Lattice Sentry 2.2 Mach-NX PFR and SFB Architecture User Guide

Technical Note

FPGA-TN-02359-1.0

April 2024

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUNDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Inclusive Language

This document was created consistent with Lattice Semiconductor’s inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice’s inclusive language FAQ 6878 for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.

Contents

Figures

Tables

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviations	Definition
AES	Advanced Encryption Standard
AHB	Advanced High-Performance Bus
AMBA	Advanced Microcontroller Bus Architecture
BMC	Baseboard Management Controller
CFG	Configuration Flash sector on Mach-NX device
CPU	Central Processing Unit
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
ECIES	Elliptic Curve Integrated Encryption Scheme
FIFO	First In First Out
FPGA	Field-Programmable Gate Array
GPIO	General Purpose Input/Output
HSP	High Speed Data Port
I^2C	Inter-Integrated Circuit
JTAG	Joint Test Action Group
MAC	Message Authentication Codes
MRoT	Main Root of Trust
OOB	Out-of-Band
PCH	Platform Controller Hub
PFR	Platform Firmware Resiliency
PIC	Programmable Interface Controller
PLD	Programmable Logic Device
QSPI	Quad Serial Peripheral Interface
RISC-V	Reduced Instruction Set Computer-V (five)
RoT	Root of Trust
RTD	Root of Trust for Detection
RTRec	Root of Trust for Recovery
RTU	Root of Trust for Update
Rx	Receiver
SFB	SoC Function Block
SHA	Secure Hash Algorithm
SoC	System on Chip
SPI	Serial Peripheral Interface
TRNG	True Random Number Generator
Tx	Transmitter
UFM	User Flash Memory

1. Introduction

The Mach™-NX device family is the next generation of Lattice Semiconductor low density programmable logic devices (PLD) including enhanced security features and a Platform Firmware Resiliency System on Chip (SoC) Function Block.

The enhanced security features include Advanced Encryption Standard (AES) AES-128/256, Secure Hash Algorithm (SHA) SHA-256/384, Elliptic Curve Digital Signature Algorithm (ECDSA), Elliptic Curve Integrated Encryption Scheme (ECIES), Hash Message Authentication Code (HMAC) HMAC-SHA256/384, Public Key Cryptography, True Random Number Generator (TRNG), and Unique Secure ID.

The Mach-NX device family is a Root of Trust hardware solution that can easily scale to protect the whole system with its enhanced bitstream security and user mode functions. With the Mach-NX device, you can implement a Platform Firmware Resiliency (PFR) solution in your system, as described in NIST Special Publication 800-193. The purpose of this document is to introduce the design methodology of the Mach-NX PFR solution using the Lattice Propel toolsets, which can largely reduce the design complexity.

1.1. PFR

NIST 800-193 Platform Firmware Resiliency (PFR) Guidelines describe the principles of supporting platform resiliency. As stated in NIST 800-193, the security guidelines are based on the following three principles:

- Protection – Mechanisms for ensuring that Platform Firmware code and critical data remain in a state of integrity and are protected from corruption, such as the process for ensuring the authenticity and integrity of firmware updates.
- Detection – Mechanisms for detecting when Platform Firmware code and critical data have been corrupted, or otherwise changed from an authorized state.
- Recovery – Mechanisms for restoring Platform Firmware code and critical data to a state of integrity in the event that any such firmware code or critical data are detected to have been corrupted, or when forced to recover through an authorized mechanism. Recovery is limited to the ability to recover firmware code and critical data.

1.2. RoT

The security mechanisms are founded in the Root of Trust (RoT). The RoT is an element that forms the basis of providing one or more security-specific functions, such as measurement, storage, reporting, recovery, verification, and update. The RoT must be designed to always behave in the expected manner because its proper functioning is essential to providing its security-specific functions and because its misbehavior cannot be detected. The RoT is typically just the first element in a Chain of Trust (CoT) and can serve as an anchor in such a chain to deliver more complex functionality.

The foundational guidelines on the Root of Trust support the subsequent guidelines for Protection, Detection, and Recovery. These guidelines are organized based on the logical component responsible for each of those security properties:

- The Root of Trust for Update (RTU) is responsible for authenticating firmware updates and critical data changes to support platform protection capabilities.
- The Root of Trust for Detection (RTD) is responsible for firmware and critical data corruption detection capabilities.
- The Root of Trust for Recovery (RTRec) is responsible for recovery of firmware and critical data when corruption is detected.

1.3. Lattice RoT Mechanism

The Mach-NX FPGA device can serve as the Root of Trust and provide the following services:

- Image Authentication – On system power-up or reset, the Mach-NX device holds the protected devices in reset while it authenticates their boot images in SPI flash. After each signature authentication passes, the Mach-NX device releases each reset, and those devices can boot from their authenticated SPI flash image. Image authentication can also be requested at any time through the I^2C Out-of-Band (OOB) communication path.
- Image Recovery – If a flash image becomes corrupted for any reason, it fails to be authenticated. The Mach-NX device can restore it to a known good state by copying from an authenticated backup image.
- SPI Flash Monitoring and Protection – The Mach-NX device can monitor multiple SPI/QSPI buses for unauthorized activity and block unauthorized accesses using external SPI quick switches. The monitors can be configured to watch for specific SPI flash commands and address ranges defined by the system designer and designate them as authorized or unauthorized.
- Event Logging – The Mach-NX device logs security events, such as unauthorized flash accesses and notifies the BMC (Baseboard Management Controller).
- I^2C Filtering – The Mach-NX device can monitor an I^2C bus for unauthorized activity and block unauthorized transactions by disabling the I^2C bus. The monitor can be configured with multiple allowlist or command list filters to watch for specific byte or bit patterns defined by the system designer and designate them as authorized or unauthorized I^2C transactions.

2. Functional Description

2.1. Overview

Figure 2.1 shows the architecture of the Mach-NX device. The system design consists of a RISC-V processor connected to a set of peripherals through the Advanced Microcontroller Bus Architecture (AMBA) bus. The software running on the processor controls the general and PFR solution peripherals and handles all the events at runtime to perform the system functionalities.

