the corresponding enable. Writing a '0' has no effect. A read of this register returns the state of the interrupt enables.

9.0 PERIPHERAL SUBSYSTEM IMPLEMENTATION - UART

The UART Peripheral Subsystem occupies its own PCI device endpoint and requires a separate device driver to function from a host system. This subsystem shares the PCI bus with all other peripheral devices (I^2C, SPI, GPIO).

The Subsystem IDs are:
- Vendor ID: 0x1055 (Microchip Technology, Inc/SMSC)
- Device ID:
  - PCI12000: 0xA002
  - PCI11010: 0xA012
  - PCI11400: 0xA032
  - PCI11101: 0xA022
  - PCI11414: 0xA042
- Class ID: 0x070002 (“Serial 16550”)

9.1 Sections

9.2 Summary of Features

- Up to four instances of UART each with a full complement of wires:
  - TXD
  - RXD
  - nRTS
  - nCTS
  - nDTR
  - nDSR
  - nDCD
  - nRI
- Programmable word length, stop bits and parity
- Programmable baud rate generator
- Interrupt generator
- Loopback mode
- Interface registers
- 256-byte Transmit FIFO
- 256-byte Receive FIFO
- Multiple clock sources
- Pin polarity control
- Low-power sleep mode

9.2.1 RS-232 SUPPORT

A typical application uses a minimum of four wires known as RS-232:
- TXD
- RXD
- nRTS
- nCTS

9.2.2 RS-485 SUPPORT

For industrial RS-485 use cases, an auto-direction feature is provided which is used to tri-state the transmitter via the nRTS or nCTS lines when no other data is available. This allows other nodes to use the shared lines.

9.3 Transmit Operation

Transmission is initiated by writing the data to be sent to the UART_TX_DATA buffer. The data is then transferred to the TX Shift Register together with a start bit and parity and stop bits as determined by settings in the Line Control Register. The bits to be transmitted are then shifted out of the TX Shift Register in the order Start bit, Data bits (LSB first), Parity bit, Stop bit, using the output from the Baud Rate Generator (divided by 16) as the clock. If enabled, a TX Holding Register Empty interrupt will be generated when the UART_TX_DATA buffer is empty after the data in the buffer is transmitted.

PCI1xxxx can store up to 256 bytes of data for transmission at a time. Transmission will continue until the TX FIFO is empty. The FIFO readiness to accept more data is indicated by interrupt.

9.4 Receive Operation

Data is sampled into the RX Shift Register using the Receive clock, divided by 16 (by default). The Receive clock is provided either by the Baud Rate Generator. A filter is used to remove spurious inputs that last for less than two periods of the Receive clock. When the complete word has been clocked into the receiver, the data bits are transferred to the UART_RX_DATA to be read by the PCIe host. (The first bit of the data to be received is placed in bit 0 of this register.) The receiver also checks that the parity bit and stop bits are as specified by the Line Control Register.

If enabled, a RX Data Received interrupt will be generated when the data has been transferred to the RX Buffer Register or when the RX Trigger Level has been reached. Interrupts can also be generated to signal RX FIFO Character Timeout, incorrect parity, a missing stop bit (frame error) or other Line Status errors. The PCI1xxxx can store up to 256 bytes of received data at a time. Depending on the selected RX Trigger Level, interrupt will go active to indicate that data is available when the RX FIFO contains 2 to 250 Bytes of data based on the following calculation:

((UART_FIFO_CTL.RECV_FIFO_TRIG[4:0] * 8) + 2) Bytes

9.5 Baud Clock

Note: The baud rate can be configured conventionally via most common UART/COM port terminal application settings (such as minicom/PuTTY) and the PCI1xxxx UART driver will set the appropriate BAUD_CLOCK_DIV* register values. There is no need to manually set these registers in a typical application.

The baud clock is derived from an internal PLL in order to maintain the required baud rate accuracy across a range of frequencies; this is selected through CLK_SEL_REG.BAUD_CLOCK_SEL[1:0] with 62.5 MHz being the default.

500 MHz and 166.667 MHz clocks are also available, but don't offer any improved accuracy or additional baud rates beyond what is available with 62.5 MHz, so the associated settings are not shown within this document.

62.5 MHz clock divisor mode is selected by setting BAUD_CLK_SEL = 00b. A divisor value is then selected from CLK_DIVISOR_REG.BAUD_CLOCK_DIV_INT[23:0] to configured the final baud rate. See Table 84 for available values.

TABLE 84: 62.5 MHZ CLOCK DIVISOR MODE BAUD RATES

Baud Rate	Actual Baud Rate	Percentage Error	Samples/ UART Bit	BAUD_CLOCK_DIV_INT[23:0] Value	BAUD_CLOCK_DIV_FRAC[7:0] Value
50	50.0000	0.0	16	0x13_12D0	0x00
75	75.0000	0.0000	16	0x0C_B735	0x55
110	110.0000	0.0000	16	0x08_AB75	0xD1
134.5	134.5000	0.0000	16	0x07_172C	0xD4
150	150.0000	0.0000	16	0x06_5B9A	0xAA
300	300.0000	0.0000	16	0x03_2DCD	0x55
600	600.0000	0.0000	16	0x01_96E6	0xAA
1.2k	1,200.0000	0.0000	16	0x00_CB73	0x55
1.8k	1,799.9999	0.0000	16	0x00_87A2	0x39

TABLE 84: 62.5 MHZ CLOCK DIVISOR MODE BAUD RATES (CONTINUED)

Baud Rate	Actual Baud Rate	Percentage Error	Samples/ UART Bit	BAUD_CLOCK_DIV_INT[23:0] Value	BAUD_CLOCK_DIV_FRAC[7:0] Value
2k	2,000.0000	0.0000	16	0x00_7A12	0x00
2.4k	2,400.0000	0.0000	16	0x00_65B9	0xAA
3.6k	3,600.0003	0.0000	16	0x00_43D1	0x1C
4.8k	4,799.9993	0.0000	16	0x00_32DC	0xD5
7.2k	7,199.9989	0.0000	16	0x00_21E8	0x8E
9.6k	9,600.0014	0.0000	16	0x00_196E	0x6A
19.2k	19,200.0029	0.0000	16	0x00_0CB7	0x35
38.4k	38,400.0058	0.0000	16	0x00_065B	0x9A
57.6k	57,599.9393	-0.0001	16	0x00_043D	0x12
115.2k	115,200.2949	0.0003	16	0x00_021E	0x88
125k	125,000.0000	0.0000	16	0x00_01F4	0x00
136.4k	136,399.8151	-0.0001	16	0x00_01CA	0x36
150k	150,000.0000	0.0000	16	0x00_01A0	0xAA
166.7k	166,699.7887	-0.0001	16	0x00_0176	0xEC
187.5k	187,500.0000	0.0000	16	0x00_014D	0x55
214.3k	214,300.1210	0.0001	16	0x00_0123	0xA5
250k	250,000.0000	0.0000	16	0x00_00FA	0x00
300k	300,000.0000	0.0000	16	0x00_00D0	0x55
375k	375,000.0000	0.0000	16	0x00_00A6	0xAA
500k	500,000.0000	0.0000	16	0x00_007D	0x00
750k	750,000.0000	0.0000	16	0x00_0053	0x55
921.6k	921,615.6826	0.0017	16	0x00_0043	0xD0
1M	999,968.6284	-0.0031	16	0x00_003E	0x80
1.5M	1,500,000.0000	0.0000	16	0x00_0029	0xAA
2M	1,999,937.2569	-0.0031	16	0x00_001F	0x40
3M	2,999,717.6736	-0.0094	16	0x00_0014	0xD5
4M	3,999,874.5137	-0.0031	8	0x00_001F	0x40
5M	5,000,000.0000	0.0000	8	0x00_0019	0x00
6M	5,999,435.3473	-0.0094	8	0x00_0014	0xD5
7M	6,999,341.2385	-0.0094	8	0x00_0011	0xDB
8M	7,999,749.0275	-0.0031	4	0x00_001F	0x40
9M	9,000,423.5493	0.0047	4	0x00_001B	0xC6
10M	10,000,000.0000	0.0000	4	0x00_0019	0x00
11M	11,000,862.8128	0.0078	4	0x00_0016	0xB9
12M	11,998,870.6945	-0.0094	4	0x00_0014	0xD5
13M	12,999,592.1697	-0.0031	4	0x00_0013	0x3B
14M	13,998,682.5	-0.0094	4	0x00_0011	0xDB
15M	15,000,000.0	0.0000	4	0x00_0010	0xAA

9.6 UART Subsystem Registers

The PERI_SUBSYSTEM_ADDR_BASE = 0x0014_0000 registers are listed in Table 85.

Note: PCI1xxxx devices have thousands of registers which are either reserved for future use or the contents are masked from public documentation because they have no end system integrator use-case (i.e. they are used by hardware to perform low level run-time tasks or are used/controlled directly by a driver during run-time). **Do not modify any contents of undocumented registers**.

TABLE 85: UART SUBSYSTEM REGISTER OFFSETS FROM PERI_SUBSYSTEM_ADDR_BASE

Register Name	Port 1	Port 2	Port 3	Port 4
ADCL_CFG_REG	0x0_0040	0x0_0440	0x0_0840	0x0_0C40
CLK_SEL_REG	0x0_0050	0x0_0450	0x0_0850	0x0_0C50
CLK_DIVISOR_REG	0x0_0054	0x0_0454	0x0_0854	0x0_0C54
FRAC_DIV_CFG_REG	0x0_0058	0x0_0458	0x0_0858	0x0_0C58
UART_PAD_CTRL_PU_REG	0x0_0060	0x0_0460	0x0_0860	0x0_0C60
UART_PAD_CTRL_PD_REG	0x0_0064	0x0_0464	0x0_0864	0x0_0C64
UART_PAD_CTRL_OD_REG	0x0_0068	0x0_0468	0x0_0868	0x0_0C68
UART_PAD_CTRL_OPOL_REG	0x0_006C	0x0_046C	0x0_086C	0x0_0C6C
UART_PAD_CTRL_WAKE_DB	0x0_0070	0x0_0470	0x0_0870	0x0_0C70
UART_PAD_C-TRL_CTS_WAKE_DB	0x0_0074	0x0_0474	0x0_0874	0x0_0C74
UART_PCI_CTRL_REG	0x0_0080			
UART_INT_REG	0x0_0084	0x0_0484	0x0_0884	0x0_0C84
UART_INT_MASK_REG	0x0_0088	0x0_0488	0x0_0888	0x0_0C88
UART_WAKE_REG	0x0_008C	0x0_048C	0x0_088C	0x0_0C8C
UART_WAKE_MASK_REG	0x0_0090	0x0_0490	0x0_0890	0x0_0C90
UART_RESET_REG	0x0_0094			
UART_LTR_VALUE_REG	0x0_0098	0x0_0498	0x0_0898	0x0_0C98

The Auto-Direction Control Register details are shown in Table 86.

TABLE 86: ADCL_CFG_REG

Auto-Direction Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:3	Reserved	R	Always ‘0’
2	ADCL_POLARITY	R/W	Selects the polarity, either high or low, that the selected signal is driven to when the output buffer of the corresponding serial port is empty or full during Auto-Direction Control (ADCL). 0b: ‘0’ means empty, ‘1’ means full. 1b: ‘1’ means empty, ‘0’ means full.
1	ADCL_PIN_SEL	R/W	An enable bit to select which pin, either nRTS or nDTR, of the serial port is affected by Auto-Direction Control (ADCL). 0b: ADCL uses nDTR. 1b: ADCL uses nRTS.

TABLE 86: ADCL_CFG_REG (CONTINUED)

Auto-Direction Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
0	ADCL_EN	R/W	Enable or disable the Auto-Direction Control (ADCL). 0b: ADCL Disabled. 1b: ADCL Enabled.

The Auto-Direction Control Register details are shown in Table 87.

TABLE 87: CLK_SEL_REG

Clock Source Select Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:2	Reserved	R	Always return ‘0’
1:0	BAUD_CLOCK_SEL[1:0]	R/W	Selects the input baud clock: 00b: 62.5 MHz (default) 01b: 166.667 MHz (500 MHz/3) 10b: 500 MHz 11b: Reserved

The Clock Divisor Register details are shown in Table 88.

TABLE 88: CLK_DIVISOR_REG

Clock Divisor Register			Default: 0x0000_0015
Bits	**Name**	**R/W**	**Description**
31:8	BAUD_CLOCK_DIV_INT[23:0]	R/W	Selects integer part of the divisor to be applied to the baud clock, selected by CLK_SEL_REG.BAUD_CLOCK_SEL[1:0], in order to generate the correct baud rate.
7:0	BAUD_CLOCK_DIV_FRAC[7:0]	R/W	Selects fractional part of the divisor to be applied to the baud clock, selected by CLK_SEL_REG.BAUD_CLOCK_SEL[1:0], in order to generate the correct baud rate. **Note:** BAUD_CLOCK_DIV_FRAC[7:0] is only valid when CLK_SEL_REG.BAUD_CLOCK_SEL [1:0] is set to 0.

The Fractional Divisor Configuration Register details are shown in Table 89.

TABLE 89: FRAC_DIV_CFG_REG

Fractional Divisor Configuration Register			Default: 0x6EF7_1000
Bits	**Name**	**R/W**	**Description**
31:28	TX_HALF_POINT[3:0]	R/W	While transmitting the UART stop bit, the ‘Nth’ sample pulse is used to transmit the UART second half STOP bit. For the 16 sample method the 6th sample is used. For the 8 sample method the 2nd sample can be used.
27:24	TX_NEXT_END_POINT[3:0]	R/W	While transmitting the UART stop bit, the ‘Nth’ sample pulse is used to transmit the UART second full STOP bit. For the 16 sample method the 14th sample is used. For the 8 sample method the 6th sample can be used.

TABLE 89: FRAC_DIV_CFG_REG (CONTINUED)

Fractional Divisor Configuration Register			Default: 0x6EF7_1000
Bits	**Name**	**R/W**	**Description**
23:20	TX_END_POINT[3:0]	R/W	While transmitting the UART bits, the 'Nth' sample pulse is used to transmit the UART data. For the 16 sample method the 15th sample is used. For the 8 sample method the 7th sample can be used. This value is also used by both RX and TX engine to Reset the step counter thus representing the 16 sample method or 8 sample method.
19:16	RX_SAMPLE_POINT[3:0]	R/W	While receiving the UART bits, the 'Nth' sample of the bit is recorded as received bit. For the 16 sample method the 7th sample is used. For the 8 sample method the 3rd sample can be used.
15:0	SAMPLE_CLK_PERIOD[15:0]	R/W	This register value represents the step increase in the baud rate generator counter.

The UART Pad Control Pull-up Register details are shown in Table 90.

TABLE 90: UART_PAD_CTRL_PU_REG

UART Pad Control Register: Internal Pull-up Resistor			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:6	Reserved	R	Always '0'.
5	CTS_PU	R/W	0b: Internal Pull-up resistor disabled on the pad 1b: Internal Pull-up resistor enabled on the pad
4	RTS_PU	R/W	0b: Internal Pull-up resistor disabled on the pad 1b: Internal Pull-up resistor enabled on the pad
3	RI_PU	R/W	0b: Internal Pull-up resistor disabled on the pad 1b: Internal Pull-up resistor enabled on the pad
2	DSR_PU	R/W	0b: Internal Pull-up resistor disabled on the pad 1b: Internal Pull-up resistor enabled on the pad
1	DCD_PU	R/W	0b: Internal Pull-up resistor disabled on the pad 1b: Internal Pull-up resistor enabled on the pad
0	DTR_PU	R/W	0b: Internal Pull-up resistor disabled on the pad 1b: Internal Pull-up resistor enabled on the pad

The UART Pad Control Pull-down Register details are shown in Table 91.

TABLE 91: UART_PAD_CTRL_PD_REG

UART Pad Control Register: Internal Pull-down Resistor			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:6	Reserved	R	Always '0'.
5	CTS_PD	R/W	0b: Internal Pull-down resistor disabled on the pad 1b: Internal Pull-down resistor enabled on the pad
4	RTS_PD	R/W	0b: Internal Pull-down resistor disabled on the pad 1b: Internal Pull-down resistor enabled on the pad

TABLE 91: UART_PAD_CTRL_PD_REG (CONTINUED)

UART Pad Control Register: Internal Pull-down Resistor			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
3	RI_PD	R/W	0b: Internal Pull-down resistor disabled on the pad 1b: Internal Pull-down resistor enabled on the pad
2	DSR_PD	R/W	0b: Internal Pull-down resistor disabled on the pad 1b: Internal Pull-down resistor enabled on the pad
1	DCD_PD	R/W	0b: Internal Pull-down resistor disabled on the pad 1b: Internal Pull-down resistor enabled on the pad
0	DTR_PD	R/W	0b: Internal Pull-down resistor disabled on the pad 1b: Internal Pull-down resistor enabled on the pad

The UART Pad Control Open-drain Register details are shown in Table 92.

TABLE 92: UART_PAD_CTRL_OD_REG

UART Pad Control Register: Open-drain Operation			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:2	Reserved	R	Always ‘0’.
1	RTS_OD	R/W	0b: open-drain operation disabled 1b: If the corresponding UARTx_RTS_N pin is configured as an open drain. When the corresponding output is a one, the output is disabled, and the pull-up is enabled. When the output is a zero, the output is enabled and a zero is driven.
0	DTR_OD	R/W	0b: open-drain operation disabled 1b: If the corresponding UARTx_DTR_N pin is configured as an open drain. When the corresponding output is a one, the output is disabled, and the pull-up is enabled. When the output is a zero, the output is enabled and a zero is driven.

The UART Pad Control Polarity Register details are shown in Table 93.

TABLE 93: UART_PAD_CTRL_OPOL_REG

UART Pad Control Register: Pin Polarity			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:2	Reserved	R	Always ‘0’.
1	RTS_OD	R/W	0b: Default Output polarity 1b: Output polarity inverted
0	DTR_OD	R/W	0b: Default Output polarity 1b: Output polarity inverted

The UART Pad Control WAKE_N Debounce Register details are shown in Table 94.

TABLE 94: UART_PAD_CTRL_WAKE_DB

UART Pad Control Register: WAKE_N Debounce			Default: 0x0000_000A
Bits	**Name**	**R/W**	**Description**
31:17	Reserved	R	Always ‘0’.

TABLE 94: UART_PAD_CTRL_WAKE_DB (CONTINUED)

UART Pad Control Register: WAKE_N Debounce			Default: 0x0000_000A
Bits	**Name**	**R/W**	**Description**
16	UART_WAKE_DB_EN	R/W	Debounce enable for UARTx_WAKE_N pad. **Note:** It is valid for this pin to be configured with the Debouncer disabled. In this case the raw input is passed through.
15:12	Reserved	R	Always ‘0’.
11:0	UART_CTS_WAKE_DB_ TIME[11:0]	R/W	This register holds the debounce timer for the UARTx_WAKE_N pad. The Debouncer starts when a transition is detected and is passed through if the pin state is maintained for the configured Debouncer time. If another transition occurs, during the Debouncer time, the Debouncer is restarted. Each count corresponds to 1 ms, with the default value being 10 ms. The timing is derived from a free running clock so will vary with edge alignment. The accuracy of the timing is limited to ± one count.

The UART Pad Control CTS Debounce Register details are shown in Table 95.

TABLE 95: UART_PAD_CTRL_CTS_WAKE_DB

UART Pad Control Register: CTS Debounce			Default: 0x0000_000A
Bits	**Name**	**R/W**	**Description**
31:17	Reserved	R	Always ‘0’.
16	UART_CTS_WAKE_DB	R/W	Debounce enable for nCTS Pad during wake-up. **Note:** Debouncing is applied when nCTS is used for waking up the UART; it is not applied during normal operation.
15:12	Reserved	R	Always ‘0’.
11:0	UART_CTS_WAKE_DB_ TIME[11:0]	R/W	This register holds the debounce timer for the nCTS Pad during wake-up. The Debouncer starts when a transition is detected and is passed through if the pin state is maintained for the configured Debouncer time. If another transition occurs, during the Debouncer time, the Debouncer is restarted. Each count corresponds to 1ms, with the default value being 10 ms. The timing is derived from a free running clock so will vary with edge alignment. The accuracy of the timing is limited to ± one count.

The UART PCI Control Register details are shown in Table 96.

TABLE 96: UART_PCI_CTRL_REG

UART PCI Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:5	Reserved	R	Always ‘0’
4	UART_MSI_VECTOR_SEL	R/W	Selects whether each instance uses a different MSI/MSI-X vector, or they all use the same MSI/MSI-X vector: 0b: all instances use the first MSI/MSI-X vector (0) 1b: each instance uses a different MSI/MSI-x vector
3:1	Reserved	R	Always ‘0’
0	UART_D3_CLK_EN	R/W	Enables the 62.5 MHz clock when the PCI function is in D3: 0b: 62.5 MHz clock is disabled when the PCI function is in D3. 1b: 62.5 MHz clock is enabled when the PCI function is in D3.

The UART WAKE_N Pin Interrupt Register details are shown in Table 97.

TABLE 97: UART_INT_REG

UART WAKE_N Pin Interrupt Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:1	Reserved	R	Always ‘0’
0	UART_WAKE_N_PIN_INT	R/W1C	When ‘1’, indicates that the UARTx_WAKE_N pin was asserted.

The UART WAKE_N Interrupt Mask Register details are shown in Table 98.

TABLE 98: UART_INT_MASK_REG

UART WAKE_N Interrupt Mask Register			Default: 0x0000_0001
Bits	**Name**	**R/W**	**Description**
31:1	Reserved	R	Always ‘0’
0	UART_WAKE_N_PIN_INT_MASK	R/W	Mask for the UART_WAKE_N_PIN_INT status bit: 0b: generate UART_IRQ event when UART_WAKE_N_PIN_INT status bit is set. 1b: prevent generation of UART_IRQ event when UART_WAKE_N_PIN_INT status bit is set.

The UART WAKE Register details are shown in Table 99.

TABLE 99: UART_WAKE_REG

UART WAKE Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:7	Reserved	R	Always ‘0’
2	UART_WAKE_N_PIN_WAKE	R/W1C	When ‘1’, indicates that the UARTx_WAKE_N pin was asserted

TABLE 99: UART_WAKE_REG (CONTINUED)

UART WAKE Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
1	UART_NCTS_WAKE	R/W1C	When '1', indicates that the nCTS pin was asserted when the UART was in suspend
0	UART_INT_WAKE	R/W1C	When '1', indicates that one or more of the UART_IIR status bits have been set when UART was not in suspend

The UART WAKE Mask Register details are shown in Table 100.

TABLE 100: UART_WAKE_MASK_REG

UART WAKE Mask Register			Default: 0x0000_00007
Bits	**Name**	**R/W**	**Description**
31:3	Reserved	R	Always '0'
2	UART_WAKE_N_PIN_WAKE_MASK	R/W	Mask for the UART_WAKE_N_PIN_WAKE status bit: 0b: generate UART_WAKE event when UART_WAKE_N_PIN_WAKE status bit is set. 1b: prevent generation of UART_WAKE event when UART_WAKE_N_PIN_WAKE status bit is set.
1	UART_NCTS_WAKE_MASK	R/W	Mask for the UART_NCTS_WAKE status bit: 0b: generate UART_WAKE event when UART_NCTS_WAKE status bit is set. 1b: prevent generation
0	UART_INT_WAKE_MASK	R/W	Mask for the UART_INT_WAKE status bit: 0b: generate UART_WAKE event when UART_INT_WAKE status bit is set. 1b: prevent generation of UART_WAKE event when UART_INT_WAKE status bit is set.

The UART Reset Register details are shown in Table 101.

TABLE 101: UART_RESET_REG

UART Reset Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:17	Reserved	R	Always '0'
16	PERI_UART_D3_RESET_DIS	R/W	D3 Reset Disable This bit modifies the subsystem Reset operation when exiting D3Cold (PCIE_PERST_N deasserts). When this bit is set and the VAUX_DET pin is high, only the PCIe interface and related logic are Reset; the UARTs are not Reset. Otherwise, the full subsystem is Reset.
15:10	Reserved	R	Always '0'

TABLE 101: UART_RESET_REG (CONTINUED)

UART Reset Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
9	PERI_UART_RELOAD_PCI	R/W	Defines the configuration to be carried out on the Peripheral UART PCIe Endpoint when PERI_UART_RELOAD is set; this corresponds to the PERI_UART_PCIE_EP_ADDR _BASE. When set to '1' the Peripheral UART PCIe Endpoint will be reloaded to PERI_UART_PCIE_EP_ADDR_BASE. When set to '0' the Peripheral UART PCIe Endpoint PF0 configuration will not be reloaded to PERI_UART_ PCIE_EP_ADDR_BASE. The System Config HW will attempt to write all configuration registers associated with a particular configuration; the subsytem will block any writes corresponding to the PCIe Endpoint at PERI_UART_ PCIE_EP_ADDR_BASE.
8	PERI_UART_RELOAD_FN	R/W	Defines the configuration to be carried out on the Peripheral UART function IP when PERI_UART_RELOAD is set; Peripheral SMBus function IP corresponds to all subsystem addresses in UART_PERI_ADDR_BASE, except for those corresponding PERI_UART_PCIE_EP_ADDR_BASE which are controlled by the PERI_UART_RELOAD_PCI bit. When set to '1' the Peripheral UART function IP configuration will be reloaded. When set to '0' the Peripheral UART function IP configuration will not be reloaded. The System Config HW will attempt to write all configuration registers associated with a particular configuration; the subsystem will block any writes corresponding to the function IP.
7:3	Reserved	R	Always '0'
2	PERI_UART_RELOAD	R/W/ SC	Initiate Peripheral UART Function RELOAD **Note:** PERI_UART_RELOAD_PCI and PERI_UART_ RELOAD_FN should be configured appropriately prior to setting PERI_UART_RELOAD. This bit self-clears once the RELOAD is complete.
1	PERI_UART_LRST	R/W/ SC	Initiate Peripheral UART Function LRST **Note:** This bit self-clears once the LRST is complete.
0	PERI_UART_SRST	R/W/ SC	Initiate Peripheral subsytem SRST **Note:** This bit self-clears once the SRST is complete.

The UART_LTR_VALUE_REG register details are shown in Table 102.

TABLE 102: UART_LTR_VALUE_REG

UART_LTR_VALUE_REG			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31	UART_NO_SNOOP_REQ	R/W	No-Snoop Requirement 0b: No latency requirement and the UART_NO_SNOOP_LATENCY_VAL[9:0] and UART_NO_SNOOP_LATENCY_SCALE[2:0] fields are ignored. 1b: Indicates that the UART_NO_SNOOP_LATENCY_VAL[9:0] and UART_NO_SNOOP_LATENCY_SCALE[2:0] fields describe the latency requirement.
30:29	Reserved	R	Always '0'
28:26	UART_NO_SNOOP_LATENCY_SCALE[2:0]	R/W	Scale of the UART_NO_SNOOP_LATENCY_VAL[9:0] field: 000b: value times 1 ns 001b: value times 32 ns 010b: value times 1024 ns 011b: value times 32768 ns 100b: value times 1048576 ns 101b: value times 33554432 ns 110b: value times not permitted 111b: value times not permitted
25:16	UART_NO_SNOOP_LATENCY_VAL[9:0]	R/W	No-Snoop Latency Value
15	UART_SNOOP_REQ	R/W	Snoop Requirement 0b: No latency requirement and the UART_SNOOP_LATENCY_VAL[9:0] and UART_SNOOP_LATENCY_SCALE[2:0] fields are ignored. 1b: Indicates that the UART_SNOOP_LATENCY_VAL[9:0] and UART_SNOOP_LATENCY_SCALE[2:0] fields describe the latency requirement.
14:13	Reserved	R	Always '0'
12:10	UART_SNOOP_LATENCY_SCALE[2:0]	R/W	Scale of the UART_SNOOP_LATENCY_VAL[9:0] field: 000b: value unit - 1 ns 001b: value unit - 32 ns 010b: value unit - 1024 ns 011b: value unit - 32768 ns 100b: value unit - 1048576 ns 101b: value unit - 33554432 ns 110b: reserved 111b: reserved
9:0	UART_SNOOP_LATENCY_VAL[9:0]	R/W	Snoop Latency Value

10.0 PERIPHERAL SUBSYSTEM IMPLEMENTATION - SMBUS CONTROLLER

The SMBus Controller Subsystem occupies its own PCI device endpoint and requires a separate device driver to function from a host system. This subsystem shares the PCI bus with all other peripheral devices (UART, SPI, GPIO).

