AN98481

Read Speed Optimization for Cypress Quad-IO SPI Flash on Zynq®-7000 Platform

AN98481 describes how to optimize the read speed of Cypress Quad SPI flash on the Zynq-7000 chipset from Xilinx®.

1 Introduction

This application note describes how to optimize the read speed of Cypress Quad SPI flash on the Zynq-7000 chipset from Xilinx®. The methods shown here are derived from the Xilinx Answer Record AR# 46880, which provides a bare metal example for Linear Addressing Mode reads with Direct Memory Access (DMA). We extend this method to U-Boot and to the Linux Memory Technology Device (MTD). We also show supporting optimizations and settings for the BootROM Header and the First Stage Boot Loader (FSBL). References to source code are provided along with benchmark results.

2 Zynq-7000 Basics

The Quad SPI flash is connected to the Processing System (PS) side of the Zynq-7000 platform where it is typically used as a boot device for initialization of the PS side and configuration of the Programmable Logic (PL) side. To achieve minimal initialization times requires optimization for read speed.

The Quad SPI (QSPI) flash controller in Zynq-7000 can operate in two basic modes: I/O mode and linear addressing mode.

- I/O Mode
  - Software interacts with a memory-mapped register and is responsible for forming correct flash command messages and for loading and unloading the FIFOs; the QSPI controller manages the serial transport across the wires in single, dual, or quad I/O.
  - This mode is flexible for any SPI flash operation, although it is not the best option for fast read speed. It is the only option for executing flash commands other than read commands, so this mode is used whenever flash status reads, flash register configuration, or flash program/erase operations are required.

  Note: I/O Mode can support any SPI flash command when the QSPI controller is operated in Single-IO mode. However, it can only support 3-byte address commands in Dual-IO or Quad-IO modes. Addressing more than 128 Mbit (16 MB) of flash requires software to manage the Bank Address Register (BAR) provided by certain high density Cypress SPI flash chips.
- Linear Addressing Mode
  - This mode maps the flash address space to the system address space enabling CPU memory fetch commands to trigger hardware-managed high performance reads from the flash.
  - This mode is only for reading, and when used with DMA it provides the most efficient read path. Linear mode is enabled by default to provide the fastest PL configuration, and can be exploited to enable the fastest PS side boot, and the fastest file system mount time.

  Note: The Linear Addressing Mode logic in the QSPI controller only supports 3-byte addressing for Single-IO, Dual-IO, or Quad-IO reads, and so only provides a system memory window of 16 MB. Software can manipulate the BAR via I/O Mode to assign the system memory window to other 16 MB banks of flash address space when high density flash is used.

The most basic code-shadow booting mode for the Zynq-7000, PS Master Non-Secure Boot to Linux from external SPI, includes the following stages:

2.1 Stage 0: BootROM Execution

The BootROM is hard-coded into the Zynq-7000 device, and is responsible for:

- Establishing communication with the external SPI flash,
- Reading the BootROM Header and Partition Table Header from the external SPI flash,
- Copying the First Stage Boot Loader (FSBL) from the FSBL Partition on the external SPI flash to On-Chip Memory (OCM, or embedded RAM), and
- Transferring control to the OCM-resident FSBL.

The only optimizations possible at this stage involve changing contents of the BootROM Header.

2.2 Stage 1: FSBL Execution

The FSBL is generated by the Xilinx tool chain and is customized to your hardware design. It is responsible for:

- Scanning the Partition Header Table on the external SPI flash to identify the Bitstream Partition and the U-Boot Partition
- Initializing the PS configuration
- Loading a bitstream from Bitstream Partition to the PL
- Loading U-Boot from the U-Boot Partition to DDR
- Transferring control to U-Boot

It is possible to optimize the FSBL for read speed.

2.3 Stage 2: U-Boot

- U-Boot is responsible for loading the Linux kernel and RAM Disk images to DDR.
- It is possible to optimize U-Boot for read speed.

2.4 Stage 3: Linux Kernel

- The Linux kernel is responsible for mounting any file partitions and initializing any applications.
- It is possible to optimize the Linux Memory Technology Device (MTD) for read speed, to enhance flash file system mounting and reading operations.

3 Optimization Details

The Xilinx Answer Record AR# 46880 introduces an optimization technique that uses Linear Addressing Mode with DMA (DMA-Linear), and provides example source code for a bare metal system. However, this optimization technique can be applied to any other software that runs on the Zynq-7000 platform such as FSBL, U-Boot, and Linux. Since the U-Boot and Linux kernel images may have significant size, read speed optimization can have a dramatic impact on booting times. We also highlight other optimization opportunities such as setting higher SPI clock speeds.

