1. Introduction

The Quad SPI interface is one of standard boot options for LayerScape (LS) products. It is designed to work with serial NOR devices. Many customers have requested NXP products to support both booting from a serial NOR/NAND device and access serial NAND devices during the normal operation. The main reason is due to the price-advantage over serial NOR for a flash that is larger than 512Mbits and smaller than 16Gbits capacities. There are also new MCP products that supports a serial NOR + serial NAND devices combination in a single package called SPISTACK. This SPISTACK configuration provides designers a viable memory solution for a reliable boot code storage in the serial NOR and storing the OS or other data configuration on the larger serial NAND device. This is all done in a single package saving board space and eliminating unnecessary extra pin traces of the board layout.

This application note describes how to access a SPISTACK device (Serial NOR + Serial NAND) as well as boot from it. The software package is based on SDK 2.0 or later SDKs. The LS1012ARDB board is used in this application note.

2. Background

The Quad SPI controller is designed to work with serial NOR flash memories. However, it can also be used to work with many serial NAND devices as long as the NAND devices support serial interface programming model. That is, the serial command based model.

For all LS products, 1-bit SPI operation (0x03 READ command) is used at least during reading the RCW and PBI phases when the chip boots from the Quad SPI interface. It also requires a 24-bit address mode to drive a NOR device. Therefore, these are three issues that need to be taken care to support boot from a serial NOR/NAND:

- Support 0x03 READ command
- Support 24-bit address mode when the device boots from the Quad SPI interface
- Operate at minimum speed of 15 MHz

The chip cannot boot from the NAND device if these three issues are not addressed. Use any of the two solutions that are listed below to resolve these issues.

- Use a combo part, such as Winbond's W25M161AW, which has 1.8V 16 Mbit serial NOR and a 1.8V 1 Gbit serial NAND with just one chip select (CS). The 1.8V 16 Mbit serial NOR device is a SPISTACK configured W25Q16FW device, while the 1.8V 1 Gbit serial NAND device is a SPISTACK configured W25N01GW device. The W25M161AW multi-chip solution is better option to support booting from a smaller size of serial NOR and normal access to a larger size of serial NAND memory.

This application note covers both solutions. The advantage of the first solution is that it is more reliable during the booting process, as it uses a NOR device. However, it is more expensive than the second solution. The bad block management and wear leveling of NAND are still needed for both solutions.

Keep in mind that there is no bad block management during booting to u-boot from a flash.

3. Software changes needed

3.1. Software changes needed with W25M161AW

Currently, the serial NAND for Quad SPI driver is not supported under U-Boot or Linux. A patch will be needed in order to support this part. Please contact NXP for details. To boot from W25M161AW, it is similar to other serial NOR as there is a NOR in W25M161AW. As guidance, the software changes have to deal with the QuadSPI Look-Up-Table (LUT). Here are the necessary LUTs besides the default entries.

W25Q16FW LUT Registers:

```
0x08180403 0x24001c08 0x00000000 0x00000000 //Read command
0x1c08049f 0x00000000 0x00000000 0x00000000 //Read JEDEC ID
0x0818040b 0x1c800c08 0x00000000 0x00000000 //Fast Read
0x08180402 0x00002040 0x00000000 0x00000000 //Page Program
0x081804d8 0x00000000 0x00000000 0x00000000 //64KB Block Erase
0x00000406 0x00000000 0x00000000 0x00000000 //Write Enable
0x1c010405 0x00000000 0x00000000 0x00000000 //Read Register 1
0x20010401 0x00000000 0x00000000 0x00000000 //Write Register 1
0x200104c2 0x00000000 0x00000000 0x00000000 // CMD C2 command (Die select)
0x0818046B 0x1E080c08 0x00000000 0x00000000 // Quad read 8 bytes (4-bit mode)
```

W25N01GW LUT Registers:

```
0x08180403 0x24001c08 0x00000000 0x00000000 //Read command
0x00000406 0x00000000 0x00000000 0x00000000 //Write Enable
0x0820040c 0x1c080c08 0x00002400 0x00000000 //Fast Read 4-byte
0x0c08049f 0x00001c08 0x00000000 0x00000000 //Read JEDEC ID
0x0c080413 0x00000810 0x00000000 0x00000000 //Page Data Read
0x08100402 0x00002040 0x00000000 0x00000000 //Load Program Data
0x0c0804d8 0x00000810 0x00000000 0x00000000 //128KB Block Erase
0x000004c7 0x00000000 0x00000000 0x00000000 //Chip Erase
0x08080405 0x00001c01 0x00000000 0x00000000 // Read Status 1
0x0808041f 0x00002001 0x00000000 0x00000000 //Write Status Register
0x200104c2 0x00000000 0x00000000 0x00000000 //Die Select
0x0A1004EB 0x1E800E04 0x00000000 0x00000000 // Quad read 128 bytes (4-bit mode)
0x0c080410 0x00000810 0x00000000 0x00000000 //Program Execute
0x08100484 0x00002040 0x00000000 0x00000000 //Random Program Data Load (64 bytes)
0x1c010405 0x1c010435 0x1c010415 0x00000000 // Read Status 1, 2, and 3
0x081004A1 0x00000810 0x00000000 0x00000000 // Bad Block Management (BBM)
0x0c0804A5 0x1CA00820 0x1CA01CA0 0x00001CA0 // Read BBM LUT (A5)
0x0c0804A9 0x00001c10 0x00000000 0x00000000 // ECC failure page address
```