The Subsystem IDs are:

- Vendor ID: 0x1055 (Microchip Technology, Inc/SMSC)
- Device ID:
  - PCI12000: 0xA003
  - PCI11010: 0xA013
  - PCI11101: 0xA023
  - PCI11400: 0xA033
  - PCI11414: 0xA043
- Class ID: 0x0C0500 (“SMBus Controller”)

The SMBus interface has 3 pins:

- SMBUS_CTLR_SCL - Clock Signal
- SMBUS_CTLR_SDA - Data Signal
- SMBUS_CTLR_ALERT_N - Alert Signal

Three speeds are supported:

- 100 kHz
- 400 kHz
- 1 MHz

Note:	The PCI1xxxx is compliant to SMBus 2.0 specification, but does include limited capabilities introduced in SMBus 3.0 specification. These include 1 MHz support and SMBus alert support.

10.1 Sections

10.2 Interface/Speed Configuration

Note:	The I^2C/SMBus driver will automatically configure the registers listed in Table 103 based on the configuration setting made via GPR0_REG written into PCI1xxxx configuration (OTP/EEPROM).

The SCL_PAD_CTRL and SDA_PAD_CTRL registers must be set appropriately to enable the I/O Pads. These registers (Table 103) gate the Pad control signals coming from the SMBus Controller Core.

Certain AC timing parameters need to be adjusted to account for internal latency of the block. For PCI1xxxx the registers in Table 103 must be reprogrammed with the correct AC timing parameters.

TABLE 103: SMBUS CONTROLLER REGISTER SETTINGS BY SPEED

Register	Field	Default Value					
		100 kHz		400 kHz		1 MHz	
		Time	Val	Time	Val	Time	Val
SR_HOLD_ TIME_REG	SR_HOLD_TIME[7:0]	4 µs	0x85	0.6 µs	0x14	0.26 µs	0x0B
IDLE_ SCALAING_REG	FAIR_BUS_IDLE_ MIN[11:0]	31 µs	0x3C9	5	0x09D	5 µs	0x09D
	FAIR_IDLE_DELAY[11:0]	32 µs	0x3E8	16	0x1F4	16 µs	0x1F4
BUS_CLOCK_REG	HIGH_PERIOD[7:0]/ LOW_PERIOD[7:0]	n/a	0x9A9C	n/a	0x2329	n/a	0x0E0F
CLKSYNC_ REGISTER	CLKSYNC[31:0]	100 ns	0x4	100	0x4	100 ns	0x4
DATA_TIMING_REG	FIRST_START_HOLD[7:0]	4 µs	0x16	0.6 µs	0x10	0.26 µs	0x06
	STOP_SETUP[7:0]	4 µs	0x9D	0.6 µs	0x14	0.26 µs	0x0C
	RESTART_SETUP[7:0]	4.7 µs	0x9D	0.6 µs	0x14	0.26 µs	0x0C
	DATA_HOLD[7:0]	0 ns	0x01	0 ns	0x01	0 ns	0x01
TO_SCALING_REG	BUS_IDLE_MIN[7:0]	4.7 µs	0xA7	1.3 µs	0x8B	0.5 µs	0x85
	CTLR_CUM_ TIME-OUT[7:0]	10 ms	0x9F	10 ms	0x9F	10 ms	0x9F
	TARGET_CUM_ TIME-OUT[7:0]	25 ms	0xC7	25 ms	0xC7	25 ms	0xC7
	CLOCK_HIGH_ TIME-OUT[7:0]	50 µs	0xCC	50 µs	0xCC	50 µs	0xCC

10.3 I²C/SMBus Buffers

When the Host software is sending data to the SMBus, it must first post the packet to the buffer, and then enable the SMBus Controller Core to transfer a data byte at a time from the buffer to the SMBus Target.

When receiving data from the I²C bus, the SMBus Controller Core will transfer a data byte at a time to the buffer. Once the received packet has terminated, the Host software has to service the buffer by reading all the bytes in the buffer.

There are two buffers. Each buffer can operate in either receive or transmit mode at a time. The software must track the transfer direction and program the buffer correctly for each transfer with SMBUS_CONTROL.TRANSFER_DIR.

The buffer is enabled when SMBUS_CONTROL.RUN is set to 1b'1. This allows the buffer to transfer a byte between itself and the SMBus Controller Core through the port. The actual byte transfer between the SMBus Controller Core and the buffer occurs when the DMA_REQ signal of the controller is asserted. When this happens either the buffer drops or picks up a byte from the controller depending on SMBUS_CONTROL.TRANSFER_DIR. After the byte transfer the SMBUS_DEV_COUNTER.DEV_COUNTER[7:0] field is incremented.

To terminate the buffer operation, the SMBus Controller Core generates an event via SMBUS_STATUS.DMA_TERM. This indicates that either the buffer has filled itself in a receive operation, or emptied itself in a transmit operation. When the buffer operation has terminated, the SMBUS_CONTROL.RUN field is automatically cleared by HW to 0b.

The software can only write or read from the buffer when the buffer is not operating or when SMBUS_CONTROL.RUN is cleared to 0b. When the buffer is enabled, only the SMBus Controller Core has access to the buffer.

The software can access the buffer in two ways:

1. In Direct Access mode, the software can directly access the buffer contents through processor (address range offset 0x80...0xFF). In direct access mode, SMBUS_CONTROL.HOST_FIFO_ENTRY is cleared.
2. When SMBUS_CONTROL.HOST_FIFO_ENTRY is set, the buffer is accessed as FIFO. In this mode, software can indirectly access the buffer through the SMBUS_FIFO_DATA register. The buffer location is indicated by the SMBUS_MCU_COUNTER.MCU_COUNTER[7:0] field. Any read or write to SMBUS_FIFO_DATA.FIFO_DATA[7:0] field causes the SMBUS_MCU_COUNTER.MCU_COUNTER[7:0] field to be incremented such that it points to the next buffer location.

10.3.1 BUFFER TRANSMIT OPERATION

Enabling the buffer to transmit data onto the SMBus involves the following steps:

1. SW must Reset both the SMBUS_MCU_COUNTER and SMBUS_DEV_COUNTER registers to zero. This can be done by either writing a zero directly to both of these registers or by setting the RESET_COUNTERS in the SMBUS_CONTROL register.
2. SW must clear the TRANSFER_DIR field in the SMBUS_CONTROL register thus putting the buffer in transmit mode.
3. SW must then post the packet to the buffer, either through FIFO entry mode or direct access mode. When using FIFO entry mode, the SMBUS_MCU_COUNTER.MCU_COUNTER[7:0] field is incremented automatically for every write to the SMBUS_FIFO_DATA.FIFO_DATA[7:0] field. When using Direct Access mode, the SMBUS_MCU_COUNTER.MCU_COUNTER[7:0] field must be written by software with the total number of bytes posted to the buffer.
4. Once the packet is posted to the buffer, the buffer is enabled by setting RUN in the SMBUS_CONTROL register. After this event, a byte is read from the buffer and passed to the SMBus Controller Core for transmission.
5. For every byte read from the buffer and transferred to the SMBus Controller Core, the SMBUS_DEV_COUNTER.DEV_COUNTER[7:0] field in the SMBUS_DEV_COUNTER register is incremented. The SMBUS_DEV_COUNTER.DEV_COUNTER[7:0] field keeps increasing until it reaches the packet size.
6. When the packet has been exhausted, BUF_EMPTY and/or DMA_TERM shall be set. The respective I^2C interrupt in the SMBUS_INTERRUPT_STATUS Register is then set if configured.
7. RUN is automatically cleared by hardware to conclude the buffer transmit operation.

Note: The size of the packet cannot exceed the amount specified by the SMBUS_BUF_DEPTH.BUF_DEPTH[7:0] field.

10.3.2 BUFFER RECEIVE OPERATION

Enabling the buffer to transmit data onto the SMBus involves the following steps:

1. SW must Reset both the SMBUS_MCU_COUNTER and SMBUS_DEV_COUNTER registers to zero. This can be done by either writing a zero directly to both of these registers or by setting the RESET_COUNTERS in the SMBUS_CONTROL register.
2. SW must set the TRANSFER_DIR field in the SMBUS_CONTROL register thus putting the buffer in Receive mode.
3. SW sets RUN in the SMBUS_CONTROL register. This enables the buffer to be filled with data from the SMBus Controller Core.
4. Whenever a byte is written into the buffer, the SMBUS_DEV_COUNTER.DEV_COUNTER[7:0] field in the SMBUS_DEV_COUNTER register is incremented.
5. The SMBUS_DEV_COUNTER.DEV_COUNTER[7:0] field keeps increasing until the receive packet has concluded.
6. After completion of the received packet RUN is cleared by HW.
7. The respective I^2C buffer interrupt is set if configured and the MCU is notified.
8. SW determines the number of bytes received by reading the SMBUS_DEV_COUNTER.DEV_COUNTER[7:0] field.
9. SW then empties the buffer with either FIFO entry or direct access mode. When using the FIFO Entry mode, the SMBUS_MCU_COUNTER.MCU_COUNTER[7:0] field in the SMBUS_MCU_COUNTER register is incremented automatically for every read from the SMBUS_FIFO_DATA.FIFO_DATA[7:0] field. After the FW empties the buffer, the buffer operation has concluded.

10.3.3 BUFFER EMPTY

Buffer empty condition can happen when the SMBus Controller Core has completely drained the buffer before the transmit operation on the SMBus has concluded. Immediately after the last byte is read from the buffer and written to the SMBus Controller Core, the BUF_EMPTY bit in the SMBUS_INTERRUPT_STATUS register is asserted and RUN is cleared in the SMBUS_CONTROL register.

If the INT_STAT_BUF_EMPTY field in the SMBUS_INTERRUPT_MASK register is cleared the respective buffer interrupt signal is triggered. SW can then service the interrupt and post the next set of transmit packet to the buffer and set RUN again. To avoid under-running SMBus, the SMBus Controller Core may automatically invoke clock stretching. This is dependent on how quickly SW can service the interrupt. Clock stretch terminates after the new data is posted into the buffer and RUN is set.

10.3.4 BUFFER FULL

A Buffer full condition can happen when the SMBus Controller Core has filled the buffer to the SMBUS_BUF_DEPTH.BUF_DEPTH[7:0] size before a receive operation has concluded on the SMBus. Immediately after the last byte is read from SMBus Controller Core and written into the buffer, and SMBUS_BUF_DEPTH.BUF_DEPTH[7:0] is reached, SMBUS_STATUS.BUF_FULL asserts. RUN is automatically cleared in the SMBUS_CONTROL register by HW.

If the INT_STAT_BUF_FULL field in the SMBUS_INTERRUPT_MASK register is cleared the respective buffer interrupt signal is triggered. SW can then service the interrupt and empty and set RUN again. To avoid under-running SMBus the SMBus Controller Core may automatically invoke clock stretching. This is dependent on how quickly SW can service the interrupt. Clock stretching terminates after the buffer is emptied and RUN is set.

10.4 Suspend

When the SMBUS_PCI_CTRL_REG.SMB_D3_CLK_EN bit is set to '0', then the 62.5 MHz clock to the SMBus module will be gated off when the SMBus SMBus Controller PCIe Endpoint function goes to D3. This is defined as the SMBus module being suspended. The clock will be restored when the PCI Function goes to D0.

When the SMBUS_PCI_CTRL_REG.SMB_D3_CLK_EN bit is set to '1', then the 62.5 MHz clock to the SMBus module will not be gated off when the SMBus SMBus Controller PCIe Endpoint function goes to D3; the module will continue to operate. This also means that CLK_REQ_REG.PERI_CRYSTAL_REQ and CLK_REQ_REG.PERI_WR_PLL_REQ must both be set to '1' even if all peripheral PCIe Endpoint functions are in the D3 state. This is defined as the SMBus module NOT being suspended.

10.5 SMBus Alert

The SMBus alert signal is a wired AND signal just as the SMBCLK and SMBDAT signals. SMBALERT# is used in conjunction with the SMBus Alert Response Address (ARA). The PCI1xxxx provides this as an input to the SMBus SMBus Controller via the SMBUS_CTLR_ALERT_N pin when this has been configured.

An SMBus Target device can signal the SMBus Host through SMBALERT# to start communication. The host processes the interrupt and simultaneously accesses all SMBALERT# devices through the Alert Response Address. Only the device(s) which pulled SMBALERT# low will acknowledge the Alert Response Address. The host performs a modified Receive Byte operation. The 7-bit device address provided by the SMBus Target transmit device is placed in the seven most significant bits of the byte. The eighth bit can be a '0' or '1'.

When the SMBUS_CTLR_ALERT_N pin is pulled low, this indicates that one or more SMBus Targets need to communicate with the SMBus Host. When the PCIe Endpoint function is in D3, a SMBUS_CTLR_WAKE event will be triggered.

When the PCIe Endpoint function is in the D0 state, a SMBUS_CTLR_WAKE interrupt is generated. Pad control for the SMBUS_CTLR_ALERT_N pin is provided in the SMBALERT_CTLR_PAD_CTRL_REG register. A time for debouncing the input used to trigger the interrupt can be set in SMBALERT_CTLR_DB_REG.SMBUS_CTLR_DB_TIME[11:0]. Both current and debounced values of the SMBUS_CTLR_ALERT_N pin are provided in the SMBALERT_CTLR_VAL_REG register.

Note:	When the SMBUS_CTLR_ALERT_N pin is used to wake-up the system without the Debouncer being enabled (SMBALERT_CTLR_PAD_CTRL_REG.SMBALERT_CTLR_DB = 0), then SMBUS_CTLR_ALERT_N needs to be asserted for a minimum of ~6µs by the SMBus Target. ~4 µs is used to request Ring Oscillator clock and 2 µs is used to generate the wake message to Host.

10.6 Interrupts

The SMBUS_CTLR_IRQ interrupt is controlled by the integrated SMBus Controller. Consult the respective specification for details on its usage. This is used to generate an MSI/MSI-X event via PCIe.

The SMBus Controller has two interrupt sources:

1. I2C_Intr - This interrupt is asserted from SMBus Controller Core.

   For example, CDONE, IDLE, AAS, PIN interrupt, etc. in the SMBUS_STATUS Register.

2. I2C_buf_ctlr_intr - Asserted from SMBus Controller Core SMBUS Network Engine operation.

   For example, BUF_FULL, BUF_EMPTY, SMBUS_STATUS.THRESHOLD_HIT and DMA_TERM interrupt, etc. in the SMBUS_INTERRUPT_STATUS Register.

The SMBUS_CTLR_IRQ interrupt is generated both for the I2C_Intr and I2C_buf_ctlr_intr. The source of the interrupt can be found by examining the SMBUS_GEN_EVENT_REG Register. The I2C_INT bit is set when the I2C_Intr interrupt is triggered, and the I2C_BUF_CTLR_INT bit is set when the I2C_bus_ctlr_intr interrupt is triggered.

When the interrupt handler is triggered, the Host Driver software checks these bits to identify the source and process the corresponding interrupt accordingly.

A SMBUS_CTLR_IRQ interrupt is also generated when the SMBUS_CTLR_ALERT_N pin is pulled low and the debounced input has settled. At this point, the SMBUS_GEN_EVENT_REG.SMBALERT_INT bit is also set.

Interrupts can be masked by setting the corresponding bit in the SMBUS_GEN_EVENT_MASK_REG register.

The SMBUS_CTLR_WAKE event is present to allow for detection of wake events.

10.7 Wake-up Support

The SMBus Controller is capable of asynchronous wake-up. The SMBus Controller indicates this occurrence via the SMBUS_CTLR_WAKE event. SMBUS_CTLR_WAKE events can occur whether or not the SMBus Controller is suspended.

When the PCIe Endpoint function is in D3 and the SMBus Controller is not in suspend:

- The SMBUS_CTLR_WAKE event is generated both for the I2C_Intr and I2C_buf_ctlr_intr. The source of the interrupt can be found by examining the SMBUS_GEN_EVENT_REG Register. The I2C_WAKE_EVENT bit is set when the I2C_Intr interrupt is triggered, and the I2C_BUF_CTLR_WAKE_EVENT bit is set when the I2C_bus_ctlr_intr interrupt is triggered.
- A SMBUS_CTLR_WAKE interrupt is also generated when the SMBUS_CTLR_ALERT_N pin is pulled low and the debounced input has settled. At this point the SMBUS_GEN_EVENT_REG.SMBALERT_WAKE_EVENT bit is set.

When the PCIe Endpoint function is in the D3 state and the SMBus Controller is suspended, the SMBUS_CTLR_ALERT_N pin has a debounce capability which can be enabled via SMBALERT_CTLR_PAD_CTRL_REG.SMBALERT_CTLR_DB for a period of time defined by SMBALERT_CTLR_DB_REG.SMBUS_CTLR_DB_TIME[11:0]. This helps prevent false wake-ups from noise which is a requirement of the I^2C specification. After filtering, when the SMBUS_CTLR_ALERT_N pin is pulled low, an SMBUS_CTLR_WAKE event is triggered and WAKE# is sent via the PCIe Endpoint. When the 62.5 MHz oscillator and BIAS are stable, the I^2C is clocked by the new clock source and the SMBUS_GEN_EVENT_REG. SMBALERT_WAKE_EVENT bit is set.

A SMBUS_CTLR_WAKE event causes PCI Wake signaling to be generated. This wake request should trigger the PCIe Host to take the SMBus Controller into the D0 device state. At this point the aggregated peripheral PLL Request CLK_REQ_REG.PERI_WR_PLL_REQ and the aggregated peripheral Crystal Request CLK_REQ_REG.PERI_CRYSTAL_REQ will both be set to true by the peripheral subsytem since one or more Physical Functions are now in D0.

Wake events are indicated by bits in the SMBUS_GEN_EVENT_REG Register and can be masked by setting the corresponding bit in the SMBUS_GEN_EVENT_MASK_REG Register.

10.8 SMBus Subsystem Registers

The PERI_SUBSYSTEM_ADDR_BASE = 0x0015_0000 registers are listed in Table 104.

Note:	PCI1xxxx devices have thousands of registers which are either reserved for future use or the contents are masked from public documentation because they have no end system integrator use-case (i.e. they are used by hardware to perform low level run-time tasks or are used/controlled directly by a driver during run-time). **Do not modify any contents of undocumented registers**.

TABLE 104: SMBUS SUBSYSTEM REGISTER OFFSETS FROM PERI_SUBSYSTEM_ADDR_BASE

Register Name	Address Offset
CONTROL_REG	0x0_0000
STATUS_REG	0x0_0004
DATA_REG	0x0_0008
SMBUS_CTLR_COMMAND_REG	0x0_000C
PEC_REG	0x0_0014
SR_HOLD_TIME_REG	0x0_0018
COMPLETION_REG	0x0_0020
IDLE_SCALING_REG	0x0_0024
CONFIG_REG	0x0_0028
BUS_CLOCK_REG	0x0_002C
BLOCK_ID_REG	0x0_0030
REVISION_REG	0x0_0037
BB_CONTROL_REG	0x0_0038
CLKSYNC_REGISTER	0x0_003C
DATA_TIMING_REG	0x0_0040
TO_SCALING_REG	0x0_0044
CTLR_TX_BUFFER_REG	0x0_0050
CTLR_RX_BUFFER_REG	0x0_0054
DEBUG_FSM_I2C	0x0_0058
DEBUG_FSM_SMB	0x0_005C
SCL_PAD_CTL	0x0_0100
SDA_PAD_CTL	0x0_0101
SMBUS_CONTROL	0x0_0200
SMBUS_STATUS	0x0_0204
SMBUS_INTERRUPT_STATUS	0x0_0208
SMBUS_INTERRUPT_MASK	0x0_020C
SMBUS_FIFO_DATA	0x0_0210
SMBUS_MCU_COUNTER	0x0_0214
SMBUS_DEV_COUNTER	0x0_0218
SMBUS_THRESHOLD_VALUE	0x0_021C
SMBUS_BUFF_DEPTH	0x0_0220
SMBALERT_CTLR_PAD_CTRL_REG	0x0_0230
SMBALERT_CTLR_DB_REG	0x0_0234
SMBALERT_CTLR_VAL_REG	0x0_0238
SMBUS_GEN_EVENT_REG	0x0_023C
SMBUS_GEN_EVENT_MASK_REG	0x0_0240
SMBus_PCI_CTRL_REG	0x0_0244
SMBUS_RESET_REG	0x0_0248
SMBUS_LTR_VALUE_REG	0x0_024C
SMBUS_DATA_BUFFER	0x0_0280
GPR0_REG	0x0_1C00

The Control Register details are shown in Table 105.

TABLE 105: CONTROL_REG

SMBus Control Register			Default: 0x81
Bits	**Name**	**R/W**	**Description**
7	PIN	W/SC	The Pending Interrupt Not (PIN) bit serves as a software Reset function. Writing the PIN bit to 1b deasserts all status bits except for the STATUS_REG.nBB bit which is not affected by the PIN bit. The PIN bit is a self-clearing bit. Writing this bit to 0b has no effect. **Note:** During SMBus Target receive mode the PIN bit in the I^2C Controller Core is asserted when the SMBus Target detects a STOP condition. If in I^2C backward compatibility mode the SMBus Target host has not deasserted the PIN bit before another valid SMBus Target address arrives then the SMBus Target will NACK the SMBus Target address because SMBus Target host has not processed the original interrupt.
6	ESO	W	The Enable Serial Output bit (ESO) enables and disables the SMBus Controller Core serial data output (SDAT). When ESO is asserted (1b1), SDAT is enabled. When ESO is not asserted (1b‘0) SDAT is disabled. The ESO bit does not affect access to, or other bit in the Registers Interface. **Note:** The ESO bit must not be deasserted when the SMBus Controller Core is actively involved in a bus transaction.
5:4	Reserved	W	Always 0b
3	ENI	W	The Enable Interrupt bit (ENI) controls the Interrupt Interface.
2	STA	W	The STA and STO bits control the generation of the I^2C Start condition and the transmission of the SMBus Target Address and R/W bit (from the DATA_REG Register), generation of repeated Start condition, and generation of the Stop condition. When set, Transmit START+address, remain CTRL/TRM if Data Register bit 0 (R/nW) =1b’0. **Note:** STA and STO must never both be set at the same time.
1	STO	W	When set, Transmit STOP go to TGT/REC mode. **Note:** STA and STO must never both be set at the same time.
0	ACK	W	The Acknowledge bit (ACK) must normally be asserted (‘1b’). This causes the controller to send an acknowledge automatically after each byte (this occurs during the 9th clock pulse). The ACK bit must not be asserted (‘0b’) when the controller is operating in SMBus controller/receiver mode and requires no further data to be sent from the SMBus Target transmitter. This causes a negative acknowledge on the I^2C bus, which halts further transmission from the SMBus Target device.

The Status Registers details are shown in Table 106.

TABLE 106: STATUS_REG

SMBus Status Register			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7	PIN	R	Pending Interrupt bit.
6	Masked	R	Function masked or not used.
5	Masked	R	Function masked or not used.
4	BER	R	When Bus Error (BER) is asserted, a misplaced START or STOP condition or Bus Time-Outs have been detected. BER de-asserts nBB (1b‘1) and immediately asserts the PIN bit (1b‘0). When BER is asserted, the SMBus Controller Core must be Reset using the Configuration Register RESET bit before taking any further action.
3	LRB_AD0	R	The “Last Received Bit” or “Address 0” (general call) bit (LRB_AD0) serves a dual function and is valid only while the PIN bit is asserted (‘0b’). When the AAS bit is not asserted (‘0n’) (i.e., not addressed as a SMBus Target), the LRB_AD0 holds the value of the last received bit over the bus. Normally this will be the value of the SMBus Target acknowledgment; thus, checking for SMBus Target acknowledgment is done via testing of the LRB_AD0 bit.
2	Masked	R	Function masked or not used.
1	Masked	R	Function masked or not used.
0	nRB	R	The Bus Busy bit (nBB) is a read-only flag indicating when the bus is in use. A zero indicates that there is a Bus Busy condition and access is not possible. This bit is asserted (‘0b’) by a Start condition and deasserted (‘1b’) following a Stop condition. The nBB bit can also be deasserted as a result of a Bus Time-Out.

The Data Register details are shown in Table 107.

TABLE 107: DATA_REG

Data Register			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:0	Data[7:0]	R/W	Data

The Controller Command Register details are shown in Table 108.

TABLE 108: SMBUS_CTLR_COMMAND_REG

SMBus Controller Command Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:24	READCOUNT[7:0]	R/W	This field is a count of the number of bytes to read in from the SMBus to the CTLR_RX_BUFFER_REG Register and must be greater than 0 in order for the SMBus Controller State Machine to initiate a read phase. It is decremented by 1 for each byte read from the SMBus into the CTLR_RX-_BUFFER_REG Register. It can be overwritten by the first byte read in from the SMBus. **Note:** The Read Count is a hard limit for block reads. Any data received beyond this value will be ignored. ReadCount[7:0] does not include a PEC byte, so the number of bytes read from the SMBus Target is (ReadCount[7:0] + Read_PEC). The number of bytes copied into to memory is (ReadCount[7:0] + 2XRead_PEC).
23:16	WRITECOUNT[7:0]	R/W	This field is a count of the number of bytes to transmit to the SMBus from the CTLR_TX_BUFFER_REG Register. It is decremented by 1 for each byte written to the SMBus from the CTLR_TX_BUFFER_REG Register.
15:14	Reserved	R	Always read ‘0’
13	READ_PEC	R/W	If this bit is ‘0b’, reading from the SMBus stops when READ-COUNT[7:0] reaches 0x00. If this bit is 1b, reading continues when READCOUNT[7:0] is 0x00 for one more byte. This enables the SMBus Controller to read a PEC returned from an external device after it has read an M-byte block. Once the PEC byte is read this field is cleared to 0b.
12	READC	R/W	If this bit is 1b’1, then the ReadCount[7:0] field is replaced by the byte that is read from the SMBus when Read-Count[7:0] is 8h’01. After ReadCount[7:0] is updated, this bit is cleared to 1b’0.
11	PEC_TERM	R/W	If this bit is 1b’1, a copy of the PEC_REG register is transmit-ted when WriteCount[7:0] is 1b’0. After the PEC_REG regis-ter is read, both the PEC_REG register and the PEC_TERM bit are cleared to 1b’0.
10	STOP	R/W	If this bit is 1b’1, send a Stop bit after the transaction com-pletes.
9	STARTN	R/W	If this bit is 1b’1, send a Start bit just before the last byte of the WriteCount[7:0] is sent to the SMBus transmitter.
8	START0	R/W	If this bit is 1b’1, send a Start bit on the SMBus before the first byte of the WriteCount[7:0] is sent to the SMBus trans-mitter.
7:2	Reserved	R	Always read ‘0b’
1	CPROCEED	R/W	When this bit is ‘0b’, the SMBus Controller State Machine does not transition out of the Idle or Pause states. When this bit is ‘1b’, the SMBus Controller State Machine immediately transitions to the WAIT-BusBusy and CRUN-Receive states, respectively. This bit is automatically cleared when the SMBus Controller State Machine enters the Pause state.