The following hardware and software are used as a baseline platform to illustrate the optimization methods and its benefits.

- Avnet® ZedBoard™ Rev D evaluation board with Cypress S25FL256S Quad SPI Flash
- Xilinx SDK 2013.3
- U-Boot and Linux from the xilinx-v14.7 tag in Xilinx git repositories

3.1 BootROM Header

The Register Initialization Table in the BootROM Header can be updated to make the Quad SPI flash controller run at a faster clock frequency instead of default clock frequency. Refer to the following documents for more details about Register Initialization Table.

- Zynq-7000 All Programmable SoC Technical Reference Manual, 6.3.3 BootROM Performance
- Zynq-7000 All Programmable SoC Software Developers Guide, Appendix A: Using Bootgen

For the target platform in this application note, the following settings are used to optimize FSBL loading:

```
.set. 0xF8000120 = 0x1F000200; // ARM_CLK_CTRL - divisor = 2 (433 MHz)
.set. 0xF8000720 = 0x00000702; // MIO_PIN_08 - QSPI FB clock, 3.3 V, Slew Fast
.set. 0xF800014C = 0x00000501; // LQSPI_CLK_CTRL - divisor = 5, IO PLL (200 MHz)
.set. 0xE000D000 = 0x800238C1; // QSPI config - divide-by-2 (100 MHz)
.set. 0xE000D038 = 0x00000020; // QSPI loopback - internal, 0 delay
.set. 0xE000D0A0 = 0x8000016B; // LQSPI_CFG - Quad READ mode, Spansion
```

3.2 FSBL

The Xilinx SDK provides an FSBL project template and the Xilinx tools automatically add a QSPI setup file, qspi.c, to the SDK project source directory.

In the default qspi.c source code the clock divider for Quad SPI is set to 8. This setting makes the Quad SPI flash controller operate at 25 MHz, given a default reference clock of 200 MHz. Changing the clock divider to 2 enables the Quad SPI flash controller to operate at 100 MHz (the maximum QSPI clock speed). At line 221 in qspi.c, change clock pre-scale from 8 to 2:

```
XQspiPs_SetClkPrescaler(QspiInstancePtr, XQSPIPS_CLK_PRESCALE_2);
```

The FSBL configures the Quad SPI flash controller as DMA-Linear to load the configuration bitstream and U-Boot. However, since the QSPI Linear Addressing Mode can only read from a 3-byte SPI address space, the accessible address range is limited to 16 MB. So to enable FSBL accesses to span 16 MB flash banks, the FSBL automatically configures the QSPI to I/O mode and uses the BAR to address additional flash memory banks. This causes a huge negative performance impact since software-managed FIFO unloading overhead is 100% or more relative to the hardware-managed burst transfer.

However, if you are sure that the FSBL only reads from the first 16 MB of flash (flash BAR = 0), you can hard code the QSPI for DMA-Linear mode. At line 242 (single flash) or 269 (dual flash), change the if-statement that checks flash size:

```
if ( 1 /*QspiFlashSize <= FLASH_SIZE_16MB*/) {
```

3.3 U-Boot

Cypress provides a U-Boot patch that configures the Xilinx hardware for DMA-Linear. Just apply the patch and rebuild U-Boot.

- Refer to http://www.wiki.xilinx.com/Build+U-Boot for detailed information about building U-Boot.
- Go to http://www.spansion.com/Support/Pages/DriversSoftware.aspx to find the Cypress U-Boot patch for Zynq-7000.

3.4 Linux

Cypress provides a Linux kernel patch that configures the Xilinx hardware for DMA-Linear. Just apply the patch and rebuild Linux.

- Refer to http://www.wiki.xilinx.com/Build+kernel for detailed information about building Linux kernel.
- Go to http://www.spansion.com/Support/Pages/DriversSoftware.aspx to find the Cypress Linux kernel patch for Zynq-7000.

