Application Note

Astra™ Machina SL2600 Series xSPI

Abstract: This application note describes the SL2600 series xSPI solution and system requirement.

Contents

Overview

The xSPI Controller is a high-performance controller designed to interface with a wide range of serial flash memory devices compliant with the JESD216 and JESD251 standards. It supports traditional SPI, Dual, Quad and Octal I/O configurations, as well as advanced protocols such as Hyper-Flash, Hyper-RAM, and SPI-NAND.

- Supports Single, Dual, Quad, and Octal I/O SPI accesses
- Integrated DMA engine for indirect (STIG) mode operations
- Dual chip select support
- Soft PHY Integration
    - Supports Single (SDR) and double data rate (DDR)
    - Supports DQS and loopback-based data sampling
- Operates at:
    - Up to 150 MHz in DDR mode with DQS
    - Up to 60 MHz without DQS strobe
- Direct and indirect access mode
- Supports 128 MB of direct addressing (extendable via indirect mode)
- 4-byte address support
- Programmable command frame formats including command, address, mode, dummy cycles, and data phases.
- Configurable interface width: 1-bit, 4-bit, or 8-bit
- Integrated TX/RX FIFOs for efficient data buffering

1.1. Block Diagram and Interface Pins

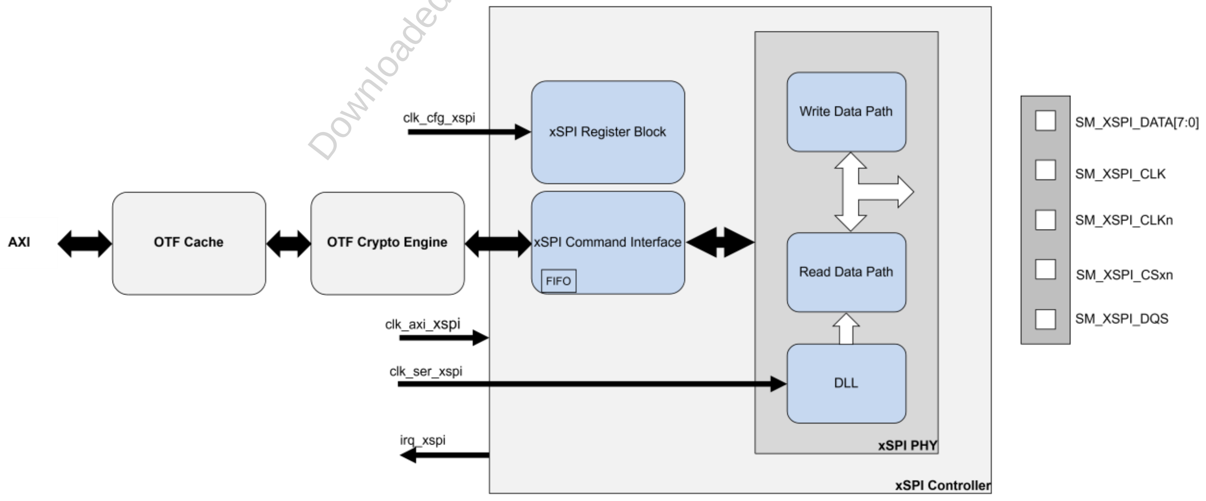

Figure 1. xSPI interface Block

1.2. Clock Configuration

The xSPI controller sources clk_mcu_io_pll clock to drives the external SM_XSPI_CLK(clk_ser_xspi) output.

