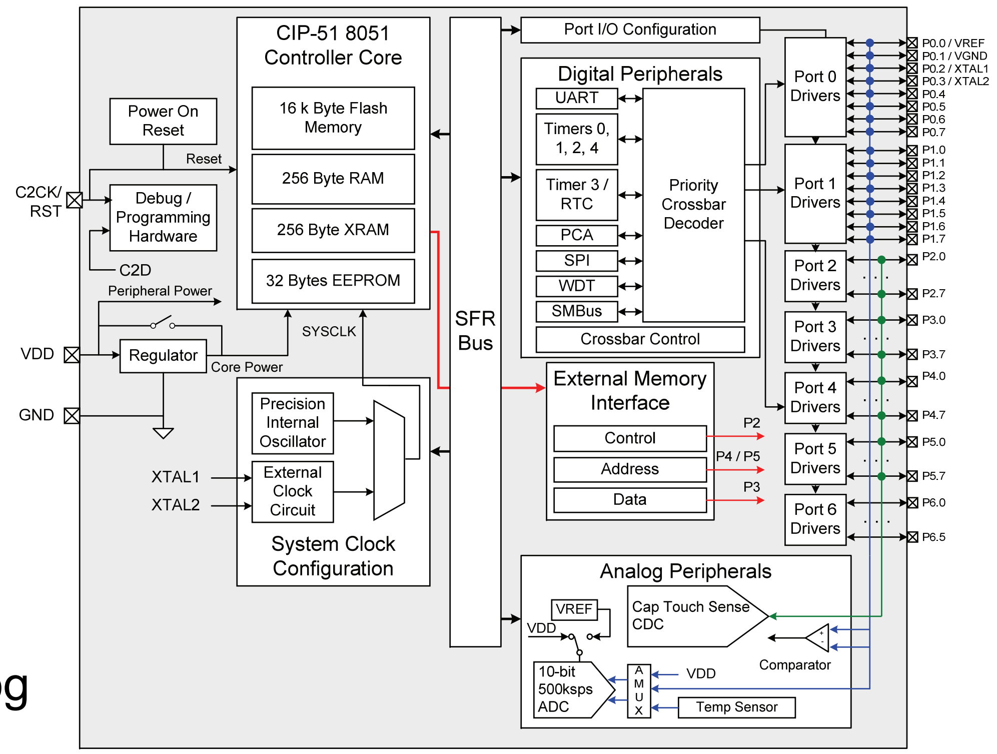
C8051F700 Serial Peripheral Interface (SPI) Overview

Agenda

- C8051F700 block diagram
- C8051F700 device features
- SPI operation overview
- SPI module overview
- Where to learn more

Introducing The C8051F700

- New patented capacitive touch sense
  - True capacitance-to-digital converter
  - Robust and responsive
  - Easy to use
- High performance MCU
  - 25 MHz 8051 CPU
  - Best in class ADC
  - 16 kB flash
  - 32 B data-EEPROM
- 54 multi-function GPIO
  - User configured as digital or analog
  - Digital crossbar assigns pins
  - Up to 32 capacitive touch sense inputs
  - Available in TQFP64, TQFP48, and QFN48 (7x7 mm) packages