4. Programming NAND flash Through U-boot

You can program the NAND flash on the board after you boot from a different bank of QSPI flash or from a different interface.

Currently, Code Warrior can be used to program W25M161AW.

4.1. Programming W25M161AW

Program W25Q16FW under u-boot is the same way described in the board Getting Started Guide. There is no memory map change. Please see 1) to 3) for details.

The following U-Boot commands are used to program new images to the QuadSPI flash on LS1012ARDB:

i2c mw 0x24 0x7 0xfc; i2c mw 0x24 0x3 0xf5 // switch to bank 2

sf probe 0:0; //probe flash

tftp 0xa0000000 $pbl_img //tftp RCW+PBI command image

sf erase 0 0x80000 //Erase flash address from 0 – 0x80000

sf write 0x80000000 0 $filesize //program RCW+PBI to flash at address 0

tftp 0x80000000 $uboot_img //tftp u-boot image

sf erase 0x100000 0x100000 //Erase flash address from 0x100000 - 200000

sf write 0x80000000 0x100000 $filesize //program u-boot to flash at address 0x100000

5. Programming NAND flash Through CodeWarrior

This section provides the formatting guidelines for headings, text, and lists. Section 6 demonstrates example of use.

The current CodeWarrior does support program serial W25M161AW. You need to install the latest CodeWarrior.

Following are steps for programming the NAND flash using LS1012ARDB board:

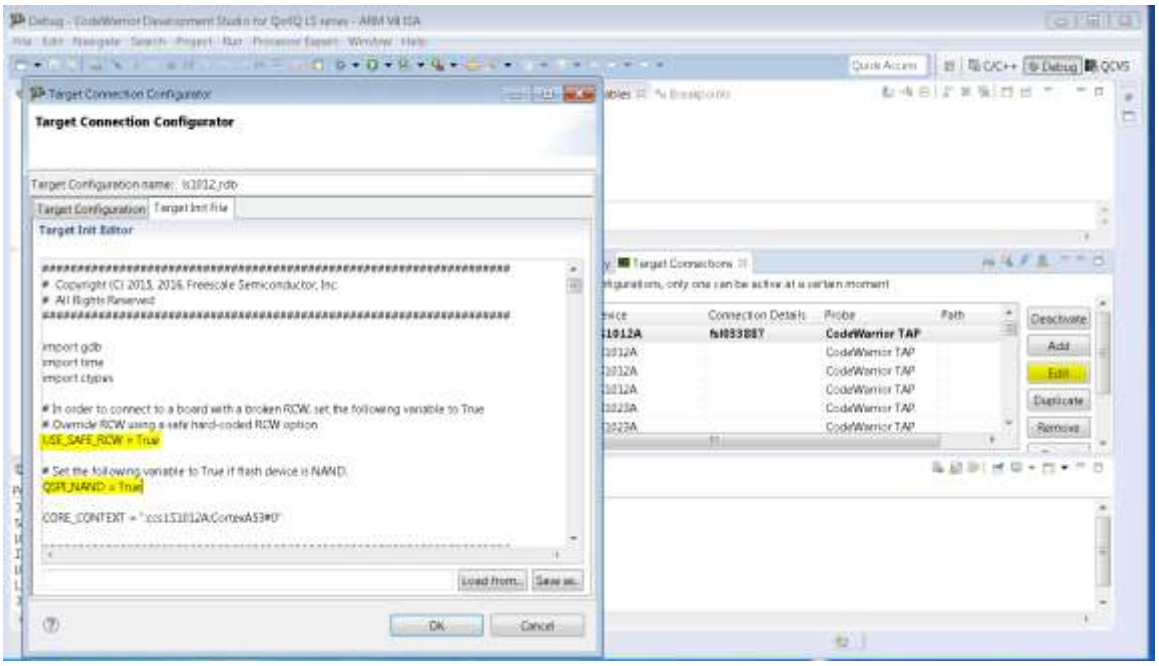
1. Start Eclipse

2. Create a new project with

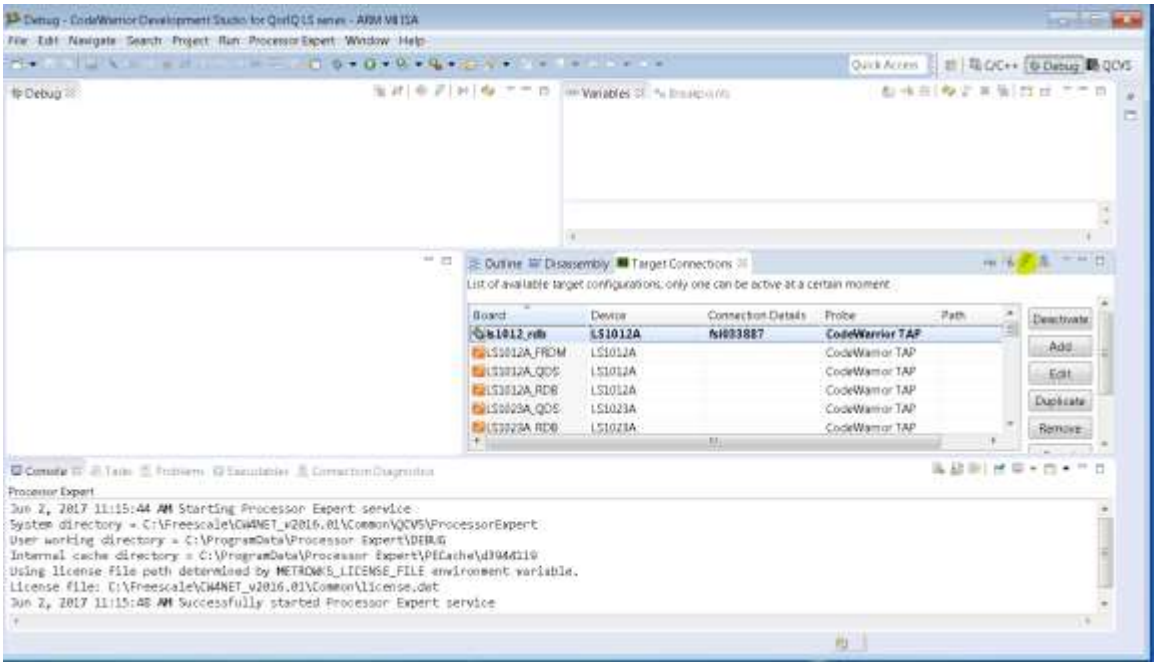
- In “Target Connections” window duplicate the LS1012A RDB connection.

- Edit the newly created connection, set connection parameters and update the initialization file:

- `Set “USE_SAFE_RCW = True”`
- `Set “QSPI_NAND = True”`
- `Click “OK” to return`



- Start Flash Programmer UI by click the highlighted tool bar



- Add unprotect action – address: 0x0, size: 0x400000
- Add program actions for RCW, SPL and ubot.
- Before adding the action check “Verify” too.

6. How to configure and boot from W25M161AW flash

The W25M161AW device has a 2 MB NOR, W25Q16FW. The configuration and booting from W25M161AW is similar to boot from W25Q16FW, a serial NOR. Select boot from Quad SPI interface for all the RCW, PBI, and software switch board configurations. For more information, see Section References.

7. References

For further information, see the following references:

1) QorIQ TWR-LS1021A Getting Started Guide (TWR-LS1021AGS)

2) QorIQ LS1012A Reference Design Board Getting Started Guide (LS1012ARDBGSG)

3) QorIQ LS1088A Reference Design Board Getting Started Guide (LS1088ARDBGSG)

8. Revision History

Version	Date	Page	Description
1.0	09/29/2017	NA	Original

Trademarks

Winbond, *SpiFlash* and *SpiStack* are trademarks of *Winbond Electronics Corporation.*
All other marks are the property of their respective owner.

Important Notice

Winbond products are not designed, intended, authorized or warranted for use as components in systems or equipment intended for surgical implantation, atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, or for other applications intended to support or sustain life. Furthermore, *Winbond* products are not intended for applications wherein failure of *Winbond* products could result or lead to a situation wherein personal injury, death or severe property or environmental damage could occur. *Winbond* customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify *Winbond* for any damages resulting from such improper use or sales.

Information in this document is provided solely in connection with Winbond products. Winbond reserves the right to make changes, corrections, modifications or improvements to this document and the products and services described herein at any time, without notice.