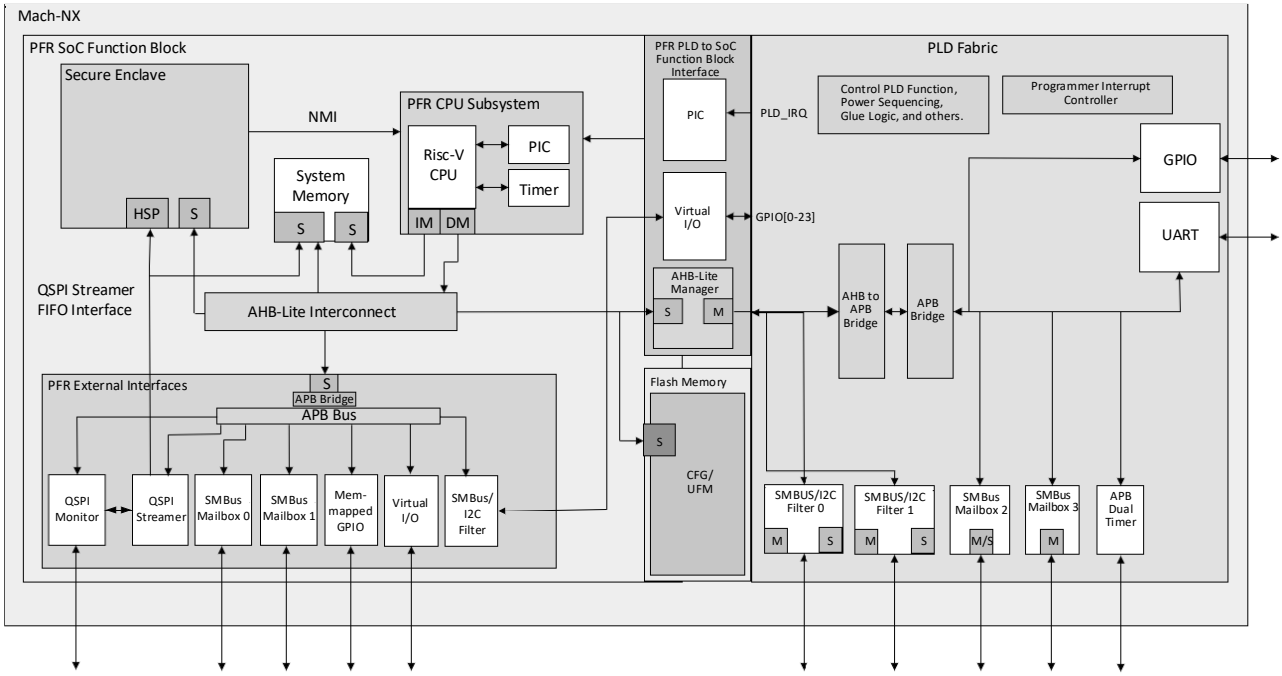

Figure 2.1. Mach-NX SoC Function Block Diagram

2.2. Module Descriptions

2.2.1. CPU Subsystem

The RISC-V Processor is based on the open source Vex RISC-V core, with integrated JTAG debugger, PIC, and Timer. The RISC-V core supports RV32I instruction set and 5-stage pipelines to fulfill the performance requirement for PFR system.

2.2.2. System Memory

The System Memory is a 176 KB dual port memory used for CPU code execution. One port connects directly to the instruction manager of the RISC-V CPU and the other port is connected to the AHB-Lite Interconnect. At boot up, the System Memory is loaded from the SPI Memory connected to QSPI Monitor 0 using the QSPI Streamer.

2.2.3. External Interface

2.2.3.1. QSPI Streamer

The QSPI Streamer is a programmable SPI controller that supports SPI and QSPI targets. QSPI Streamer incorporates an SPI FIFO controller that provides significant performance improvement by supporting data read and write transactions of programmable length, allowing an entire SPI flash device to be read in one SPI transaction. For processor access, it contains FIFOs for Tx and Rx data, which enable it to support full page SPI transactions of 256 bytes. For image

authentication, the external Rx FIFO interface is connected directly to the Security Subsystem, bypassing the AHB-Lite interconnect to allow faster image authentication and SPI read transactions up to 2 MB.

2.2.3.2. QSPI Monitor

The QSPI Monitor is a programmable security module that monitors up to two SPI, DSPI, or QSPI buses for unauthorized activity and blocks transactions by controlling the chip select signal and external quick switch devices. In addition to monitoring, it can connect external SPI/QSPI buses to the QSPI Streamer through a programmable mux/demux block. The QSPI Monitor watches the external buses for allowed flash commands and flash addresses. The QSPI monitor provides fine grain control over the set of allowed commands and supports up to eight configurable address spaces. These address spaces can be configured to as follows:

- all access – Read, Write, and Erase transactions are allowed;
- read access – allows Read transactions only;
- no access – Read, Write, and Erase transactions are blocked.

All non-defined addresses are read-only.

2.2.3.3. I^2C Filter

The I^2C Filter is a programmable security module that monitors traffic on an I^2C bus to identify unauthorized activity. When the I^2C Filter detects an unauthorized activity, the I^2C bus is disabled and the firmware is notified so that an event can be logged.

2.2.3.4. SMBus Mailboxes

There are two SMBus Mailboxes to provide Out-of-Band communication interfaces to the BMC and Platform Controller Hub (PCH).

2.2.3.5. GPIO

There are 16 Memory Mapped General Purpose I/O available which are controlled by the PFR CPU.

2.2.3.6. Virtual I/O

There are 24 virtual I/O available that are controlled by the PLD Fabric.

2.2.4. Secure Enclave

The Secure Enclave provides a set of security services for the Mach-NX device. The Secure Enclave has two interfaces for sending and receiving data: a register interface and a FIFO-based High Speed Data Port (HSP). The Secure Enclave provides the following major functions:

- Secure Hash Algorithm (SHA) — 256/384 bits
- Elliptic Curve Digital Signature Algorithm (ECDSA) — Generation and Verification
- Message Authentication Codes (MACs) — Hash-based MAC (HMAC)
- Elliptic Curve Diffie-Hellman (ECDH) Scheme
- Elliptic Curve Cryptography (ECC) Key Pair Generation — Public Key/Private Key
- Elliptic Curve Integrated Encryption Scheme (ECIES) Encryption/Decryption
- True Random Number Generator (TRNG)
- Advanced Encryption Standard (AES) — 128/256 bits
- Authentication controller for configuration engine
- AHB-Lite interface to user logic
- High Speed Port (HSP) for FIFO-based streaming data transfer
- Unique Secure ID

2.2.5. Flash Memory

The Mach-NX device provides a configuration flash sector (CFG)/user flash memory (UFM) block that can be used for a variety of applications including storing the PLD configuration image, initializing Embedded Block RAMs (EBR) to store PROM data, or as a general purpose user Flash memory.