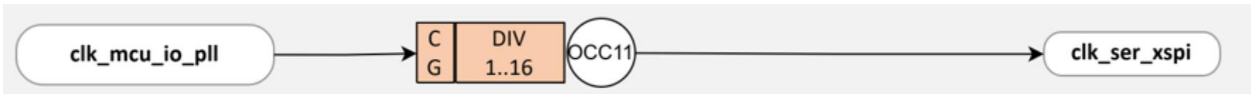

Figure 2. xSPI interface Clock Tree

Table 1 clk_ser_xspi Clock configuration

Clock	Clock Function	Source	clock divider	Max clock frequency [MHz]
clk_ser_xspi	XSPI Core Clock	clk_mcu_io_pll	DIV1-DIV16	150

1.3. xSPI Clock Programming

The following example demonstrates how to configure SM_XSPI_CLK to 150 MHz, assuming the source clock clk_mcu_io_pll is running at 300 MHz:

```
// Configure SM_XSPI_CLK = 150 MHz, derived from clk_mcu_io_pll = 300 MHz

// 1. Enable the Divider clock source

clk_ser_xspi_ctrl.clk_en = 1;

// 2. Set divider value: divide-by-2 → dctrl = 1

clk_ser_xspi_ctrl.dctrl = 0x1;

// 3. Latch the new divider value

clk_ser_xspi_ctrl.div_chng_en = 1;
```

The divider value is set via the dctrl field (bits [5:2]) in the mcu_gbl_cfg.clk_gen.clk_ser_xspi_ctrl register. A value of 0 corresponds to divide-by-1, 1 to divide-by-2, and so on.

1.4. xSPI Differential Clock Output Programming

Some xSPI Flash memory support the usage of a differential clock for improved signal integrity. The xSPI controller supports this configuration by enabling a differential clock mode that drives both *SM_XSPI_CLK* and *SM_XSPI_CLKN* signals.

The differential clock output is enabled by the *DQS_REMOD_EN* bit in the mcu_gbl_cfg.perif_ctrl_XSPI_CTRL1 register.

When differential clock is enabled, make sure that the PHY read sampling mode is consistent with the use of differential signaling (see section **Error! Reference source not found.**).

1.5. Hardware Interface

The following table lists the external interface pins of the xSPI controller. These signals connect the controller to the memory device and are used for command, address and data transfer.

Table 2 xSPI Interface pin overview

Pin	Direction	Description
SM_XSPI_CS[1:0]n	Output	Chip Select for the memory device. Up to 2 chip select lines are supported
SM_XSPI_CLK	Output	Clock output from the controller, used as the SPI clock signal for memory devices
SM_XSPI_CLKn/ REBAR	Output/ Output	Differential clock (CKn) for memory, used in HyperRAM where needed. In loopback mode, REBAR act as the internal DQS source
SM_XSPI_DQS	Bidirectional	xSPI data strobe
SM_XSPI_DATA[7:0]	Bidirectional	xSPI data

1.6. SPI Protocols

The xSPI Flash Controller supports a wide range of SPI protocols, making it compatible with a wide variety of serial memory devices across different performance and interface tiers. Supported protocols include:

- Standard SPI: Single I/O line for command, address, and data.
- Dual SPI: Two I/O lines used simultaneously for command, address, and data phases, increasing data throughput.
- Quad SPI: Four I/O lines used in parallel for all transaction phases, offering significantly higher bandwidth than standard or dual SPI.
- Octal SPI: Eight I/O lines used concurrently for high-speed memory operations.
- Octal DDR (Dual Data Rate): Extends Octal SPI by transmitting data on both clock edges, doubling effective data rate. Requires use of a DQS-based sampling.
- HyperBus (HyperFlash / HyperRAM): supports xSPI Profile 2 timing with dedicated DQS.
- SPI-NAND: Page-based addressing with ECC and error status report support.

1.6.1. SPI Modes

The xSPI controller supports SPI Mode 0 and SPI Mode 3, which differ primarily in the polarity of the serial clock (SCK) when the bus is idle, as illustrated in Figure 3.

In Single Data Rate (SDR) mode, data transfers between the controller and the SPI memory device can be configured to use either Mode 0 (where spi_clock_mode=0) or Mode 3 (spi_clock_mode=1). This setting is configured through clock_mode_settings register at offset 0x1008 in the xSPI-PHY.