4 Benchmarks

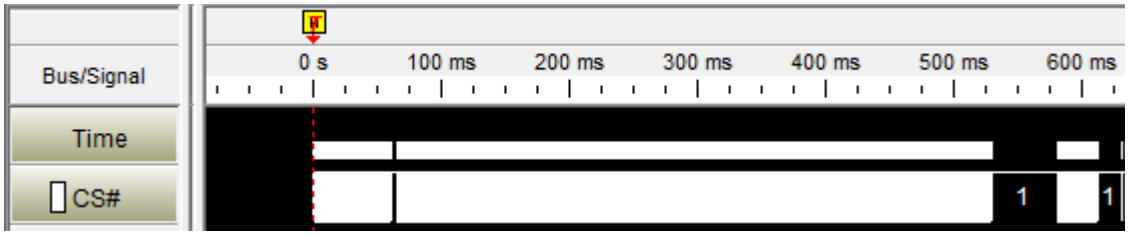
This chapter shows before and after boot time measurements to quantify how much the optimizations described in this document improve the system level performance. Table 1 shows our measurements on the baseline platform.

Table 1. Measurement Result

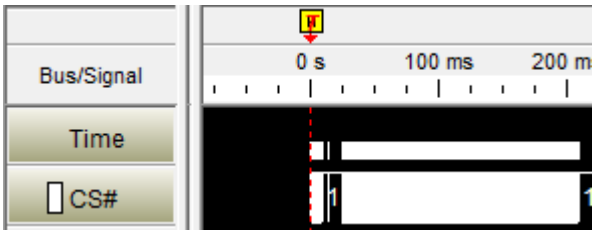
Measurement Scope	Partition Size (MB)	Before Optimization (ms)	After Optimization (ms)	Improvement (ms)	Improvement (%)
BootROM Read, FSBL Load and Execute	4.2	610	210	400	190
U-Boot (Kernel, Ramdisk Load)	11	1620	340	1280	377
Linux (JFFS2 Mount)	16	1330	340	990	291
Total	31.2	3560	890	2670	300

4.1 Benefit for FSBL Load and FSBL Execute

Without the optimizations described here, it takes 610 ms to read the BootROM header, and to load and execute the FSBL. Executing the FSBL means loading the PL configuration bitstream and loading U-Boot.

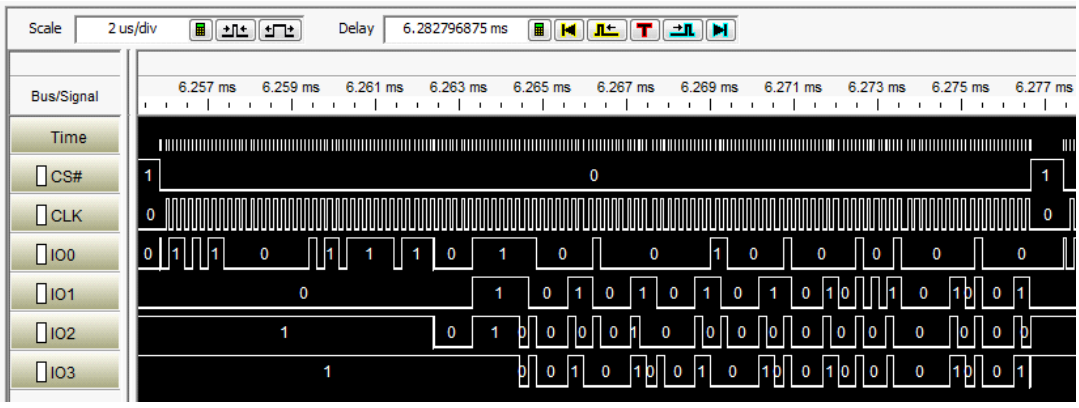


After applying the optimizations described here for the BootROM and the FSBL, the load and execution time is reduced to 210 ms.