C8051F700 Product Family Selection

Part Number	MIPS (peak)	FLASH Memory (bytes)	Data EEPROM (bytes)	RAM (bytes)	Digital Port I/O Pins	Serial Buses	Timers (16-bit)	PCA Chnls	Internal Osc	Cap Touch Sense	ADC0	Temp Sensor	VREF	Comp.	Package
C8051F700-GQ	25	15kB	32	512	54	UART, I2C, SPI	4	3	2%	Y	10-Bit	Y	Y	1	QFP64
C8051F701-GQ	25	15kB	32	512	54	UART, I2C, SPI	4	3	2%	Y	-			1	QFP64
C8051F702-GQ	25	16kB	-	512	54	UART, I2C, SPI	4	3	2%	Y	10-Bit	Y	Y	1	QFP64
C8051F703-GQ	25	16kB	-	512	54	UART, I2C, SPI	4	3	2%	Y	-			1	QFP64
C8051F704-GQ	25	15kB	32	512	39	UART, I2C, SPI	4	3	2%	Y	10-Bit	Y	Y	1	QFP48
C8051F704-GM	25	15kB	32	512	39	UART, I2C, SPI	4	3	2%	Y	10-Bit	Y	Y	1	QFN48
C8051F705-GQ	25	15kB	32	512	39	UART, I2C, SPI	4	3	2%	Y	-			1	QFP48
C8051F705-GM	25	15kB	32	512	39	UART, I2C, SPI	4	3	2%	Y	-			1	QFN48
C8051F706-GQ	25	16kB	-	512	39	UART, I2C, SPI	4	3	2%	Y	10-Bit	Y	Y	1	QFP48
C8051F706-GM	25	16kB	-	512	39	UART, I2C, SPI	4	3	2%	Y	10-Bit	Y	Y	1	QFN48
C8051F707-GQ	25	16kB	-	512	39	UART, I2C, SPI	4	3	2%	Y	-			1	QFP48
C8051F707-GM	25	16kB	-	512	39	UART, I2C, SPI	4	3	2%	Y	-			1	QFN48
C8051F708-GQ	25	8kB	32	512	54	UART, I2C, SPI	4	3	2%	Y	10-Bit	Y	Y	1	QFP64
C8051F709-GQ	25	8kB	32	512	54	UART, I2C, SPI	4	3	2%	Y	-			1	QFP64
C8051F710-GQ	25	8kB	-	512	54	UART, I2C, SPI	4	3	2%	Y	10-Bit	Y	Y	1	QFP64
C8051F711-GQ	25	8kB	-	512	54	UART, I2C, SPI	4	3	2%	Y	-			1	QFP64
C8051F712-GQ	25	8kB	32	512	39	UART, I2C, SPI	4	3	2%	Y	10-Bit	Y	Y	1	QFP48
C8051F712-GM	25	8kB	32	512	39	UART, I2C, SPI	4	3	2%	Y	10-Bit	Y	Y	1	QFN48
C8051F713-GQ	25	8kB	32	512	39	UART, I2C, SPI	4	3	2%	Y	-			1	QFP48
C8051F713-GM	25	8kB	32	512	39	UART, I2C, SPI	4	3	2%	Y	-			1	QFN48
C8051F714-GQ	25	8kB	-	512	39	UART, I2C, SPI	4	3	2%	Y	10-Bit	Y	Y	1	QFP48
C8051F714-GM	25	8kB	-	512	39	UART, I2C, SPI	4	3	2%	Y	10-Bit	Y	Y	1	QFN48
C8051F715-GQ	25	8kB	-	512	39	UART, I2C, SPI	4	3	2%	Y	-			1	QFP48
C8051F715-GM	25	8kB	-	512	39	UART, I2C, SPI	4	3	2%	Y	-			1	QFN48

- 24 unique part numbers
  - Choice of flash size
  - Can select EEPROM (in larger flash size, EEPROM is traded for 1 kB flash)
  - ADC or no-ADC
  - Capacitive touch sense option