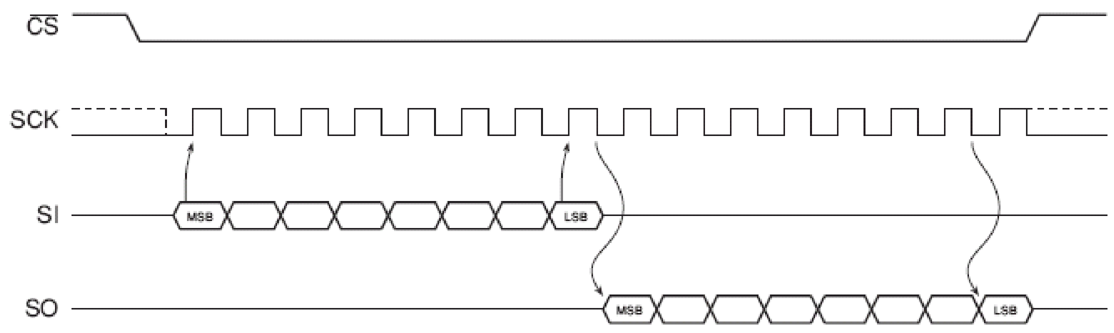

Figure 3. SPI Mode 0/3

In Mode 0, the SCK line idles low (logic 0), whereas in Mode 3, it idles high (logic 1).

1.6.2. *Standard SPI*

The SPI bus interface consists of four primary signals: Serial Clock (SCLK), Chip Select (CS), Serial Data Output (SO), and Serial Data Input (SI). All instructions, addresses, and data are transferred to and from the flash device most significant bit (MSB) first.

Serial input data is sampled on the first rising edge of SCLK after CS is driven low. Following this, the one-byte instruction code is shifted into the device on the SI line, with each bit latched on the rising edge of SCLK. Every SPI transaction begins with this instruction byte, which may be followed by address bytes, data bytes, or both—depending on the specific command.

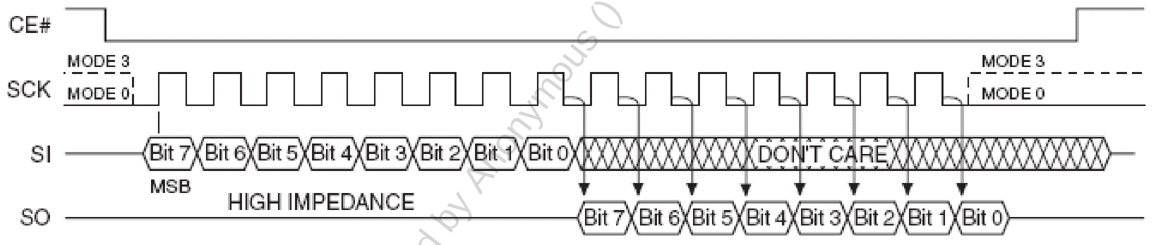

Figure 4. SPI Protocol

1.6.3. *Dual I/O SPI Protocol*

The Dual I/O SPI protocol transmits instructions, addresses, and data across two data lines: DQ0 and DQ1. This differs from Extended SPI dual I/O mode, where only the address and data phases are transferred over dual lines, while the instruction phase remains on a single data line.

