Document Number:C29xPCIeQS
Rev 1, 03/2014

C29x PCIe Card Quick Start Guide

1 Introduction

This C29x PCIe Card Quick Start Guide explains Freescale C29x crypto coprocessor board settings and physical connections needed to boot the board.

The document provides steps for configuring switches, connectors, jumpers, push buttons, and LEDs and connecting peripheral devices.

NOTE

The C29x PCIe card is modified to use the Power from the external 12V Power supply or the 2x3 ATX Power adaptor. The PCIe card does NOT draw the power from the PCIe gold finger.

NOTE

By default, the C29x PCIe runs in standalone mode. For C29x PCIe End Point Mode support, refer to the the dip switch setting in the PCIe Endpoint Mode section.

2 Related Documentation

The table below lists and explains the additional documents that you can refer to, for more information about C29x PCIe.

Contents

Some of the documents listed below may be available only under a non-disclosure agreement (NDA). To request access to these documents, contact your local field applications engineer or sales representative.

Table 1. Related documentation

Document	Description
C29x Crypto Coprocessor Family Reference Manual	Defines the functionality of the Freescale C29x family. Freescale C29x family consists of three highly integrated security processors, optimized for public key acceleration and secure key management. Each family member combines a Power Architecture processor core with a high performance security engine, network and high-speed serial interfaces, DDR and non-volatile memory controllers.
C29x Crypto Offload User Guide	Explains the procedure to build, configure, and use different software components for the Freescale C29x crypto coprocessor device.
C29x PCIe Card Getting Started Guide	Explains C29x PCIe board settings and physical connections needed to boot the board.
Freescale C29x Crypto Coprocessor Family Product Brief	Provides an overview of the Freescale C29x family of crypto coprocessor features, and examples of C29x usage.

3 C29x PCIe Board Top View

The figure below shows the C29x PCIe board top view.

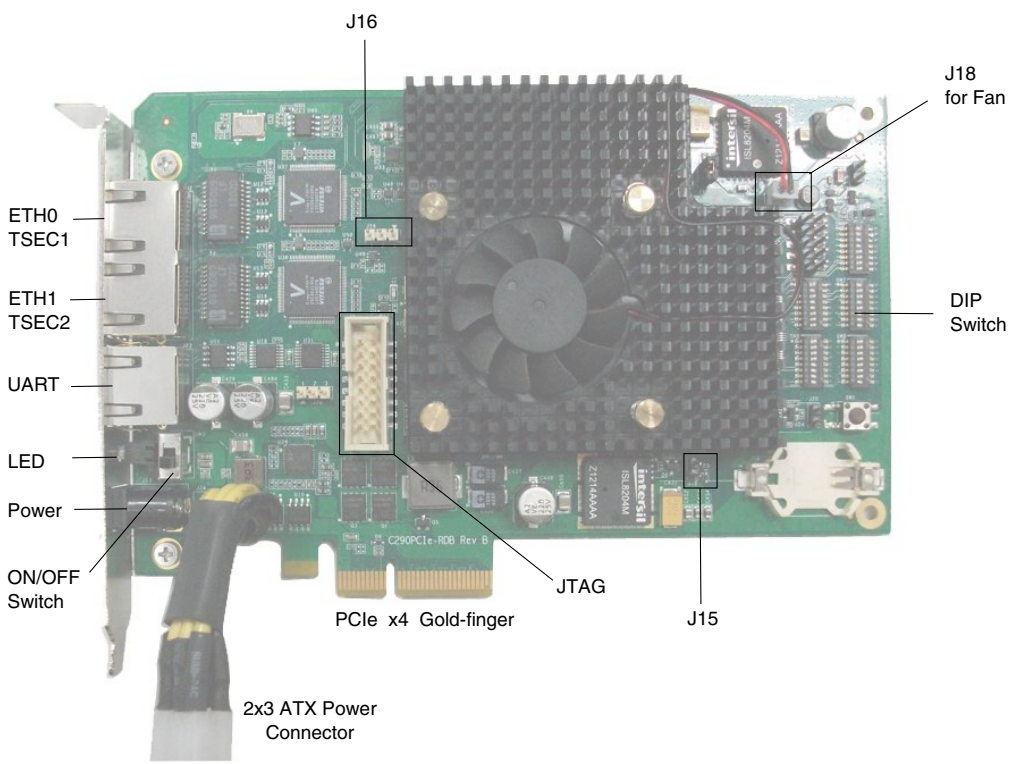

Figure 1. C29x PCIe Board

4 PCIe Use Cases

This section covers the following use cases for C29x PCIe card:

- PCIe endpoint mode
- Standalone host mode

4.1 PCIe Endpoint Mode

The figure below shows the C29x PCIe card, operating in the PCIe endpoint mode.