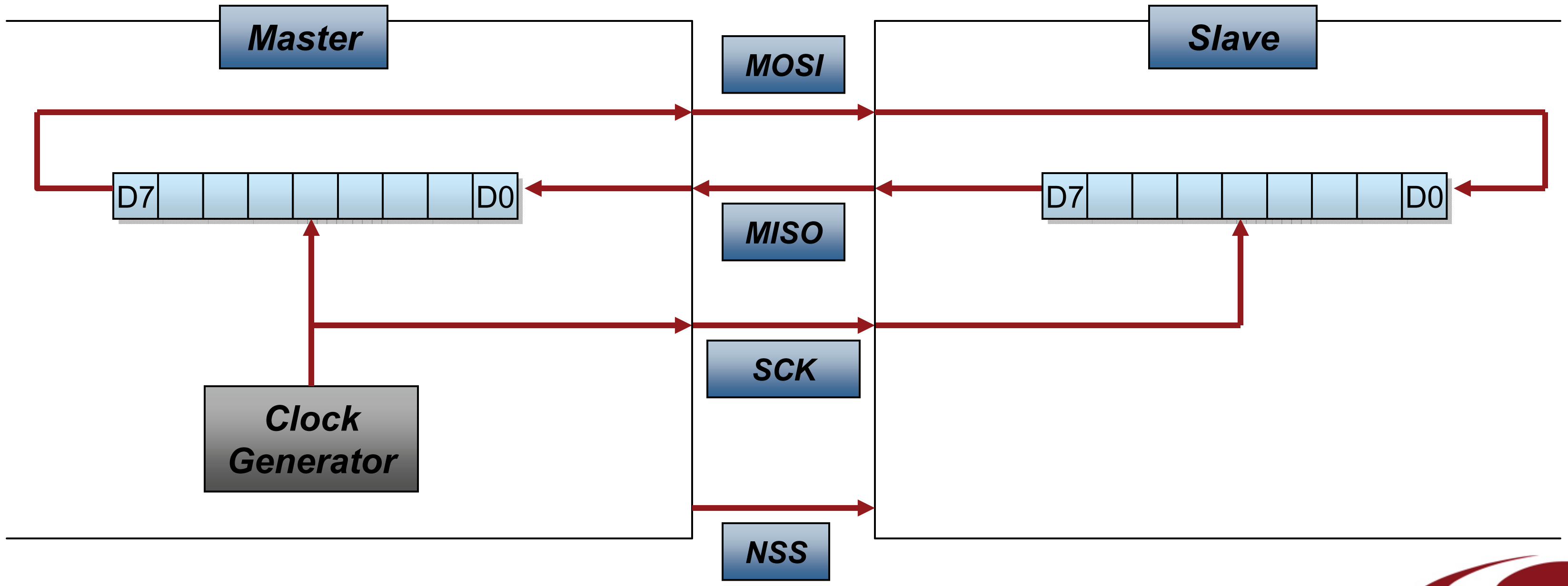
SILICON LABS

SILICON LABS

Serial Peripheral Interface (SPI)

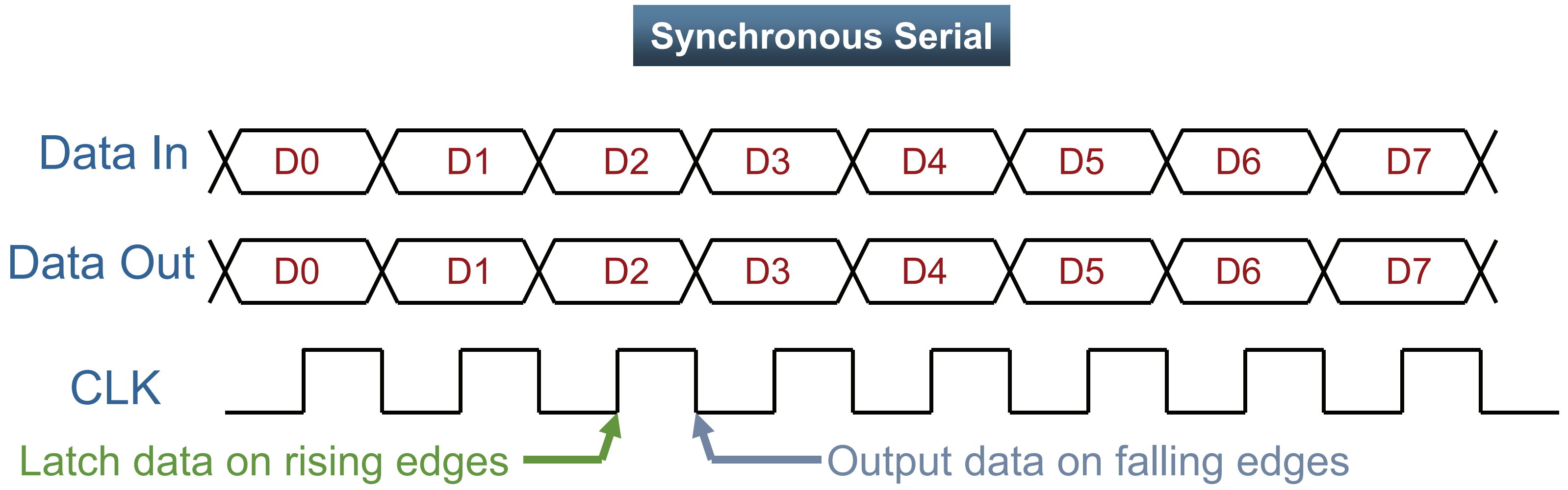
SPI Overview

- **Serial Peripheral Interface (SPI)**
  - Master/Slave operation
  - Full duplex or single wire operation
  - Programmable transmit bit rates
  - Double buffered