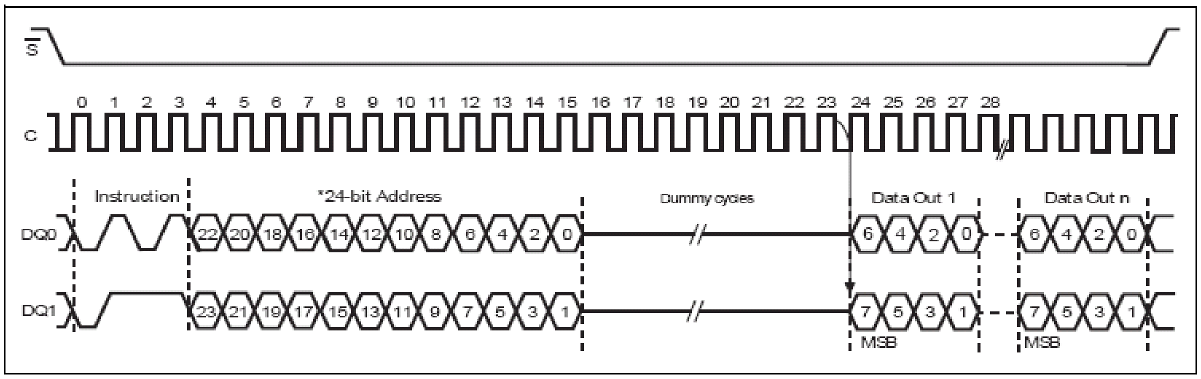

Figure 5. Dual SPI Protocol

1.6.4. Quad I/O SPI Protocol

The Quad SPI (QSPI) protocol transmits instructions, addresses, and I/O data over four data lines: DQ0, DQ1, DQ2, and DQ3, significantly increasing bandwidth compared to Single or Dual SPI modes. When the device is configured with Quad Enable (typically set in a non-volatile configuration register), the Write Protect (WP#) and HOLD# (or RESET#) functions are repurposed to serve as data lines, effectively disabling their original functionality to allow full 4-bit data transfers.

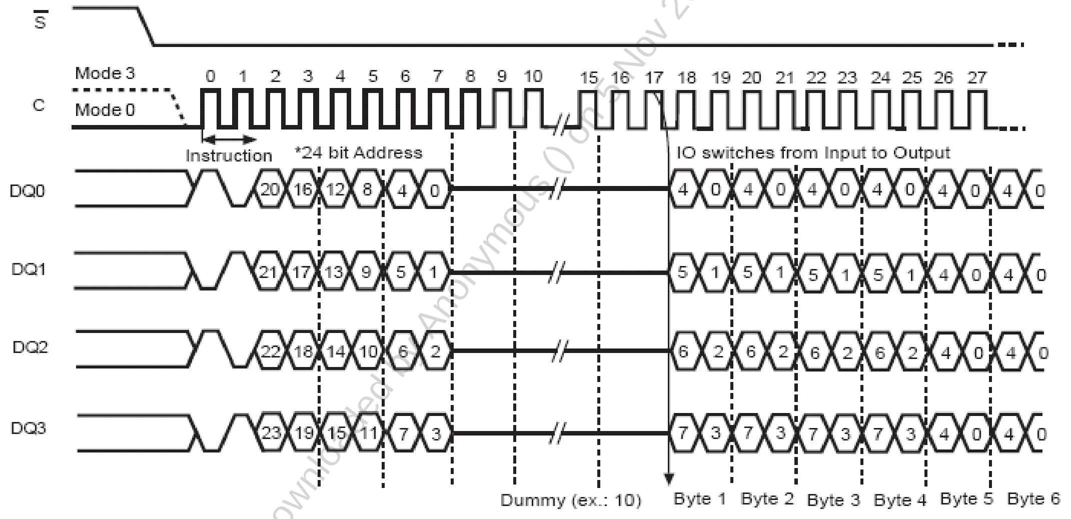

Figure 6. Quad SPI Protocol

1.6.5. Octa I/O SPI Protocol

The Octal SPI (Octa) protocol is a high-speed serial interface that uses eight data lines (DQ[7:0]) to transfer command, address, and data in parallel, significantly increasing throughput compared to standard SPI. In Octal SDR (Single Data Rate) mode, data is sampled on one clock edge, while Octal DDR (Double Data Rate) mode samples data on both rising and falling edges of the clock, effectively doubling bandwidth. Commands are typically 1 byte, followed by multi-byte addresses and optional mode bits. The number of dummy cycles is device-dependent and must be configured per datasheet. A DQS signal may be used in DDR modes for accurate data sampling, or alternatively loopback mode (SM_XSPI_CLKn) can be used. Octa protocols support advanced features like XIP (Execute-in-Place) and quad/octal mode entry via configuration registers. The controller manages these sequences using pre-programmed configuration registers to align timings, byte order, and latency. Octa mode is compliant with JEDEC xSPI Profile 1 and Profile 2 specifications and is widely used in modern high-density NOR flash and HyperFlash devices.