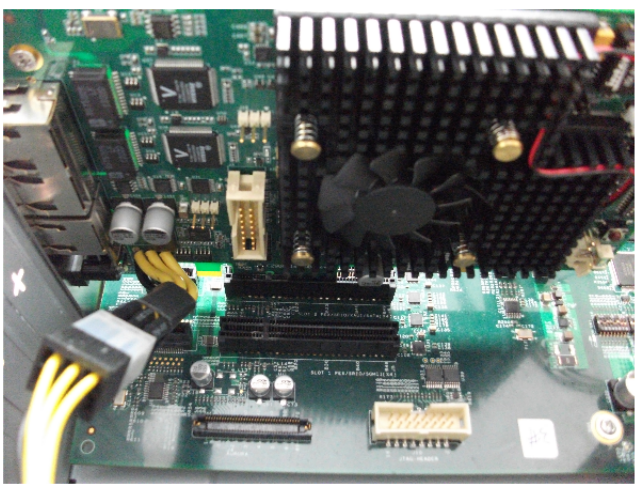

Figure 2. PCIe endpoint mode

Perform the following steps to use the C29x PCIe card in the PCIe endpoint mode:

1. Connect the heat sink fan power line to J18.
2. Plug C29x PCIe into the PCIe slot on the motherboard. C29x PCIe supports x1, x2, and x4 configurations.
3. Connect 2x3 ATX power connector to the ATX power supply of the computer, if ATX power supply is installed in the computer; otherwise, connect the 12V power adapter.
4. Connect the RJ45 console line (RJ45 to DB9 cable used) to UART port.
5. Configure the serial port of the attached computer with the following values:
    - Data rate: 115200 bps
    - Number of data bits: 8
    - Parity: None
    - Number of stop bits: 1
    - Flow control: Hardware/None
6. Connect network cable to TSEC1.
7. Power on host motherboard and put the power switch to ON position. Now, you will see the boot up message on the computer console.

Following are the device configurations required for this use case by setting DIP switch:

- `SW7[1] ON, cfg_cpu_boot=0`
- `SW7[5] ON, cfg_host_agt=0`

The table below shows dual in-line package (DIP) switch settings of the C29x PCIe card in PCIe endpoint mode (800 MHz core, 400 MHz platform, PCIe-x4 configuration).

Table 2. PCIe endpoint mode DIP switch settings

SW4[1..8]	0101 1000	ON OFF ON OFF OFF ON ON ON
SW5[1..8]	1111 0000	OFF OFF OFF OFF ON ON ON ON
SW6[1..8]	0000 1111	ON ON ON ON OFF OFF OFF OFF
SW7[1..8]	0001 0111	**ON** ON ON OFF ON OFF OFF OFF
SW8[1..8]	0000 1011	ON ON ON ON OFF ON OFF OFF

NOTE

If your host does not recognize the card when performing lspci, then try changing the PCIe lane configuration.

4.1.1 PKCAL/SKMM Mode

PKCAL means Public Key Calculator, which is one use case of PCIe endpoint mode. SKMM means Secure Key Management Module. In this document, SKMM only means the core does NOT boot from internal SDRAM. In PKCAL mode, the board holds and wait for the host to load the image into its internal SDRAM, then the host, which is released, lets the board to run.

In SKMM mode, the board runs the u-boot and loads the Linux by itself while working in PCIe end point mode. The host can communicate with PCIe interface. The PKCAL/SKMM mode can be enabled by setting the SW8[8] switch. To enable the PKCAL mode, set SW8[8] to 0. To enable the SKMM mode, set SW8[8] to 1.

The PKCAL mode can be started from the PCIe endpoint mode. In PKCAL mode, a C29x processor only uses internal SDRAM, instead of DDR3/NOR flash/NAND flash. Therefore, DDR/NOR flash/NAND flash should not be initialized in the source code. In addition, the PKCAL mode requires the board to be booted from the PCIe slot.

For the PKCAL use case, the device boots from either the internal SRAM or the L2 SRAM. The device is in PCI Express agent mode and e500 is in boot hold off. Following are the device configurations required for this use case by setting DIP switch:

- `SW7[1] ON, cfg_cpu_boot=0`
- `SW7[5] ON, cfg_host_agt=0`
- `SW8[8] ON, test_sel_b=0`

By default, the C29x PCIe card works in the SKMM mode.

4.2 Standalone Host Mode

The figure below shows the C29x PCIe card, operating in standalone host mode (without the host computer).

Figure 3. Standalone host mode

Perform the following steps to use the C29x PCIe card in the standalone host mode:

1. Remove PCIe bracker, and install four plastic posts.
2. Connect heat sink fan power line to J18.
3. Connect 12V power supply.
4. Connect RJ45 console line to UART port.
5. Configure the serial port of the attached computer with the following values:
   - Data rate: 115200 bps
   - Number of data bits: 8
   - Parity: None
   - Number of stop bits: 1
   - Flow control: Hardware/None
6. Connect network cable to TSEC1.
7. Turn on ON/OFF switch to power on C29x PCIe. Now, you will see C29x PCIe boot up message on the computer console.

Following are the device configurations required for this use case by setting DIP switch:

- `SW7[1] OFF, cfg_cpu_boot=1`

The table below shows DIP switch settings of the C29x PCIe board in the standalone mode (800 MHz core, 400 MHz platform, PCIe-x4 configuration). As compared to the end point mode, in the standalone mode, SW7[1] is in OFF position.

Table 3. Standalone host mode DIP switch settings

SW4[1..8]	0101 1000	ON OFF ON OFF OFF ON ON ON
SW5[1..8]	1111 0000	OFF OFF OFF OFF ON ON ON ON
SW6[1..8]	0000 1111	ON ON ON ON OFF OFF OFF OFF
SW7[1..8]	1001 1000	**OFF** ON ON OFF ON OFF OFF OFF

Table continues on the next page...

Table 3. Standalone host mode DIP switch settings (continued)

SW8[1..8]	0000 1011	ON ON ON ON OFF ON OFF OFF

5 Secure Boot Mode

C29x supply Secure Boot mode to protect customer system. The customer can put secured encoded u-boot and other image into flash, and put the secure key into C29x silicon. If secured keys are matched, the image can be loaded and run. The secure boot mode can be enabled by setting SW7[6] to ON and cfg_sb_dis to 0.