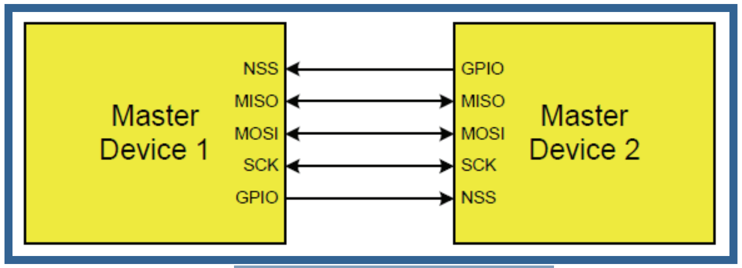
Synchronous Serial Communication

- Serial communication implies sending data bit by bit over a single wire
- Synchronous serial requires the clock signal to be transmitted from the source along with the data
- Data rate for the link must be the same for the transmitter and the receiver

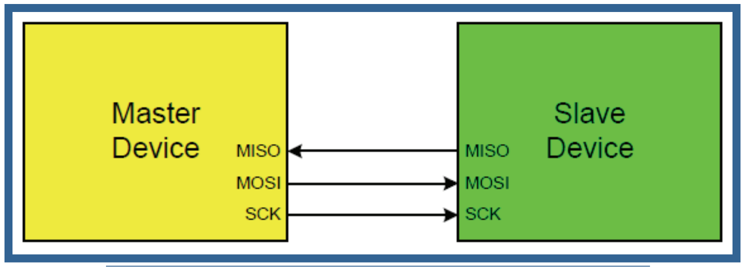


SPI Configurations

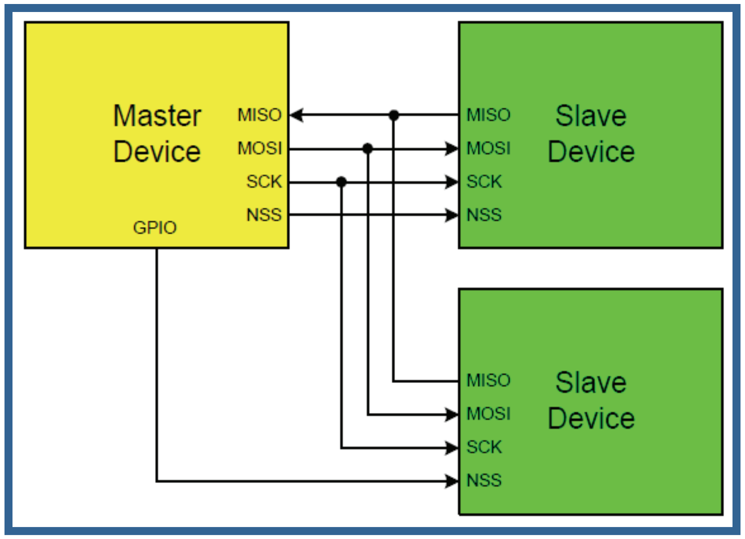
- Multi-master
- Single master with single slave
- Single master with several slaves