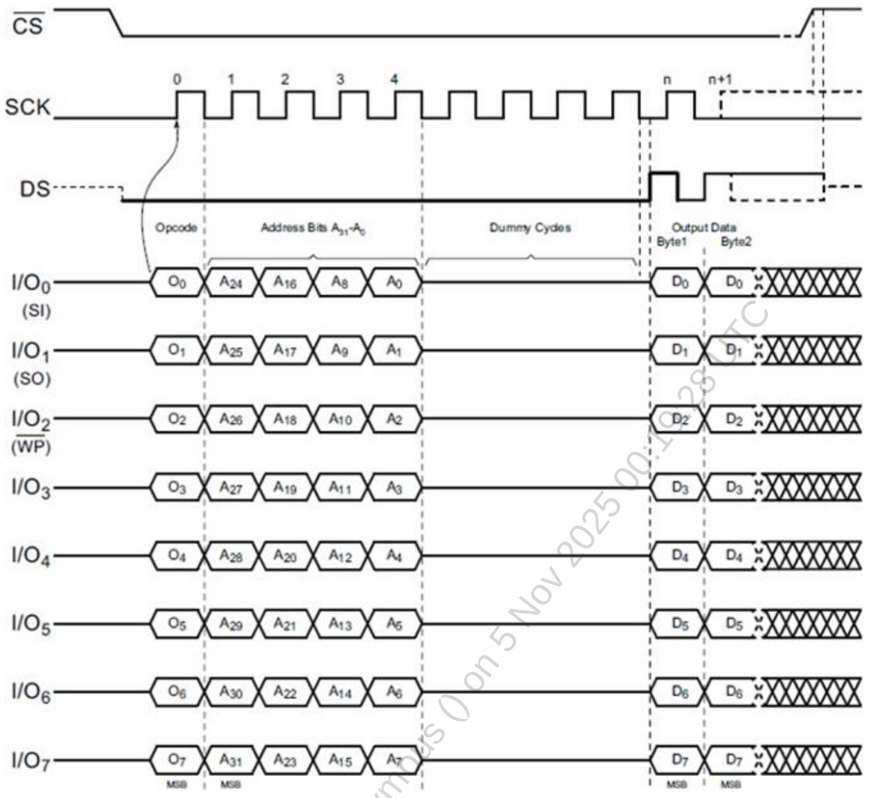

Figure 7. Octa SPI Protocol

1.7 RDK Recommended xSPI devices

RDK platform supports on-board xSPI-SD Card boot. And below is the recommended xSPI devices.

Table 3 RDK recommended xSPI

Vendor	SPI device	Type	Footprint
GigaDevice	GD25LQ128DSIGR	128MBIT SPI/QUAD	wson8ep_1p27mm_5x6mm
Winbond	W25Q64JWZPIM	64MBIT SPI/QUAD	wson8ep_1p27mm_5x6mm
Winbond	W25Q128JWPIM	128MBIT SPI/QUAD	wson8ep_1p27mm_5x6mm

2. References

- Astra Machina Foundation Series Quick Start Guide (PN: 511-001404-01)
- Astra Machina SL1620 Developer Kit User Guide (PN: 511-001407-01)
- Astra Machina SL1640 Developer Kit User Guide (PN: 511-001405-01)
- Astra Machina SL1680 Developer Kit User Guide (PN: 511-001403-01)
- Astra Machina SL2600 Series Developer Kit User Guide (PN:511-001453-01)

3. Revision History

Revision	Description
A	Initial release.