2.2.6. PLD to SoC Function Block Interface

2.2.6.1. AHB-Lite Manager

The AHB-Lite Manager provides an interface for the RISC-V processor to manage customer logic in the PLD fabric.

2.2.6.2. Programmable Interrupt Control Interface

The Programmable Interrupt Control Interface provides the PLD fabric with eight interrupts to the RISC-V processor.

2.2.6.3. Virtual I/O Interface

The Virtual I/O Interface provides the PLD access to the PFR SoC Function Block's GPIO.

2.2.7. PLD Fabric

The PLD Fabric contains programmable logic available for design customization.

2.3. Signal Description

Table 2.1. PFR SoC Function Block External Interface

Signal	Direction	Description
QSPI Monitor		
QSPI_MONx_CLK	Bidir	SPI/QSPI Clock
QSPI_MONx_CSN_INTSW_MOSI	Output	• External Switch: Chip Select High Impedance during monitoring • Internal Switch: MOSI
QSPI_MONx_DIS_A	Output	Quick Switch Disable Flash A • 0=enabled • 1=disabled
QSPI_MONx_DIS_B	Output	Quick Switch Disable Flash B • 0=enabled • 1=disabled
QSPI_MONx_DQ0	Bidir	• SPI: MOSI • QSPI: serial data input and output
QSPI_MONx_DQ1	Bidir	• SPI: MISO • QSPI: serial data input and output

QSPI_MONx_DQ2_INTSW_FLASHB_MISO	Bidir	• External Switch:
		  • SPI: unused
		  • QSPI: serial data input and output
		  High Impedance during monitoring
		• Internal Switch: MISO for Flash B

QSPI_MONx_DQ3_INTSW_FLASHA_MISO	Bidir	• External Switch:
		  • SPI: unused
		  • QSPI: serial data input and output
		  High Impedance during monitoring
		• Internal Switch: MISO for Flash A

QSPI_MONx_PRE_CSN	Input	QSPI/SPI Chip select before quick switch
QSPI_MONx_RST_O	Output	Reset

Signal	Direction	Description

QSPI_MONx_SWI_EN_INTSW_CLK	Output	• External Switch:
		Quick Switch Output Enable
		  • 0=disabled
		  • 1=enabled
		This signal is enabled when the QSPI Monitor is protecting the SPI Flash and when the QSPI Monitor is switched to the internal controller.
		• Internal Switch: SPI Clock Out

QSPI_MONx_SWI_ISO	Output	Quick Switch Isolation • 0=disabled • 1=enabled This optional signal is used when a flash has switching logic to select between multiple SPI Controllers, such as BMC and PCH. This signal is enabled when the QSPI Monitor is switched to the internal controller.
I²C Filter		
SMB_filterx_sclm	Bidir	Clock connected to the controller device
SMB_filterx_sdam	Bidir	Data connected to the controller device
SMB_filterx_scls	Bidir	Clock connected to the target device(s)
SMB_filterx_sdas	Bidir	Data connected to the target device(s)
SMBus Mailbox		
SMB_ALERT_N	Output	Interrupt, active low
SMB_SCL	Bidir	Clock
SMB_SDA	Bidir	Data
Memory Mapped GPIO		
GPIO_MMxx	Bidir	16 General Purpose I/O
Virtual GPIO		
GPIO_xx	Bidir	24 General Purpose I/O

3. Hardware Considerations

3.1. SPI Monitoring

3.1.1. External Switching

The external hardware is required for the Mach-NX device to protect and access the firmware images in the QSPI Flash. A basic implementation is shown in Figure 3.1.

A switch is required for each QSPI signal to provide isolation from the CPU when the QSPI Monitor is blocking access for protection or accessing the firmware for authentication or recovery. The switches are controlled by the QSPI Monitor using the QSPI_MONx_SWI_EN. QSPI_MONx_PRE_CSN is required to monitor the CSn coming from the CPU before the switch. All other signals should monitor their corresponding signal after the switches.

For a dual flash configuration, the flash selection logic is controlled using OR gates and controlled by QSPI_MONx_DIS_A and QSPI_MONx_DIS_B. QSPI_MON0_DIS_A should be pulled down to ground with a 1 kΩ resistor, this allows the PFR firmware to be loaded from the BMC's primary flash. QSPI_MON1_DIS_A and QSPI_MONx_DIS_B should be pulled to VCC. For Single Flash configuration, the OR gates can be removed and QSPI_MONx_DIS_A/B are not used.