Multi-master



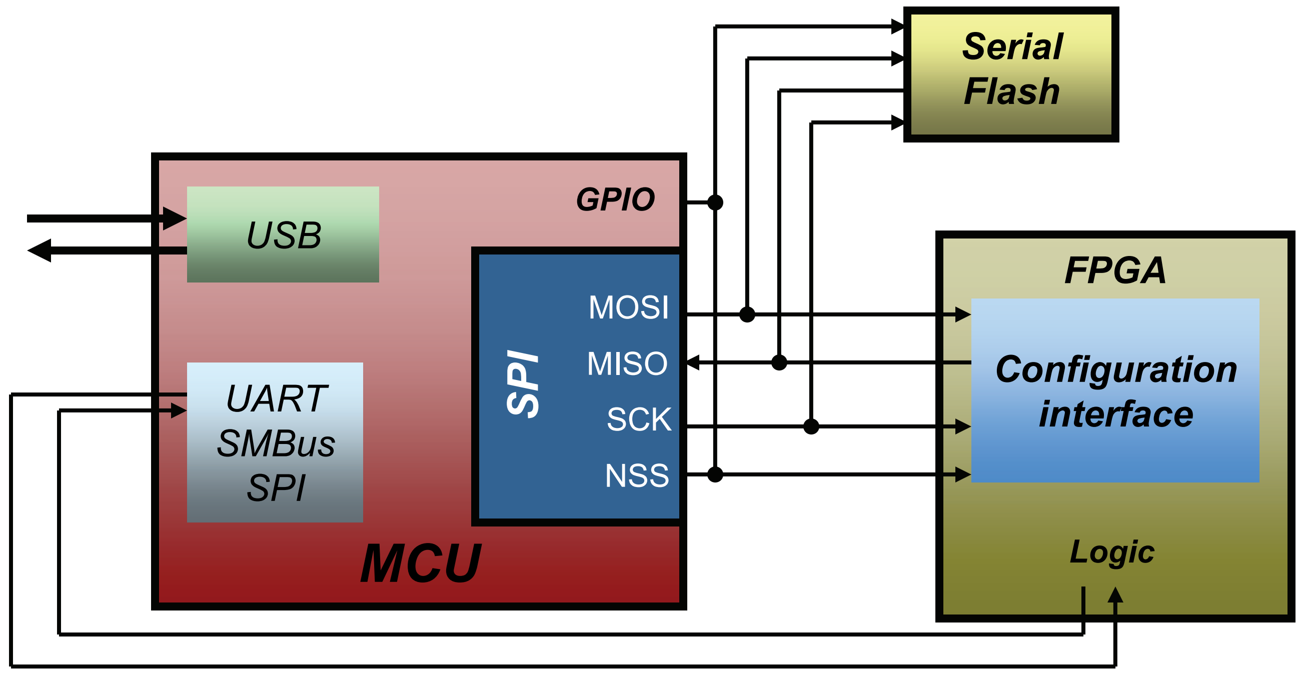
Single master/single slave



Single master/multiple slaves

SPI Interface Application Example

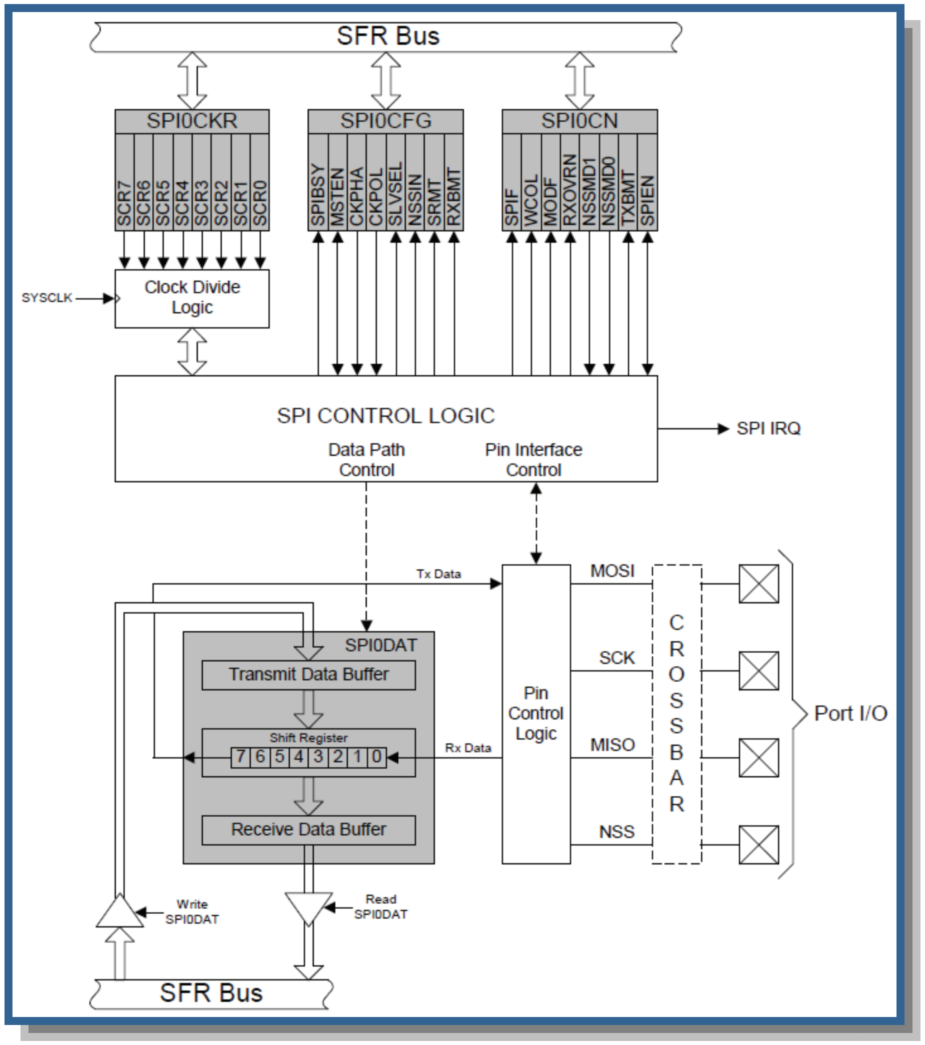
- Need a solution that provides the capability to download the FPGA configuration file
  - MCU provides the bridge between a host application that has the configuration
  - MCU can retrieve configuration file from on board serial flash
  - MCU can provide additional functionality to the FPGA after configuration
    - USB 2 UART bridge