TABLE 108: SMBUS_CTLR_COMMAND_REG (CONTINUED)

SMBus Controller Command Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
0	CRUN	R/W	While this bit is 1b, transfer bytes over SMBus. As long as WRITECOUNT[7:0] is non-zero, a byte from the SMBus Controller Transmit Buffer is transmitted to the SMBus Target device and WRITECOUNT[7:0] is decremented. If WRITECOUNT[7:0] is 0x00, the controller asserts COMPLETION_REG.CDONE, clears CPROCEED, and waits for software to restart the State Machine. When restarted by writing CPROCEED, each byte received from the SMBus Target device is written into the SMBus Controller Receive Buffer. In both the write and read phases the SMBus Controller drives the SMBus clock. The State Machine asserts COMPLETION_REG.CDONE after the receive phase is completed. This bit is cleared and the transaction completes when: • Read_PEC is 0b and both READCOUNT[7:0] and WRITECOUNT[7:0] are 0x00 • Read_PEC is 1b, READCOUNT7:0] is 0x00 and WRITECOUNT[7:0] is 0xFF • A NACK was received from the SMBus Target device • The SMBus Controller lost Bus Arbitration • A SMBus Controller Timeout or other bus error occurred This bit is not cleared when the SMBus Controller State Machine counts the WRITECOUNT[7:0] to 8h'00 and READCOUNT[7:0] is greater than 8h'00. The COMPLETION_REG.CDONE interrupt is issued and firmware can restart the SMBus Controller State Machine by writing a 1b to the CPROCEED bit in this register. In order to transfer out of the Idle state, software must set both CRUN and CPROCEED to '1b'.

The Packet Error Register details are shown in Table 109.

TABLE 109: PEC_REG

Packet Error Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
7:0	PEC[7:0]	R/W	Contains the PEC value if PEC function is used.

The Hold Time Register details are shown in Table 110.

TABLE 110: SR_HOLD_TIME_REG

Hold Time Register			Default: 0x85
Bits	**Name**	**R/W**	**Description**
7:0	SR_HOLD_TIME[7:0]	R/W	This is used to control the hold time of the clock until the Hold Time for the repeated Start Bit has been satisfied.

The Completion Register details are shown in Table 111.

TABLE 111: COMPLETION_REG

Completion Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31	Masked	R	Do not modify
30	CDONE	R/W1C	If this bit is 1b, SMBus Controller State Machine completed operation and returned to the Idle state. This bit is cleared when written with a 1b. Writes of a 0b have no effect.
29	IDLE	R/W1C	This bit is set when the I^2C bus becomes idle (on the rising edge of STATUS_REG.nBB). It is also set if the bus was idle when either the Idle interrupt is enabled (on the rising edge of the ENIDI interrupt) or when the I^2C core is enabled. This bit is cleared when written with a 1b. Writes of a 0b have no effect.
28:26	Reserved	R	Always read ‘0’
25	CTR	R	SMBus Controller Transmit/Receive. This bit reports the phase of the SMBus Controller State Machine when it asserts CDONE. 0b: SMBus Controller has just finished the receive phase of a transaction. 1b: SMBus Controller has just finished the transmit phase of a transaction.
24	CNAKX	R/WC	If this bit is 1b, the SMBus Controller State Machine received a NACK from the receiving SMBus Target while the SMBus Controller was transmitting data over the SMBus interface.
23:22	Reserved	R	Always read ‘0’
21:19	Masked	R	Do not modify
18	Reserved	R	Always read ‘0’
17:16	Masked	R	Do not modify
15	Reserved	R	Always read ‘0’
14:13	Masked	R	Do not modify
12	CHDH	R/WC	CHDH is the bus idle time-out detect bit (Clock High Data High).
11	CHDL	R/WC	CHDL is the clock high time-out detect bit (Clock High Data Low).
10	Masked	R	Do not modify
9	CCTO	R/WC	CCTO is the SMBus Controller Cumulative Time-out bit.
8	DTO	R/WC	DTO is the Device Time-out bit. The bus idle period should be programmed to the TIDLE_WINDOW time. This field defines the number of ticks of the baud clock required to satisfy the fairness protocol. The default value for this field sets the idle window to 31 µs, which is the appropriate value for a 100 kHz bus.
7	Reserved	R	Always read ‘0’

TABLE 111: COMPLETION_REG (CONTINUED)

Completion Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
6	TIMERR	R	The Time-out Error Detected bit (TIMERR) is asserted (1b) whenever any of the enabled time-out error detect status bits (CHDH, CHDL, TCTO, CCTO and DTO) are asserted. **Note:** The time-out error detect status bit enables are BIDEN, CCEN, and DTEN. TIMERR is not asserted when all of the enabled time-out error detect status bits are not asserted. If all of the time-out error detect status bit enables are 0b, TIMERR cannot be asserted.
5	BIDEN	R	The BIDEN bit enables Bus Idle Detect Time-Out checking. When BIDEN is asserted (1b), Bus Idle Detect Time-Out checking is enabled. When BIDEN is not asserted (0b), Bus Idle Detect Time-Out checking is disabled.
4	Masked	R	Do not modify
3	CCEN	R/W	The CCEN bit enables SMBus Controller Cumulative Time-Out checking. When CCEN is asserted (1b), SMBus Controller Cumulative Time-Out checking is enabled. When CCEN is not asserted (0b), SMBus Controller Cumulative Time-Out checking is disabled.
2	DTEN	R/W	The DTEN bit enables Device Time-out checking. When DTEN is asserted (1b), Device Time-out checking is enabled. When DTEN is not asserted (0b), Device Time-out checking is disabled.
1:0	Reserved	R	Always 0

The Idle Scaling Register details are shown in Table 112.

TABLE 112: IDLE_SCALING_REG

Idle Scaling Register			Default: 0x03E8_03C9
Bits	**Name**	**R/W**	**Description**
31:28	Reserved	R	Always read ‘0’
27:16	FAIR_IDLE_DELAY[11:0]	R/W	This field establishes the MCTP T IDLE_DELAY period. When the MCTP Fairness protocol is enabled, this field causes the start of SMBus Controller transaction to be delayed by an additional period of TIDLE_DELAY after the State Machine has detected an idle period of FAIR BUS IDLE MIN. This field defines the number of ticks of the baud clock required to program the delay.
15:13	Reserved	R	Always read ‘0’
11:0	FAIR_BUS_IDLE[11:0]	R/W	The bus idle period should be programmed to the TIDLE_WINDOW time. This field defines the number of ticks of the baud clock required to satisfy the fairness protocol.

The Configuration Register details are shown in Table 113.

TABLE 113: CONFIG_REG

SMBus Configuration Register			Default: 0x0000_2000
Bits	**Name**	**R/W**	**Description**
31	ENSI	R/W	If this bit is 1, the SMBus Target Done interrupt is enabled. If this bit is 0, the SMBus Target Done interrupt is disabled.
30	ENMI	R/W	If this bit is 1, the SMBus Controller Done interrupt is enabled. If this bit is 0, the SMBus Controller Done interrupt is disabled.
29	ENIDI	R/W	If this bit is 1, the Idle interrupt is enabled. If this bit is 0, the Idle interrupt is disabled. **Note:** The Idle interrupt should not be enabled if the SMBus Controller State Machine is operating. (The CRUN bit in the SMBUS_CTLR_COMMAND_REG Register was set to 1.)
28	Masked	R	Do not modify
27:20	Reserved	R	Always 0
19	FLUSH_MRBUF	W	A write of a 1 to this bit forces the CTLR_RX_BUFFER_REG Register to be marked empty. A write of zero (0) has no effect. This is a self-clearing bit.
18	FLUSH_MXBUF	W	A write of a 1 to this bit forces the CTLR_TX_BUFFER_REG Register to be marked empty. A write of zero (0) has no effect. This is a self-clearing bit.
17	FLUSH_SRBUF	W	A write of a 1 to this bit forces the CTLR_RX_BUFFER_REG Register to be marked empty. A write of zero (0) has no effect. This is a self-clearing bit.
16	FLUSH_SXBUF	W	A write of a 1 to this bit forces the CTLR_TX_BUFFER_REG Register to be marked empty. A write of zero (0) has no effect. This is a self-clearing bit.
15:11	Masked	R	Do not modify
10	ENAB	R/W	When ENAB (Enable) is not asserted (‘0b’) (default), the SMBus Controller Core is disabled and in the lowest power consumption state (Disabled state). When ENAB is not asserted, all registers can be accessed normally. The ENAB bit must be asserted (‘1b’) for normal operation. The BBEN bit in the Bit-Bang Control Register must not be asserted when the ENAB bit is asserted. **Note:** It is up to the host to guarantee that the SMBus Controller Core is not in use before the ENAB bit is deasserted.

TABLE 113: CONFIG_REG (CONTINUED)

SMBus Configuration Register			Default: 0x0000_2000
Bits	**Name**	**R/W**	**Description**
9	RESET	R/W	When RESET is asserted (1b‘1), all logic and registers except for the RESET bit itself are initialized to the Power-on default state. RESET is not self-clearing and can be deasserted by the host after one Input Host Clock period when the Input Host Clock is less than or equal to the I^2C Baud Clock period, or otherwise after one I^2C Baud Clock period to resume normal operation. Register reads while RESET is asserted return default register values.
8	FEN	R/W	When the Input Filtering Enable bit (FEN) is asserted (1b), the Input Filter is enabled. When FEN is not asserted (0b) (default), the Input Filter is bypassed. In applications that do not include external filtering on the Hardware Interface, FEN is required to be asserted.
7	PECEN	R/W	When the PEC Enable bit (PECEN) is asserted (1b), Hardware PEC Support is enabled.
6	SS_DET_CLK_SEL	R	START STOP DETECT CLOCK SELECT bit: 0b: use system clock for detection. 1b: use core clock for detection.
5	CONFIG_SLOW_CLK	R	When this bit is 1b, the base period for the Bus Clock Register is multiplied by 4, and thus the frequency is divided by 4. It does not affect other timing calculations (such as those in the Data Timing Register or the Time-Out Scaling Register).
4	TCEN	R/W	When the Timing Check Enable bit (TCEN) is asserted (1b), Bus Time-Outs are enabled.
3:0	PORT_SEL[3:0]	R/W	The PORT_SEL[3:0] bits determine which one of 16 possible bus ports apply to the active two-wire SDAT and SCLK bus pair. For example, when the PORT_SEL[3:0] bits are 4b’0000 (default), SDAT and SCLK message signaling appears on SDAT_IN [0], SDAT_OUT [0], SDAT_EN_N [0], SCLK_IN [0], SCLK_OUT [0] and SCLK_EN_N [0].

The Bus Clock Register details are shown in Table 114.

TABLE 114: BUS_CLOCK_REG

Bus Clock Register			Default: 0x9A9C
Bits	**Name**	**R/W**	**Description**
15:8	HIGH_PERIOD[7:0]	R/W	This field defines the number of I^2C Baud Clock periods that make up the high phase of the I^2C/SMBus bus clock. The number of clock periods is one greater than the contents of this field. For example, setting this field to 0x27 implies a phase that is 0x28 baud clock periods long. In I^2C Fast mode (400 kHz operation), HIGH_PERIOD[7:0] should define a period of at least 0.6 µs. In I^2C Fast mode Plus (1 MHz operation), HIGH_ PERIOD[7:0] should define a period of at least 0.26 µs.
7:0	LOW_PERIOD[7:0]	R/W	This field defines the number of I^2C Baud Clock periods that make up the low phase of the I^2C/SMBus bus clock. The number of clock periods is one greater than the contents of this field. For example, setting this field to 0x27 implies a phase that is 0x28 baud clock periods long. **Note:** In I^2C Fast mode (400 KHz operation), LOW_ PERIOD[7:0] should define a period of at least 1.3 µs. In I^2C Fast mode Plus (1 MHz operation), LOW_PERIOD [7:0] should define a period of at least 0.5 µs.

The Block ID details are shown in Table 115.

TABLE 115: BLOCK_ID_REG

SMBus Block ID			Default: 0x11
Bits	**Name**	**R/W**	**Description**
7:0	ID[7:0]	R/W	Block ID

The Block Revision details are shown in Table 116.

TABLE 116: REVISION_REG

SMBus Block Revision			Default: 0xXX
Bits	**Name**	**R/W**	**Description**
7:0	REVISION[7:0]	R/W	Revision

The Bit-Bang Control Register details are shown in Table 117.

TABLE 117: BB_CONTROL_REG

Bit-Bang Control Register			Default: 0x60
Bits	**Name**	**R/W**	**Description**
7	Reserved	R	Always ‘0b’.
6	BBDATI	R	Bit-Bang Data In. The BBDATI bit always returns the state of SDAT.
5	BBCLKI	R	Bit-Bang Clock In. The BBCLKI bit always returns the state of SCLK.
4	BBDAT	R/W	Bit-Bang Data. The BBDAT bit controls the state of SDAT when BBEN = and DADIR = 1b‘1.
3	BBCLK	R/W	Bit-Bang Clock. The BBCLK bit controls the state of SCLK when BBEN = and CLDIR = 1b‘1.
2	DADIR	R/W	Bit-Bang Data Direction. The DADIR bit controls the direction of SDAT. 0b: Input 1b: Output
1	CLDIR	R/W	Bit-Bang Clock Direction. The CLDIR bit controls the direction of SCLK. 0b: Input 1b: Output
0	BBEN	R/W	Bit-Bang Mode Enable. The BBEN bit enables and disables the Bit-bang mode: 0b: Bit Bang Mode Disabled 1b: Bit Bang Mode Enabled

The Clock Sync Register details are shown in Table 118.

TABLE 118: CLKSYNC_REGISTER

Clock Sync Register			Default: 0x0000_0004
Bits	**Name**	**R/W**	**Description**
31:0	CLKSYNC[31:0]	R/W	CLKSYNC[31:0] allows “n” number of clock cycles to sync up to the external clock before we start comparing the internal and external clocks for clock stretching.

The Data Timing Register details are shown in Table 119.

TABLE 119: DATA_TIMING_REG

Data Timing Register			Default: 0x169D_9D01
Bits	**Name**	**R/W**	**Description**
31:24	FIRST_START_HOLD[7:0]	R/W	The STOP_SETUP[7:0] timer determines the SCLK hold time following SDAT driven low during the first START bit in a transfer. Repeated START hold time is determined by the SR_HOLD_TIME_REG Register.
23:16	STOP_SETUP[7:0]	R/W	The STOP_SETUP[7:0] timer determines the SDAT setup time from the rising edge of SCLK for a Stop condition.

TABLE 119: DATA_TIMING_REG (CONTINUED)

Data Timing Register			Default: 0x169D_9D01
Bits	**Name**	**R/W**	**Description**
15:8	RESTART_SETUP[7:0]	R/W	The RESTART_SETUP[7:0] timer determines the SDAT setup time from the rising edge of SCLK for a repeated Start condition.
7:0	DATA_HOLD[7:0]	R/W	The DATA_HOLD[7:0] timer determines the SDAT hold time following SCLK driven low.

The Time-out Scaling Register details are shown in Table 120.

TABLE 120: TO_SCALING_REG

Time-out Scaling Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:24	BUS_IDLE_MIN[7:0]	R/W	Bus Idle Minimum time. At 100 kHz I^2C bus rate, Bus Idle Minimum time is 4.7 µs or greater.
23:16	CTLR_CUM_TIME-OUT[7:0]	R/W	SMBus Controller Cumulative Time-Out duration = CTLR _CUM_TIME-OUT[7:0] x Baud_Clock_Period x 2048. At 100 kHz I^2C bus rate, SMBus Controller Cumulative Time-Out duration is 10 ms (max).
15:8	TARGET_CUM_TIME-OUT[7:0]	R/W	SMBus Target Cumulative Time-Out duration = TARGET _CUM_TIME-OUT[7:0] x Baud_Clock_Period x 4096 At 100 kHz I^2C bus rate, SMBus Target Cumulative Time-Out duration is 25 ms (max).
7:0	CLOCK_HIGH_TIME-OUT[7:0]	R/W	Clock High Time-out period = CLOCK_HIGH_TIME-OUT[7:0] x Baud_Clock_Period x 8 At 100 kHz I^2C bus rate, Clock High time-out period is 50 µs (max).

The Controller Transmit Buffer Register details are shown in Table 121.

TABLE 121: CTLR_TX_BUFFER_REG

SMBus Controller Transmit Buffer Reg			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:0	CTLR_TRANSMIT_BUFFER[7:0]	R/W	SMBus Controller Transmit Buffer

The Controller Receive Buffer Register details are shown in Table 122.

TABLE 122: CTLR_RX_BUFFER_REG

SMBus Controller Receive Buffer Reg			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:0	CTLR_RECEIVE_BUFFER[7:0]	R/W	SMBus Controller Receive Buffer

The Debug I²C State Machine Register details are shown in Table 123.

TABLE 123: DEBUG_FSM_I²C

Debug I²C State Machine Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:28	28 TIMER_MBI[3:0]	R	0000b: MBI_IDLE 0001b: MBI_COUNT 0010b-1111b: Reserved. Operation is undefined for these values.
27:24	TIMER_SCTO[3:0]	R	0000b: SC_IDLE 0001b: SC_COUNT 0010b-1111b: Reserved. Operation is undefined for these values
23:20	TIMER_MCTO[3:0]	R	State for Timer MCTO: 0000b: MC_IDLE 0001b: MC_COUNT 0010b-1111b: Reserved. Operation is undefined for these values.
19:16	PHY[3:0]	R	State for I²C PHY: 0000b: IDLE 0001b: CLKHIGH 0010b: STARTSTOP 0011b: CLKLOW 0100b: SDATCTRL 0101b: ARBLOSS 0110b-1111b: Reserved. Operation is undefined for these values.
15:8	Reserved	R	Always read ‘0’
7:0	I2C_CTLR[7:0]	R	State for I²C Controller: 0000b: IDLE 0001b: W4 START 0010b: ADDR PHASE 0011b: CHK ACK 0100b: RX DATA 0101b: ACK NACK DATA 0110b: TX DATA 0111b: TX DATA LD 1000b: WAIT ACK 1001b: W4 STOP 1010b: LOST ARB 1011b: LOST ARB REPSTART 1100b: LOST ARB REPSTART DLY1 1101b: LOST ARB REPSTART DLY2 1110b: W4 START HOLD 1111b: Reserved. Operation is undefined for this value.

The Debug SMBus State Machine Register details are shown in Table 124.

TABLE 124: DEBUG_FSM_SMB

Debug SMBus State Machine Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:24	Reserved	R	Always read '0'
23:16	SMB_FAIR[7:0]	R	State for SMB Fair: 0000b: IDLE 0001b: BUSY 0010b: WINDOW 0011b: DELAY 0100b: WAIT 0101b: WAIT DONE 0110b: ACTIVE 0111b-1111b: Reserved. Operation is undefined for these values.
15:8	Reserved	R	Always read '0'
7:0	SMB_CTLR[7:0]	R	State for SMBus Controller: 0000b: IDLE 0001b: SOP 0010b: START 0011b: START PIN 0100b: WDATA 0101b: WPEC 0110b: RSTART 0111b: RSTART PIN 1000b: RDATA N 1001b: RDATA PEC 1010b: RPEC 1011b: PAUSE 1100b: STOP 1101b: EOP 1110b-1111b: Reserved. Operation is undefined for these values.

The Clock Pad Control Register details are shown in Table 125.

TABLE 125: SCL_PAD_CTL

SMBus Clock Pad Control Register			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:5	Reserved	R	Always read '0'
4	FAST_OPEN_DRAIN	R/W	This bit is set for fast open drain operation. 0b: output is driven low 1b: on a 0 to 1 transition, the output is driven high for one system clock before being tri-stated.
3	PULLUP_EN	R/W	Enable pull-up resister when set to 1b.
2	PULLDOWN_EN	R/W	Enable pull-down resister when set to 1b.
1	INPUT_EN	R/W	Enable input buffer of Pad when set to 1b. Output of input buffer defaults to 1b when INPUT_EN is cleared to 0b.
0	OUTPUT_EN	R/W	Enable output buffer of Pad when set to 1b.

The Data Pad Control Register details are shown in Table 126.

TABLE 126: SDA_PAD_CTL

SMBus Data Pad Control Register			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:5	Reserved	R	Always read '0'
4	FAST_OPEN_DRAIN	R/W	This bit is set for fast open drain operation. 0b: output is driven low 1b: on a 0 to 1 transition, the output is driven high for one system clock before being tri-stated.
3	PULLUP_EN	R/W	Enable pull-up resister when set to 1b.
2	PULLDOWN_EN	R/W	Enable pull-down resister when set to 1b.
1	INPUT_EN	R/W	Enable input buffer of Pad when set to 1b. Output of input buffer defaults to 1b when INPUT_EN is cleared to 0b.
0	OUTPUT_EN	R/W	Enable output buffer of Pad when set to 1b.

The SMBus Control details are shown in Table 127.

TABLE 127: SMBUS_CONTROL

SMBus Control			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:4	Reserved	R	Always read '0'
3	RESET_COUNTERS	R/W/SC	Setting this bit will Reset both SMBUS_MCU_COUNTER.MCU_COUNTER[7:0] and SMBUS_DEV_COUNTER.DEV_COUNTER[7:0] to 8'h00. The bit is set for only one clock period, and then clears itself automatically. Clearing this bit has no effect.
2	TRANSFER_DIR	R/W	0b: Data transfer is from MCU to device. This means MCU is filling the buffer while the I^2C controller is draining the buffer. 1b: Data transfer is from device to MCU. This means the I^2C controller is filling the buffer while the MCU is draining the buffer.
1	HOST_FIFO_ENTRY	R/W	0b: MCU has direct access to the buffer. 1b: MCU has access to the buffer through SMBUS_FIFO_DATA.FIFO_DATA[7:0].
0	RUN	R/W	Setting this bit will enable the DMA emulation logic and the I^2C controller will either fill or drain the buffer depending on TRANSFER_DIR setting. Only software can set this bit. Hardware and software can both clear this bit. Hardware will clear this bit when SMBUS_STATUS.DMA_TERM is 1b. Hardware will also clear out the run control register bit if SMBUS_STATUS.BUF_EMPTY is set to 1b in a send operation or if SMBUS_STATUS.BUF_FULL is set to 1b in a receive operation. When software clears this bit, the DMA emulation logic will be aborted irrespective of a live I^2C transfer simultaneously occurring.

The SMBus Status details are shown in Table 128.

TABLE 128: SMBUS_STATUS

SMBus Status			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7	DMA_TERM	R	The DMA_TERM bit is hooked up directly to the DMA_TERM signal coming from the I^2C/SMBus Controller. The DMA_TERM bit being set to '1b' can only generate an SMBUS_CTLR_IRQ event when SMBUS_INTERRUPT_MASK.INT_MASK_DMA_TERM is set to '0b'.
6	DMA_REQ	R	Register bit is hooked up directly to the SMB_MDMA_REQ signal coming from the I^2C/SMBus Controller.
5:3	Reserved	R	Always read '0'
2	THRESHOLD_HIT	R	Register bit is set when SMBUS_DEV_COUNTER.DEV_COUNTER[7:0] is greater than or equal to SMBUS_THRESHOLD_VALUE.THRESHOLD[7:0]. Otherwise it's cleared. This bit can assert the SMBUS_CTLR_IRQ signal when SMBUS_INTERRUPT_MASK.INT_MASK_THRESHOLD is '0b'.
1	BUF_FULL	R	Register bit is set when either SMBUS_MCU_COUNTER.MCU_COUNTER[7:0] or SMBUS_DEV_COUNTER.DEV_COUNTER[7:0] is greater than or equal to SMBUS_BUF_DEPTH.BUF_DEPTH[7:0] when filling the buffer. Otherwise this bit is cleared. TRANSFER_DIR determines which counter is keeping track of the fill operation. This bit can assert the SMBUS_CTLR_IRQ signal when SMBUS_INTERRUPT_MASK.INT_MASK_BUF_FULL is 1b'0.
0	BUF_EMPTY	R	Register bit is only set when SMBUS_MCU_COUNTER.MCU_COUNTER[7:0] equals SMBUS_DEV_COUNTER.DEV_COUNTER[7:0] and neither counter is at the zero value, and it is cleared in all other cases. This bit can assert the SMBUS_CTLR_IRQ signal when SMBUS_INTERRUPT_MASK.INT_MASK_BUF_EMPTY is 1b'0.

The SMBus Interrupt Status details are shown in Table 129.

TABLE 129: SMBUS_INTERRUPT_STATUS

SMBus Interrupt Status			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7	INT_STAT_DMA_TERM	R/W1C	Interrupt for DMA_TERM status bit. A 0 to 1 transition on the DMA_TERM status bit will set this bit to ‘1b’. Software can only clear this bit by writing a ‘1b’.
6:3	Reserved	R	Always read ‘0’
2	INT_STAT_THRESHOLD	R/W1C	Interrupt for SMBUS_STATUS.THRESHOLD_HIT status bit. A 0 to 1 transition on the SMBUS_STATUS.THRESHOLD_ HIT status bit will set this bit to 1b’1. Software can only clear this bit by writing a ‘1b’.
1	INT_STAT_BUF_FULL	R/W1C	Interrupt for BUF_FULL status bit. A 0 to 1 transition on the BUF_FULL status bit will set this bit to ‘1b’. Software can only clear this bit by writing a ‘1b’.
0	INT_STAT_BUF_EMPTY	R/W1C	Interrupt for BUF_EMPTY status bit. A 0 to 1 transition on the BUF_EMPTY status bit will set this bit to ‘1b’. Software can only clear this bit by writing a ‘1b’.

The SMBus Interrupt Mask details are shown in Table 130.

TABLE 130: SMBUS_INTERRUPT_MASK

SMBus Interrupt Mask			Default: 0x87
Bits	**Name**	**R/W**	**Description**
7	INT_MASK_DMA_TERM	R/W	When 1b’1, prevents the generation of interrupt for SMBUS_STATUS.DMA_TERM status bit = 1b’1.
6:3	Reserved	R	Always read ‘0’
2	INT_MASK_THRESHOLD	R/W	When ‘1b’, prevents the generation of interrupt for SMBUS_STATUS.THRESHOLD_HIT status bit = ‘1b’
1	INT_MASK_BUF_FULL	R/W	When ‘1b’, prevents the generation of interrupt for SMBUS_STATUS.BUF_FULL status bit = ‘1b’
0	INT_MASK_BUF_EMPTY	R/W	When ‘1b’, prevents the generation of interrupt for SMBUS_STATUS.BUF_EMPTY status bit = ‘1b’.