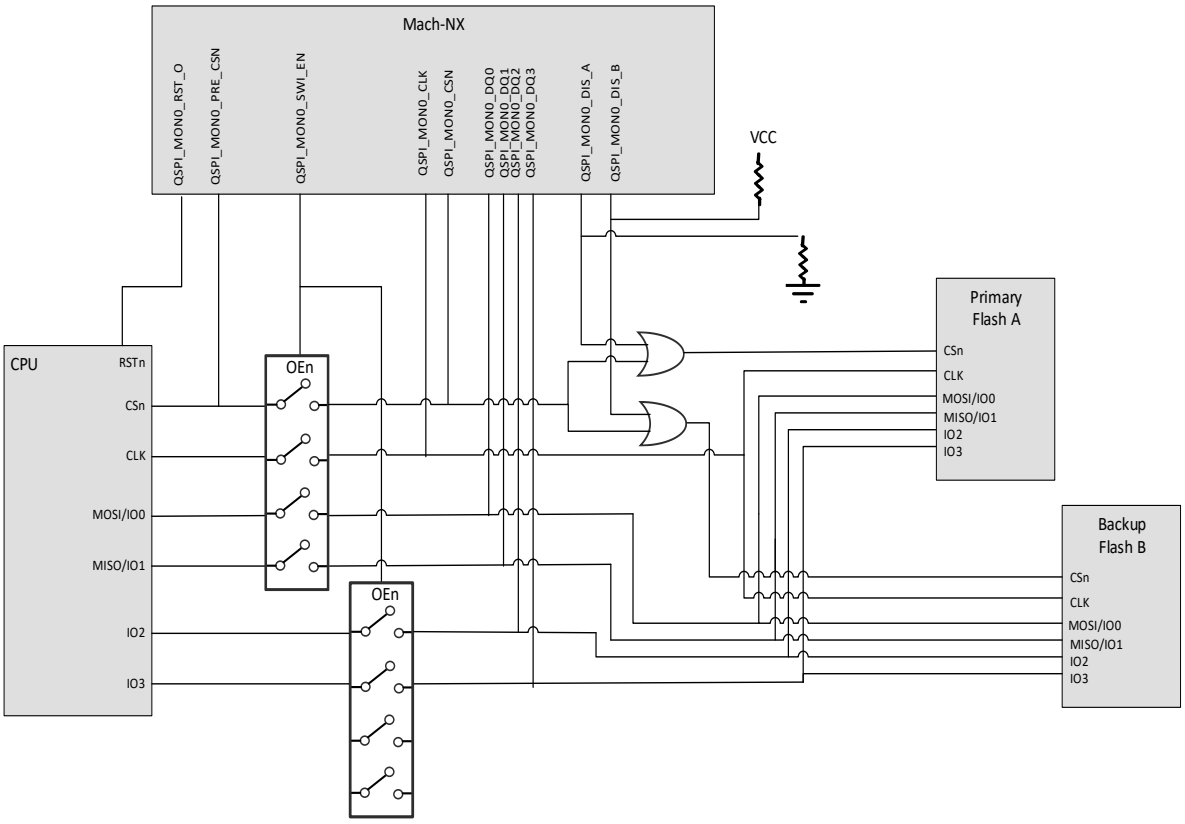

Figure 3.1. Dual Flash Configuration

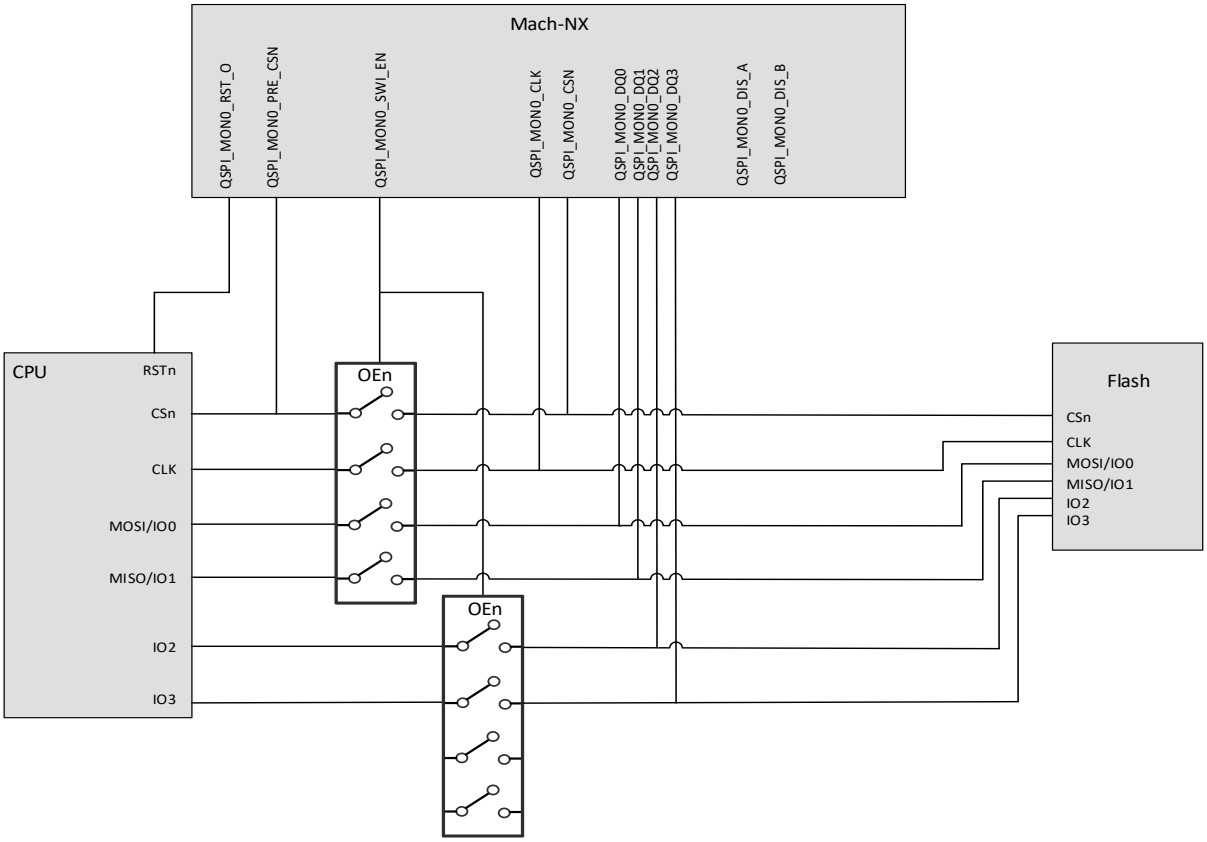

Figure 3.2. Single Flash Configuration

When the QSPI Flash is being accessed by multiple CPUs, a multiplexer is required to select the proper active CPU, as shown in Figure 3.3. If the multiplexer can tristate the output, the switch for the QSPI data signals can be removed by connecting QSPI_MONx_SWI_ISO to the OEn of the multiplexer, as shown in Figure 3.4.

Figure A.1 to Figure A.4 show the sample schematics of the single and dual flash configurations for the BMC and PCH.