The secure boot mode can be started either from the PCIe endpoint mode or the standalone host mode. To start the secure boot mode from the PCIe endpoint mode or standalone host mode:

1. Connect J15 (pin 1) and J16 (pin 2).
2. Install the battery. The board now supports secure boot from NOR flash, NAND flash, or SPI EEPROM.

Following are the device configurations required for this use case by setting DIP switch:

```
SW7[6] = ON, cfg_sb_dis = 0
```

The table below lists the secure boot POR settings example in NOR flash boot mode (800 MHz core, 800 MHz DDR, PCIe-x4 agent).

Table 4. Secure POR settings

SW4[1..8]	0101 1000	ON OFF ON OFF OFF ON ON ON
SW5[1..8]	1111 1000	OFF OFF OFF OFF OFF ON ON ON
SW6[1..8]	0000 1111	ON ON ON ON OFF OFF OFF OFF
SW7[1..8]	0001 0011	ON ON ON OFF ON ON OFF OFF
SW8[1..8]	0000 1011	ON ON ON ON OFF ON OFF OFF

6 POR Configuration

The C29x PCIe card has user selectable switches or registers, for evaluating different frequency and boot configuration for this device. The table below shows how POR configuration is done through switches.

Table 5. POR configuration through switches

Switch	POR configuration	Signal name	Default setting	Signal meaning	Settings
SW4[1]	cfg_sys_pll[0]	IFC_AD0	ON	Rate between SYSCLK input and CCB clock (platform clock)	SW4[1,2,3] ON ON ON (000): 4:1
SW4[2]	cfg_sys_pll[1]	IFC_AD1	OFF		SW4[1,2,3] ON ON OFF (001): 5:1
SW4[3]	cfg_sys_pll[2]	IFC_AD2	ON		SW4[1,2,3] ON OFF ON (010): 6:1 Others are reserved. SYSCLK on this board is 66.67 MHz.

Table continues on the next page...

Table 5. POR configuration through switches (continued)

Switch	POR configuration	Signal name	Default setting	Signal meaning	Settings
SW4[4]	cfg_sys_speed	READY	OFF		0: SYSCLK frequency is at or below 66 MHz. 1: SYSCLK frequency is above 66 MHz.
SW4[5]	cfg_core_pll[0]	IFC_AD3	OFF	Ratio between the e500 core clock and e500 CCB clock	SW4[5,6,7] ON ON ON (000): reserved SW4[5,6,7] ON ON OFF (001): reserved SW4[5,6,7] ON OFF ON (010): 1:1 SW4[5,6,7] ON OFF OFF (011): 1.5:1 SW4[5,6,7] OFF ON ON (100): 2:1 SW4[5,6,7] OFF ON OFF (101): 2.5:1 SW4[5,6,7] OFF OFF ON (110): 3:1 SW4[5,6,7] OFF OFF OFF (111): 3.5:1
SW4[6]	cfg_core_pll[1]	IFC_AD4	ON		
SW4[7]	cfg_core_pll[2]	IFC_AD5	ON		
SW4[8]	cfg_core_speed	IFC_AD6	ON	Core speed configuration input	ON(0): Core clock frequency is greater than or equal to 600 MHz and less than 1001 MHz. OFF(1): Core clock frequency is greater than or equal to 1001 MHz and less than 1201 MHz.
SW5[1]		SW_CFG_ROM_LOC0	OFF	Boot ROM location	SW5[1:4] ON ON OFF ON (0010): 8b NAND (8k page size) SW5[1:4] ON OFF ON ON (0100): DDR controller SW5[1:4] ON OFF OFF ON (0110): SPI SW5[1:4] ON OFF OFF OFF (0111): SDHC (SD/MMC) SW5[1:4] OFF ON ON ON (1000): 8b NAND (512 page size) SW5[1:4] OFF ON ON OFF (1001): 8b NAND (2k page size) SW5[1:4] OFF ON OFF OFF (1011): 8b NOR SW5[1:4] OFF OFF ON ON (1100): 16b NAND (512 page size) SW5[1:4] OFF OFF ON OFF (1101): 16b NAND (2k page size) SW5[1:4] OFF OFF OFF OFF (1111): 16b NOR Others are reserved.
SW5[2]		SW_CFG_ROM_LOC1	OFF		
SW5[3]		SW_CFG_ROM_LOC2	OFF		
SW5[4]		SW_CFG_ROM_LOC3	OFF		
SW5[5]		VCORE_MGN	ON		ON(0): CPU will run at v1.0 core voltage

Table continues on the next page...

Table 5. POR configuration through switches (continued)