The SMBus FIFO Data details are shown in Table 131.

TABLE 131: SMBUS_FIFO_DATA

SMBus FIFO Data			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:0	FIFO_DATA[7:0]	R/W	Writes to FIFO_DATA[7:0] involve passing DATA to the buffer array pointed to in SMBUS_MCU_ COUNTER.MCU_COUNTER[7:0], and immediately thereafter the SMBUS_MCU_COUNTER.MCU_COUNTER[7:0] is incremented, unless SMBUS_MCU_COUNTER.MCU_COUNTER[7:0] hits the maximum buffer depth. **Note:** Reads from DATA involves reading the buffer array pointed by the SMBUS_MCU_COUNTER.MCU_ COUNTER[7:0], and immediately thereafter the SMBUS_MCU_COUNTER. MCU_COUNTER[7:0] is incremented, unless SMBUS_MCU_COUNTER.MCU_ COUNTER[7:0] hits the maximum buffer depth.

The SMBus Microcontroller Counter details are shown in Table 132.

TABLE 132: SMBUS_MCU_COUNTER

SMBus Microcontroller Counter			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:0	MCU_COUNTER[7:0]	R/W	This register is used as the address for the buffer array. Software can read and write to this register anytime. However it's recommended for software to write this register only when SMBUS_CONTROL.RUN bit is cleared to 1b’0 (i.e. DMA emulation logic is disabled), otherwise DMA operation will be compromised. Hardware will increment this register up to the SMBUS_BUF_DEPTH.BUF_DEPTH[7:0] any time the SMBUS_FIFO_DATA.FIFO_DATA[7:0] is written or read from. **Note:** Software can also Reset MCU_COUNTER[7:0].

The SMBus Device Counter details are shown in Table 133.

TABLE 133: SMBUS_DEV_COUNTER

SMBus Device Counter			Default: 0x00
Bits	**Name**	**R/W**	**Description**
7:0	DEV_COUNTER[7:0]	R/W	This register is used as the address for the buffer array. Software can read and write to this register anytime. However it's recommended for software to write this register only when SMBUS_CONTROL.RUN bit is cleared to '0b'. Hardware will increment this register up to the SMBUS_BUF_DEPTH.BUF_DEPTH[7:0] whenever the I^2C controller is requesting a read or write operation to the buffer array. Software can also Reset DEV_COUNTER[7:0] to 8h'00 by writing '1b' to SMBUS_CONTROL.RESET_COUNTERS.

The SMBus Threshold Value details are shown in Table 134.

TABLE 134: SMBUS_THRESHOLD_VALUE

SMBus Threshold Value			Default: 0x7F
Bits	**Name**	**R/W**	**Description**
7:0	THRESHOLD[7:0]	R/W	This register holds the threshold value. When the SMBUS_DEV_COUNTER.DEV_COUNTER[7:0] is greater than or equal to SMBUS_THRESHOLD_VALUE .THRESHOLD[7:0], SMBUS_STATUS.THRESHOLD_HIT will be asserted.

The SMBus Buffer Depth details are shown in Table 135.

TABLE 135: SMBUS_BUFF_DEPTH

SMBus Buffer Depth			Default: 0x80
Bits	**Name**	**R/W**	**Description**
7:0	BUF_DEPTH[7:0]	R/W	Returns configured buffer depth. Should always be 128 Bytes.

The SMBus Alert Pad Control Register details are shown in Table 136.

TABLE 136: SMBALERT_CTLR_PAD_CTRL_REG

SMBus Alert Pad Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:3	Reserved	R	Always '0b'.

TABLE 136: SMBALERT_CTLR_PAD_CTRL_REG (CONTINUED)

SMBus Alert Pad Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
2	SMBALERT_CTLR_DB	R/W	Debounce enable for SMBUS_CTLR_ALERT_N Pad. 0b - Debounce disabled 1b - Debounce enabled **Note:** It is valid for this pin to be configured with the Debouncer disabled. In this case the raw input is passed through.
1	SMBALERT_CTRL_PD	R/W	Pull down enable for SMBUS_CTLR_ALERT_N Pad. 0b - Pull-down disabled 1b - Pull-down enabled
0	SMBALERT_CTRL_PU	R/W	Pull up enable for SMBUS_CTLR_ALERT_N Pad. 0b - Pull-up disabled 1b - Pull-up enabled **Note:** SMBALERT# is an active-low signal.

The SMBus Alert Debounce Register details are shown in Table 137.

TABLE 137: SMBALERT_CTLR_DB_REG

SMBus Alert Debounce Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:12	Reserved	R	Always ‘0b’.
11:0	SMBUS_CTLR_DB_TIME[11:0]	R/W	This register holds the debounce timer for the SMBUS_ CTLR_ALERT_N Pad. The Debouncer starts when a transition is detected and is passed through if the pin state is maintained for the configured Debouncer time. If another transition occurs, during the Debouncer time, the Debouncer is restarted. Each count corresponds to 1ms, with the default value being 10ms. The timing is derived from a free running clock so will vary with edge alignment. The accuracy of the timing is limited to ± one count.

The SMBus Alert Value Register details are shown in Table 138.

TABLE 138: SMBALERT_CTLR_VAL_REG

SMBus Alert Value Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:2	Reserved	R	Always ‘0b’.
1	SMBALERT_CTLR_DB_VAL	R	Contains the debounced value of the SMBUS_CTLR_ ALERT_N pin. **Note:** The debounced value is used to generate the SMBALERT_INT interrupt.
0	SMBALERT_CTLR_VAL	R	Contains the current value of the SMBUS_CTLR_ALERT_N pin.

The SMBus Event Generation Register details are shown in Table 139.

TABLE 139: SMBUS_GEN_EVENT_REG

SMBus Event Generation Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:11	Reserved	R	Always ‘0b’.
10	SMBALERT	R/W	When set to 1b indicates that the SMBUS_CTLR_ALERT_N pin has been pulled low. Write 1b to clear this bit.
9	I2C_BUF_CTLR_INT	R/W	When set to 1b indicates that there has been an I2C_buf_ctlr_intr from the SMBus Network Engine. Details of the interrupt can be found in the SMBUS_INTERRUPT_STATUS Register. Write 1b to clear this bit.
8	I2C_INT	R/W	When set to 1b indicates that there has been an I2C_intr from the SMBus Controller Core. Details of the interrupt can be found in the STATUS_REG Register. Write 1b to clear this bit.
7:3	Reserved	R	Always ‘0b’.
2	SMBALERT_WAKE	R/W	When set to 1b indicates that the SMBUS_CTLR_ALERT_N pin has been pulled low. Write 1b to clear this bit.
1	I2C_BUF_CTLR_WAKE	R/W	When set to 1b indicates that there has been an I2C_buf_ctlr_intr from the SMBus Network Engine. Write 1b to clear this bit.
0	I2C_WAKE	R/W	When set to 1b indicates that there has been an I2C_intr from the SMBus Controller Core. Write 1b to clear this bit.

The SMBus Event Generation Mask Register details are shown in Table 140.

TABLE 140: SMBUS_GEN_EVENT_MASK_REG

SMBus Event Generation Mask Register			Default: 0x0000_0707
Bits	**Name**	**R/W**	**Description**
31:11	Reserved	R	Always ‘0b’.
10	SMBALERT_INT_MASK	R/W	Mask for the SMBALERT_INT status bit: 0b: generate SMBUS_CTLR_IRQ event when SMBALERT_INT status bit is set. 1b: prevent generation of SMBUS_CTLR_IRQ event when SMBALERT_INT status bit is set.

TABLE 140: SMBUS_GEN_EVENT_MASK_REG (CONTINUED)

SMBus Event Generation Mask Register			Default: 0x0000_0707
9	I2C_BUF_CTLR_INT_MASK	R/W	Mask for the I2C_BUF_CTLR_INT status bit: 0b: generate SMBUS_CTLR_IRQ event when I2C_BUF_CTLR_INT status bit is set. 1b: prevent generation of SMBUS_CTLR_IRQ event when I2C_BUF_CTLR_INT status bit is set.
8	I2C_INT_MASK	R/W	Mask for the I2C_INT status bit: 0b: generate SMBUS_CTLR_IRQ event when I2C_INT status bit is set. 1b: prevent generation of SMBUS_CTLR_IRQ event when I2C_INT status bit is set.
7:3	Reserved	R	Always ‘0b’.
2	SMBALERT_WAKE_MASK	R/W	Mask for the SMBALERT_WAKE_EVENT status bit: 0b: generate SMBUS_CTLR_WAKE event when SMBALERT_WAKE_EVENT status bit is set. 1b: prevent generation of SMBUS_CTLR_WAKE event when SMBALERT_WAKE_EVENT status bit is set.
1	I2C_BUF_CTLR_WAKE_MASK	R/W	Mask for the I2C_BUF_CTLR_WAKE_EVENT status bit: 0b: generate SMBUS_CTLR_WAKE event when I2C_BUF_CTLR_WAKE_EVENT status bit is set. 1b: prevent generation of SMBUS_CTLR_WAKE event when I2C_BUF_CTLR_WAKE_EVENT status bit is set.
0	I2C_WAKE_MASK	R/W	Mask for the I2C_WAKE_EVENT status bit: 0b: generate SMBUS_CTLR_WAKE event when I2C_WAKE_EVENT status bit is set. 1b: prevent generation of SMBUS_CTLR_WAKE event when I2C_WAKE_EVENT status bit is set.

The SMBus PCI Control Register details are shown in Table 141.

TABLE 141: SMBUS_PCI_CTRL_REG

SMBus PCI Control Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:0	Reserved	R	Always ‘0b’.

TABLE 141: SMBUS_PCI_CTRL_REG (CONTINUED)

SMBus PCI Control Register			Default: 0x0000_0000
0	SMB_D3_CLK_EN	R/W	Enables the 62.5 MHz clock when the PCI function is in D3: 0b: 62.5 MHz clock is disabled when the PCI function is in D3. 1b: 62.5 MHz clock is enabled when the PCI function is in D3. The Peripheral PCIe Endpoint uses aggregated clock request bits CLK_REQ_REG.PERI_CRYSTAL_REQ and CLK_REQ_REG.PERI_WR_PLL_REQ to indicate its need for the crystal clock and the AB WR PLL, When SMB_D3_CLK_EN is '1b' CLK_REQ_REG.PERI_CRYSTAL_REQ and CLK_REQ_REG.PERI_WR_PLL_REQ shall not be set to '0b' even if all Physical Functions are in D3.

The SMBus Reset Register details are shown in Table 142.

TABLE 142: SMBUS_RESET_REG

SMBus Reset Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:17	Reserved	R	Always '0b'.
16	PERI_SMBUS_D3_RESET_DIS	R/W	D3 Reset Disable This bit modifies the subsytem Reset operation when exiting D3Cold (PCIE_PERST_N deasserts). When this bit is set and the VAUX_DET pin is a high only the PCIe interface and related logic are Reset; the SMBus Controller is not Reset. Otherwise, the full subsystem is Reset.
15:10	Reserved	R	Always '0b'.
9	PERI_SMBUS_RELOAD_PCI	R/W	Defines the configuration to be carried out on the Peripheral SMBUS PCIe Endpoint PF1 when PERI_SMBUS_RELOAD is set; this corresponds to the PERI_SMBUS_PCIE_EP_ADDR_BASE. When set to '1' the Peripheral SMBus PCIe Endpoint PF1 will be reloaded to PERI_SMBUS_PCIE_EP_ADDR_BASE. When set to '0' the Peripheral SMBus PCIe Endpoint PF1 configuration will not be reloaded to PERI_SMBUS_PCIE_EP_ADDR_BASE. The System Config HW will attempt to write all configuration registers associated with a particular configuration; the Subsystem will block any writes corresponding to the PCIe Endpoint at PERI_SMBUS_PCIE_EP_ADDR_BASE.

TABLE 142: SMBUS_RESET_REG (CONTINUED)

SMBus Reset Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
8	PERI_SMBUS_RELOAD_FN	R/W	Defines the configuration to be carried out on the Peripheral SMBus function IP when PERI_SMBUS_RELOAD is set; Peripheral SMBus function IP corresponds to all Subsytem addresses in SMBUS_PERI_ADDR_BASE, except for those corresponding PERI_SMBUS_PCIE_EP_ADDR_ BASE, which are controlled by the PERI_SMBUS_RELOAD _PCI bit. When set to '1' the Peripheral SMBus function IP configuration will be reloaded. When set to '0' the Peripheral SMBus function IP configuration will not be reloaded. The System Config HW will attempt to write all configuration registers associated with a particular configuration; the Subsystem will block any writes corresponding to the function IP.
7:3	Reserved	R	Always '0b'.
2	PERI_SMBUS_RELOAD	R/W/ SC	Initiate Peripheral SMBus Function RELOAD PERI_SMBUS_RELOAD_PCI and PERI_SMBUS_RELOAD _FN should be configured appropriately prior to setting PERI_SMBUS_RELOAD. This bit self-clears once the RELOAD is complete.
1	PERI_SMBUS_LRST	R/W/ SC	Initiate Peripheral SMBus Function LRST. This bit self-clears once the LRST is complete.
0	PERI_SMBUS_SRST	R/W/ SC	Initiate Peripheral SMBus Function SRST. This bit self-clears once the SRST is complete. Setting this bit will Reset the entire peripheral subsystem due to the PCI link being taken down.

The SMBus No Snoop Latency Value Register details are shown in Table 143.

TABLE 143: SMBUS_LTR_VALUE_REG

SMBus No Snoop Latency Value Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31	SMBUS_NO_SNOOP_REQ	R/W	No-Snoop Requirement When set, indicates that the SMBUS_NO_SNOOP_ LATENCY_VAL[9:0] and SMBUS_NO_SNOOP_LATENCY _SCALE[2:0] fields describe the latency requirement. When cleared, there is no latency requirement and the SMBUS_NO_SNOOP_LATENCY_VAL[9:0] and SMBUS_NO_SNOOP_LATENCY_SCALE[2:0] fields are ignored.

TABLE 143: SMBUS_LTR_VALUE_REG (CONTINUED)

SMBus No Snoop Latency Value Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
30:29	Reserved	R	Always ‘0b’.
28:26	SMBUS_NO_SNOOP_LATENCY_SCALE[2:0]	R/W	No-Snoop Latency Scale Scale of the SMBUS_NO_SNOOP_LATENCY_VAL[9:0] field: 000b - value times 1 ns 001b - value times 32 ns 010b - value times 1024 ns 011b - value times 32768 ns 100b - value times 1048576 ns 101b - value times 33554432 ns 110b - value times not permitted 111b - value times not permitted
25:16	SMBUS_NO_SNOOP_LATENCY_VAL[9:0]	R/W	No-Snoop Latency Value Latency value field.
15	SMBUS_SNOOP_REQ	R/W	Snoop Requirement When set, indicates that the SMBUS_SNOOP_LATENCY_VAL[9:0] and SMBUS_SNOOP_LATENCY_SCALE[2:0] fields describe the latency requirement. When cleared, there is no latency requirement and the SMBUS_SNOOP_LATENCY_VAL[9:0] and SMBUS_SNOOP_LATENCY_SCALE[2:0] fields are ignored.
14:13	Reserved	R	Always ‘0b’.
12:10	SMBUS_SNOOP_LATENCY_SCALE[2:0]	R/W	Snoop Latency Scale Scale of the SMBUS_SNOOP_LATENCY_VAL[9:0] field: 000b - value times 1 ns 001b - value times 32 ns 010b - value times 1024 ns 011b - value times 32768 ns 100b - value times 1048576 ns 101b - value times 33554432 ns 110b - value times not permitted 111b - value times not permitted
9:0	SMBUS_SNOOP_LATENCY_VAL[9:0]	R/W	Snoop Latency Value Latency value field.

The SMBus Data Buffer Register details are shown in Table 144.

TABLE 144: SMBUS_DATA_BUFFER

SMBus Data Buffer Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31	SMBUS_BUFFER	R/W	SMBus Data Buffer

The General Purpose Register details are shown in Table 145.

TABLE 145: GPR0_REG

General Purpose Register			Default: 0x0000_0000
Bits	**Name**	**R/W**	**Description**
31:2	Reserved	R	Always 0's
1:0	SMBUS_SPEED_SEL	R/W	Selects the Speed of the SMBus interface. 00b: 400 kHz "Fast mode" 01b: 100 kHz "Standard" 10b: 1M Hz "Fast mode Plus" 11b: User-defined - all SMBus controller timings are defined in PCI1XXXX configuration and the I2C/SMBus driver will not overwrite. This is a software defined register. The I2C/SMBus device driver reads this register. If a value of 00b, 01b, or 10b is read, then the driver automatically sets all register settings per Table 103 to the selected value. If a value of 11b is read, the driver does not modify any speed settings of the interface.

10.9 SMBus Configuration Example

10.9.1 PCI11414 SMBUS SET UP WITH A0 SILICON WORKAROUND

1. Select programmable pins to function as SMBus.

In programmable pin mux table (Table 146), identify which PF pins your system shall use to expose each of SMBUS _CTLR_SCL, SMBUS_CTLR_SDA, and SMBUS_CTLR_ALERT_N (if Alert input function is necessary).

In SYSTEM_REG_ADDR_BASE.PPCTL_ADDR_BASE, locate the offset corresponding to the PFx pins being used to expose the SMBUS function pins and write the corresponding function selection to PPx_CTL_REG.SELECT[3:0] and drive strength to PPx_CTL_REG.DRIVE_STRENGTH[1:0].

TABLE 146: PROGRAMMABLE PIN MUX

	Clock	Data	Alert
Pin	PF32	PF33	PF34
Register	0x0024_0480	0x0024_0484	0x0024_0488
Value	0x0000_2006	0x0000_2003	0x0000_0001
Remarks	Select Function 6 (SMBUS_CTLR_SCL) & set DRIVE_STRENGTH=8mA	Select Function 6 (SMBUS_CTLR_SDA) & set DRIVE_STRENGTH=8mA	Select Function 1 (SMBUS_CTLR_ALERT_N) & set DRIVE_STRENGTH=2mA (**Note:** Drive strength not applicable for Input pin)

2. Program GPR0_REG (Address: 0x0015_1C00) with SCL speed selection for the driver to program the SMBus controller timing registers appropriately. For A0 silicon, this must be set to 0x0000_00003 at all times.
3. Write each of the values in Table 103 to the desired speed selection.
4. The DATA_TIMING_REG.DATA_HOLD[7:0] must be overwritten to value: 0x02 in order to prevent arbitration loss in SMBus controller logic, otherwise SMBus controller is inoperable.

11.0 PERIPHERAL SUBSYSTEM IMPLEMENTATION - SPI

Note: The details outline in this section are not necessary for most end system integrators. This information is outlined as it may be useful for debugging, qualifying functions manually via direct PCIe register access, and correlating what is occurring on the physical hardware level when examining a PCIe protocol trace.

The SPI Controller Subsystem occupies its own PCI device endpoint and requires a separate device driver to function from a host system. This subsystem shares the PCI bus with all other peripheral devices (UART, SMBus, and GPIO).

The Subsystem IDs are:

- Vendor ID: 0x1055 (Microchip Technology, Inc / SMSC)
- Device ID:
  - PCI12000: 0xA004
  - PCI11010: 0xA014
  - PCI11101: 0xA024
  - PCI11400: 0xA034
  - PCI11414: 0xA044
- Class ID: 0x0C8000 (“Other Interface”)

The SPI subsystem supports up to 7 devices through 7 separate ‘chip enable’ signals and multiple baud clock frequencies which are derived from a 62.5 MHz clock. The frequencies supported are listed in Table 147.

TABLE 147: PROGRAM FUNCTION REGISTER FORMAT

Approximate Frequency	Actual Frequency	62.5 MHz Divider Value
30 MHz	31.25 MHz	2
20 MHz	20.83 MHz	3
15 MHz	15.63 MHz	4
12 MHz	12.50 MHz	5
10 MHz	10.42 MHz	6
2 MHz	2.08 MHz	7

Note: The 62.5 MHz clock must be divided by some factor greater than 1. 62.5 MHz SPI operation is not supported.

11.1 Sections

11.2 Obtaining Access Without SPI Device Driver Installed

Before direct access to the PCIe registers are allowed from the host, you must:

1. Ensure that the COMMAND field of the SPI peripheral general PCI Configuration space is properly set. A value of 0x0000 will block all access to the PCI register space. When a driver is not loaded for the SPI peripheral, the COMMAND field is left at 0x00 and must be manually overridden. The recommended value is 0x0406.
2. Set the respective SYS_LOCK register for the SPI subsystem.

11.3 Summary of Operation

The SPI Interface uses command and response buffers, along with some control registers to control SPI access.

- **SPI_CMD_BUF[319:0]:** 320 Byte command buffer. There is a 320 byte response buffer.

- **SPI_RESP_BUF[319:0]:** 320 Byte Response buffer. There is a length register counting out the number of bytes.
- **SPI_CMD_LEN:** Length counting register in Bytes.
- **SPI_CTL:** Control register containing a self clearing GO bit which initiates SPI access. Once the GO bit is set, the SPI hardware interface asserts the selected chip enable pin (SPI_CEN), and starts oscillating the clock, transmitting the SPI_CMD_BUF buffer, and storing received data to the SPI_RESP_BUF. The clock will oscillate for SPI_CMD_LEN * 8 clock cycles.

11.3.1 ARBITRARY LENGTH SPI ACCESS EXAMPLE

Note:	During an arbitrary length SPI access, the software sets chip enable, using the SPI_CTLR_CTL_REG. FORCE_CE bit, and successively writes data to the transmit buffer. At the end of the transmission, the software must clear the SPI_CTLR_CTL_REG.FORCE_CE bit to release chip enable.

For a SPI access of arbitrary length:

1. Software sets SPI_CTLR_CTL_REG.MODE_SEL to the desired mode.
2. Software sets SPI_CTLR_CTL_REG.DEV_SEL[2:0] to the desired device.
3. Software sets SPI_CTLR_CTL_REG.CMD_LEN[7:0] to the desired length.
4. Software sets SPI_CTLR_CTL_REG.FORCE_CE bit. This forces the chip enable low.
5. Software writes the output byte(s) to SPI_CTLR_CMD_BUF.SPI_CMD_BUF[0:319].
6. If an interrupt is needed, software clears the SPI_CTLR_CTL_REG.SPI_INT_MASK.
7. Software sets the SPI_CTLR_CTL_REG.GO bit.
8. PCI1xxxx transmits the data from SPI_CTLR_CMD_BUF.SPI_CMD_BUF[0:319] and receives the response into the SPI_CTLR_RSP_BUF.SPI_RSP_BUF[0:319].
9. When this process has completed, PCI1xxxx clears the SPI_CTLR_CTL_REG.GO bit.
10. If SPI_CTLR_CTL_REG.SPI_INT_MASK is cleared, PCI1xxxx signals a SPI_CTLR_IRQ.
11. If SPI_CTLR_CTL_REG.SPI_INT_MASK is set when the SPI_CTLR_CTL_REG.GO bit clears, the software reads the response from the SPI_CTLR_RSP_BUF.SPI_RSP_BUF[0:319].
12. If SPI_CTLR_CTL_REG.SPI_INT_MASK is set, the software waits for the SPI_CTLR_IRQ and reads the response from the SPI_CTLR_RSP_BUF.SPI_RSP_BUF[0:319].
13. If there is more data, go to step 5.
14. Once all data has been transmitted, software clears the SPI_CTLR_CTL_REG.FORCE_CE bit; this releases chip enable.

11.3.2 FIXED LENGTH SPI ACCESS EXAMPLE

Note:	During a fixed length SPI access SW writes one block of data to SPI_CTLR_CMD_BUF.SPI_CMD_BUF[0:319] for transmission. When the software sets the SPI_CTLR_CTL_REG.GO bit, PCI1xxxx sets chip enable for the duration of the transmission. PCI1xxxx releases chip enable once the transmission is complete (HW does not set the SPI_CTLR_CTL_REG.FORCE_CE bit).

For a SPI access of fixed length:

1. Software sets SPI_CTLR_CTL_REG.MODE_SEL to the desired mode.
2. Software sets SPI_CTLR_CTL_REG.DEV_SEL[2:0] to the desired device.
3. Software sets SPI_CTLR_CTL_REG.CMD_LEN[7:0] to the desired length.
4. Software writes the output byte(s) to SPI_CTLR_CMD_BUF.SPI_CMD_BUF[0:319].
5. If an interrupt is needed, software clears the SPI_CTLR_CTL_REG.SPI_INT_MASK.
6. Software sets the SPI_CTLR_CTL_REG.GO bit.
7. PCI1xxxx forces the chip enable low.
8. PCI1xxxx transmits the data from SPI_CTLR_CMD_BUF.SPI_CMD_BUF[0:319] and receives the response into the SPI_CTLR_RSP_BUF.SPI_RSP_BUF[0:319].
9. When this process has completed, HW forces the chip enable high and clears the SPI_CTLR_CTL_REG.GO bit.
10. If SPI_CTLR_CTL_REG.SPI_INT_MASK is cleared, PCI1xxxx signals a SPI_CTLR_IRQ.
11. If SPI_CTLR_CTL_REG.SPI_INT_MASK is set when the SPI_CTLR_CTL_REG.GO bit clears, the software

reads the response from the SPI_CTLR_RSP_BUF.SPI_RSP_BUF[0:319].

12. If SPI_CTLR_CTL_REG.SPI_INT_MASK is set, the software waits for the SPI_CTLR_IRQ and reads the response from the SPI_CTLR_RSP_BUF.SPI_RSP_BUF[0:319].

11.4 Suspend

When the SPI_PCI_CTRL_REG.SPI_D3_CLK_EN bit is set to '0' then the 62.5 MHz clock to the SPI Controller instance will be gated off when the SPI Controller PCIe Endpoint function goes to D3. This is defined as the SPI module being suspended. The clocks will be restored when the SPI Controller PCI Function goes to D0.

When the SPI_PCI_CTRL_REG.SPI_D3_CLK_EN bit is set to '1' then the 62.5 MHz clock to the SPI Controller instance will not be gated off when the SPI Controller PCIe Endpoint function goes to D3; the module will continue to operate.

This also means that CLK_REQ_REG.PERI_CRYSTAL_REQ and CLK_REQ_REG.PERI_WR_PLL_REQ must both be set to '1' even if all peripheral PCIe Endpoint functions are in the D3 state. This is defined as the SPI module not being suspended.

11.5 Interrupts and Wake

The SPI Controller block will generate a SPI_CTLR_IRQ when this is unmasked; this triggers an MSI/MSI-X/Legacy interrupt via the Peripheral PCIe Endpoint. The SPI Controller block will generate a SPI_CTLR_WAKE when this is unmasked; this triggers PCI Wake via the Peripheral PCIe Endpoint. See Table 148.

Note: SPI_CTLR_IRQ and SPI_CTLR_WAKE will be asserted when enabled by the appropriate interrupt enable/mask register regardless of the PCIe Peripheral Device state or Link state. On the assertion the PCIe Endpoint will determine whether PCI Wake#/PME events are needed based on the present PCIe Link state. If PME is enabled and PME support is configured for current PMCSR D-state asserting this signal causes the controller to wake from either L1 or L2 state (if needed). When the controller has transitioned back to the L0 state it transmits a PME message and sets the PME_Status. Upon receiving the PME message the PCIe Root Complex should clear the PME_Status and change the D-state back to D0.