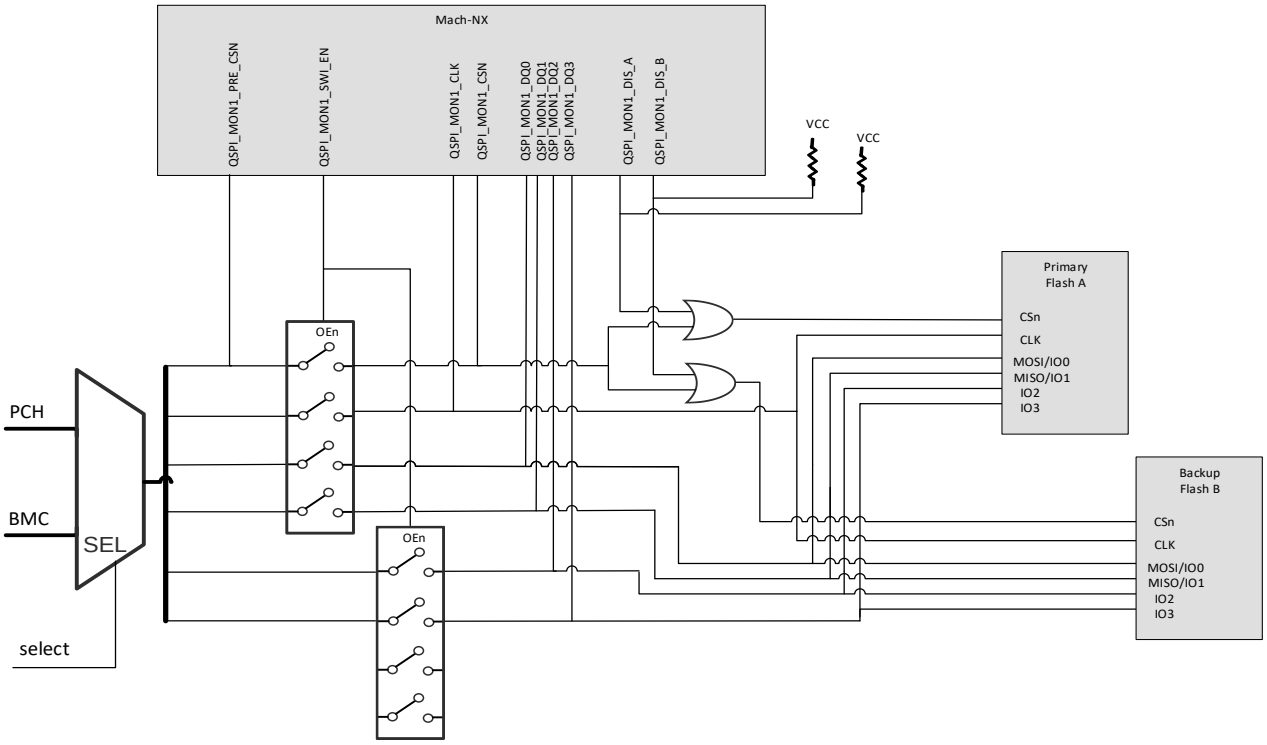

Figure 3.3. Multi-CPU Configuration

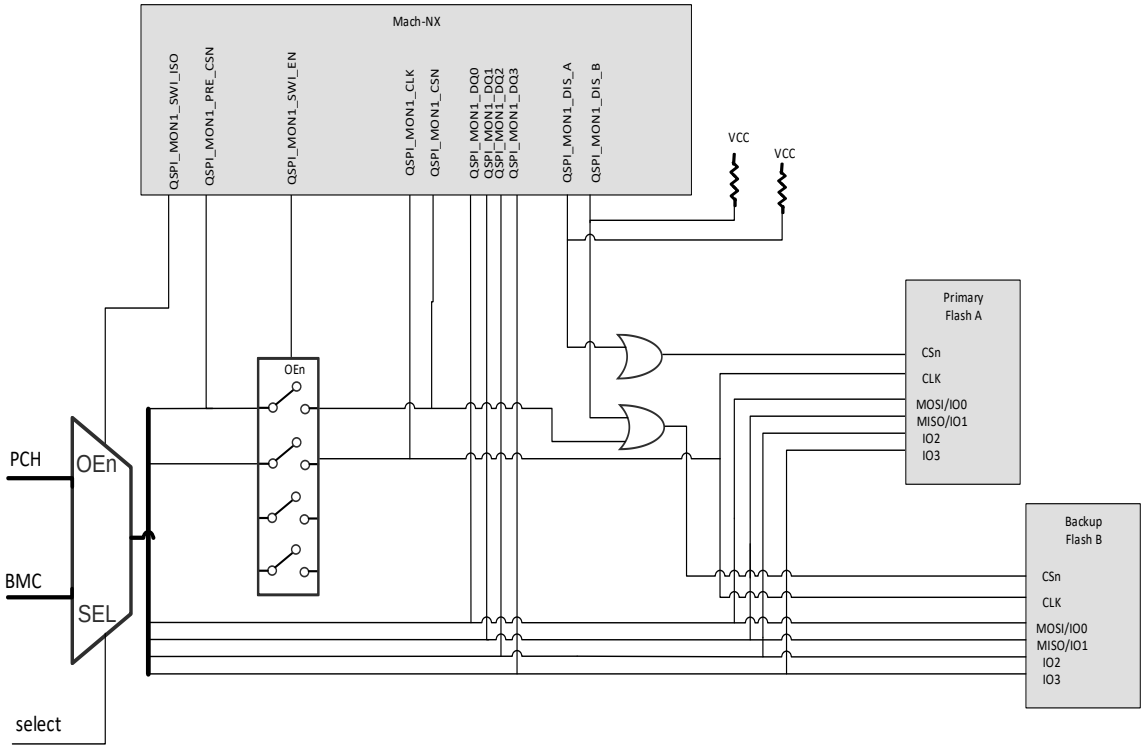

Figure 3.4. Alternate Multi-CPU Configuration

4. Boot Sequence

The Mach-NX device configures the FPGA from the CFG memory, and the SoC is configured from the primary SPI Flash connected to the QSPI Monitor0.