Switch	POR configuration	Signal name	Default setting	Signal meaning	Settings
					OFF(1): CPU will run at v1.05 core voltage
SW5[6]		BOOT_FLASH_SEL	ON	CS0/1 select	ON (0): CS0 is connected to NOR flash; CS1 is connect to NAND flash. OFF (1): CS0 is connected to NAND flash; CS1 is connected to NOR flash.
SW5[7]		FBANK_SEL1	ON	NOR boot section choose	Set which section works as a boot section.
SW5[8]		FBANK_SEL2	ON		
SW6[1]	cfg_ddr_pll[0]	SW_TSEC1_TXD0	ON	Clock ratio between 100 MHz OSC clock input and DDR complex clock	SW6[1,2,3] ON ON ON (000): 8:1 SW6[1,2,3] ON ON OFF (001): 10:1 SW6[1,2,3] ON OFF ON (010): 12:1
SW6[2]	cfg_ddr_pll[1]	SW_TSEC1_TXD1	ON		
SW6[3]	cfg_ddr_pll[2]	UART1_RTS_N	ON		
SW6[4]	cfg_ddr_speed[0]	1588_CLK_OUT	ON	DDR complex speed configuration input	cfg_ddr_speed[0]: ON (0): DDR data rate is less than 967 MHz. OFF (1): DDR data rate is greater than or equal to 967 MHz. cfg_ddr_speed[1]: ON (0): When cfg_ddr_speed[0]=1 and cfg_ddr_pll=10 OFF (1): When cfg_ddr_speed[0]=0 and (cfg_ddr_pll=8 or cfg_ddr_pll=10 or cfg_ddr_pll>=12) or when cfg_ddr_speed[0]=1 and cfg_ddr_pll>=12 NA: When cfg_ddr_speed[0]=1 and cfg_ddr_pll=8
SW6[5]	cfg_ddr_speed[1]	1588_PULSE_OUT	OFF		
SW6[6]	cfg_plat_speed	IFC_PAR1	OFF	Platform speed configuration input	ON (0): Platform clock frequency is greater than or equal to 267 MHz and less than 320 MHz. OFF (1): Platform clock frequency is greater than or equal to 320 MHz and less than 401 MHz.
SW6[7]	cfg_boot_seq[0]	IFC_A26	OFF	Boot sequencer configuration options	SW6[7,8] ON OFF (01): Normal I^2C addressing mode is used. Boot sequencer is enabled and loads configuration information from a ROM on the I^2C1 interface. A valid ROM must be present. SW6[7,8] OFF ON (10): Extended I^2C addressing mode is used. Boot sequencer is enabled and loads
SW6[8]	cfg_boot_seq[1]	IFC_A19	OFF		

Table continues on the next page...

Table 5. POR configuration through switches (continued)

Switch	POR configuration	Signal name	Default setting	Signal meaning	Settings
					configuration information from a ROM on the I^2C1 interface. A valid ROM must be present. SW6[7,8] OFF OFF (11): Boot sequencer is disabled. No I^2C ROM is accessed. This is the default setting.
SW7[1]	cfg_cpu_boot	DMA_DDONE0_N	OFF	CPU boot configuration inputs	ON (0): CPU boot hold off mode. The e500 core is prevented from booting until configured by an external master. OFF (1): The e500 core is allowed to boot without waiting for configuration by an external master.
SW7[2]	cfg_io_port[0]	IFC_AD13	ON	Different I/O ports active on the SerDes	SW7[2:4] ON ON ON (000): PCIe-x4 (5 GHz) SW7[2:4] ON ON OFF (001): PCIe-x4 (2.5 GHz) SW7[2:4] ON OFF ON (010): PCIe-x2 (5 GHz) SW7[2:4] ON OFF OFF (000): PCIe-x2 (2.5 GHz) SW7[2:4] OFF ON ON (100): PCIe-x1 (5 GHz) SW7[2:4] OFF ON OFF (101): PCIe-x1 (2.5 GHz) SW7[2:4] OFF OFF ON (110): Reserved SW7[2:4] OFF OFF OFF (111): Disabled
SW7[3]	cfg_io_port[1]	IFC_AD14	ON		
SW7[4]	cfg_io_port[2]	IFC_BCTL	OFF		
SW7[5]	cfg_host_agt	IFC_A23	ON	Host/agent reset configuration input	ON (0): Agent on PCI express interface OFF (1): Host/RC on PCI express interface
SW7[6]	cfg_sb_dis	HRESET_REQ_N	OFF		ON(0): Secure boot enabled OFF(1):Secure boot disabled (POR)
SW7[7]	cfg_svr[0]	SW_TSEC2_TXD0	OFF	To be used in the future to control functionality.	SW7[7:8] OFF OFF (11): Default operation Others are reserved.
SW7[8]	cfg_svr[1]	CKSTP_OUT_N	OFF		
SW8[1]	cfg_gpinput[0]	IFC_AVD	ON	General-purpose POR configuration vector to be placed in GPPORCR	Software can then use this value to inform the operating system about initial system configuration. Typical interpretations include circuit board type, board ID number, or a list of available peripherals.
SW8[2]	cfg_gpinput[1]	IFC_WE_N	ON		
SW8[3]	cfg_gpinput[2]	IFC_CLE	ON		
SW8[4]	cfg_gpinput[3]	UART0_TXD	ON		
SW8[5]	cfg_eng_use[0]	SW_EC_MDC	OFF	To be used in the future to control functionality.	SW8[5:6] OFF OFF (11): Default operation

Table continues on the next page...

Table 5. POR configuration through switches (continued)

Switch	POR configuration	Signal name	Default setting	Signal meaning	Settings
SW8[6]	cfg_eng_use[1]	UART0_RTS_N	ON		Others are reserved.
SW8[7]			OFF		
SW8[8]		TEST_SEL_N	OFF	PKCAL/SKMM mode	ON (0): PKCAL mode OFF (1): SKMM mode

The table below shows the POR configuration through registers.