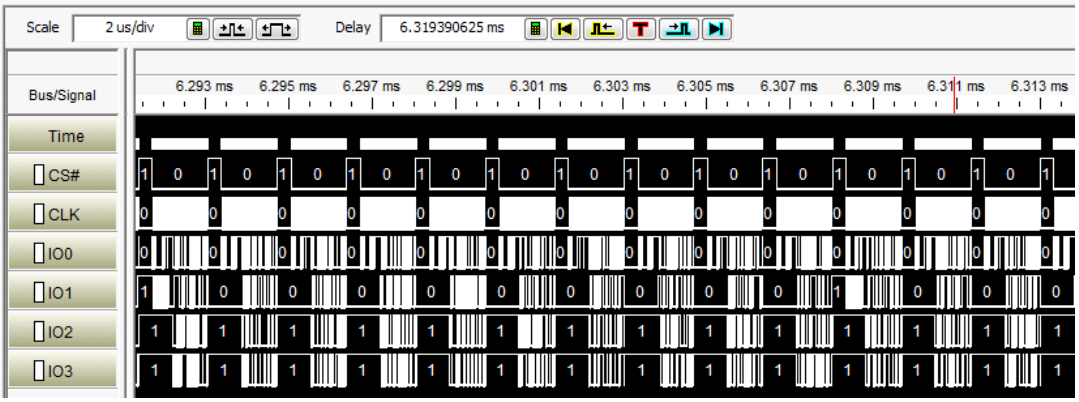


4.1.1 FSBL Load

By default, the FSBL load is performed by Quad Read in Linear mode, but the SPI clock frequency is 5 MHz.

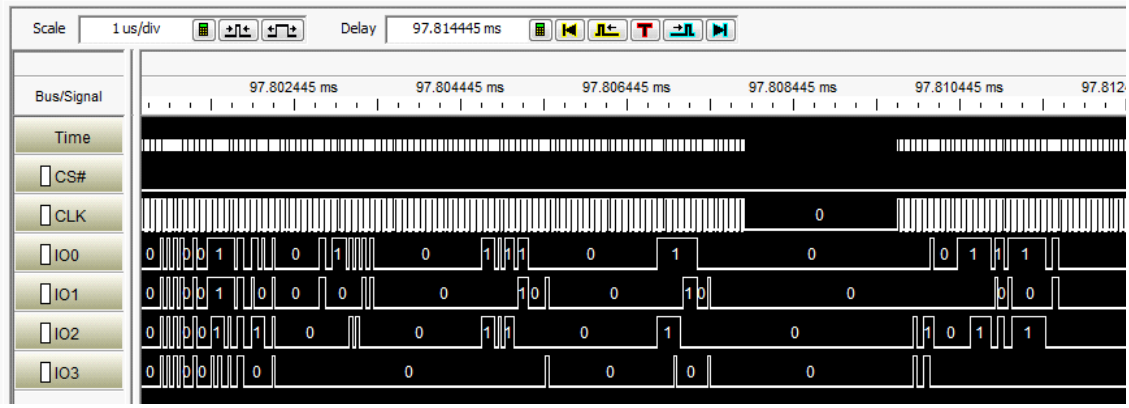


After adding register initializations into the BootROM Header, the SPI clock frequency is changed to 100 MHz.