The following components are required for the SoC Function Block to boot properly:

- FPGA configuration – The FPGA image with the SoC Function Block Interface IP stored in CFG0 and/or CFG1 of the Mach-NX device. This image is created by Lattice Diamond software. The FPGA image can be signed using customer ECC256 private/public key pair.
- Flash Address Map – The Flash Address Map created by the Lattice Propel Flash Address tool is stored in the Flash Address Map UFM3 flash sector. The Flash Address Map is created using the Flash Address Tool in Lattice Propel design environment. The Flash Address Map needs to be configured with the primary and secondary location of the SoC Function Block Configuration Bitstream and the primary and secondary location of the PFR Firmware image.
- SoC Function Block Configuration Bitstream – The SoC Function Block Configuration Bitstream is a signed and encrypted configuration image used to configure the SFB. It is provided by Lattice after a system is built with the SoC Function Block. The SoC Function Configuration Bitstream should be programmed into the primary flash monitored by SPI Monitor0. The location of the primary and backup SoC Function Block Configuration Bitstream should be programmed into the Flash Address Map. The SoC Function Block is securely signed and encrypted using ECC256/AES256 with Lattice Keys.
- PFR Firmware Image – The PFR Firmware Image is the binary that runs on the RISC-V processor in the SoC Function Block. This image is provided after building the PFR software in the Propel SDK software. The location of the primary and backup PFR Firmware Image should be programmed into the Flash Address Map. The PFR Firmware image can be signed using the same customer ECC256 Private/Public Key pair as the FPGA configuration image.

  Note: When the customer public key is programmed into the Mach-NX device, both the FPGA configuration and PFR Firmware images must be signed.

The following is the boot sequence at power up and configuration:

1. FPGA is configured based on programmed boot sequence. If the public key is programmed, ECC256 ECDSA authentication takes place.
2. After the FPGA is configured properly, with the SoC Function Block Interface included in the FPGA image, the Mach-NX device reads the Flash Address Map UFM and retrieves primary/secondary address of SoC Function Block Configuration Bitstream and performs AES256 decryption and ECC256 ECDSA authentication using Lattice's keys. If authentication/decryption is successful, the SoC Function Block Configuration Bitstream is loaded into the SoC Function Block.
3. After the SoC Function Block is configured properly, the Mach-NX device reads Flash Address Map UFM3 and retrieves the primary/secondary address of PFR Firmware.
   a. If a public key is programmed into the Mach-NX device, ECC256 ECDSA authentication takes place. If authentication is successful, the SoC Function Block loads the PFR Firmware into the system memory of the SoC Function Block.
   b. If no public key is programmed into the Mach-NX device, the SoC Function Block loads the PFR Firmware into the system memory of the SoC Function Block.
4. RISC-V processor is released from reset and starts executing the PFR Firmware.

References

For more information, refer to:

- Lattice Propel 1.1 SDK User Guide (FPGA-UG-02115)
- Lattice Propel 1.1 Builder User Guide (FPGA-UG-02116)
- Lattice Diamond 3.12 User Guide
- Lattice Sentry Solution Stack web page
- Mach-NX Devices web page
- Boards, Demos, IP Cores, and Reference Designs for Mach-NX Devices
- Lattice Insights for Lattice Semiconductor training courses and learning plans