Table 6. POR configuration through registers

Register	POR configuration	Signal name	Default setting	Signal meaning	Setting
R247	cfg_dram_type	1588_ALARM_OUT	NC (1)	Different voltage level from DDR3L	0: DDR3L 1.35V, CKE low at reset 1: DDR3 1.5V, CKE low at reset
R248	cfg_ddr_pll_backup	TSEC1_TXD3	NC (1)		0: Disabled 1: Enabled
R249	cfg_ddr_half_full_mode	TSEC2_TXD1	NC (1)		0: Half mode 1: Full mode
R250	cfg_ec1_prtc	TSEC2_TXD3	NC (1)	Ethernet interface mode	0: RMII 1: RGMII
R251	cfg_ec2_prtc	TSEC1_TXD2	NC (1)	Ethernet interface mode	0: RMII 1: RGMII
R265	cfg_ifc_pb[0]	IFC_AD9	NC (1)	Corresponding pages per block if NAND flash is used for booting	000: Reserved 001: 2k pages per block 010: 1k pages per block 000: 512 pages per block 100: 256 pages per block 101: 128 pages per block 110: 64 pages per block 111: 32 pages per block
R266	cfg_ifc_pb[1]	IFC_AD10	4.7k (0)		
R267	cfg_ifc_pb[2]	IFC_AD11	NC (1)		
R273	cfg_ifc_ecc_mode[0]	IFC_A25	NC (1)	IFC ECC correction mode if NAND flash is used for booting	00: 4b correction per 520 Byte sector 01: 8b correction per 520 Byte sector 10: 24b correction per 520 Byte sector 11: 40b correction per 520 Byte sector
R274	cfg_ifc_ecc_mode[1]	IFC_A27	NC (1)		

Table continues on the next page...

Table 6. POR configuration through registers (continued)

Register	POR configuration	Signal name	Default setting	Signal meaning	Setting
R275	cfg_ifc_ecc_dec_en	IFC_A21	4.7k (0)	Enable IFC ECC checking on boot if NAND flash is used for booting	0: ECC decoding disabled 1: ECC decoding enabled
R276	cfg_ifc_flash_mode	IFC_A22	NC (1)	Type of NOR/ NAND flash used for booting	0: For NOR, multiplexed NOR flash (AVD type); for NAND, bad block indicator is at page 0 and at last page of each block. 1: For NOR, normal asynchronous NOR flash; for NAND, bad block indicator is at page 0 and page 1 of each block.
R277	cfg_ifc_adm_mode	IFC_AD15	NC (1)	Which address bits are multiplexed with IFC data if NOR flash is used for booting	0: Lower order address bits are multiplexed with data on IFC_AD[0:15] 1: Higher order address bits are multiplexed with data on IFC_AD[0:15]
R278	cfg_ifc_te	IFC_TE	NC (1)		IFC transciever enabled
R279	cfg_srds_refclk	IFC_AD12	NC (1)	Input SerDes reference clock	0: SerDes expects a 125 MHz reference clock frequency. 1: SerDes expects a 100 MHz reference clock frequency. This is the default value.
R493	cfg_srds_pll_timeout_en	ASLEEP	NC (1)	Enable SerDes PLL timeout	0: Disabled 1: Enabled
R280	cfg_por_bist	IFC_OE_N	NC (1)		
R284	cfg_fuse_rd_en	IFC_PAR0	NC (1)	Enable security fuse read	0: Disabled 1: Enabled
R286	cfg_test_port_mux_sel	UART1_TXD	NC (1)	Test port MUX select	0: Not selected 1: Selected
R287	cfg_test_port_dis	IFC_WP_N	NC (1)	Disable test port	1: Disabled 0: Enabled
R289	cfg_60x	TSEC2_TXD2	NC (1)		
R290	cfg_pcc_drowsy_en	IFC_A20	4.7k (0)	Enable PPC drowsy	0: Disabled 1: Enabled
R425	cfg_sdram_drawsy_en	IRQ_OUT_N	4.7k (0)	Enable SDRAM drowsy	0: Disabled 1: Enabled

In the above tables, ON indicates 0 and OFF indicates 1.

7 POR Settings for different Boot Modes

This section provides POR settings for three boot modes: NOR flash, NAND flash, and SPI flash.

7.1 NOR Flash POR Settings

The table below shows default POR settings in NOR flash boot mode (800 MHz core, 800 MHz DDR, PCIe-x4 agent).

Table 7. NOR flash POR settings for 800 MHz core

SW4[1..8]	0101 1000	ON OFF ON OFF OFF ON ON ON
SW5[1..8]	1111 0000	OFF OFF OFF OFF ON ON ON ON
SW6[1..8]	0000 1111	ON ON ON ON OFF OFF OFF OFF
SW7[1..8]	1001 0111	OFF ON ON OFF ON OFF OFF OFF
SW8[1..8]	0000 1011	ON ON ON ON OFF ON OFF OFF

The table below shows POR settings in NOR flash boot mode for 1.2 GHz core (1.2 GHz core, 800 MHz DDR, PCIe-x4 agent).

Table 8. NOR flash POR settings for 1.2 GHz core

SW4[1..8]	0101 1101	ON OFF ON OFF OFF OFF ON OFF
SW5[1..8]	1111 0000	OFF OFF OFF OFF ON ON ON ON
SW6[1..8]	0000 1111	ON ON ON ON OFF OFF OFF OFF
SW7[1..8]	1001 0111	OFF ON ON OFF ON OFF OFF OFF
SW8[1..8]	0000 1011	ON ON ON ON OFF ON OFF OFF

NOR flash POR DIP settings:

- SW5[1..4] = 1111 (16-bit NOR)
- SW5[6] =0 (CS0 is connected to NOR)

7.2 NAND Flash POR Settings

The table below shows POR settings in NAND flash boot mode (800 MHz core, 800 MHz DDR, PCIe-x4 agent).