TABLE 148: INTERRUPT AND WAKE OPERATION

SPI EVENT	Masked	Link State	Device State	PCIe Event
SPI_CTLR_WAKE	Yes	All	All	None
	No	L0	D0	MSI/MSI-X/Legacy Interrupt
	No	L1/L1ss/L2	D0	Undefined
	No	L0	D3	Undefined
SPI_CTLR_WAKE	Yes	All	All	None
	No	L0	D3	Send PME_Status
	No	L1/L1ss/L2	D3	Generate WAKE# followed by PME_Status on transition to L0.
	No	All	D0	Undefined

11.6 Interrupts

The SPI Controller interrupt is triggered whenever the SPI_CTLR_CTL_REG.GO bit is cleared at the end of a transaction. This interrupt is indicated by the SPI_CTLR_EVENT_REG.SPI_INT bit being set and can be masked by setting the SPI_CTLR_EVENT_MASK_REG.SPI_INT_MASK bit.

The SPI Controller Alert interrupt is triggered whenever the SPIALERT_CTLR_VAL_REG.SPIALERT_CTLR_VAL_REG bit is set. This interrupt is indicated by the SPI_CTLR_EVENT_REG.SPI_ALERT_INT bit being set and can be masked by setting the SPI_CTLR_EVENT_MASK_REG.SPI_ALERT_INT_MASK bit.

When the SPI PCIe Endpoint function is in D0 the SPI_CTLR_IRQ output indicates if a SPI interrupt is pending. This is used to generate an MSI or MSI-X event or legacy PCI interrupt via PCIe; if operating in MSI/MSI-x with separate vectors, the interrupt vector indicating which SPI instance, 0 or 1, generated the interrupt.

When the SPI PCIe Endpoint function is in D3 the SPI_CTLR_WAKE output indicates if a SPI Controller interrupt is pending. This is used to generate a wake event via PCIe.

11.6.1 WAKE-UP SUPPORT

The SPI Controller is capable of asynchronous wake-up. The SPI Controller indicates this occurrence via the SPI_CTLR_WAKE event. SPI_CTLR_WAKE events can occur either when the SPI Controller is in suspend or when it is not suspended.

When the PCIe Endpoint function is in D3 and the SPI Controller is not in suspend:

- The SPI_CTLR_WAKE event is generated when the SPI_CTLR_CTL_REG.GO bit is cleared at the end of a transaction.
- An SPI_CTLR_WAKE event is also generated when the SPI_CTLR*_ALERT_N pin is pulled low and the debounced input has settled. At this point the SPI_CTLR_EVENT_REG.SPI_ALERT_WAKE bit is also set.

When the PCIe Endpoint function is in D3 and the SPI Controller is suspended the SPI_CTLR_WAKE event is generated when the SPI_CTLR*_ALERT_N pin is pulled low and the debounced input has settled. At this point the SPI_CTLR_EVENT_REG.SPI_ALERT_WAKE bit is also set.

A SPI_CTLR_WAKE event causes PCI Wake signalling to be generated. This wake request should trigger the PCIe Host to take the SPI Controller into the D0 device state. At this point the aggregated peripheral PLL Request CLK_REQ_REG.PERI_WR_PLL_REQ and the aggregated peripheral Crystal Request CLK_REQ_REG.PERI_CRYSTAL_REQ will both be set to true by the peripheral subsytem since one or more Physical Functions are now in D0.

Wake events are indicated by bits in the SPI_CTLR_EVENT_REG Register and can be masked by setting the corresponding bit in the SPI_CTLR_EVENT_MASK_REG Register.

11.6.2 SPI ALERT

The PCI1xxxx provides this as an input from the SPI Peripheral via the SPI_CTLRx_ALERT_N pins when these have been configured.

A SPI Peripheral-only device can signal the host through SPIALERT# that it wants to talk. When the SPI_CTLRx_ALERT_N pin is pulled low this means that the SPI Peripheral needs to communicate with the SPI Controller. When the SPI Controller is in a suspend state a SPI_CTLR_WAKE event will be triggered. When the SPI Controller is not in suspend a SPI_CTLR_IRQ interrupt is generated.

Pad control for the SPI_CTLRx_ALERT_N pin is provided in the SPI_CTLR_PAD_CTL_REG Register; a time for debouncing the input used to trigger the interrupt can be set in SPIALERT_CTLR_DB_REG.SPI_CTLR_DB_TIME[11:0].

Both current and debounced values of the SPI_CTLRx_ALERT_N pin are provided in the SPIALERT_CTLR_VAL_REG Register.

When the SPI_CTLRx_ALERT_N pins are used to wake-up the system, without the De-bouncer being enabled (SPI_CTLR_PAD_CTL_REG.SPIALERT_CTLR_DB=0), then SPI_CTLRx_ALERT_N needs to be asserted for a minimum of ~6 µs by the SPI Peripheral. ~4 µs is used to request Ring Oscillator clock to start up and 2 µs is used to generate the wake message to Host.

11.6.3 PRESERVING FUNCTIONAL CONTEXT ON D3COLD EXIT

A Warm Reset occurs when the device is operating on auxiliary power or multiplexed main / auxiliary power (VAUX_DET pin is configured and is high) and PCIE_PERST_N is asserted and then deasserted.

Normally, the entire PCIe Endpoint (excluding the PME context) is Reset upon exiting the D3cold state. System software (typically the OS) is required to reinitialize the PCIe configuration registers. Driver software is required to reinitialize the PCIe Endpoint functionality.

Although the saving of SMBus Controller functional context during D3cold is possible, the subsequent device Reset will cause the contents of the receive FIFOs and registers to be cleared.

A Reset option is available which, when enabled, suppresses the Reset of the SMBus Controller's functional context by a Warm Reset. When the SPI_RESET_REG.PERI_SPI_D3_RESET_DIS bit is set, only the PCIe Endpoint controller, configuration registers and related logic are Reset; the SMBus Controller's functionality is not Reset. Once the OS reinitializes the PCIe configuration registers and places the PCIe Endpoint into the D0a state, the PCIe Endpoint, under driver control, can continue with normal operation, with no loss of the functional context.

11.7 Register Map

There are two instances of the SPI Controller each with its own instance of the SPI Controller register set. The memory offsets for these memory blocks with respect to BAR 0/1 for PF0 of the Physical Function are shown below. The offsets also give the address with respect to SPI_CONTROLLER_ADDR_BASE when accessed via JTAG/SPI/SMBUS.

Certain registers apply to all SPI instances and not just one instance. The following Registers only appear in the SPI0 register set; applications should not read or write these same register offset in the SPI1 register set.

- SPI_PCI_CTRL_REG
- SPI_RESET_REG

There are a total of two SPI interfaces present with identical register maps, see Table 149.

TABLE 149: SPI CONTROLLER REGISTERS

SPI0 Offset Address	SPI1 Offset Address	Name	R/W	Description
0x000-0x13Fh	0x800-0x93F	SPI_CTLR_CMD_BUF	R/W	Command Output Buffer up to 320 Bytes. *Default Value = 0x0000_0000*
0x140 - 0x1FF	0x840 - 0x8FF	Reserved	R	Reserved
0x200 - 0x23F	0x900 - 0x93F	SPI_CTLR_RSP_BUF	R/W	Response Input Buffer up to 320 Bytes. *Default Value = 0x0000_0000*
0x340 - 0x3FF	0xA40 - 0xAFF	Reserved	R	Reserved

TABLE 149: SPI CONTROLLER REGISTERS (CONTINUED)

SPI0 Offset Address	SPI1 Offset Address	Name	R/W	Description
0x400	0xB00	SPI_CTLR_CTL_REG	R/W/SC	SPI Mode Control Default Value = 0x0000_0040 Bits [31:28] - Reserved (all 0's) Bits [27:25] - DEV_SEL[2:0] Selects Device to Access • 000b - SPIx_CEN0 • 001b - SPIx_CEN1 • 010b - SPIx_CEN2 • 011b - SPIx_CEN3 • 100b - SPIx_CEN4 • 101b - SPIx_CEN5 • 110b - SPIx_CEN6 • 111b - Reserved Bits [24:17] - Reserved (all 0's) Bits [16:8] - CMD_LN[7:0] Bits [7:5] - SPI_SPEED[2:0] Selects SPI Speed through 62.5 MHz divider factor • 000b - Off • 001b - Reserved • 010b - 30 MHz • 011b - 20 MHz • 100b - 15 MHz • 101b - 12 MHz • 110b - 10 MHz • 111b - 2 MHz Bit [4] - FORCE_CE When this bit is set, it forces the SPI chip enable low. This bit should only be set when using the SPI command buffer. It should never be set when doing direct accesses from the bus. Bit [3] - Reserved (0b) Bit [2] - MODE_SEL Selects SPI Clock mode • 1b - Mode 0 • 0b - Mode 3 Bit [1] - Reserved (0b) Bit [0] - GO (Self Clearing Bit) Setting this bit initiates the SPI transaction
0x404 - 0x41F	0xB04 - 0xB1F	Reserved	R	Reserved

TABLE 149: SPI CONTROLLER REGISTERS (CONTINUED)

SPI0 Offset Address	SPI1 Offset Address	Name	R/W	Description
0x420	0xB20	SPI_CTLR_CTL_REG	R/W1C	Event Register Default Value = 0x0000_0000 Bits [31:10] - Reserved (all 0's) Bit [2] - SPI_ALERT_INT This bit is set every time the SPIALERT_CTLR_VAL_REG.SPIALERT_CTLR_DB_VAL bit is set to 1b'1. This bit causes an SPI_CTLR_IRQ interrupt to be triggered. Write 1b'1 to clear. Bit [2] - SPI_INT This bit is set every time the SPI_CTLR_CTL_REG.GO bit is cleared at the end of a transaction. This bit causes an SPI_CTLR_IRQ interrupt to be triggered. Write 1b'1 to clear. Bits [7:2] - Reserved (all 0's) Bit [1] - SPI_ALERT_WAKE When 1b'1, indicates the wake event was due to a SPI alert being asserted (see Section 11.6.2, "SPI Alert"). This bit causes an SPI_CTLR_WAKE event to be triggered. Write 1b'1 to clear. Bit [0] - SPI_INT_WAKE When 1b'1, indicates the wake event was due to the SPI_CTLR_CTL_REG.GO bit being cleared at the end of a transaction. This bit causes an SPI_CTLR_WAKE event to be triggered. Write 1b'1 to clear.
0x424	0xB24	SPI_CTL-R_EEVENT_REG	R/W	Event Mask Register Default Value = 0x0000_0303
0x428 - 0x45F	0xB28 - 0xB5F	Reserved	R	Reserved
0x460	0xB60	SPI_CTLR_	R/W	SPI Controller Pad Control Register Default Value = 0x0000_0008
0x464	0xB64	SPI_CTLR_	R/W	SPIALERT# Pad Debounce Register Default Value = 0x0000_000A
0x468	0xB68	SPI_CTLR_	R	SPI ALTER# Value Register Default Value = 0x0000_0003
0x470 - 0x47F	0xB70 - 0xB7F	Reserved	R	Reserved
0x480	n/a	SPI_PCI_CTRL_REG	R/W	SPI PCI Control Register Default Value = 0x0000_0000
0x484	n/a	SPI_RESET_REG	R/W /SC	SPI Reset Register Default Value = 0x0000_0000
0x488	0xB88	SPI_LTR_VAL UE_REG	R/W	SPI LTR Value Register Default Value = 0x0000_0000
0x48C - 0x7FF	0xB8C - 0xFFF	Reserved	R	Reserved

12.0 PERIPHERAL SUBSYSTEM IMPLEMENTATION - GPIO

The GPIO Subsystem occupies its own PCI device endpoint and requires a separate device driver to function from a host system. This subsystem shares the PCI bus with all other peripheral devices (UART, SMBus, and SPI).

The Subsystem IDs are:
- Vendor ID: 0x1055 (Microchip Technology, Inc./SMSC)
- Device ID:
  - PCI12000: 0xA005
  - PCI11010: 0xA015
  - PCI11101: 0xA025
  - PCI11400: 0xA035
  - PCI11414: 0xA045
- Class ID: 0x130000 (“Non-Essential”)

GPIO registers will only affect a PROG pin when that PROG pin function selection is set to GPIO (always FUNCTION 0). A GPIO may be configured as an output or an input.

12.1 Sections

12.2 Base Addresses

Configuration and control of the GPIO subsystem requires correct usage of different base addresses depending on the use-case. Typically, users will utilize the function calls enabled via the peripheral system drivers. This knowledge is only essential when constructing configuration files using raw register modification (instead of MPLAB Connect GUI interface selections) or when accessing PCI registers directly. See Table 150.

TABLE 150: PROGRAM FUNCTION REGISTER FORMAT

Access Pathway	Base Address
Configuration (OTP or EEPROM)	0x3_0000
Run-time Access via PCIe	Obtained via CPU assignment during system enumeration. Locate the peripheral using PCI bus examination tools and obtain the “BAR0” address.
Run-time Access via SMBus/SPI	0x3_0000

12.3 GPIO as Output

Any GPIO can be configured as either a push/pull or open drain output. The output enable must be set in the PIO32_OUT_EN, PIO64_OUT_EN, and PIO96_OUT_EN register(s). The Pin mode is set as push/pull by default, but can be optionally set to open drain via the PIO32_OD, PIO64_OD, and PIO96_OD registers.

The Output state is controlled directly through the PIO32_OUT, PIO64_OUT, and PIO96_OUT registers.

12.4 GPIO as Input

When a GPIO is configured as an input, its status may be monitored either through or a polling mechanism (i.e.: periodically reading the input status) or the pin may be configured to monitor for events and generate an interrupt.

12.4.1 GPIO INPUT EVENTS

A GPIO, configured as an input, can be configured to trigger events on either the raw or debounced input. Events (interrupts or wake events) from the PIOs can be programmed to be edge or level based.

For edge-based events, the trigger can be rising edge, falling edge, or both.

For level-based events, either level can be selected. The level mask and polarity must be set.

Falling Edge, Rising Edge or Level PIO events are recorded in the PIOxx_STATUS registers depending on the settings of the PIOxx_MODE, PIOxx_HI_TO_LO_EDGE_CONFIG, PIOxx_LO_TO_HI_EDGE_CONFIG, PIOxx_LEVEL_CONFIG and PIOxx_LEVEL_MSK Registers for the particular PIO.

PIOxx_STATUS register must be cleared by software explicitly. Writing 1b to a bit clears the bit and enables the detection of the next level transition.

Note:	If the PIO is configured for level via PIOxx_MODE, and the level of the pin remains active, the PIOxx_STATUS will remain set when software writes 1b to it.

A 1b in any bit in the PIOxx_STATUS register will force either:

- A PIO_IRQ interrupt event if the Peripheral General Function is in D0 if the interrupt has not been masked by the corresponding bit int the PIOxx_INT_MASK Register.
- A PIO_WAKE event if the Peripheral General function is in D3 if wake-up has not been masked by the corresponding bit in the PIOxx_WAKE_MSK register.

Note:	PIO_IRQ and PIO_WAKE will be triggered when enabled by the appropriate interrupt enable/mask register regardless of the PCIe Peripheral Device state or Link state. For Wake the PCIe Endpoint will determine whether Wake/PME events are needed based on the present PCIe Link state.

12.4.2 GPIO INPUT WAKE-UP SUPPORT

The PIOs are capable of asynchronous wake-up while the PIO PCIe Endpoint function is in D3. The PIO indicates this occurrence via the PIO_WAKE event. PIO_WAKE events can occur either when the PIO is in suspend or when it is not suspended.

A PIO_WAKE event causes PCI Wake signaling to be generated. This wake request should trigger the PCIe Host to take the PIO PCIe Endpoint function into the D0 device state. At this point the aggregated peripheral PLL Request CLK_REQW_REG.PERI_WR_PLL_REQ and the aggregated peripheral Crystal Request CLK_REQ_REG.PERI_CRYSTAL_REQ will both be set to true by the peripheral subsystem since once or more Physical Functions are now in D0.

Wake events can be masked by setting the corresponding bit in the PIOx_WAKE_INT_MSK Register. When PIO_WAKE event occurs on a pin, without the Debouncer being enabled (PIOxx_DEBOUNCE[xx] = 0), then the signal needs to be asserted for a minimum of 6 µs. 4 µs is used to request ROSC clock, and 2 µs is used to generate the wake message to Host.

12.5 GPIO Operation During Suspend

When the PIO_PCI_CTRL_REG.PIO_D3_CLK_EN bit is set to '0' and the PIO PCIe Endpoint function goes to the PCIe D3 state then the 62.5 MHz clock to the PIOs will be gated off. When this occurs, the PIO module considered 'suspended'.

The clocks will be restored when the PIO PCI Function goes back to D0. When the PIO_PCI_CTRL_REG.PIO_D3_CLK_EN bit is set to '1', then the 62.5 MHz clock to the PIOs will not be gated off when the PIO PCIe Endpoint function goes to D3. The module will continue to operate. This also means that CLK_REQ_REG.PERI_CRYSTAL_REQ and CLK_REQ_REG.PERI_WR_PLL_REQ must both be set to '1' even if all peripheral PCIe Endpoint functions are in the D3 state.

12.6 GPIO Configuration Registers

These GPIO Configuration Registers allow properties to be set to modify the behavior of the GPIO pins. For each setting, there are three 32-bit registers where one bit corresponds to one of the 96 GPIO pins. Settings include input/output enable, debounce enable, pin status, interrupt enable, pull-up/pull-down enable and more as seen in Table 151.

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS

Offset Address	Name	R/W	Description	Default Value
0x0003_4000	PIO32_OUT_EN	R/W	GPIO Output Enable Register 0b - Output Disabled 1b - Output Enabled Bit Mapping: • Bit 0 - PIO1 • Bit 1-30 - PIO2 • Bit 31 - PIO32	0x0000_0000
0x0003_4004	PIO64_OUT_EN	R/W	GPIO Output Enable Register 0b - Output Disabled 1b - Output Enabled Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0x0000_0000
0x0003_4008	PIO96_OUT_EN	R/W	GPIO Output Enable Register 0b - Output Disabled 1b - Output Enabled Bit Mapping: • Bit 0 - PIO65 • Bit 1-27 - PIO66 • Bit 28 - PIO93 • Bit 29-31 - Unused	0x0000_0000
0x0003_400C - 0x0003_400F	Reserved	R	Reserved	All 0's
0x0003_4010	PIO32_INP_EN	R/W	GPIO Input Enable Register 0b - Input Disabled 1b - Input Enabled Bit Mapping: • Bit 0 - PIO1 • Bit 1-30 - PIO2 • Bit 31 - PIO32	0x0000_0000

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS (CONTINUED)

Offset Address	Name	R/W	Description	Default Value
0x0003_4014	PIO64_INP_EN	R/W	GPIO Input Enable Register 0b - Input Disabled 1b - Input Enabled Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0x0000_0000
0x0003_4018	PIO96_INP_EN	R/W	GPIO Input Enable Register 0b - Input Disabled 1b - Input Enabled Bit Mapping: • Bit 0 - PIO65 • Bit 1-27 - PIO66 • Bit 28 - PIO93 • Bit 29-31 - Unused	0x0000_0000
0x0003_401C - 0x0003_401F	Reserved	R	Reserved	All 0’s
0x0003_4020	PIO32_OUT	R/W	GPIO Output State Register 0b - Drive Low 1b - Drive High Bit Mapping: • Bit 0 - PIO1 • Bit 1-30 - PIO2 • Bit 31 - PIO32	0x0000_0000
0x0003_4024	PIO64_OUT	R/W	GPIO Output State Register 0b - Drive Low 1b - Drive High Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0x0000_0000
0x0003_4028	PIO96_OUT	R/W	GPIO Output State Register 0b - Drive Low 1b - Drive High Bit Mapping: • Bit 0 - PIO65 • Bit 1-27 - PIO66 • Bit 28 - PIO93 • Bit 29-31 - Unused	0x0000_0000
0x0003_402C - 0x0003_402F	Reserved	R	Reserved	All 0’s

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS (CONTINUED)

Offset Address	Name	R/W	Description	Default Value
0x0003_4040	PIO32_IN	R/W	GPIO Input State Register 0b - Input Low 1b - Input High Bit Mapping: • Bit 0 - PIO1 • Bit 1-30 - PIO2 • Bit 31 - PIO32	0x0000_0000
0x0003_4044	PIO64_IN	R/W	GPIO Input State Register 0b - Input Low 1b - Input High Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0x0000_0000
0x0003_4048	PIO96_IN	R/W	GPIO Input State Register 0b - Input Low 1b - Input High Bit Mapping: • Bit 0 - PIO65 • Bit 1-27 - PIO66 • Bit 28 - PIO93 • Bit 29-31 - Unused	0x0000_0000
0x0003_403C - 0x0003_403F	Reserved	R	Reserved	All 0’s
0x0003_4040	PIO32_PU	R/W	Enable Internal Pull-up Resistor 0b - Pull-up disabled 1b - Pull-up enabled Bit Mapping: • Bit 0 - PIO1 • Bit 1-30 - PIO2 • Bit 31 - PIO32	0x0000_0000
0x0003_4044	PIO64_PU	R/W	Enable Internal Pull-up Resistor 0b - Pull-up disabled 1b - Pull-up enabled Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0x0000_0000

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS (CONTINUED)

Offset Address	Name	R/W	Description	Default Value
0x0003_4048	PIO96_PU	R/W	Enable Internal Pull-up Resistor 0b - Pull-up disabled 1b - Pull-up enabled Bit Mapping: • Bit 0 - PIO65 • Bit 1-27 - PIO66 • Bit 28 - PIO93 • Bit 29-31 - Unused	0x0000_0000
0x0003_404C - 0x0003_404F	Reserved	R	Reserved	All 0's
0x0003_4050	PIO32_PD	R/W	Enable Internal Pull-down Resistor 0b - Pull-down disabled 1b - Pull-down enabled Bit Mapping: • Bit 0 - PIO1 • Bit 1-30 - PIO2 • Bit 31 - PIO32	0x0000_0000
0x0003_4054	PIO64_PD	R/W	Enable Internal Pull-down Resistor 0b - Pull-down disabled 1b - Pull-down enabled Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0x0000_0000
0x0003_4058	PIO96_PD	R/W	Enable Internal Pull-down Resistor 0b - Pull-down disabled 1b - Pull-down enabled Bit Mapping: • Bit 0 - PIO65 • Bit 1-27 - PIO66 • Bit 28 - PIO93 • Bit 29 - 31 - Unused	0x0000_0000
0x0003_405C - 0x0003_405F	Reserved	R	Reserved	All 0's
0x0003_4060	PIO32_OD	R/W	Sets Pin mode as Open-drain (if configured as output) 0b - Push/Pull 1b - Open Drain Bit Mapping: • Bit 0 - PIO1 • Bit 1-30 - PIO2 • Bit 31 - PIO32	0x0000_0000

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS (CONTINUED)

Offset Address	Name	R/W	Description	Default Value
0x0003_4064	PIO64_OD	R/W	Sets Pin mode as Open-drain (if configured as output) 0b - Push/Pull 1b - Open-drain Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0x0000_0000
0x0003_4068	PIO96_OD	R/W	Sets Pin mode as Open-drain (if configured as output) 0b - Push/Pull 1b - Open-drain Bit Mapping: • Bit 0 - PIO65 • Bit 1-27 - PIO66 • Bit 28 - PIO93 • Bit 29-31 - Unused	0x0000_0000
0x0003_406C - 0x0003_406F	Reserved	R	Reserved	All 0's
0x0003_4070	PIO32_WAKE_ MSK	R/W	0b: Enables an event to generate a GPIO interrupt 1b: Prevents an event from generating a GPIO interrupt. Bit Mapping: • Bit 0 - PIO31 • Bit 1-30 - PIO33 • Bit 31 - PIO63	0xFFFF_FFFF
0x0003_4074	PIO64_WAKE_ MSK	R/W	0b: Enables an event to generate a GPIO interrupt 1b: Prevents an event from generating a GPIO interrupt. Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0xFFFF_FFFF
0x0003_4078	PIO96_WAKE_ MSK	R/W	0b: Enables an event to generate a GPIO interrupt 1b: Prevents an event from generating a GPIO interrupt. Bit Mapping: • Bit 0 - PIO65 • Bit 1-27 - PIO66 • Bit 28 - PIO93 • Bit 29-31 - Unused	0x1FFF_FFFF

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS (CONTINUED)

Offset Address	Name	R/W	Description	Default Value
0x0003_407C - 0x0003_407F	Reserved	R	Reserved	All 0's
0x0003_4080	PIO32_MODE	R/W	0b - GPIO event Edge-trigger mode 1b - GPIO event Level-trigger mode Bit Mapping: • Bit 0 - PIO31 • Bit 1-30 - PIO33 • Bit 31 - PIO63	0x0000_0000
0x0003_4084	PIO64_MODE	R/W	0b - GPIO event Edge-trigger mode 1b - GPIO event Level-trigger mode Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0x0000_0000
0x0003_4088	PIO96_MODE	R/W	0b - GPIO event Edge-trigger mode 1b - GPIO event Level-trigger mode Bit Mapping: • Bit 0 - PIO65 • Bit 1-27 - PIO66 • Bit 28 - PIO93 • Bit 29-31 - Unused	0x0000_0000
0x0003_408C - 0x0003_408F	Reserved	R	Reserved	All 0's
0x0003_4090	PIO32_LO_TO_HI_EDGE_CONFGIG	R/W	0b - Enables an event to be generated on a low to high transition of the corresponding PIO line. 1b - Prevents an event from being generated on a low to high transition of the corresponding PIO line. Bit Mapping: • Bit 0 - PIO31 • Bit 1-30 - PIO33 • Bit 31 - PIO63 **Note:** For edge-triggered, if both high and low edge mask bits are set, the PIO edge events will not occur. The event only occurs in the direction that is not masked.	0xFFFF_FFFF

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS (CONTINUED)

Offset Address	Name	R/W	Description	Default Value
0x0003_4094	PIO64_LO_TO_HI_EDGE_CONFGIG	R/W	0b - Enables an event to be generated on a low to high transition of the corresponding PIO line. 1b - Prevents an event from being generated on a low to high transition of the corresponding PIO line. Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64 **Note:** For edge-triggered, if both high and low edge mask bits are set, the PIO edge events will not occur. The event only occurs in the direction that is not masked.	0xFFFF_FFFF
0x0003_4098	PIO96_LO_TO_HI_EDGE_CONFGIG	R/W	0b - Enables an event to be generated on a low to high transition of the corresponding PIO line. 1b - Prevents an event from being generated on a low to high transition of the corresponding PIO line. Bit Mapping: • Bit 0 - PIO65 • Bit 1-27 - PIO66 • Bit 28 - PIO93 • Bit 29-31 - Unused **Note:** For edge-triggered, if both high and low edge mask bits are set, the PIO edge events will not occur. The event only occurs in the direction that is not masked.	0x1FFF_FFFF
0x0003_409C - 0x0003_409F	Reserved	R	Reserved	All 0’s
0x0003_40A0	PIO32_HI_TO_LOW_EDGE_CONFIG	R/W	0b - Enables an event to be generated on a high to low transition of the corresponding PIO line. 1b - Prevents an event from being generated on a high to low transition of the corresponding PIO line. Bit Mapping: • Bit 0 - PIO1 • Bit 1-30 - PIO33 • Bit 31 - PIO32	0xFFFF_FFFF

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS (CONTINUED)