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

Appendix A. Schematic Diagrams

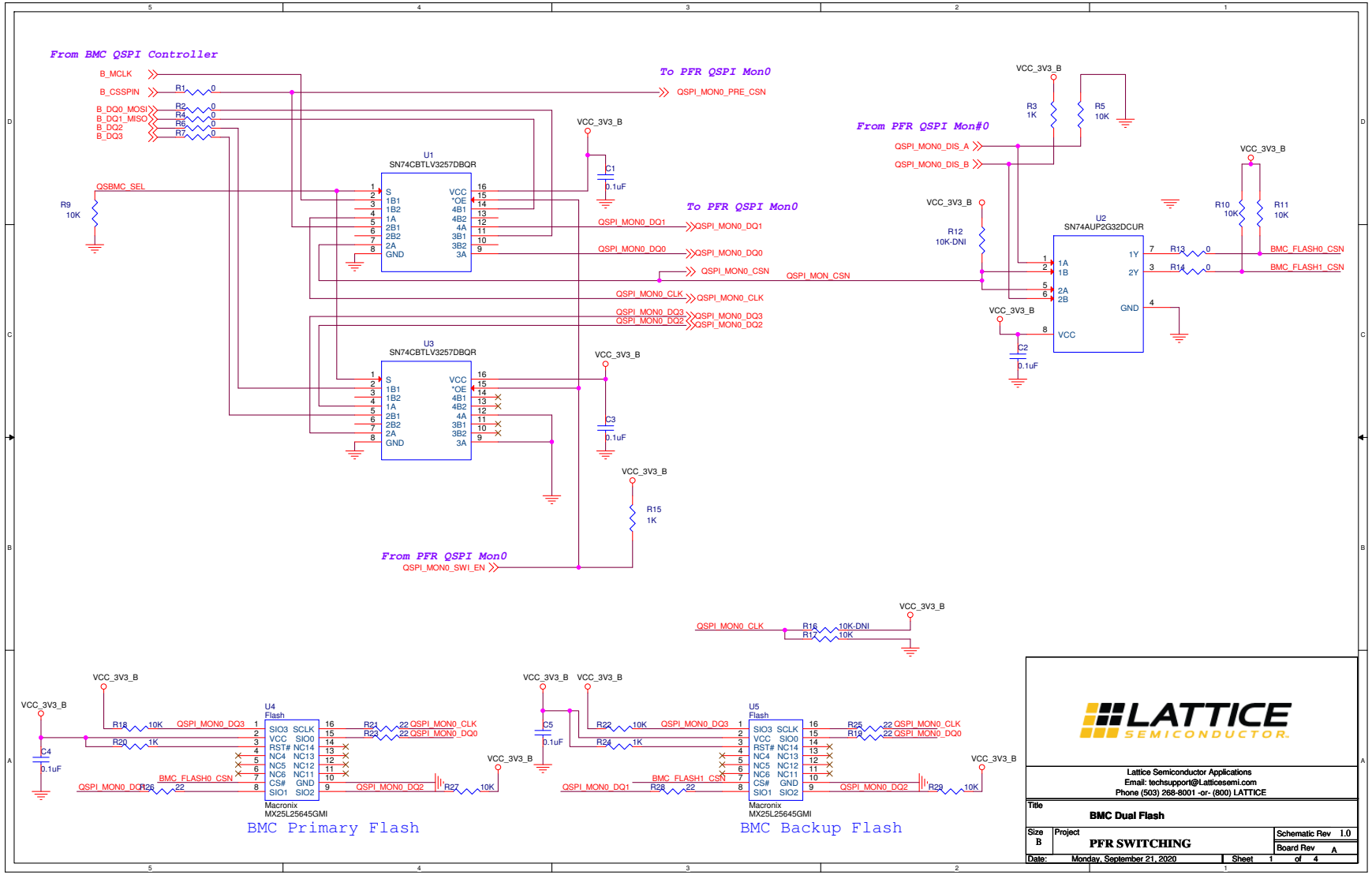

Figure A.1. BMC Dual Flash Schematic

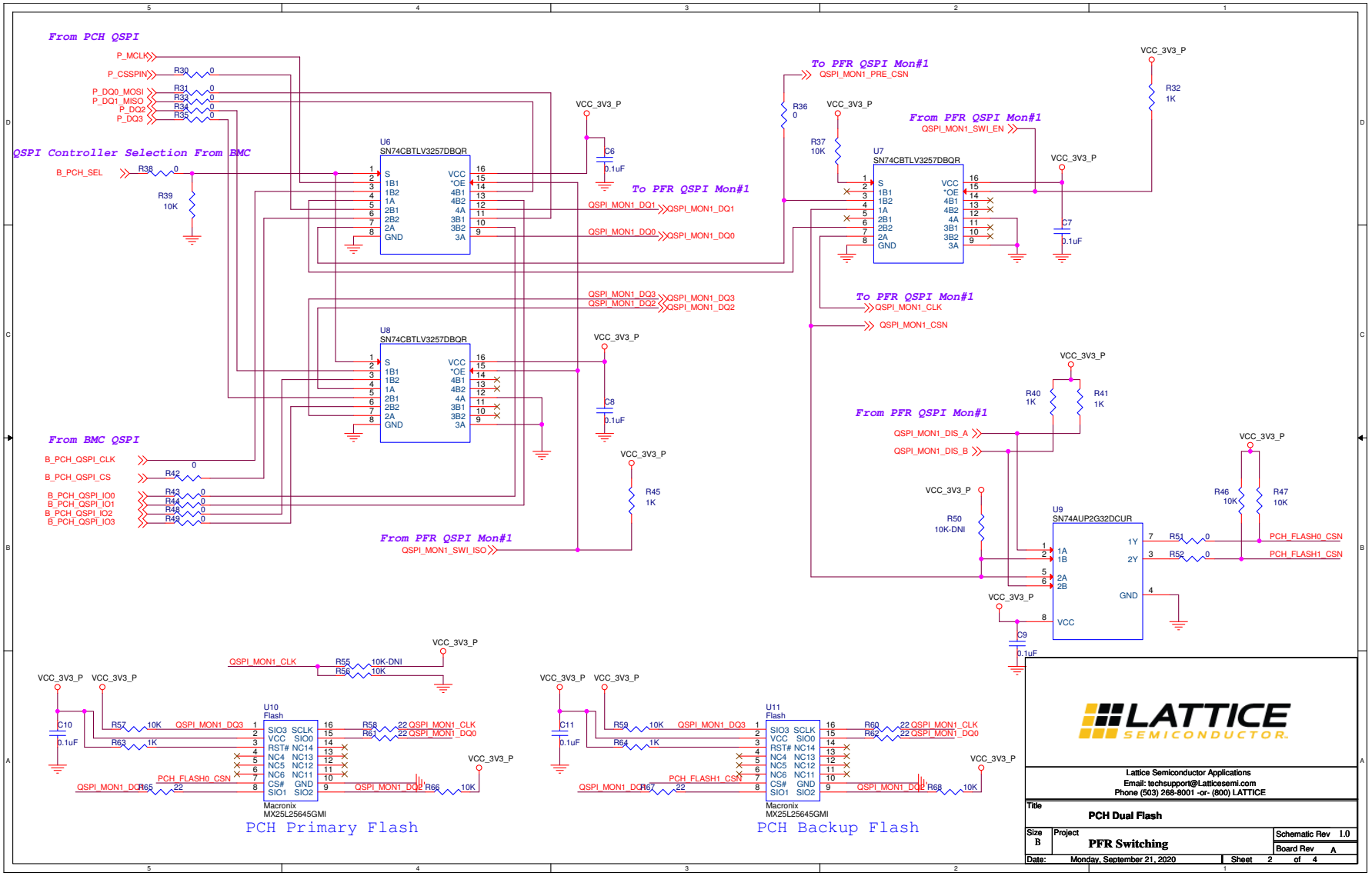

Figure A.2. PCH Dual Flash Schematic