Table 9. NAND flash POR settings

SW4[1..8]	0101 1000	ON OFF ON OFF OFF ON ON ON
SW5[1..8]	0010 0100	**ON ON OFF ON** ON **OFF** ON ON
SW6[1..8]	0000 1111	ON ON ON ON OFF OFF OFF OFF
SW7[1..8]	1001 0111	OFF ON ON OFF ON OFF OFF OFF
SW8[1..8]	0000 1011	ON ON ON ON OFF ON OFF OFF

NAND flash POR DIP setting:

- SW5[1..4]=0010 (8-bit with 8k page size NAND flash)
- SW5[6]=1 (CS0 is connected to NAND)

7.3 SPI Flash POR Settings

The table below shows POR settings in SPI flash boot mode (800 MHz core, 800 MHz DDR, PCIe-x4 agent).

Table 10. SPI flash POR settings

SW4[1..8]	0101 1000	ON OFF ON OFF OFF ON ON ON
SW5[1..8]	0110 0100	**ON OFF OFF ON** ON **OFF** ON ON
SW6[1..8]	0000 1111	ON ON ON ON OFF OFF OFF OFF
SW7[1..8]	1001 0111	OFF ON ON OFF ON OFF OFF OFF
SW8[1..8]	0000 1011	ON ON ON ON OFF ON OFF OFF

SPI flash POR DIP setting:

- SW5[1..4]=0110

8 Programming U-Boot on a Board having no U-Boot Installed

To program u-boot on a board that has no u-boot installed:

1. Select standalone mode.
2. Connect 110V/220 power to power adapter.
3. Connect RJ45 console line to UART port.
4. Connect network cable into Ethernet port.
5. Connect USB TAP to JTAG port on C29x PCIe.
6. Open the CodeWarrior IDE for Power Architecture, and select P1010 processor with `P1010RDB_core_init.tcl`.
7. Power on board and attach C29x PCIe.
8. Memory download `images/spiimage`, available on C29x PCIe DVD, to 0x10FFFC00.
9. Set the computer to 0x1107f000, and run. When u-boot outputs message on console, enter u-boot environment.
10. Set u-boot variables, such as `ipaddr`, `serverip`, `ethaddr`, and `eth1addr`.
11. Perform one of the following steps to program NOR/NAND/SPI u-boot:
    a. Use the following commands to program NOR u-boot:

```
tftp 1000000 u-boot.bin
protect off all
erase eff80000 effffff
cp.b 1000000 eff80000 80000
```

    b. Use the following commands to program NAND u-boot:

```
tftp 1000000 u-boot-nand.bin
nand erase 0 80000
nand write 1000000 0 80000
```

    c. Use the following commands to program SPI u-boot:

```
sf probe 0
sf erase 0 80000
sf write 11000000 0 80000
```

9 Appendix

This section lists the `C29x_core_init.tcl` script for C29x PCIe card.