Offset Address	Name	R/W	Description	Default Value
0x0003_40A4	PIO64_HI_TO_LOW_EDGE_CONFIG	R/W	0b - Enables an event to be generated on a high to low transition of the corresponding PIO line. 1b - Prevents an event from being generated on a high to low transition of the corresponding PIO line. Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0xFFFF_FFFF
0x0003_40A8	PIO96_HI_TO_LOW_EDGE_CONFIG	R/W	0b - Enables an event to be generated on a high to low transition of the corresponding PIO line. 1b - Prevents an event from being generated on a high to low transition of the corresponding PIO line. Bit Mapping: • Bit 0 - PIO65 • Bit 1-27 - PIO66 • Bit 28 - PIO93 • Bit 29-31 - Unused	0x1FFF_FFFF
0x0003_40AC - 0x0003_40AF	Reserved	R	Reserved	All 0’s
0x0003_40B0	PIO32_LEVEL_CONFIG	R/W	0b - Polarity of event is unchanged in Level mode; an event is detected when the pin is high. 1b - Polarity of event is inverted in Level mode; an event is detected when the pin is low. Bit Mapping: • Bit 0 - PIO1 • Bit 1-30 - PIO33 • Bit 31 - PIO32	0x0000_0000
0x0003_40B4	PIO64_LEVEL_CONFIG	R/W	0b - Polarity of event is unchanged in Level mode; an event is detected when the pin is high. 1b - Polarity of event is inverted in Level mode; an event is detected when the pin is low. Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0x0000_0000

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS (CONTINUED)

Offset Address	Name	R/W	Description	Default Value
0x0003_40B8	PIO96_LEVEL_CONFIG	R/W	0b - Polarity of event is unchanged in Level mode; an event is detected when the pin is high. 1b - Polarity of event is inverted in Level mode; an event is detected when the pin is low. Bit Mapping: • Bit 0 - PIO65 • Bit 1-27 - PIO66 • Bit 28 - PIO93 • Bit 29-31 - Unused	0x0000_0000
0x0003_40BC - 0x0003_40BF	Reserved	R	Reserved	All 0's
0x0003_40C0	PIO32_LEVEL_MSK	R/W	0b - Level mode events are not masked. 1b - Level mode events are masked. Bit Mapping: • Bit 0 - PIO1 • Bit 1-30 - PIO33 • Bit 31 - PIO32	0xFFFF_FFFF
0x0003_40C4	PIO64_LEVEL_MSK	R/W	0b - Level mode events are not masked. 1b - Level mode events are masked. Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0xFFFF_FFFF
0x0003_40C8	PIO96_LEVEL_MSK	R/W	0b - Level mode events are not masked. 1b - Level mode events are masked. Bit Mapping: • Bit 0 - PIO65 • Bit 1-27 - PIO66 • Bit 28 - PIO93 • Bit 29-31 - Unused	0x1FFF_FFFF
0x0003_40CC - 0x0003_40CF	Reserved	R	Reserved	All 0's

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS (CONTINUED)

Offset Address	Name	R/W	Description	Default Value
0x0003_40D0	PIO32_STATUS	R/W 1C	0b - No event change has occurred. 1b - An event change has occurred. When in Edge mode, this bit will be set when a transition (low-to-high or high-to-low) of the input pin occurs. When in Level mode, this bit will be set when the pin state is high if PIOxx_LEVEL_ CONFIG = 0b or low if PIOxx_LEVEL_ CONFIG = 1b. Events are cleared by writing 1b to the bit. However, in Level mode, if the pin state is still in the asserted state, this status bit will remain set. Bit Mapping: • Bit 0 - PIO1 • Bit 1-30 - PIO33 • Bit 31 - PIO32	0x0000_0000
0x0003_40D4	PIO64_STATUS	R/W 1C	0b - No event change has occurred. 1b - An event change has occurred. When in Edge Mode, this bit will be set when a transition (low-to-high or high-to-low) of the input pin occurs. When in Level Mode, this bit will be set when the pin state is high if PIOxx_LEVEL_CON-FIG = 0b or low if PIOxx_LEVEL_CONFIG = 1b. Events are cleared by writing 1b to the bit. However, in level mode, if the pin state is still in the asserted state, this status bit will remain set. Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0x0000_0000

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS (CONTINUED)

Offset Address	Name	R/W	Description	Default Value
0x0003_40D8	PIO96_STATUS	R/W 1C	0b - No event change has occurred. 1b - An event change has occurred. When in Edge mode, this bit will be set when a transition (low-to-high or high-to-low) of the input pin occurs. When in Level Mode, this bit will be set when the pin state is high if PIOxx_LEVEL_CONFIG = 0b or low if PIOxx_LEVEL_CONFIG = 1b. Events are cleared by writing 1b to the bit. However, in Level mode, if the pin state is still in the asserted state, this status bit will remain set. Bit Mapping: • Bit 0 - PIO65 • Bit 1-26 - PIO66 • Bit 27 - PIO92 • Bit 28-31 - Unused	0x0000_0000
0x0003_40DC - 0x0003_40DF	Reserved	R	Reserved	All 0's
0x0003_40E0	PIO32_DEBOUNCE	R/W	0b - No debounce applied and the raw input is passed through. 1b - If a bit is set, and the corresponding PIO is configured as an input, then the PIO input is debounced by the amount specified in PIO_GLOBAL_CONFIG.PIO_DEBOUNCE[11:0]. Bit Mapping: • Bit 0 - PIO1 • Bit 1-30 - PIO33 • Bit 31 - PIO32	0x0000_0000
0x0003_40E4	PIO64_DEBOUNCE	R/W	0b - No debounce applied and the raw input is passed through. 1b - If a bit is set, and the corresponding PIO is configured as an input, then the PIO input is debounced by the amount specified in PIO_GLOBAL_CONFIG.PIO_DEBOUNCE[11:0]. Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0x0000_0000

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS (CONTINUED)

Offset Address	Name	R/W	Description	Default Value
0x0003_40E8	PIO96_ DEBOUNCE	R/W	0b - No debounce applied and the raw input is passed through. 1b - If a bit is set, and the corresponding PIO is configured as an input, then the PIO input is debounced by the amount specified in PIO_ GLOBAL_CONFIG.PIO_DEBOUNCE[11:0]. Bit Mapping: • Bit 0 - PIO65 • Bit 1-26 - PIO66 • Bit 27 - PIO92 • Bit 28-31 - Unused	0x0000_0000
0x0003_40EC - 0x0003_40EF	Reserved	R	Reserved	All 0's
0x0003_40F0	PIO_GLOBAL_ CONFIG	R/W	Bits [31:18] - Reserved Bit [17] - PIO_WAKE_MASTER_MASK 0b - PIO Wake events are not masked. 1b - All PIO Wake events are masked. This is a final gate over the single wake event (an OR-function of 93 possible individual PIO wake events) that is fed to the PCIe Endpoint. Bit [16] - PIO_INT_MASTER_MASK 0b - PIO Interrupt events are not masked. 1b - All PIO Interrupt events are masked. This is a final gate over the single wake event (an OR-function of 93 possible individual PIO wake events) that is fed to the PCIe Endpoint. Bits [15:12] - Reserved Bits [11:0] PIO_DEBONCE[11:0] This field holds the debounce timer for the GPIOs. The Debouncer starts when a transition is detected and is passed through if the pin state is maintained for the configured Debouncer time. If another transition occurs, during the Debouncer time, the Debouncer is restarted. Each count corresponds to 1 ms, with the default value being 10 ms. The timing is derived from a free-running clock and will vary with edge alignment. The accuracy of the timing is limited to ± one count.	0x0003_000A

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS (CONTINUED)

Offset Address	Name	R/W	Description	Default Value
0x0003_40F4	PIO_PCI_ CTRL_REG	R/W	Bits [31:1] - Reserved Bit [0] - PIO_D3_CLK_EN Enables the 62.5 MHz clock when the PCI function is in D3: 0b - 62.5 MHz clock is disabled when the PCI function is in D3. 1b - 62.5 MHz clock is enabled when the PCI function is in D3.	0x0000_0000

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS (CONTINUED)

Offset Address	Name	R/W	Description	Default Value
0x0003_40F8	PIO_LTR_ VALUE_REG	R/W	Bit [31] - PIO_NO_SNOOP_REQ 0b - No latency requirement and the PIO_NO_SNOOP_LATENCY_VAL[9:0] and PIO_NO_SNOOP_LATENCY_SCALE[2:0] fields are ignored. 1b - PIO_NO_SNOOP_LATENCY_VAL[9:0] and PIO_NO_SNOOP_LATENCY_SCALE[2:0] fields describe the latency requirement. Bits [30:29] - Reserved Bits [28:26] - PIO_NO_SNOOP_LATENCY_ SCALE[2:0] Scale of the PIO_NO_SNOOP_LATEN-CY_VAL[9:0] field: 000b - value times 1 ns 001b - value times 32 ns 010b - value times 1024 ns 011b - value times 32768 ns 100b - value times 1048576 ns 101b - value times 33554432 ns 110b through 111b - value times not permitted Bits [25:16] - PIO_NO_SNOOP_LATENCY_ VAL[9:0] Latency Value Field Bit [15] - PIO_SNOOP_REQ 0b - no latency requirement and the PIO_SNOOP_LATENCY_VAL[9:0] and PIO_SNOOP_LATENCY_SCALE[2:0] fields are ignored. 1b - PIO_SNOOP_LATENCY_VAL[9:0] and PIO_SNOOP_LATENCY_SCALE[2:0] fields describe the latency requirement. Bits [14:13] - Reserved Bits [12:10] - PIO_SNOOP_LATENCY_ SCALE[2:0] Scale of the PIO_SNOOP_LATENCY_ VAL[9:0] field: 000b - value times 1 ns 001b - value times 32 ns 010b - value times 1024 ns 011b - value times 32768 ns 100b - value times 1048576 ns 101b - value times 33554432 ns 110b through 111b - value times not permitted Bits [9:0] - PIO_SNOOP_LATENCY_VAL[9:0] Snoop Latency Value	0x0000_0000

TABLE 151: DEVICE PROPERTIES AND PROGRAMMING OPTIONS (CONTINUED)

Offset Address	Name	R/W	Description	Default Value
0x0003_40FC - 0x0003_40FF	Reserved	R	Reserved	All 0's
0x0003_4100	PIO32_INT_MSK	R/W	0b - Interrupt not masked. 1b - Interrupt is masked. Bit Mapping: • Bit 0 - PIO1 • Bit 1-30 - PIO33 • Bit 31 - PIO32	0xFFFF_FFFF
0x0003_4104	PIO64_INT_MSK	R/W	0b - Interrupt not masked. 1b - Interrupt is masked. Bit Mapping: • Bit 0 - PIO33 • Bit 1-30 - PIO34 • Bit 31 - PIO64	0xFFFF_FFFF
0x0003_4108	PIO96_INT_MSK	R/W	0b - Interrupt not masked. 1b - Interrupt is masked. Bit Mapping: • Bit 0 - PIO65 • Bit 1-27 - PIO66 • Bit 28 - PIO93 • Bit 29-31 - Unused	0x1FFF_FFFF
0x0003_410C - 0x0003_43FF	Reserved	R	Reserved	All 0's

13.0 CONFIGURATION FILE FORMATTING

13.1 Sections

- Section 13.2, "Configuration Memory Structure (for OTP and EEPROM)"
- Section 13.3, "Configuration Command Formatting"

13.2 Configuration Memory Structure (for OTP and EEPROM)

Configuration Data is divided into two main regions:

- The MEM_STAT region which has fixed addresses for specific parameters and is used to indicate to the PCI1xxxx which memory regions contain configuration data blocks, and which hardware subsystems actually have configuration data which modifies the hardware settings from the default settings.
- The Configuration Data region contains the actual configuration data blocks. Each configuration data block can have variable length depending on how many configuration settings are required for the specific application.

To configure OTP, do the following steps:

1. Configure MEM_STAT memory:
   - Write the MAGIC_BYTE to indicate that memory is configured.
   - Write the next bit in OTP_PROG_COUNT to indicate how many times OTP has been programmed.

Note:	If all of the bits in OTP_PROG_COUNT have been set, OTP memory can still be configured. The only impact is that additional OTP programming events cannot be tracked using this counter any longer.

   - Write the MEMORY_SIZE memory to indicate the memory size.
   - Write the bits in MEM_REGION_PROG memory to indicate which memory regions will be programmed; one region is required per subsytem to be programmed.
   - If any memory regions are to be invalidated then set the corresponding bit in OTP_REGION_INV. This is typically done to replace the configuration block for one subsystem with a new one. In this case the memory region, which has been previously programmed, will be set to invalid and a new region will be set in MEM_REGION_PROG.
   - For each memory region configure the memory region descriptor MEM_REGION_DESCRx. Write the MEM_TAG[7:0] to indicate the subsystem that the memory region corresponds to and the MEM_START_ADDRESS[12:0] with the starting address of the configuration block in OTP.

Note:	MEM_START_ADDRESS[12:0] should be selected such that there is sufficient memory for each of the configuration blocks without the memory regions overlapping.

2. Configure each of the memory regions pointed to by the MEM_REGION_DESCRx following the format defined in Section 13.3, "Configuration Command Formatting".
3. Any memory regions which need to be write-locked are configured by setting the bit corresponding to the memory region in OTP_REGION_WR_PROT.

Note:	After a region has been marked write-protected it cannot ever be changed or extended. Write-protect memory only after it is confirmed that additional programming or extension may not be required in the future.

Any memory regions which need to be read-locked are configured by setting the bit corresponding to the memory region in OTP_REGION_RD_PROT, see Table 152.

TABLE 152: CONFIGURATION MEMORY STRUCTURE

OFFSET	SIZE	NAME	DESCRIPTION
0x00h	1	MAGIC_BYTE	Indicates that the OTP has been configured when programmed to A5h. All other values indicate the OTP is not programmed or not programmed properly and should be ignored.
0x01	1	Reserved	Reserved

TABLE 152: CONFIGURATION MEMORY STRUCTURE (CONTINUED)

OFFSET	SIZE	NAME	DESCRIPTION
0x02 - 0x03	2	OTP_PROG_ COUNT	OTP: Number of times OTP Memory has been programmed. Each individual bit indicates one instance of OTP programming, so the counter can only track up to 16 OTP programming events. Setting of bits should begin at the LSB after programming OTP the first time, and increment until the 16th bit is set after programming 16 times. If bits are skipped, then the meaning of the counter becomes uncertain/corrupted. Examples: • 0000_0000_0000_0001b: OTP has been programmed once • 0000_0000_0000_1111b: OTP has been programmed 4 times • 1111_1111_1111_1111b: OTP has been programmed 16 times.
0x04 - 0x07	4	MEMORY_SIZE	MEMORY_SIZE gives the maximum size of the OTP in bytes; this is used by HW to detect over-runs. Memory beyond MEMORY_SIZE should not be configured or accessed.
0x08 - 0x0B	4	MEM_ REGION_PROG	A 32-bit area of memory indicating which of the 32 memory regions have been programmed; each bit corresponds to one of the 32 memory regions 0 to 31. For example, if the value is 0x0000_9221, this indicates memory regions 0, 5, 9, 12 and 15 have been programmed and should be loaded by the hardware.
0x0C - 0x0F	4	OTP_REGION_INV	An OTP specific field enabling configuration blocks to be invalidated after they have been programmed. It is a 32-bit area of memory indicating which of the 32 memory regions have been invalidated; each bit corresponds to one of the 32 memory regions 0 to 31. For example, to invalidate region 12, set OTP_REGION_INV = 0x0000_8000.
0x10 - 0x13	4	OTP_RE- GION_WR_PROT	An OTP specific field allowing certain areas of OTP memory to be write locked. It is a 32-bit area of memory indicating which of the 32 memory regions have been invalidated; each bit corresponds to one of the 32 memory regions 0 to 31. For example, to write lock region 0, set OTP_REGION _WR_PROT = 0x0000_0001.
0x14 - 0x17	4	OTP_REGION_RD_PROT	An OTP specific field allowing certain areas of OTP memory to be read locked. It is a 32-bit area of memory indicating which of the 32 memory regions have been read locked; each bit corresponds to one of the 32 memory regions 0 to 31. For example, to read lock region 0, set OTP_REGION_RD_ PROT = 0x0000_0001.
0x18 - 0x2F	17	Reserved	Reserved

TABLE 152: CONFIGURATION MEMORY STRUCTURE (CONTINUED)

OFFSET	SIZE	NAME	DESCRIPTION
0x30 - 0x33	4	MEM_REGION_DESCR0	Memory Region Descriptor. This is used by the hardware to know the memory location of the configuration data block, and to which hardware subsystem the configuration data block is attributed to. **Bits [31:29]:** Reserved **Bits [28:16]:** MEM_START_ADDRESS: Start address of the memory region. **Bits [15:8]:** Reserved **Bits [7:0]:** MEM_TAG - Indicates which subsystem the memory region relates to, and a pointer to the configuration block MEM_TAG can be one of the following values: • 0x00: Reserved • 0x01: USB Subsystem tag • 0x02: Ethernet Subsystem tag • 0x03: UART Physical Function Subsystem Tag • 0x04: SMBus Physical Function Subsystem Tag • 0x05: SPI Physical Function Subsystem Tag • 0x06: General Physical Function Subsystem Tag • 0x07: PCIe Switch Subsystem Main Tag • 0x08: PCIe Switch Subsystem Port 0 Tag • 0x09: PCIe Switch Subsystem Port 1 Tag • 0x0A: PCIe Switch Subsystem Port 2 Tag • 0x0B: PCIe Switch Subsystem Port 3 Tag • 0x0C: PCIe Switch Subsystem Port 4 Tag • 0x0D: System Register Address Base • 0x0E: PCIe PHY A Tag • 0x0F: PCIe PHY B Tag • 0x10: PCIe PHY C Tag • 0x11 - 0xFF: Reserved
0x34 - 0x37	4	MEM_REGION_DESCR1	
0x38 - 0x3B	4	MEM_REGION_DESCR2	
0x3C - 0x3F	4	MEM_REGION_DESCR3	
0x40 - 0x43	4	MEM_REGION_DESCR4	
0x44 - 0x47	4	MEM_REGION_DESCR5	
0x48 - 0x4B	4	MEM_REGION_DESCR6	
0x4C - 0x4F	4	MEM_REGION_DESCR7	
0x50 - 0x53	4	MEM_REGION_DESCR8	
0x54 - 0x57	4	MEM_REGION_DESCR9	
0x58 - 0x5B	4	MEM_REGION_DESCR10	
0x5C - 0x5F	4	MEM_REGION_DESCR11	
0x60 - 0x63	4	MEM_REGION_DESCR12	
0x64 - 0x67	4	MEM_REGION_DESCR13	
0x68 - 0x6B	4	MEM_REGION_DESCR14	
0x6C - 0x6F	4	MEM_REGION_DESCR15	
0x70 - 0x73	4	MEM_REGION_DESCR16	
0x74 - 0x77	4	MEM_REGION_DESCR17	
0x78 - 0x7B	4	MEM_REGION_DESCR18	
0x7C - 0x7F	4	MEM_REGION_DESCR19	
0x80 - 0x73	4	MEM_REGION_DESCR20	
0x84 - 0x77	4	MEM_REGION_DESCR21	
0x88 - 0x7B	4	MEM_REGION_DESCR22	
0x8C - 0x7F	4	MEM_REGION_DESCR23	
0x90 - 0x93	4	MEM_REGION_DESCR24	
0x94 - 0x97	4	MEM_REGION_DESCR25	
0x98 - 0x9B	4	MEM_REGION_DESCR26	
0x9C - 0x9F	4	MEM_REGION_DESCR27	
0xA0 - 0xA3	4	MEM_REGION_DESCR28	
0xA4 - 0xA7	4	MEM_REGION_DESCR29	
0xA8 - 0xAB	4	MEM_REGION_DESCR30	
0xAC - 0xAF	4	MEM_REGRION_DESC31	
0xB0 - 0xFF	79	Reserved	Reserved
0100h - 1F40h	4,096	Configuration Data Blocks	All Configuration Data Blocks are programmed into this memory region. The precise location of each configuration data block is not critical or enforced, as the memory start region is programmed into the MEM_REGION_DESCx memory. Generally, each configuration region is programmed next each other with no blank memory left in between, however, it is also possible to leave buffer space in between each configuration memory region to allow for future additions and upgradability.

13.3 Configuration Command Formatting

The configuration data contains only register modifications which need to be changed from the default value. Registers which do not need to be changed should not be included in the configuration data block. Configuration data is implemented through a series of commands with user-defined lengths. Each block of configuration data is intended to configure only a signal hardware block. Each configuration data block is programmed into one of the 32 available memory region, see Table 153.

The three primary OpCodes:

- STOP (0x00): Signals the end of the configuration data block.
- SET_MEMORY_ADRESS (0x80): Sets the memory address pointer to the selected address, and then writes data of a variable length to memory and all contiguous memory addresses incrementing from the selected start address.
- SKIP_MEMORY (0x81-0xFF): Skips the memory address pointer by up to 127 memory locations, and then writes data of a variable length to memory and all contiguous memory addresses incrementing from the selected start address.

TABLE 153: OTP STORAGE COMMANDS

Command	OPCODE	Parameters	Description
STOP	0x00	N/A	The last opcode in the configuration block shall be STOP. To extend a configuration block at any time STOP can be overwritten with an additional command provided that this does not cause the configuration block to overlap any others and that the additional command itself is followed by STOP to terminate the configuration block. **Note:** The default value for each byte of OTP memory is 0x00. Hence, STOP is defined with this value so that configuration blocks have a natural termination value (OTP default) and can therefore be easily extended if required.
SET_MEMORY_ ADDRESS	0x80	MEMORY_ ADDRESS	Set the current address location to the address specified.
		LENGTH	Number of consecutive DWORDS to be written to memory starting at memory space selected in MEMORY_ADDRESS parameter. May be 0x01-0x7F (1 to 127 DWORDs). 0x00 is an illegal value.
		DATA	DWORDs to be written. The length of data must be equal to the LENGTH parameter. Data is ordered in little endian format.

TABLE 153: OTP STORAGE COMMANDS (CONTINUED)

Command	OPCODE	Parameters	Description
SKIP_MEMORY	0x81 - 0xFF	SKIP_DWORDS	Skip the memory address pointer by the value of the command used minus 0x80. Allows OTP commands to save non-volatile memory space. May be specified as 0x81 - 0xFF (Skip 1 to 127 addresses) Example: 0x84 will skip the address pointer by 4 DWORDs. If the starting address was 0x0FF0_0000 and the opcode of the SKIP_DWORDS command is 0x84, then the memory address to be configured will be 0x0FF0_0010h
		LENGTH	Number of consecutive DWORDS to be written to memory starting at memory space selected in MEMORY_ADDRESS parameter. May be 0x01-0x7F (1 to 127 DWORDs). 0x00 is an illegal value.
		DATA	DWORDs to be written. The length of data must be equal to the LENGTH parameter. Data is ordered in little endian format.

13.4 Configuration Command Examples

1. Writing Value to a Single Memory Address.

To write 0xAAAB_ACAD into the register 0xBF80_3000, see Table 154.

TABLE 154: EXAMPLE 1-OTP CONFIGURATION COMMAND

Byte Offset	Byte	Description
0	0x80	OpCode for SET_MEMORY_ADDRESS
1	0x00	MEMORY_ADRESS Parameter
2	0x30	
3	0x80	
4	0xBF	
5	0x01	LENGTH
6	0xAD	Data to be written to the MEMORY_ADDRESS
7	0xAC	
8	0xAB	
9	0xAA	
10	0x00	STOP

2. Writing Value to a Two Memory Addresses.

To write 0xAAAB_ACAD to the register located at 0xBF80_3000 and also write 0x5556_5758 to the register located at 0xBF80_300C, see Table 155.

TABLE 155: EXAMPLE 2-OTP CONFIGURATION COMMAND

Byte Offset	Byte	Description
0	0x80	OpCode for SET_MEMORY_ADDRESS
1	0x00	MEMORY_ADRESS Parameter
2	0x30	
3	0x80	
4	0xBF	
5	0x01	LENGTH
6	0xAD	Data to be written to the MEMORY_ADDRESS
7	0xAC	
8	0xAB	
9	0xAA	
10	0x82	Opcode for SKIP_MEMORY; skip 2 Memory addresses (0x82-0x80 DWORDs).
11	0x01	LENGTH
12	0x58	Data to be written to the MEMORY_ADDRESS + 2
13	0x57	
14	0x56	
15	0x55	
16	0x00	STOP

14.0 OTP PROGRAMMING (READ AND WRITE)

14.1 Sections

14.2 Loading OTP Memory at Boot

PCI1XXXX takes the following steps when loading from OTP at boot:

1. Check that the first byte is the MAGIC_BYTE. If the MAGIC_BYTE is present, then continue. Otherwise, OTP is not configured and loading from OTP is skipped (whether or not hardware defaults will be used depends whether there are other configuration options present or not).
2. Latch in MEMORY_SIZE and use this as a validation check to ensure that PCI1xxxx does not read past the end of configured memory due to a poor configuration. When PCI1xxxx reaches the end of memory it shall stop reading and assume that configuration has been completed.
3. Latch in MEM_REGION_PROG to determine which memory areas have been programmed. Each bit in MEM_REGION_PROG when set indicates that the corresponding memory descriptor has been programmed.
4. Latch in OTP_REGION_INV to determine which memory areas are valid. Each bit in OTP_REGION_INV when set indicates that the corresponding memory descriptor has been invalidated.
5. PCI1xxxx loops through each memory descriptor, MEM_REGION_DESRx, which has been programmed according to MEM_REGION_PROG and has not been invalidated according to OTP_REGION_INV. Other memory descriptors are ignored.
6. PCI1xxxx reads the MEM_TAG[7:0] memory which indicates the subsytem or function to be configured. PCI1xxxx verifies that this subsystem is valid for the selected device part number and that the subsystem or function is actually requesting configuration by checking GENERAL_SYS_CONFIG_REQ_REG [7:0], before proceeding with configuring the subsystem or function. If the subsytem is not to be configured, then PCI1xxxx goes to step 10.
7. For the subsystem to be configured, PCI1xxxx reads the MEM_START_ADDRESS[12:0] memory in the MEM_REGION_DESRx which points to the configured area of memory. HW sets CFG_BYTE = MEM_START_ADDRESS[12:0] and MEM_DWORD = 0x0000_0000.
8. HW reads the byte stored at CFG_BYTE:

   8.1 If the byte is 0x00, this is a STOP opcode. There is no more data to read from the configuration block. HW goes to step 10.

   8.2 If the byte is 0x80, this is a SET_MEMORY_ADDRESS opcode. HW increments CFG_BYTE by 1 and reads the MEMORY_ADDRESS which it stores in MEM_DWORD. HW increments CFG_BYTE by 4.

   8.3 If the byte is 0x81...0xFF, this is a SKIP_DWORDS opcode. If MEMORY_ADDRESS = 0x0000_0000, there has been no SET_MEMORY_ADDRESS opcode prior to the SKIP_DWORDS opcode, HW sets the relevant error bit in GENERAL_SYS_OTP_CONFIG_ERR_REG for OTP and the corresponding MEM_TAG[7:0] and goes to step 10. HW increments the value stored in MEM_DWORD by 4*(opcode-0x80). HW increments CFG_BYTE by 1.