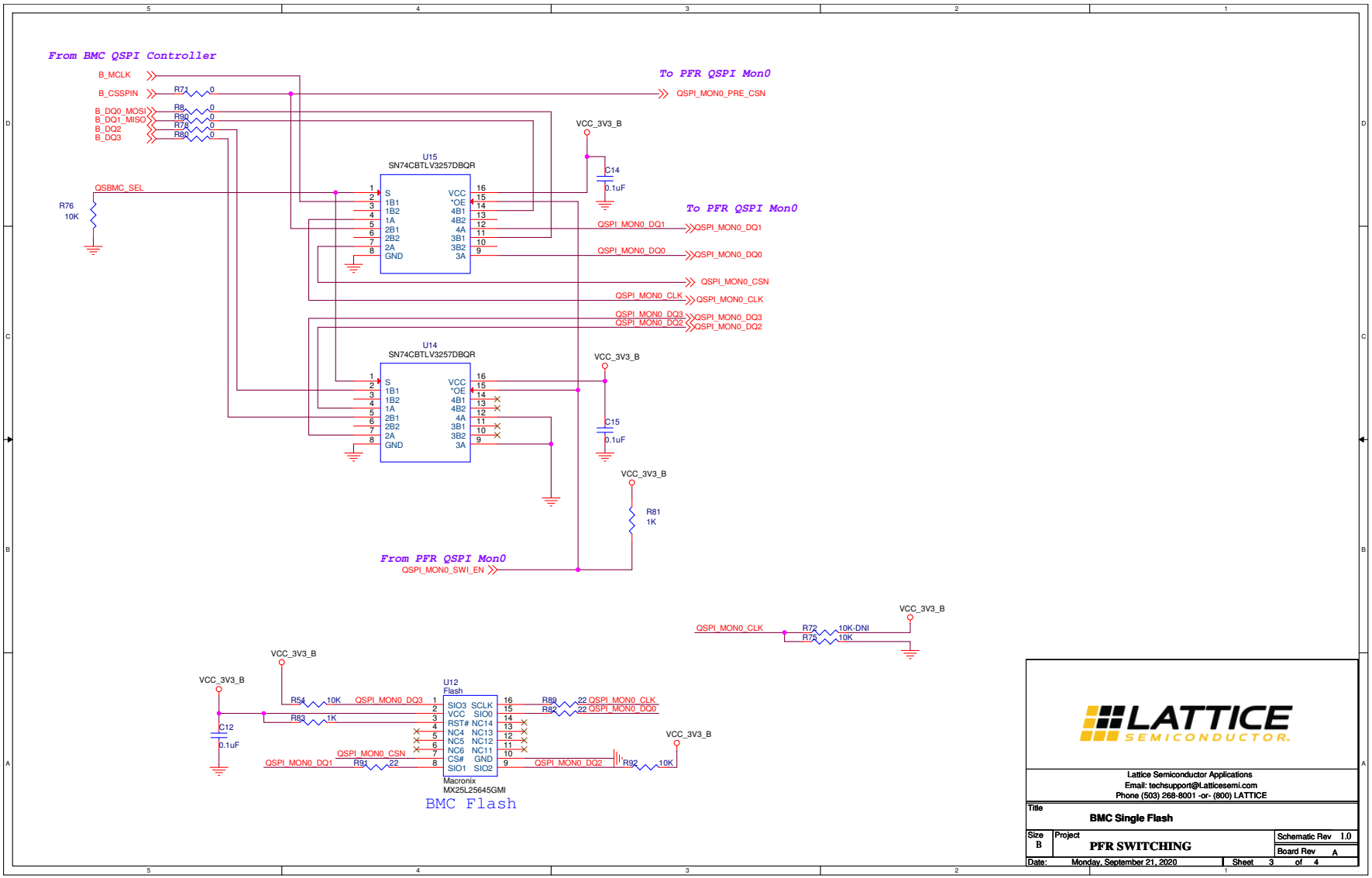

Figure A.3. BMC Single Flash Schematic

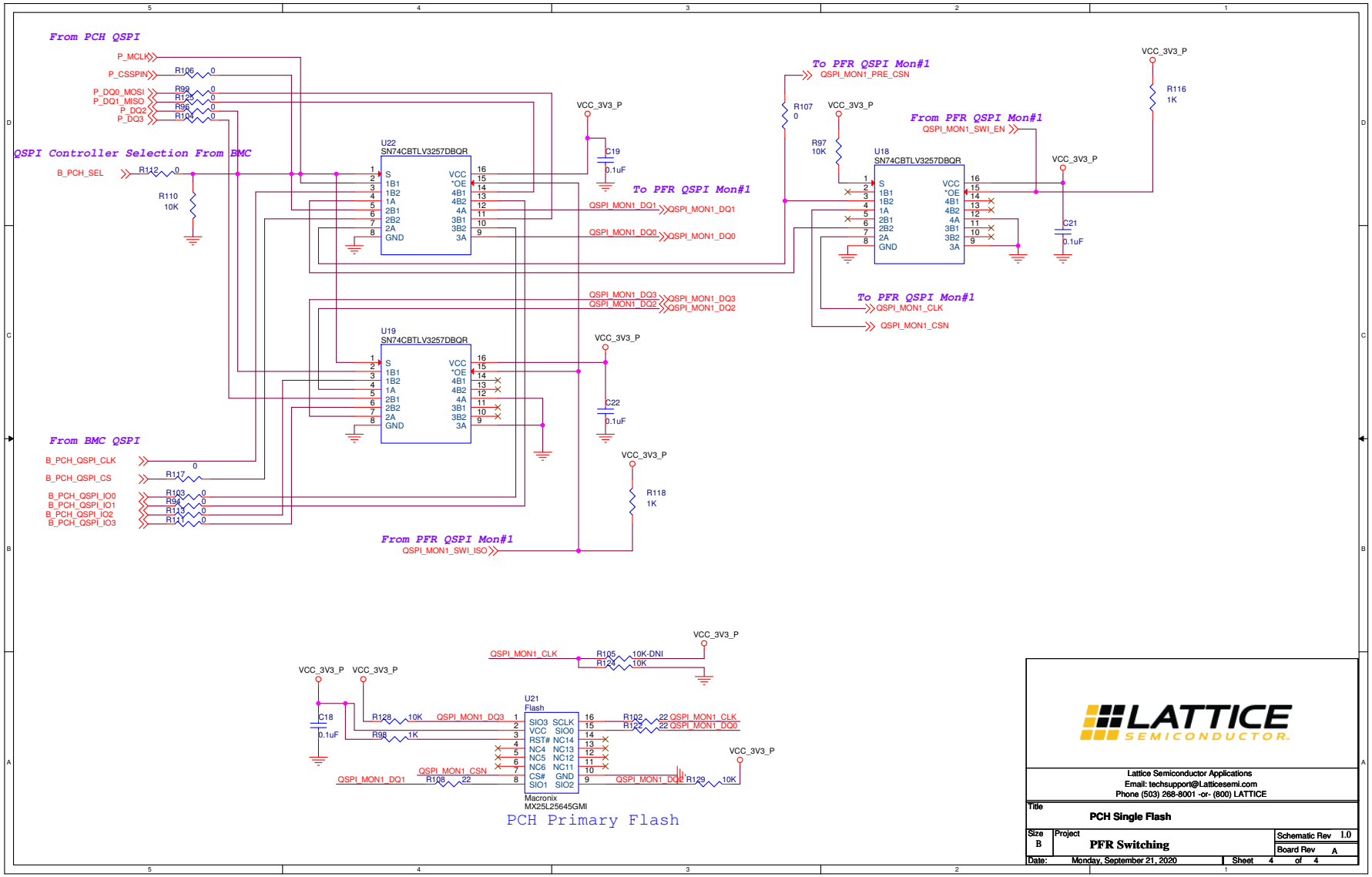

Figure A.4. PCH Single Flash Schematic

Revision History

Revision 1.0, April 2024

Section	Change Summary
All	Production release.

www.latticesemi.com