C29x_core_init.tcl

```
#####################################################################
# Initialization file for P1010RDB board
variable CCSRBAR 0xff700000

proc CCSR {reg_off} {
      global CCSRBAR
      return p:0x[format %x [expr {$CCSRBAR + $reg_off}]]
}

proc apply_e500v2_workaround {} {
      # After reset, e500 cores need to run before being able to enter debug mode.
      # Work-around: set a HW BP at reset address and run the core; after the core hits the
BP, it enters debug mode
      # e500 cores need to have valid opcode at the interrupt vector
      variable SPR_GROUP "e500 Special Purpose Registers/"
      ####################################################################
      # Set a breakpoint at the reset address
      reg ${SPR_GROUP}IAC1 = 0xfffffffc
      reg ${SPR_GROUP}DBCR0 = 0x40800000
      reg ${SPR_GROUP}DBCR1 = 0x00000000
      # Run the core
      config runcontrolsync off
      go
      wait 50
      config runcontrolsync on
      stop
      # Clear affected registers
      reg ${SPR_GROUP}DBSR  = 0x01CF0000
      reg ${SPR_GROUP}DBCR0 = 0x41000000
      reg ${SPR_GROUP}IAC1  = 0x00000000
      reg ${SPR_GROUP}CSRR0 = 0x00000000
      reg ${SPR_GROUP}CSRR1 = 0x00000000
}

proc init_P1010 {} {
      global CCSRBAR
      variable SPR_GROUP "e500 Special Purpose Registers/"
      variable SPR "e500 Special Purpose Registers/"
      variable SSPR "Standard Special Purpose Registers/"
      variable GPRS "General Purpose Registers/"
      # get ROM_LOC from PORBMSR
      variable ROM_LOC       0x[format %x [expr {([mem [CCSR 0xE0004] -np] & 0x0f000000) >>
24}]]
      ####################################################################
      # CCSRBAR
      # bit 8 - 23 - BASE_ADDR
      mem [CCSR 0x0] = 0x000ff700
      set CCSRBAR 0xff700000

      ####################################################################
      # invalidate BR0
      # CSPR0
      mem [CCSR 0x1E010] = 0x00000100
      # ABIST off
      # L2ERRDIS[MBECCDIS]=1 L2ERRDIR[SBECCDIS]=1
      mem [CCSR 0x20E44] = 0x0000000C
      # activate debug interrupt and enable SPU
      reg ${SSPR}MSR = 0x02000200
```

```
      ######################################################################
      #      Memory Map
      variable CAM_GROUP "regPPCTLB1/"
      # MMU  initialization
      # define 1MB    TLB1 entry 2: 0xE0000000 - 0xE00FFFFF; for CCSR Space, cache
inhibited, guarded
      reg ${CAM_GROUP}L2MMU_CAM1 = 0x500003CA1C080000FF700000FF700001
      # define 64M   TLB1 entry 3: 0xFC000000 - 0xFFFFFFFF; for Local Bus, cache inhibited,
guarded
      reg ${CAM_GROUP}L2MMU_CAM2 = 0x800007CA1C080000FC000000FC000001
      # define 1GB    TLB1 entry 8: 0x00000000 - 0x3FFFFFFF; DDR
      reg ${CAM_GROUP}L2MMU_CAM8 = 0xA0007FC01C08000000000000000000001
      #delete CAM0
      reg ${CAM_GROUP}L2MMU_CAM0 = 0x00000000000000000000000000000000
      #delete CAM1
      reg ${CAM_GROUP}L2MMU_CAM3 = 0x00000000000000000000000000000000
      #delete CAM10
      reg ${CAM_GROUP}L2MMU_CAM4 = 0x00000000000000000000000000000000
      reg ${CAM_GROUP}L2MMU_CAM5 = 0x00000000000000000000000000000000
      reg ${CAM_GROUP}L2MMU_CAM6 = 0x00000000000000000000000000000000
      reg ${CAM_GROUP}L2MMU_CAM7 = 0x00000000000000000000000000000000
      reg ${CAM_GROUP}L2MMU_CAM9 = 0x00000000000000000000000000000000
      reg ${CAM_GROUP}L2MMU_CAM10 = 0x00000000000000000000000000000000
      #delete CAM11
      reg ${CAM_GROUP}L2MMU_CAM11 = 0x00000000000000000000000000000000
      #delete CAM12
      reg ${CAM_GROUP}L2MMU_CAM12 = 0x00000000000000000000000000000000
      #delete CAM13
      reg ${CAM_GROUP}L2MMU_CAM13 = 0x00000000000000000000000000000000
      #delete CAM14
      reg ${CAM_GROUP}L2MMU_CAM14 = 0x00000000000000000000000000000000
      #delete CAM15
      reg ${CAM_GROUP}L2MMU_CAM15 = 0x00000000000000000000000000000000
      # disable & invalidate all core caches
      reg ${SPR_GROUP}L1CSR0 = 0x2
      reg ${SPR_GROUP}L1CSR1 = 0x0
      reg ${SPR_GROUP}L1CSR1 = 0x2
      # L2CTL
      # bit 0 = 0             - L2E: L2 disabled
      # bit 1 = 1                  - L2I: L2 flash invalidate
      mem [CCSR 0x20000] = 0x[format %x [expr {[mem [CCSR 0x20000] -np] & 0x7FFFFFFF |
0x40000000}]]

      ######################################################################
      #      Memory Windows
      # configure local access windows
      # LAWBAR0 - Local Bus
      # bit 8 - 31 = 0xFC000000 - base addr
      mem [CCSR 0xC08] = 0x000fc000
      # LAWAR0
      # bit 0 = 1 - enable window
      # bit 7-11 = 00100 - Local Bus
      # bit 26 - 31 =  011011 64M - size
      mem [CCSR 0xC10] = 0x80400019
      # LAWBAR11 - DDR
      # bit 8 - 31 = 0x00000000 - base addr
      mem [CCSR 0xD68] = 0x00000000
      # LAWAR11
      # bit 0 = 1 - enable window
      # bit 7-11 = 01111 - DDR
      # bit 26 - 31 =  011101 1G - size
      mem [CCSR 0xD70] = 0x80F0001d
      #disable LAW 7
      mem [CCSR 0xCE8] = 0x00000000
      mem [CCSR 0xCF0] = 0x00000000
      #disable LAW 8
      mem [CCSR 0xD08] = 0x00000000
      mem [CCSR 0xD10] = 0x00000000
      #disable LAW 9
      mem [CCSR 0xD28] = 0x00000000
```

```
      mem [CCSR 0xD30] = 0x00000000
      #disable LAW 10
      mem [CCSR 0xD48] = 0x00000000
      mem [CCSR 0xD50] = 0x00000000
      ########################################
      #      CPLD INIT
      config MemAccess 8
      config MemWidth 8
      mem v:0xefb0000a = 0x00
      mem v:0xefb00009 = 0x00
      mem v:0xefb00013 = 0x00
      config MemAccess 32
      config MemWidth 32
      ########################################
      #      DDRC INITIALIZATION