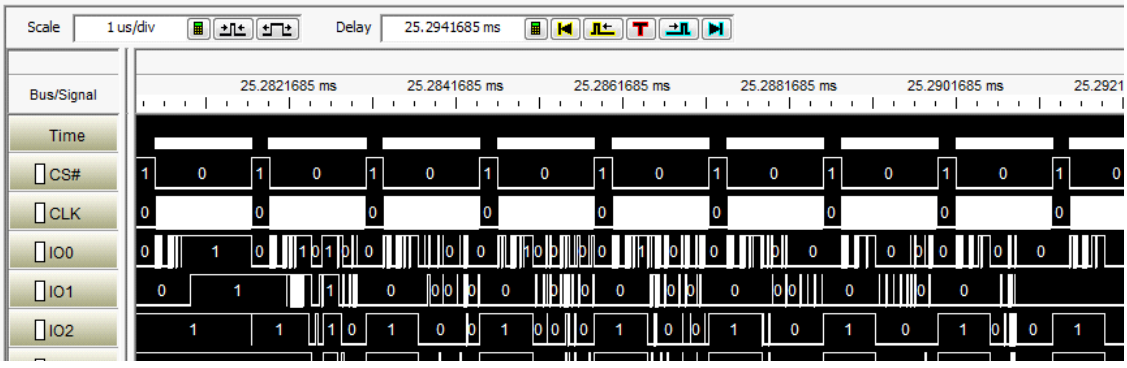


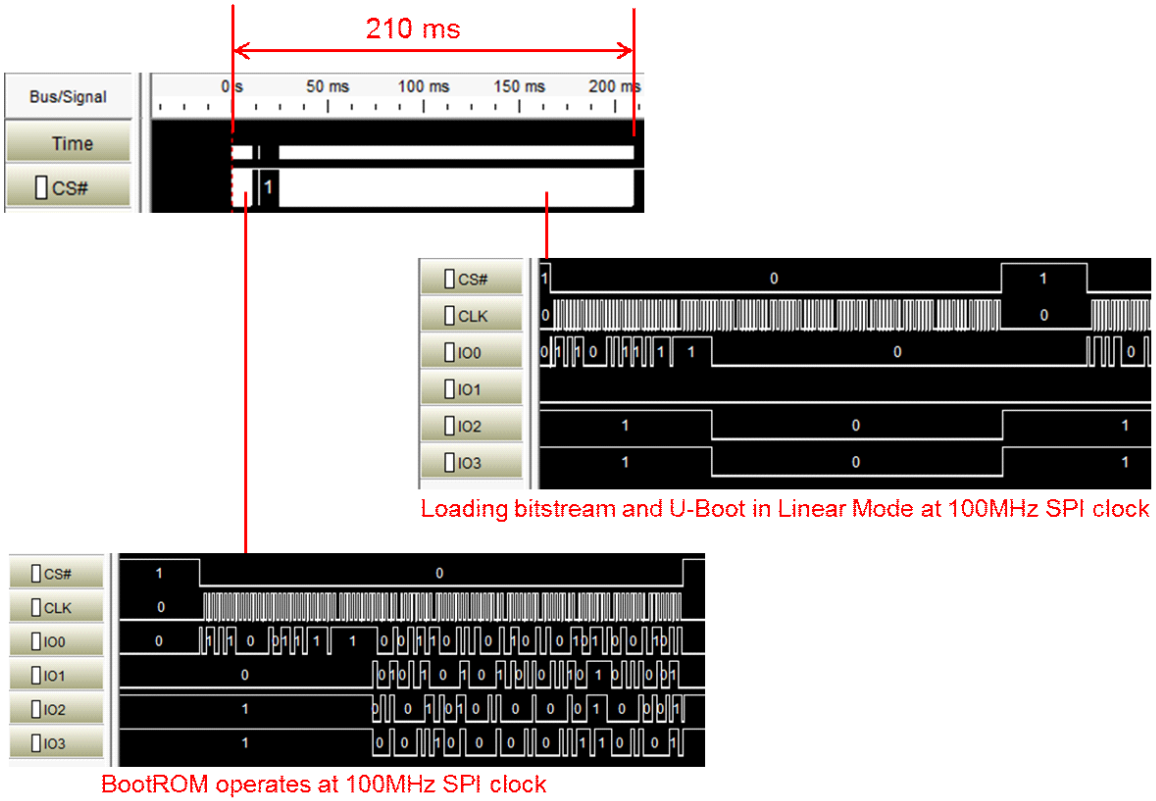
4.1.2 FSBL Execute (PL Bitstream Load and U-Boot Load)

By default, loading the PL bitstream and U-Boot is performed by Quad Read in IO mode at 25 MHz. FIFO unloading overhead can be seen as clock to clock gap below.



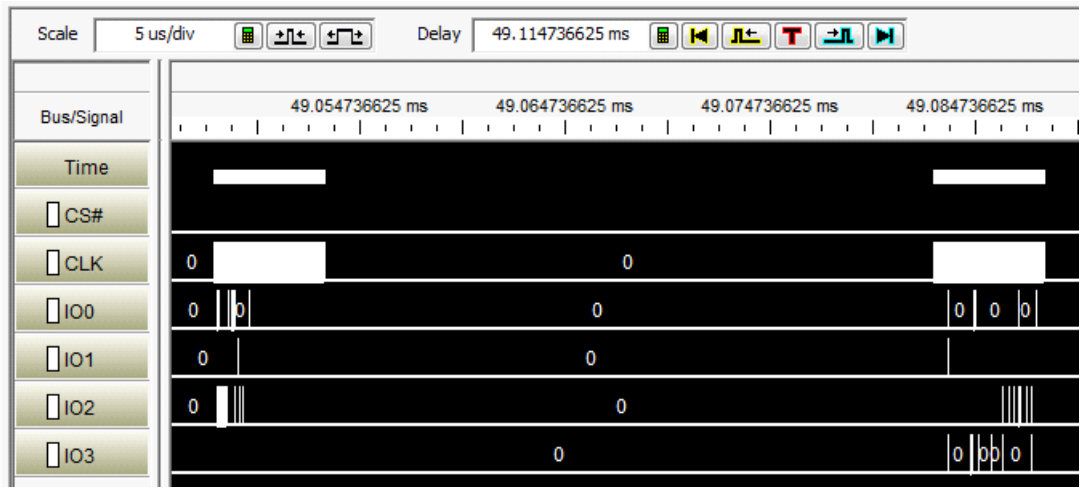
After changing FSBL code, loading PL bitstream and U-Boot is performed by Quad Read in Linear mode at 100 MHz. FIFO unloading overhead almost disappears.