   8.4 HW reads the LENGTH.

   8.5 If LENGTH = 0, this is an error. HW sets the relevant error bit in GENERAL_SYS_OTP_CONFIG_ERR_REG for OTP and the corresponding MEM_TAG[7:0] (bits 0...7) and goes to step 10.

   8.6 HW increments CFG_BYTE by 1 and reads the DATA which is of LENGTH DWORDS.

   8.7 HW increments CFG_BYTE by 4*LENGTH.

   8.8 HW writes LENGTH DWORDS of DATA to MEMORY_ADDRESS. If LENGTH = 1, HW will use bits 0...1 of MEMORY_ADDRESS as a byte selector (see Table 160).

   8.9 HW increments the value stored in MEM_DWORD by 4*LENGTH.

8.10 HW goes back to step 8.

9. If at any point the CFG_BYTE being read is past the end of the memory indicated by MEMORY_SIZE, the HW sets the GENERAL_SYS_OTP_CONFIG_ERR_REG.OTP_MEM_OVERSHOOT_ERR bit and goes immediately to step 10.
10. If there are more subsystems to configure, HW goes back to step 5.
11. If there are no straps present indicating configuration in EEPROM or via SPI/SMBUS, the HW will bring up the subsystems at this point.

14.3 OTP Read/Write Protection

An area at the start of OTP memory is reserved by hardware to store values used to control access to JTAG and to protect regions of OTP memory in order to make them permanently non-modifiable.

OTP_REGION_WR_PROT and OTP_REGION_RD_PROT each have 1-bit corresponding to each of the 32 memory regions described by the MEM_REGION_DESCRx descriptors. When the bit corresponding to a memory region in OTP_REGION_WR_PROT is set, this means that the memory region is write protected by OTP hardware block and can no longer be written to.

When the bit corresponding to a memory region in OTP_REGION_RD_PROT is set, this means that the memory region is read protected by OTP hardware block and can no longer be read externally from the chip. If read protected, the read-protected OTP values can only be read by PCI1xxxx itself during configuration of the system when loading from OTP.

14.4 Manual OTP Programming

All OTP addresses have an unprogrammed default state of zero. Programming is done one bit at a time. The OTP_PROGRAM command programs the content of the OTP Data register to the memory location specified by the OTP address.

1. Take the OTP out of power down by clearing OTP_PWRDN_N in the OTP_PWR_DN register.
2. Program bit or byte mode programming accordingly by setting OTP_PGM_MODE_BYTE in the OTP_PRGM_MODE register.
3. Program the various pulse width CSRs.
4. Write to the OTP_ADDR_HIGH/OTP_ADDR_LOW/OTP_ADDR_BITS registers. If Byte mode is utilized, the OTP_ADDR_BITS register is ignored.
5. Write data pattern to be programmed into the OTP_PRGM_DATA register. Although the OTP only supports bit writes the HW adjusts for this when byte programming is selected. HW only attempts to program values of one as OTP bit cells default to zero. If bit programming is enabled then only bit 0 of this register is valid.
6. Program OTP_MAX_PROG[4:0] in the OTP_MAX_PRG register to set the number of programming cycles.
7. Set the OTP_PROGRAM Command bit in the OTP_FUNC_CMD register. This bit will be self-cleared by hardware after the operation completes.
8. Set the OTP_GO Command bit in the OTP_CMD_GO register to initiate the OTP_PROGRAM command. This bit will be self-cleared by hardware after the operation completes.
9. The OTP_BUSY bit in the OTP_STATUS register asserts while the operation is pending. HW will automatically drive the inputs to the OTP.
10. After the programming cycle completes, the OTP_RD_DATA register is updated to include the value actually programmed into the OTP. HW obtains this value by sampling the D port of the OTP after the deassertion of CPUMPEN. HW shall account for bit and byte write when updating this register.
11. Poll the OTP_STATUS register until OTP_BUSY bit is cleared.
12. Operation completes and the OTP_BUSY bit deasserts.
13. Read the OTP_RD_DATA_FLD[7:0] from the OTP_RD_DATA register and determine if the OTP programmed. successfully. It should match the contents of the OTP_PRGM_DATA register. If bit programming is enabled, only bit 0 of the register is valid. Otherwise, each bit in the OTP_RD_DATA register corresponds to the respective bit in the OTP_PRGM_DATA register.
14. If a location did not program successfully, it must be reprogrammed. The IP provider recommends that after eleven failures, it is determined that the location has a failure and that there is a manufacturing fault in the OTP.

Note: The default value of an OTP byte is 0x00. There is no need to program a target byte of 0x00.

14.5 SMART OTP Programming

SMART mode programming is used to reduce the programming time and should be used to program all the bits that need to be logic 1.

1. Take the OTP out of power down by clearing the OTP_PWRDN_N bit in the OTP_PWR_DN register.
2. Program bit mode programming by setting '0' into the OTP_PGM_MODE_BYTE.
3. Program the various pulse width CSRs.
4. Write to the OTP_ADDR_HIGH/OTP_ADDR_LOW/OTP_ADDR_BITS and OTP_PRGM_DATA registers.
5. Write 1 to the OTP_FW_SMART_PGM_EN bit in the OTP_FUNC_CMD register.
6. Write 1 to the OTP_FW_SMART_PGM_INIT_ACCESS bit if the next PROGRAM/TESTDEC command is the first in SMART programming sequence to the programmed address. Otherwise, this bit shall set to '0'.
7. Set the OTP_PROGRAM command bit in OTP function command register. This bit is self-cleared by HW after the operation completes.
8. Set the OTP_GO Command bit in OTP_CMD_GO register to initiate the PROGRAM command. This bit is self-cleared after the operation completes.
9. Check the OTP_BUSY bit deassertion in the OTP_STATUS register.
10. Read the OTP_RD_DATA_FLD[7:0] from the OTP_RD_DATA register and determine if the OTP programmed successfully. It should match the contents of the OTP_PRGM_DATA register. If bit programming is enabled, only bit '0' of the register is valid. Otherwise, each bit in the OTP_RD_DATA register corresponds to the respective bit in the OTP_PRGM_DATA register
11. If the OTP_RD_DATA matches OTP_PRGM_DATA, continue programming operation with new address. Otherwise, continue programming with same address as said above from step 5 and within the maximum number of programming as mentioned in SMART programming flow.

14.6 External Read-Back of OTP Memory

The READ command reads data from the OTP internal DATA register. The data will then be available in the OTP_RD_DATA register. Below is the sequence of a typical READ. The hardware will automatically drive the READEN signal during the access.

1. Take the OTP out of power down by clearing the OTP_PWRDN_N bit in the OTP_PWR_DN register.
2. Program the OTP_RSTB_PW_HIGH/OTP_RSTB_PW_LOW and OTP_READ_PW_HIGH/OTP_READ_PW_LOW registers.
3. Write to the OTP_ADDR_HIGH/OTP_ADDR_LOW/OTP_ADDR_BITS register.
4. Set the OTP_READ Command bit in the OTP_FUNC_CMD register. This bit will be self-cleared by hardware after command is accepted by the OTP controller.
5. Set the OTP_GO Command bit in the OTP_CMD_GO register to initiate the READ command. This bit will be self-cleared by hardware after command is accepted by the OTP controller.
6. The OTP_BUSY bit asserts while the operation is pending.
7. Poll the OTP_STATUS register until the OTP_BUSY bit is cleared.
8. Operation completes and the OTP_BUSY bit deasserts.
9. Read the OTP_RD_DATA register.

Read operations are always byte wide. The OTP is addressed on a byte boundary for reads. The OTP_ADDR_BITS register is ignored by the OTP. The OTP shall be placed in power-down after all planned operations have completed.

14.7 OTP Programming via Linux Memory-Mapped Device

On Linux, the PCI1xxx enumerates OTP and EEPROM as two separate 8 kB disks through the GPIO subsystem driver.

This allows the OTP and EEPROM to be programmed using the Linux dd command.

The driver also provides a custom IOCTL interface which enables programming of a variable length byte stream from any offset. The driver always enumerates the OTP always whereas the EEPROM is enumerated only if it is present on the board.

OTP is enumerated as:

```
/dev/PCI1xxxxOTPn
```

EEPROM is enumerated as:

```
/dev/PCI1xxxxE2Pn
```

To program a configuration file (called ‘input.bin’ in this example) to OTP via dd commands, refer to the following example:

```
sudo dd if=input.bin of=/dev/PCI1xxxxOTPn bs=512 count=16 oflag=direct
```

To read-back the OTP via dd commands, refer to the following example:

```
sudo dd if=/dev/PCI1xxxxOTPn of=output.bin bs=512 count=16 iflag=direct
```

15.0 EEPROM MEMORY PROGRAMMING (READ AND WRITE)

15.1 Sections

15.2 EEPROM Memory Load at Boot

PCI1xxxx first reads configurations from OTP if present in all cases. If an EEPROM is available and configured, PCI1xxxx subsequently takes additional settings from EEPROM.

PCI1xxxx takes the following steps when reading from EEPROM:

1. Check that the first byte is the MAGIC_BYTE. If the MAGIC_BYTE is present then continue, otherwise, EEPROM is not configured and EEPROM loading is skipped.
2. Latch in MEMORY_SIZE and use this as a validation check to ensure that HW does not read past the end of configured memory due to a poor configuration. When PCI1xxxx reaches the end of memory it shall stop reading and assume that configuration has been completed.
3. Latch in MEM_REGION_PROG to determine which memory areas have been programmed. Each bit in MEM_REGION_PROG when set indicates that the corresponding memory descriptor has been programmed.
4. PCI1xxxx loops through each memory descriptor, MEM_REGION_DESRx, which has been programmed according to MEM_REGION_PROG. Other memory descriptors are ignored.
5. PCI1xxxx reads the MEM_TAG[7:0] memory which indicates the subsytem to be configured. PCI1xxxx verifies that this subsystem is valid for the CPN defined by CPN_SEL[2:0] and that the subsystem is actually requesting configuration by checking bits [7:0] of the GENERAL_SYS_CONFIG_REQ_REG before proceeding with configuring the subsystem. If the subsystem is not to be configured, then PCI1xxxx goes to step 10.
6. For the subsystem to be configured, PCI1xxxx reads the MEM_START_ADDRESS[12:0] memory in the MEM_REGION_DESRx which points to the configured area of memory.
7. PCI1xxxx reads the byte stored at CFG_BYTE:

   7.1 If the byte is 0x00, this is a STOP opcode. There is no more data to read from the configuration block. HW goes to step 9.

   7.2 If the byte is 0x80, this is a SET_MEMORY_ADDRESS opcode. PCI1xxxx increments CFG_BYTE by 1 and reads the MEMORY_ADDRESS which it stores in MEM_DWORD. Then the PCI1xxxx increments CFG_BYTE by 4.

   7.3 If the byte is 0x81...0xFF, this is a SKIP_DWORDS opcode. If MEMORY_ADDRESS = 0x0000_0000 there has been no SET_MEMORY_ADDRESS opcode prior to the SKIP_DWORDS opcode, HW sets the relevant error bit in GENERAL_SYS_EEPROM_CONFIG_ERR_REG for EEPROM and the corresponding MEM_TAG[7:0] (bits 15...21) and goes to step 9. PCI1xxxx increments the value stored in MEM_DWORD by 4*(opcode-0x80) and increments CFG_BYTE by 1.

   7.4 PCI1xxxx reads the LENGTH.

   7.5 If LENGTH = 0, this is an error. PCI1xxxx sets the relevant error bit in GENERAL_SYS_EEPROM_CONFIG_ERR_REG for EEPROM and the corresponding MEM_TAG[7:0] (bits 0...7) and goes to step 9.

   7.6 PCI1xxxx increments CFG_BYTE by 1 and reads the DATA which is of LENGTH DWORDS.

   7.7 PCI1xxxx increments CFG_BYTE by 4*LENGTH.

   7.8 PCI1xxxx writes LENGTH DWORDS of DATA to MEMORY_ADDRESS. If LENGTH = 1, HW will use bits 0...1 of MEMORY_ADDRESS as a byte selector.

   7.9 PCI1xxxx increments the value stored in MEM_DWORD by 4*LENGTH.

   7.10 PCI1xxxx goes back to step 7.
8. If at any point the CFG_BYTE being read is past the end of the memory indicated by MEMORY_SIZE, PCI1xxxx sets the GENERAL_SYS_EEPROM_CONFIG_ERR_REG.EEPROM_MEM_OVERSHOOT_ERR bit and goes immediately to step 9.
9. If there are more subsystems to configure, PCI1xxxx goes back to step 6.
10. If there are no straps present indicating configuration via SPI/SMBUS, PCI1xxx will continue boot process and activate hardware subsystems.

15.3 EEPROM Programming via Linux Memory-Mapped Device

On Linux, the PCI1xxx enumerates OTP and EEPROM as two separate 8 kB disks through the GPIO subsystem driver.

This allows the OTP and EEPROM to be programmed using the Linux dd command.

The driver also provides a custom IOCTL interface which enables programming of a variable length byte stream from any offset. The driver always enumerates the OTP, whereas the EEPROM is enumerated only if it is present on the board.

OTP is enumerated as:

```
/dev/PCI1xxxxOTPn
```

EEPROM is enumerated as:

```
/dev/PCI1xxxxE2Pn
```

To program a configuration file (called ‘input.bin’ in this example) to EEPROM via dd commands, refer to the following example:

```
sudo dd if=input.bin of=/dev/PCI1xxxxE2Pn bs=512 count=16 oflag=direct
```

To read-back the EEPROM via dd commands, refer to the following example:

```
sudo dd if=/dev/PCI1xxxxE2Pn of=output.bin bs=512 count=16 iflag=direct
```

16.0 SMBUS/I2C TARGET CONFIGURATION

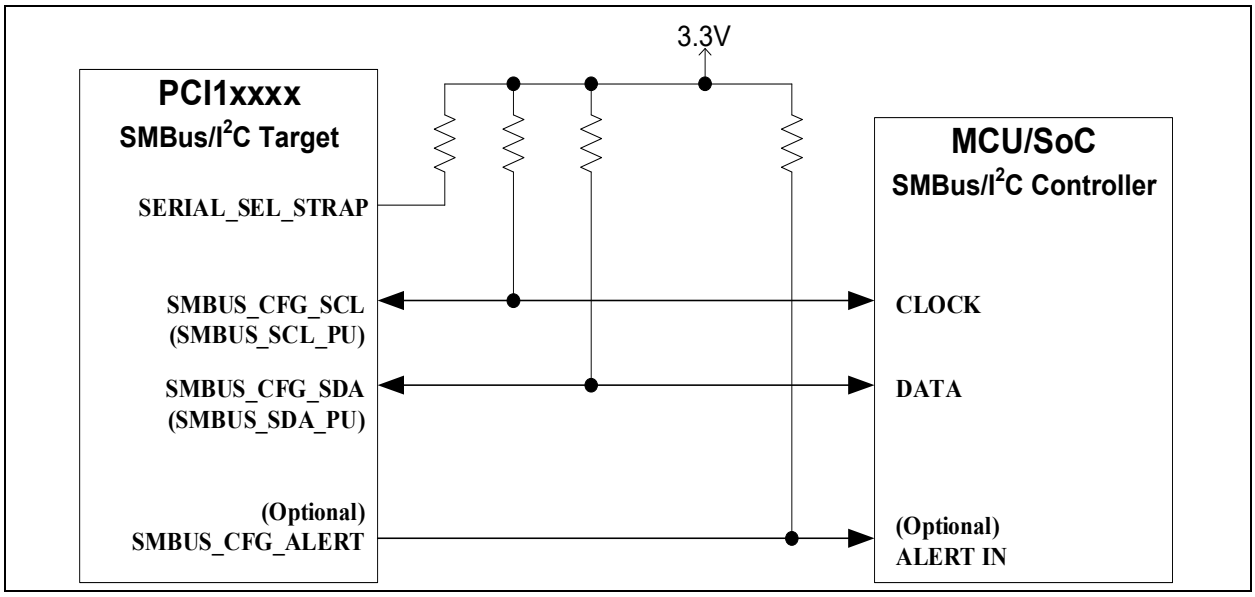
16.1 Sections

16.2 Configuring PCI1XXXX for SMBus/I2C Configuration

In order to configure the PIC1XXXX for SMBus/I2C configuration, the hardware configuration strap pins must be configured appropriately.

See Figure 7 for correct hardware strap and interface handling.

FIGURE 7: SMBUS/I2C CONFIGURATION INTERFACE - SCHEMATIC EXCERPT



These hardware configuration straps vary by device. Refer to Table 156 for pinout by device.

TABLE 156: HARDWARE STRAPS REQUIRED TO ENABLE SPI CONFIGURATION INTERFACE - PIN MAPPING BY DEVICE

Device	SERIAL SEL_STRAP	SMBUS_SCL_PU	SMBUS_SDA_PU
PCI12000	Pin 24 (PROG33)	Pin 38 (PROG64)	Pin 39 (PROG65)
PCI11010	Pin 64 (PROG68)	Pin 60 (PROG64)	Pin 61 (PROG65)
PCI11101	Pin 78 (PROG68)	Pin 74 (PROG64)	Pin 75 (PROG65)
PCI11400	Pin 70 (PROG68)	Pin 66 (PROG64)	Pin 67 (PROG65)
PCI11414	Pin 93 (PROG68)	Pin 89 (PROG64)	Pin 90 (PROG65)

The SMBus/I2C interface pin mappings vary by device. Refer to Table 157 for pinout by device.

TABLE 157: SMBUS/I2C INTERFACE REQUIRED TO ENABLE SPI CONFIGURATION INTERFACE - PIN MAPPING BY DEVICE

Device	SMBUS_CFG_SCL	SMBUS_CFG_SCL	SMBUS_CFG_ALERT_N
PCI12000	Pin 38 (PROG64)	Pin 39 (PROG65)	Pin 40 (PROG66)
PCI11010	Pin 60 (PROG64)	Pin 61 (PROG65)	Pin 62 (PROG66)
PCI11101	Pin 74 (PROG64)	Pin 75 (PROG65)	Pin 76 (PROG66)
PCI11400	Pin 66 (PROG64)	Pin 67 (PROG65)	Pin 68 (PROG66)
PCI11414	Pin 89 (PROG64)	Pin 90 (PROG65)	Pin 91 (PROG66)

16.3 SMBus/I2C Alert Configuration

The SMBUS_CFG_ALERT_N pin is an optional alert function which must be specifically enabled and configured before use.

SMBUS_CFG_ALERT_N is used in conjunction with the SMBus Alert Response Address (ARA). The PCI1xxxx provides this as an output from the SMBus target interface via the SMBUS_SLV_ALERT_N pin when this has been configured.

The PCI1xxxx can signal the controller through SMBUS_SLV_ALERT_N that it has an update to communicate to the controller. The SMBus/I2C controller processes the interrupt and simultaneously accesses all SMBUS_SLV_ALERT_N devices through the Alert Response Address.

Only the device(s) which pulled SMBALERT# low will acknowledge the Alert Response Address. The host performs a modified Receive Byte operation.

The 7-bit device address provided by the target transmit device is placed in the 7 most significant bits of the byte. The 8-bit can be a 0 or 1.

The SMBUS_SLV_ALERT_N pin shall be pulled low when:

Note 1: Any EXT_SYS_CONFIG_REQ_REG.EXT_*CONFIG_REQ bit is set (transitions from '0' to '1').

2: Any EXT_SYS_CONFIG_REQ_REG.EXT_*CONFIG_REQ bit is '1' and the corresponding block becomes not ready (GENERAL_SYS_READY_REG.*_RDY transitions from '0' to '1').

When the I2C_ALERT_SC bit is set to '1', the SMBUS_CFG_ALERT_N pin, when asserted, will be automatically cleared after receiving an acknowledge (ACK) from the controller in response to its address.

When the I2C_ALERT_SC bit is set to '0', the SMBUS_CFG_ALERT_N pin, when asserted, has to be manually cleared by host software writing '1' to I2C_SLV_ALERT.

16.4 SMBus/I2C Instructions

SMBus/I2C is a bidirectional two-wire data protocol. A device that sends data is defined as a transmitter and a device that receives data is defined as a receiver. The bus is controlled by a controller which generates the SCL clock, controls bus access and generates the start and stop conditions. Either a controller or target may operate as a transmitter or receiver as determined by the controller.
Both the clock (SCL) and data (SDA) signals have digital input filters that reject pulses that are less than 100 ns.

The following Bus states exist:

- **Idle**: Both SDA and SCL are high when the bus is idle.
- **Start & Stop Conditions**: A start condition is defined as a high to low transition on the SDA line while SCL is high. A stop condition is defined as a low to high transition on the SDA line while SCL is high. The bus is considered to be busy following a start condition and is considered free 4.7 µs/1.3 µs (for 100 kHz and 400 kHz operation, respectively) following a stop condition. The bus stays busy following a repeated start condition (instead of a stop condition). Starts and repeated starts are otherwise functionally equivalent.
- **Data Valid**: Data is valid, following the start condition, when SDA is stable while SCL is high. Data can only be changed while the clock is low. There is one valid bit per clock pulse. Every byte must be 8 bits long and is transmitted MSB first.

- **Acknowledge**: Each byte of data is followed by an acknowledge bit. The controller generates a ninth clock pulse for the acknowledge bit. The transmitter releases SDA (high). The receiver drives SDA low so that it remains valid during the high period of the clock, taking into account the setup and hold times. The receiver may be the controller or target depending on the direction of the data. Typically, the receiver acknowledges each byte. If the controller is the receiver, it does not generate an acknowledge on the last byte of a transfer. This informs the target to not drive the next byte of data so that the controller may generate a stop or repeated start condition.

16.5 SMBus/I²C Target Operation

When in SMBus/I²C managed mode, the SMBus/I²C target interface is used for CPU/MCU/SoC management of the device. All device registers are accessible to the CPU/MCU/SoC in these modes. The SMBus/I²C target controller implements the low level SMBus/I²C target serial interface (start and stop condition detection, data bit transmission and reception and acknowledge generation and reception), handles the target command protocol and performs system register reads and writes. The SMBus/I²C target controller conforms to SMBus/I²C specifications.

The SMBus/I²C target serial interface consists of a data wire (SMBUS_CFG_SDA) and a serial clock (SMBUS_CFG_SCL). The serial clock is driven by the controller, while the data wire is bi-directional. Both signals are open-drain and require external pull-up resistors.

The SMBus/I²C target serial interface supports the Standard-mode speed of up to 100 kHz and the Fast-mode speed of 400 kHz.

Refer to the SMBus/I²C specifications for detailed timing information with the following modifications:

- $t_{VD;DAT}$ maximum (SDA data output valid from SCL falling) is 3000 ns and 700 ns for standard and fast modes respectively.
- $t_{VD;ACK}$ maximum (SDA acknowledge output valid from SCL falling) is 3000 ns and 700 ns for standard and fast modes respectively.
- t_{SP} maximum (input spike suppression on SCL and SDA) is 100 ns.

Note:	The minimum pulse filter width is programmable via the I2C_SLV_CONFIG_REG.I2C_PULSE_FILTER [3:0] field.

- $t_{HD;DAT}$ minimum (SDA data and acknowledge output hold from SCL falling) is 100 ns.

Note:	The $t_{HD;DAT}$ parameter is 100 ns larger than that in the I²C specification. The t_{HD} parameter is specified from the $V_{IL(max)}$ point of SCL. The fall time of SCL (from $V_{IH(min)}$ to $V_{IL(max)}$) is potentially 300 ns. Since the SDA output could change at any point when SCL is below $V_{IH(min)}$, it could change before SCL is valid low. To compensate for this, the device must provide an additional 300 ns of hold time (total of 400 ns) to meet the desired 100 ns.

16.5.1 SMBUS/I²C TARGET COMMAND FORMAT

The SMBus/I²C target serial interface supports single register and multiple register read and write commands. A read or write command is started by the controller first sending a start condition, followed by a control byte. The control byte consists of a 7-bit target address and a 1-bit read/write indication (R/~W). The default target address used by the device is 0001010b, written as SA6 (first bit on the wire) through SA0 (last bit on the wire). Alternatively, the SMBus/I²C target address may be configured to another address by configuring the I2C_CFG_CONFIG_REG.I2C_CFG_ADDR[6:0] field with the desired value. Assuming the target address in the control byte matches this address, the control byte is acknowledged by the device. Otherwise, the entire sequence is ignored until the next start condition.

If the read/write indication (R/~W) in the control byte is a 0 (indicating a potential write), the next 32 bits sent by the controller is the register address. After the address DWORD is acknowledged by the target, the controller may either send 4 data bytes to be written or it may send another start condition (to start the reading of data) or a stop condition. The latter two will terminate the current write (without writing any data), but will have the affect of setting the internal register address which will be used for subsequent reads. If the read/write indication in the control byte is a 1 (indicating a read), the target will start sending data following the control byte acknowledgment.

Until the device has been initialized to the point where the various configuration inputs are valid, the SMBus/I²C target interface must not respond to or be affected by any external pin activity. If the device initialization completes during an active cycle, the trailing end of the cycle should be ignored. Internal registers should not be affected, and the state of the I²C target interface must not change.

16.5.1.1 SMBus/I2C Target Read Polling for Initialization Complete

Before device initialization, the SMBus/I²C interface will not return valid data. To determine when the I2C target interface is functional, the Byte Order Test Register (BYTE_TEST_REG) should be polled. Once the correct pattern is read, the interface can be considered functional.

Note:	The SMBus/I2C controller should only use single register reads (one data cycle per Start/Stop) while polling the BYTE_TEST_REG register.

16.5.2 ACCESS DURING AND FOLLOWING POWER MANAGEMENT

During Low-power mode, reads and writes will be ignored. The SMBus/I2C target interface will not respond to or be affected by any external pin activity. To determine when the I2C target interface is functional, the Byte Order Test Register (BYTE_TEST_REG) should be polled. Once the correct pattern is read, the interface can be considered functional.

Note:	The SMBus/I2C controller should only use single register reads (one data cycle per Start/Stop) while polling the BYTE_TEST_REG register.

16.5.3 READ COMMAND DETAIL

Following the device addressing, a register is read from the target when the controller sends a start condition and control byte with the R/~W bit set. Assuming the target address in the control byte matches the target address, the control byte is acknowledged by the target. Otherwise, the entire sequence is ignored until the next start condition. Following the acknowledgment, the device sends 4 bytes of data. The first 3 bytes are acknowledged by the controller and on the fourth, the controller sends a no-acknowledge followed by the Stop condition. The no-acknowledge informs the device not to send the next 4 bytes (as it would in the case of a multiple read). The internal register address is unchanged following the single read.

Multiple reads are performed when the controller sends an acknowledge on the fourth byte. The internal address is incremented and the next register is shifted out. Once the internal address reaches its maximum, it rolls over to 0. The multiple read is concluded when the controller sends a no-acknowledge followed by a Stop condition. The no acknowledge informs the target not to send the next 4 bytes. The internal register address is incremented for each read including the final.

For both single and multiple reads, in the case that the controller sends a no-acknowledge on any of the first three bytes of the register, the device will stop sending subsequent bytes. If the controller sends an unexpected start or Stop condition, the target will stop sending immediately and will respond to the next sequence as needed.

SMBus/I2C reads from unused register addresses return all zeros.

EXAMPLE 1: READ COMMAND EXAMPLE

An example register read of BYTE_TEST_REG, which is located at address ‘0000_0120h’, is shown in Table 158. In the example, the expected return value of ‘8765_4321h’ is returned.