      # DDR_SDRAM_CFG
      mem [CCSR 0x2110] = 0x470c0000
      # CS0_BNDS
      mem [CCSR 0x2000] = 0x0000003F
      # CS0_CONFIG
      mem [CCSR 0x2080] = 0x80014302
      # TIMING_CFG_0
      mem [CCSR 0x2104] = 0x00110004
      # TIMING_CFG_1
      mem [CCSR 0x2108] = 0x5d59e544
      # TIMING_CFG_2
      mem [CCSR 0x210c] = 0x0fa890cd
      # TIMING_CFG_3
      mem [CCSR 0x2100] = 0x00010000
      # DDR_SDRAM_CFG_2
      mem [CCSR 0x2114] = 0x04401010
      # DDR_SDRAM_MODE
      mem [CCSR 0x2118] = 0x00441210
      # DDR_SDRAM_MODE_2
      mem [CCSR 0x211C] = 0x00000000
      # DDR_SDRAM_MD_CNTL
      mem [CCSR 0x2120] = 0x00000000
      # DDR_SDRAM_INTERVAL
      mem [CCSR 0x2124] = 0x0a280000
      # DDR_DATA_INIT
      mem [CCSR 0x2128] = 0xDEADBEEF
      # DDR_SDRAM_CLK_CNTL
      mem [CCSR 0x2130] = 0x03000000
      # TIMING_CFG_4
      mem [CCSR 0x2160] = 0x00000001
      # TIMING_CFG_5
      mem [CCSR 0x2164] = 0x03402400
      # DDR_ZQ_CNTL
      mem [CCSR 0x2170] = 0x89080600
      # DDR_WRLVL_CNTL
      mem [CCSR 0x2174] = 0x8655a608
      # ERR_INT_EN
      mem [CCSR 0x2E48] = 0x00000000
      # ERR_SBE
      mem [CCSR 0x2E58] = 0x00000000
      # DDRCDR_1
      mem [CCSR 0x2B28] = 0x00000000
      # DDRCDR_2
      mem [CCSR 0x2B2C] = 0x00000000
      #delay before enable
      wait 500
      # DDR_SDRAM_CFG
      mem [CCSR 0x2110] = 0xC70C0000
      #wait for DRAM data initialization
      wait 500
#SPI init
      # SPMODE
      mem [CCSR 0x7000] = 0x80000403
      # SPIM - catch all events
```

```
      mem [CCSR 0x7008] = 0x00000000
      # SPMODE1
      mem [CCSR 0x7020]  = 0x28170008
      ###################################################################
      # configure IFC controller
wait 2000
#             # CS0 - NOR Flash settings
#             # AMASK0 64M NOR
            mem [CCSR 0x1E0A0] = 0xFC000000
#             # CSOR0
            mem [CCSR 0x1E130] = 0x00008000
#             #FTIM0_CS0
            mem [CCSR 0x1E1C0] = 0x40050005
#             #FTIM1_CS0
            mem [CCSR 0x1E1C4] = 0x1e000f00
#             #FTIM2_CS0
            mem [CCSR 0x1E1C8] = 0x0410001c
#             #FTIM3_CS0
            mem [CCSR 0x1E1CC] = 0x00000000
#             # CSPR0
            mem [CCSR 0x1E010] = 0xFC000101
      ###################################################################
      # interrupt vectors initialization
      # IVPR (default reset value)
      reg ${SPR}IVPR = 0x00000000
      # interrupt vector offset registers
      # IVOR0 - critical input
      reg      ${SPR}IVOR0 = 0x00000100
      # IVOR1 - machine check
      reg      ${SPR}IVOR1 = 0x00000200
      # IVOR2 - data storage
      reg      ${SPR}IVOR2 = 0x00000300
      # IVOR3 - instruction storage
      reg      ${SPR}IVOR3 = 0x00000400
      # IVOR4 - external input
      reg      ${SPR}IVOR4 = 0x00000500
      # IVOR5 - alignment
      reg      ${SPR}IVOR5 = 0x00000600
      # IVOR6 - program
      reg      ${SPR}IVOR6 = 0x00000700
      # IVOR8 - system call
      reg      ${SPR}IVOR8 = 0x00000c00
      # IVOR10 - decrementer
      reg      ${SPR}IVOR10 = 0x00000900
      # IVOR11 - fixed-interval timer interrupt
      reg      ${SPR}IVOR11 = 0x00000f00
      # IVOR12 - watchdog timer interrupt
      reg      ${SPR}IVOR12 = 0x00000b00
      # IVOR13 - data TLB errror
      reg      ${SPR}IVOR13 = 0x00001100
      # IVOR14 - instruction TLB error
      reg      ${SPR}IVOR14 = 0x00001000
      # IVOR15 - debug
      reg      ${SPR}IVOR15 = 0x00001500
      # IVOR32 - SPE-APU unavailable
      reg      ${SPR}IVOR32 = 0x00001600
      # IVOR33 - SPE-floating point data exception
      reg      ${SPR}IVOR33 = 0x00001700
      # IVOR34 - SPE-floating point round exception
      reg      ${SPR}IVOR34 = 0x00001800
      # IVOR35 - performance monitor
      reg      ${SPR}IVOR35 = 0x00001900
      # put a valid opcode at debug and progrm exception vector address
      mem v:0x00000700 = 0x48000000
      mem v:0x00001500 = 0x48000000
      ###################################################################
      apply_e500v2_workaround
      # enable floating point
      reg ${SSPR}MSR = 0x02001200
      #############
```

```
      # time base enable & MAS7 update
      # HID0
      reg      ${SPR}HID0 = 0x00004080
      ######
      # CW debugger settings
      #Trap debug event enable
      reg      ${SPR}DBCR0 = 0x41000000
      # for debugging starting at program entry point when stack is not initialized
      reg      ${GPRS}SP = 0x0000000F
}

proc envsetup {} {
      # Environment Setup
#      radix x
      config hexprefix 0x
      config MemIdentifier v
      config MemWidth 32
      config MemAccess 32
      config MemSwap off
}
#-------------------------------------------------------------------
# Main
#-------------------------------------------------------------------
  envsetup
  init_P1010
```

10 Revision History

The table below provides revision history of this document.

Table 11. Revision history

Revision number	Date	Topic cross-reference	Change description
Rev. 1	03/2014	Secure Boot Mode	Updated steps to start secure boot mode on the PCIe endpoint mode and standalone host mode.
		POR Configuration	Updated the settings corresponding to SW6[1,2,3] and SW7[7:8] in the "POR configuration through switches" table.
		NOR Flash POR Settings	Removed NOR flash POR settings for 1.4 GHz core.
		Programming U-Boot on a Board having no U-Boot Installed	Updated the code in step 11(a).
Rev. 0	10/2013		Initial public release.

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number: C29xPCIeQS, REV1
Agile Number: 926-27781