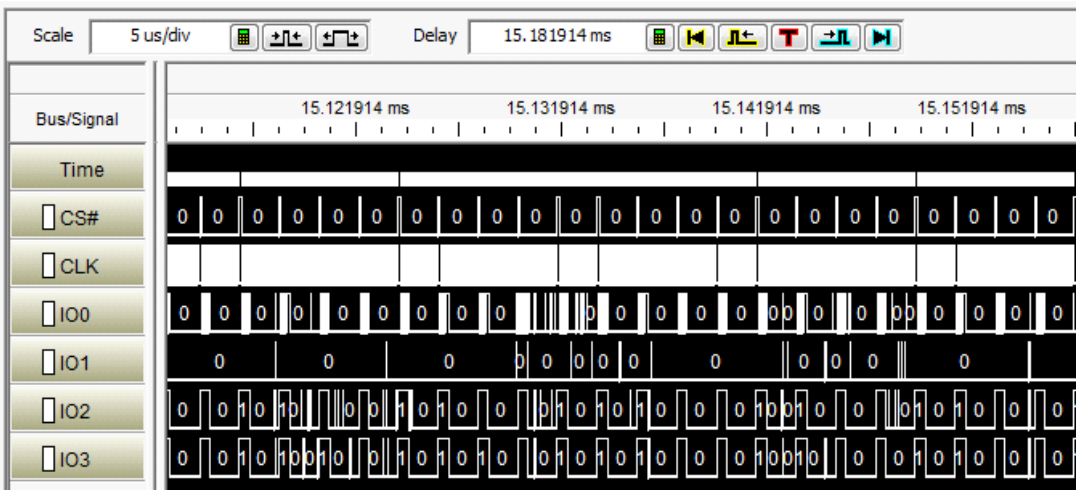


4.2 Benefit for U-Boot

The U-Boot loads a total of 11 MB including Linux kernel image, Ramdisk image, and Device Tree. Without optimization, loading the 11 MB data in I/O Mode at 100 MHz SPI clock takes 1620 ms. In I/O Mode, the software is responsible for loading the transmit FIFO and unloading the receive FIFO of the Quad SPI flash controller, which results in a huge overhead. In the following logic analyzer capture, there is a time gap between SPI transfers, which is consumed by loading/unloading these FIFOs.



Enabling DMA-Linear Mode by applying the U-Boot patch drastically improves the loading time to 360 ms. In the DMA-Linear Mode, the data flow is controlled by hardware that eliminates the FIFO loading/unloading overhead.



4.3 Benefit for Linux

Optimizing flash read speed is expected to improve Flash File System mount speed, which typically happens during the system boot process. Mounting a Flash File System partition takes 1330 ms in I/O Mode and 340 ms in DMA-linear Mode, under the following conditions.

- Flash File System type: JFFS2
- The size of Flash File System partition: 16 MB
- The occupancy rate (how much the partition is used): 40%
- 100 MHz SPI Clock Frequency

5 Conclusion

The Zynq-7000 has a Quad SPI Flash controller that operates in I/O Mode or in Linear Addressing Mode. Using Linear Addressing Mode with DMA improves the read speed drastically. Since the Quad SPI flash can be used as a boot device and many reads happen during the boot process, using this optimization technique is expected to improve the system boot performance.

6 References

- Xilinx Zynq-7000 All Programmable SoC Technical Reference Manual
- Xilinx Zynq-7000 All Programmable SoC Software Developers Guide
- Xilinx AR# 46880 Zynq-7000 Example Design – Linear QSPI Performance (Max Effective Throughput)
- Cypress S25FL256S Flash Memory Data Sheet

Document History Page

Document Title: AN98481 - Read Speed Optimization for Cypress Quad-IO SPI Flash on Zynq®-7000 Platform Document Number: 001-98481				
Rev.	**ECN No.**	**Orig. of Change**	**Submission Date**	**Description of Change**
**	–	–	03/16/2015	New Application Note
*A	4955529	MSWI	10/09/2015	Updated in Cypress template
*B	5844214	AESATMP8	08/04/2017	Updated logo and Copyright.

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6

Cypress Developer Community

Forums | WICED IOT Forums | Projects | Video | Blogs | Training | Components

Technical Support

cypress.com/support

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709