Note:	Since the register reads are always performed in blocks of 4 bytes, the last two bits of the register address are discarded and only bits 9:2 are used. The two most significant bits are not necessary either since the total register space of PCI1xxxx does not utilize them. In this example, the target address is 120h = 0001_0010_0000b. With the two most significant and two least significant bits discarded. This becomes 01001000b.

TABLE 158: SMBUS/I²C REGISTER WRITE EXAMPLE

Action	Control	Write Control Byte (w/ Target SMBus/I²C Address)								Status	Register Address Byte (Bits 9:2)								Status
Bit	START	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0	WRITE	ACK	REG ADDR9	REG ADDR8	REG ADDR7	REG ADDR6	REG ADDR5	REG ADDR4	REG ADDR3	REG ADDR2	ACK
Controller	S	0	0	0	1	0	1	0	0	z	0	1	0	0	1	0	0	0	
Target	z	z	z	z	z	z	z	z	z	0	z	z	z	z	z	z	z	z	0

Action	Control	Read Control Byte (w/ Target SMBus/I²C Address)								Status	Data Byte (Bits 31:24)								Status
Bit	START	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0	READ	ACK	D31	D30	D29	D28	D27	D26	D25	D24	ACK
Controller	S	0	0	0	1	0	1	0	1	z	z	z	z	z	z	z	z	z	0
Target	z	z	z	z	z	z	z	z	z	0	1	0	0	0	0	1	1	1	z

Action	Data Byte (Bits 23:16)								Status	Data Byte (Bits 15:8)								Status
Bit	D23	D22	D21	D20	D19	D18	D17	D16	ACK	D15	D14	D13	D12	D11	D10	D9	D8	ACK
Controller	z	z	z	z	z	z	z	z	0	z	z	z	z	z	z	z	z	0
Target	0	1	1	0	0	1	0	1	z	0	1	0	0	0	0	1	1	z

Action	Data Byte (Bits 7:0)								Status	Control
Bit	D7	D6	D5	D4	D3	D2	D1	D0	NOT ACK	STOP
Controller	z	z	z	z	z	z	z	z	z	P
Target	0	0	1	0	0	0	0	1	z	z

16.5.4 WRITE COMMAND DETAIL

Following the target addressing, a register is written to the device when the controller continues to send data bytes. Each byte is acknowledged by the target. Following the fourth byte of the sequence, the controller may either send another Start condition or halt the sequence with a Stop condition. The internal register address is unchanged following a single write. Multiple writes are performed when the controller sends additional bytes following the fourth acknowledge. The internal address is automatically incremented and the next register is written. Once the internal address reaches its maximum value, it rolls over to 0. The multiple write is concluded when the controller sends another Start or Stop condition. The internal register address is incremented for each write including the final. This is not relevant for subsequent writes, since a new register address would be included on a new write cycle. However, this does affect the internal register address if it were to be used for reads without first resetting the register address.
For both single and multiple writes, if the controller sends an unexpected start or Stop condition, the device will stop immediately and will respond to the next sequence as needed.
The data write to the register occurs after the 32 bits are input. In the event that 32 bits are not written (controller sends

a Start or a Stop condition occurs unexpectedly), the write is considered invalid and the register is not affected. Multiple registers may be written in a multiple write cycle, each one being written after 32 bits. SMBus/I^{2}C writes must not be performed to unused register addresses.

EXAMPLE 2: WRITE COMMAND EXAMPLE

An example register write of GENERAL_SYS_CONFIG_DONE_REG, which is located at address '058h', is shown in Table 159. In the example, the configuration change flags for each hardware component are set = 1, so that configuration changes take place for every system component.

Note: Since the register writes are always performed in blocks of 4 bytes, the last two bits of the register address are discarded and only bits 9:2 are used. The two most significant bits are not necessary either since the total register space of PCI1xxxx does not utilize them. In this example, the target address is 058h = 0000_0101_0100b. With the two most significant and two least significant bits discarded, this becomes 00010101b.

TABLE 159: SMBUS/I^{2}C REGISTER WRITE EXAMPLE

Action	Control	Write Control Byte (w/ Target SMBus/I^{2}C Address)								Status	Register Address Byte (Bits 9:2)								Status
Bit	START	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0	WRITE	ACK	REG ADDR9	REG ADDR8	REG ADDR7	REG ADDR6	REG ADDR5	REG ADDR4	REG ADDR3	REG ADDR2	ACK
Controller	S	0	0	0	1	0	1	0	0	z	0	0	0	1	0	1	0	1	
Target	z	z	z	z	z	z	z	z	z	0	z	z	z	z	z	z	z	z	0

Action	Data Byte (Bits 31:24)								Status	Data Byte (Bits 23:16)								Status
Bit	D31	D30	D29	D28	D27	D26	D25	D24	ACK	D23	D22	D21	D20	D19	D18	D17	D16	ACK
Controller	0	0	0	0	0	0	0	1	z	0	0	1	1	1	1	1	1	z
Target	z	z	z	z	z	z	z	z	0	z	z	z	z	z	z	z	z	0

Action	Data Byte (Bits 15:8)								Status	Data Byte (Bits 7:0)								Status	Control
Bit	D15	D14	D13	D12	D11	D10	D9	D8	ACK	D7	D6	D5	D4	D3	D2	D1	D0	ACK	STOP
Controller	0	0	1	1	1	1	1	1	z	0	0	1	1	1	1	1	1	z	P
Target	z	z	z	z	z	z	z	z	0	z	z	z	z	z	z	z	z	0	z

16.6 Configuration Complete

If the PCI1XXXX is configured for I^{2}C Configuration, it will wait within the configuration stage indefinitely until the EXT_SYS_CONFIG_DONE_REG bits are all set.

17.0 SPI CONFIGURATION

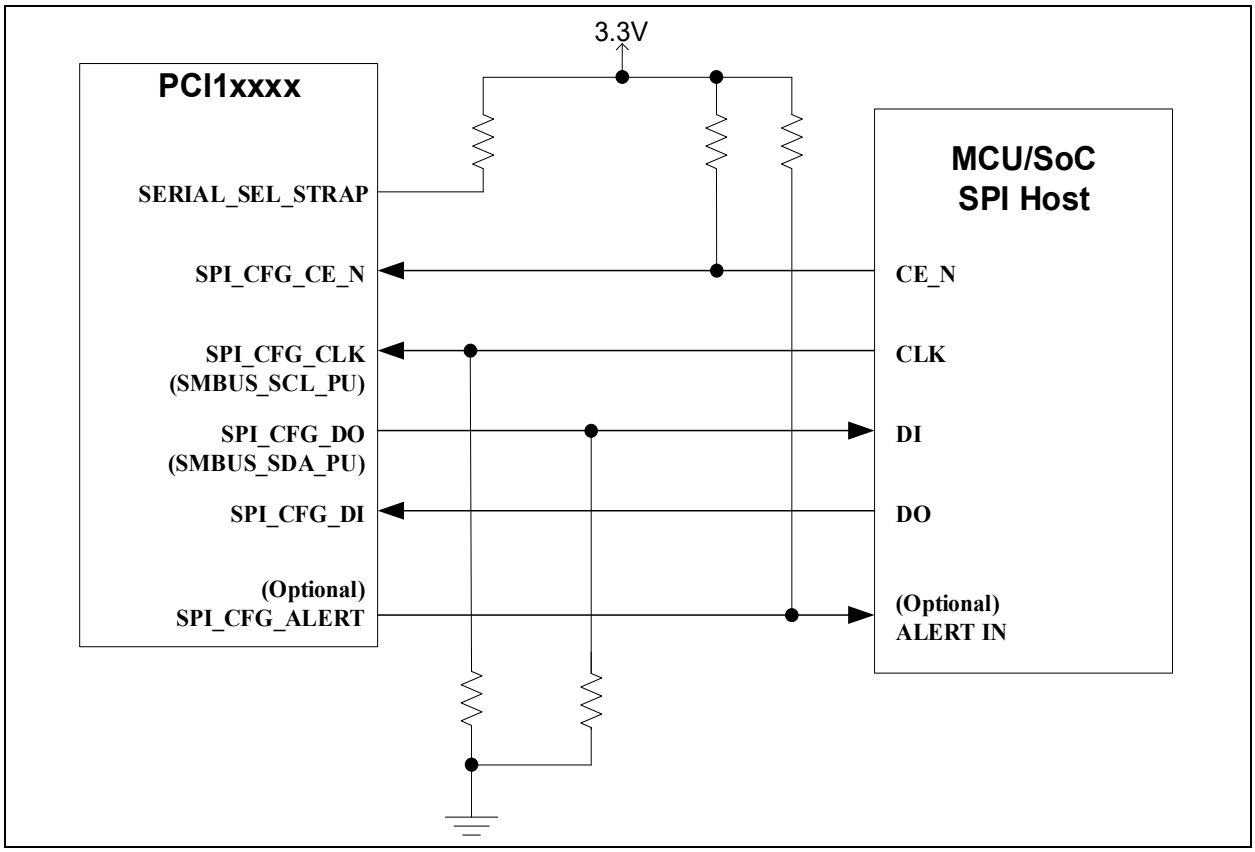
17.1 Sections

17.2 Configuring PCI1XXXX for SPI Configuration

To configure the PIC1XXXX for SPI configuration, the hardware configuration strap pins must be configured appropriately.

See Figure 8 below for correct hardware strap and interface handling.

FIGURE 8: SPI CONFIGURATION INTERFACE - SCHEMATIC EXCERPT



These hardware configuration straps vary by device. Refer to Table 160 for pinout by device.

TABLE 160: HARDWARE STRAPS REQUIRED TO ENABLE SPI CONFIGURATION INTERFACE - PIN MAPPING BY DEVICE

Device	SERIAL SEL_STRAP	SMBUS_SCL_PU	SMBUS_SDA_PU
PCI12000	Pin 24 (PROG33)	Pin 38 (PROG64)	Pin 39 (PROG65)
PCI11010	Pin 64 (PROG68)	Pin 60 (PROG64)	Pin 61 (PROG65)
PCI11101	Pin 78 (PROG68)	Pin 74 (PROG64)	Pin 75 (PROG65)
PCI11400	Pin 70 (PROG68)	Pin 66 (PROG64)	Pin 67 (PROG65)
PCI11414	Pin 93 (PROG68)	Pin 89 (PROG64)	Pin 90 (PROG65)

The SPI interface pin mappings vary by device. Refer to Table 161 for pinout by device.

TABLE 161: SPI INTERFACE REQUIRED TO ENABLE SPI CONFIGURATION INTERFACE - PIN MAPPING BY DEVICE

Device	SPI_CFG_CLK	SPI_CFG_DO	SPI_CFG_DI	SPI_CFG_CE_N	SPI_CFG_ALERT_N
PCI12000	Pin 38 (PROG64)	Pin 39 (PROG65)	Pin 40 (PROG66)	Pin 41 (PROG67)	Configurable
PCI11010	Pin 60 (PROG64)	Pin 61 (PROG65)	Pin 62 (PROG66)	Pin 63 (PROG67)	Configurable
PCI11101	Pin 74 (PROG64)	Pin 75 (PROG65)	Pin 76 (PROG66)	Pin 78 (PROG67)	Configurable
PCI11400	Pin 66 (PROG64)	Pin 67 (PROG65)	Pin 68 (PROG66)	Pin 69 (PROG67)	Configurable
PCI11414	Pin 89 (PROG64)	Pin 90 (PROG65)	Pin 91 (PROG66)	Pin 92 (PROG67)	Configurable

17.3 SPI Alert Configuration

The SPI_CFG_ALERT_N pin is an optional alert function which must be mapped to an available PROGx pin. This pin may be mapped to any of spare PROGx pins shown in Table 162.

TABLE 162: SPI CONFIGURATION INTERFACE ALERT PIN MAPPINGS BY DEVICE

Device	PROG0	PROG1	PROG2	PROG3	PROG4	PROG5	PROG6	PROG7	PROG8	PROG9	PROG10	PROG17	PROG18	PROG19	PROG20	PROG21	PROG22	PROG23	PROG24
PCI12000																			
PCI11010	X	X	X	X								X	X						
PCI11101	X	X	X		X	X	X					X	X	X	X	X			
PCI11400	X	X	X	X	X	X	X	X				X	X	X	X				
PCI11414	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
	PROG46	PROG47	PROG48	PROG49	PROG50	PROG51	PROG67	PROG68	PROG73	PROG79	PROG81	PROG86	PROG87						
PCI12000							X	X											
PCI11010	X	X	X	X	X	X	X	X		X	X								
PCI11101	X	X	X	X	X	X	X	X	X	X	X								
PCI11400	X	X	X	X			X	X		X	X								
PCI11414	X	X	X	X	X	X	X	X		X	X	X	X						

The PCI1xxxx provides this as an output from the SPI Configuration Channel via the SPI_CFG_ALERT_N pin when this has been specifically configured.

The SPI_CFG_ALERT_N pin shall be pulled low when:

- Any EXT_SYS_CONFIG_REQ_REG.EXT_*CONFIG_REQ bit is set (transitions from '0' to '1').
- Any EXT_SYS_CONFIG_REQ_REG.EXT_*CONFIG_REQ bit is '1' and the corresponding block becomes not

ready (GENERAL_SYS_READY_REG.*_RDY transitions from '0' to '1').

When the SPI_CFG_CONFIG_REG.SPI_ALERT_SC bit is set to '1', the SPI_CFG_CE_N pin, when asserted, will be automatically cleared when the SPI_CFG_CE_N pin goes from high to low. After the SPI_CFG_CE_N pin is asserted by the SPI Configuration Host (goes from '0' to '1'), the target device must stop pulling down on SMBUS_CFG_ALERT_N pin.

When the SPI_CFG_CONFIG_REG.SPI_ALERT_SC bit is set to '0', the SPI_CFG_CE_N pin, when asserted, has to be manually cleared by host software writing '1' to SPI_CFG_STATUS_REG.SPI_CFG_ALERT. After the SPI_CFG_STATUS_REG.SPI_CFG_ALERT is cleared by host software, the target device must stop pulling down on SMBUS_CFG_ALERT_N pin.

17.4 SPI Instructions

The SPI Interface supports basic Read and Write commands only.

Before PCI1xxxx initialization, the SPI interface will not return valid data. To determine when the SPI interface is functional, the Byte Order Test Register (BYTE_TEST_REG) should be polled. Once the correct pattern is read, the interface can be considered functional. Use only single register reads during this polling period.

Input data on the SPI_CFG_DI pin is sampled on the rising edge of the SCK input clock. Output data is sourced on the SPI_CFG_DO pin with the falling edge of the clock.

The SPI Interface supports basic Read and Write commands only, see Table 163.

TABLE 163: SUPPORT SPI INSTRUCTIONS

Instruction	Code	Bytes			Max Frequency
		Address	Dummy	Data	
Read	03h	4	2	4+	30 MHz
Write	02h	4	0	4+	80 MHz

Note: The following commands are not supported: FASTREAD, SDOR, SDIOR, SQOR, SQIOR, SDDW, SDADW, SQDW, SQADW.

17.4.1 READ COMMAND DETAIL

The READ instruction inputs the instruction code and address bytes one bit per clock and outputs the data one bit per clock. This instruction is supported in SPI bus protocol only with clock frequencies up to 30 MHz.

The SPI target interface is selected by first bringing SPI_CFG_CE_N active. The 8-bit READ instruction, 03h, is input into the SPI_CFG_DO pin, followed by the four address bytes and two dummy bytes. The address bytes specify a BYTE address within the device. The two dummy bytes add wait states corresponding to the maximum 512 ns allowed for reading an internal register (16*33 ns).

On the falling clock edge following the rising edge of the last dummy bit, the SPI_CFG_DI pin is driven starting with the most-significant bit (msb) of the least-significant byte (LSB) of the selected register. The remaining register bits are shifted out on subsequent falling clock edges. The SPI_CFG_CE_N input is brought inactive to conclude the cycle. The SPI_CFG_DI pin is tri-stated (Hi-Z) at this time.

EXAMPLE 3: SPI READ COMMAND EXAMPLE

An example register read of BYTE_TEST_REG, which is located at address '0000_0120h', is shown in Table 164. In the example, the expected return value of '8765_4321h' is returned.

TABLE 164: SPI SINGLE REGISTER READ EXAMPLE

Name	Instruction	Address				Dummy Bytes		Return Data			
Byte	1	2	3	4	5	6	7	8	9	10	11
SI	02h	00h	00h	01h	20h	00h	00h	x	x	x	x
SO	z	z	z	z	z	z	z	21	43	65	87

17.4.2 WRITE COMMAND DETAIL

The WRITE instruction inputs the instruction code and address and data bytes one bit per clock. This instruction is supported in SPI bus protocols with clock frequencies up to 30 MHz.

The SPI target interface is selected by first bringing SPI_CFG_CE_N active. For SPI mode, the 8-bit WRITE instruction, 02h, is input into the SI/SIO[0] pin, followed by the four address bytes. The address bytes specify a BYTE address within the device.

The data follows the address bytes. The data is input into the SPI_CFG_DI pin starting with the msb of the LSB. The data write to the register occurs after the 32 bits are input. In the event that 32 bits are not written when the SPI_CFG_CE_N is returned high, the write is considered invalid and the register is not affected.

The SPI_CFG_CE_N input is brought inactive to conclude the cycle.

EXAMPLE 4: SPI WRITE COMMAND EXAMPLE

An example register write of GENERAL_SYS_CONFIG_DONE_REG, which is located at address ‘0058h’, is shown in Table 165. In the example, the configuration change flags for each hardware component are set = 1, so that configuration changes take place for every system component.

TABLE 165: SPI SINGLE REGISTER WRITE EXAMPLE

Name	Instruction	Address				Dummy Bytes		Write Data			
Byte	1	2	3	4	5	6	7	8	9	10	11
SI	02h	00h	00h	00h	58h	00h	00h	3F	3F	3F	01
SO	z	z	z	z	z	z	z	z	z	z	z

17.5 Configuration Complete

If the PCI1xxxx is configured for SPI Configuration, it will wait within the configuration stage indefinitely until the EXT_SYS_CONFIG_DONE_REG bits are all set. See Table 165 for the example of a write to this register.

18.0 RUN-TIME CONFIGURATION

During this Run-time, the system may asynchronously reconfigure PCI1xxxx within a limited set of registers. Common functions which may be reconfigured at Run-time include:

- Ethernet MAC Address
- Ethernet PHY configuration parameters (e.g. full/half-duplex, auto-negotiation)
- USB Port Disable/Enable
- UART Speed (baud rate)
- I^2C/SPI Speed

Completion of a reconfiguration must be indicated by assertion of a corresponding Run-time Configuration can be managed by three pathways:

- PCIe Host
- I^2C Configuration Channel
- SPI Configuration Channel

APPENDIX A: SYSTEM LOCK REGISTER (SYS_LOCK_REG)

The PCI1XXXX allows multiple drivers to access common system registers from different PCIe memory spaces. The SYS_LOCK registers provide a safe mechanism to prevent conflicts from occurring due to multiple concurrent accesses by different drivers.

Most users of PCI1XXXX devices will not need to be aware of the SYS_LOCK_REG registers and operation, but during certain debug scenarios (like testing hardware without a driver loaded) may require understanding and manual access to the registers.

A.1 Driver Use Model for SYS_LOCK_REG

1. Read SYS_LOCK_REG.
2. If any bit is set, wait some time and go back to step 1.
3. If no other driver has a lock on any system component, write SYS_LOCK_REG with a 1b to its corresponding function bit.
4. Read SYS_LOCK_REG to ensure lock was obtained.
5. If the expected lock bit was not set, go back to step 1 again (lock acquisition failed as another driver obtained lock first).
6. Else, if the expect lock bit has been set, then lock has been acquired. Perform any desired operations on System registers until fully done accessing them or otherwise software is in a state where other software stepping in between its usage is guaranteed not to cause problems.
7. Write SYS_LOCK_REG with a 0b to its corresponding function bit (Release lock).

Caution:	Failure to follow the above procedure will result in System register writes being ignored, and System register reads always returning 0x00000000. A driver is responsible for preventing any lock-ups due to the bits not being cleared.

A.2 Hardware Operation

The SYS_LOCK_REG follows these rules:

- Each bit of the SYS_LOCK_REG register can only be written - either set (lock acquired) or cleared (lock released) from its corresponding function's memory space.
- If one bit of the SYS_LOCK_REG has been set by one of the functions, no other function can set its corresponding bit until after that bit is cleared.
- The SYS_LOCK_REG register can always be read by any function regardless of whether its bit is set or not.
- To access any System register (except for the SYS_LOCK_REG) through its function's memory space, that function must have previously acquired the lock. If the lock has not been previously acquired any System register (except for the SYS_LOCK_REG) access by that function will be handled as follows:
  - Any write operation will be discarded
  - Any read operation will return 0x00000000.
- The SYS_LOCK_REG cannot be written to until the COMMAND bytes of the device's general PCI configuration space is configured properly (not left at default 0x0000 value). This is always done by a properly functioning driver, but will need to be set manually in a driverless scenario.

A.3 SYS_LOCK_REG Register Definition

The SYS_LOCK_REG is accessible from any of the PCIe function's memory spaces (i.e.: the USB subsystem, Ethernet system, or one of the peripheral subsystems), see Table 166 and Table 167.

To locate the register via PCI register access, you must first locate the BAR0 address for the selected peripheral by reading back the device configuration space (See **Appendix B: "PCI Configuration Space"**).

TABLE 166: SYS_LOCK_REG

Base Address		Peripheral Base Address		SYS_LOCK Offset
Obtain for BAR0 for selected Address	+	USB_SYSTEM _REG_ADDR_BASE = 0x0001_2000 ENET_SYSTEM_REG_ADDR_BASE = 0x0000_4000 PERI_UART_SYSTEM_REG_ADDR_BASE = 0x0000_1000 PERI_SMBUS_SYSTEM_REG_ADDR_BASE = 0x0001_1000 PERI_SPI_SYSTEM_REG_ADDR_BASE = 0x0002_1000 PERI_GEN_SYSTEM_REG_ADDR_BASE = 0x0003_1000	+	0x0A0

TABLE 167: SYS_LOCK_REG

SYS_LOCK_REG Offset: 0x0A0			System Lock Register Default = 0x0000_0000
BIT(s)	**Name**	**R/W**	**Description**
31:8	Reserved	R	Always read ‘0’
7	MAIN_LOCK	R/W	Bit for locking via SYSTEM_REG_ADDR_BASE 0b = Not Locked 1b = Locked
6	Reserved	R	Always read ‘0’
5	GEN_PERI_LOCK	R/W	Bit for locking via GENERAL_PERI_ADDR_BASE 0b = Not Locked 1b = Locked
4	SPI_PERI_LOCK	R/W	Bit for locking via SPI_PERI_ADDR_BASE 0b = Not Locked 1b = Locked
3	SMBUS_PERI_LOCK	R/W	Bit for locking via SMBUS_PERI_ADDR_BASE 0b = Not Locked 1b = Locked
2	UAR_PERI_LOCK	R/W	Bit for locking via UART_PERI_ADDR_BASE 0b = Not Locked 1b = Locked
1	ENET_SS_LOCK	R/W	Bit for locking via ENET_SUBSYSTEM_ADDR_BASE 0b = Not Locked 1b = Locked
0	USB_SS_LOCK	R/W	Bit for locking via USB_SUBSYSTEM_ADDR_BASE 0b = Not Locked 1b = Locked

APPENDIX B: PCI CONFIGURATION SPACE

All PCI devices, except host bus bridges, provide 256 bytes of configuration registers referred to as Configuration Address Space. This allows for each device to be initialized and configured entirely from the software level.

The configuration space memory mapping varies by device type. All peripherals available on PCI1xxxx, except for the PCIe switch, use header type 0x0.

The mapping of the configuration space is shown in the Table 168. Refer to relevant PCI specifications for the lower-level bit details. Byte order is always little-endian. Fields which occupy multiple adjacent bytes will have the least significant values at the lower addresses.

TABLE 168: CONFIGURATION SPACE MEMORY MAPPING

Offset	Bits[31:24]	Bits [23:16]	Bits [15:8]	Bits [7:0]
0x00	Device ID		Vendor ID	
0x04	Status		Command	
0x08	Class Code	Subclass	Prog IF	Revision ID
0x0C	BIST	Header Type	Latency Timer	Cache Size
0x10	BAR0 (Base Address 0)			
0x14	BAR1			
0x18	BAR2			
0x1C	BAR3			
0x20	BAR4			
0x24	BAR5			
0x28	Cardbus CIS Pointer			
0x2C	Subsystem Device ID		Subsystem Vendor ID	
0x30	Expansion ROM Base Address			
0x34	Reserved			Capabilities Pointer
0x38	Reserved			
0x3C	Max Latency	Min Grant	Interrupt Pin	Interrupt Line

B.1 Read Access of Configuration Space in Linux

“lspci” is a flexible utility for displaying information about all devices on the PCI bus. This tool allows for easy read-back of the configuration spaces of all devices on the PCI bus. To read the configuration space of all devices on the PCI bus, enter the following command into a Console terminal:

```
lspci -xxx
```

This command will return the full configuration space dump of all devices on the PCI bus, and will appear in the terminal as:

```
29:00.0 Serial controller: Microchip Technology / SMSC Device a042 (rev a0)
00: 55 10 42 a0 06 04 10 00 a0 02 00 07 10 00 80 00
10: 04 80 01 fc 00 00 00 00 04 70 02 fc 00 00 00 00
20: 04 60 02 fc 00 00 00 00 00 00 00 00 55 10 42 a0
30: 00 00 00 00 40 00 00 00 00 00 00 00 0b 01 00 00
```

Additionally, you can filter for only devices which match the Microchip Vendor ID by using the “-d [<vendor>]:[<device>]” option:

```
lspci -xxx -d 1055:*
```

B.2 Write Access of Configuration Space in Linux

The “setpci” is a utility for reading and setting PCI device configuration space. You can select a device either through the selection of the domain:bus:slot.function, or through the Vendor ID and Device ID:

```
setpci -s [[[[<domain>]:]<bus>]:][<slot>][.[<func>]]

setpci -d [<vendor>]:[<device>]
```

Finally, you can select the memory offset and width of data to set in the configuration space via either .b, for 1 Byte, .w for 2 Bytes, and .L for 4 bytes.

For example, to set the COMMAND register to 0x0406, and the device domain:bus:slot.function is known to be 29:00.3:

```
sudo setpci -s 29:00.3 04.w=0406
```

APPENDIX C: APPLICATION NOTE REVISION HISTORY

TABLE C-1: REVISION HISTORY

Revision Level & Date	Section/Figure/Entry	Correction
DS00005213B (11-06-24)	Table 16	Updated PHY and logical port mapping details.
	Table 47	Changed “BASE ADDRESS + 0x1_A220” to “BASE ADDRESS + 0x5_6004.”
	All	Removed “Confidential” marking from the footer. Updated data in R/W columns of register tables. Made minor text and formatting changes.
DS00005213A (04-01-24)	Initial document release	

NOTES:

THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at 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. 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 products:

- 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, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip product is strictly prohibited and may violate the Digital Millennium Copyright Act.
- 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. Microchip is committed to continuously improving the code protection features of our products.

ISBN: 978-1-6683-0501-0

For information regarding Microchip's Quality Management Systems, please visit 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

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 - Hod Hasharon
Tel: 972-9-775-5100

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