Example SPI Application – FPGA configuration interface

Silicon Labs Enhanced SPI Module

- Full duplex synchronous serial communications
- Master or slave operation
  - Supports multiple masters or slave on a single SPI bus
- 3 or 4 wire operation
- Up to 12.5 Mbps operation in master mode
- 2.5 Mbps operation in slave mode
  - 6.25 Mbps operation in half duplex mode
- Programmable clock phase and polarity settings



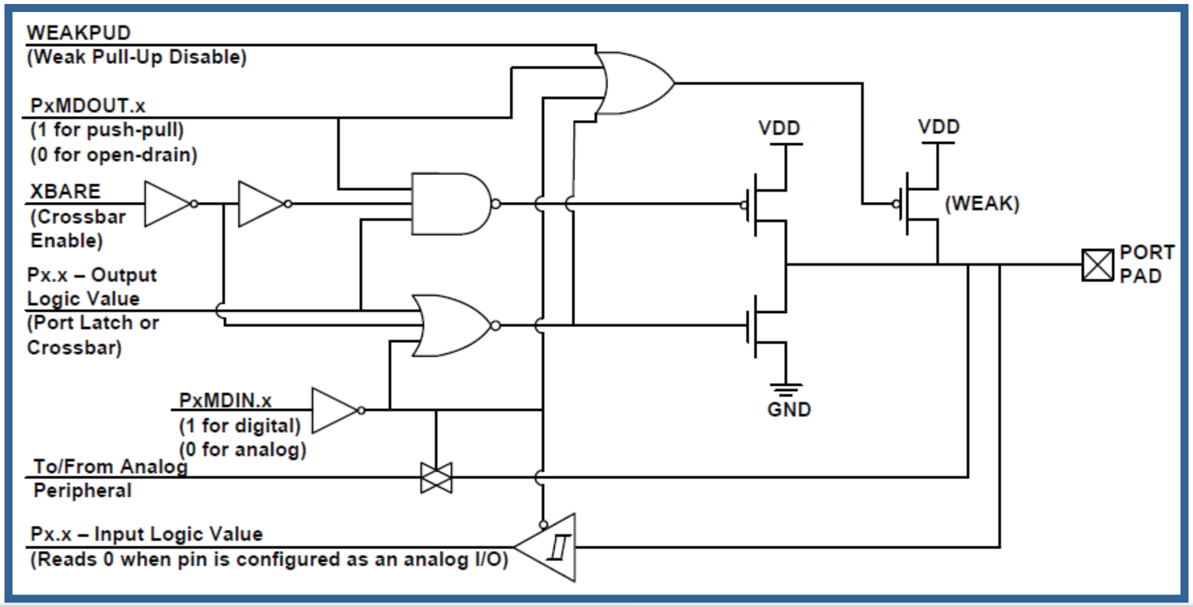
SPI Programming Steps

- ***Step 1***: Set port pins to digital inputs and push-pull outputs using the port I/O registers
- ***Step 2***: Enable the SPI module in the crossbar using the XBAR register
- ***Step 3***: Set the master or slave operating mode of the SPI peripheral using the SPI0CFG register as well as the clock phase and polarity
- ***Step 4***: Select the SPI interface as 3 or 4 wire slave or master and enable the SPI peripheral
- ***Step 5***: Set the SPI clock speed using the SPI0CKR register
- ***Step 6***: Enable the interrupts

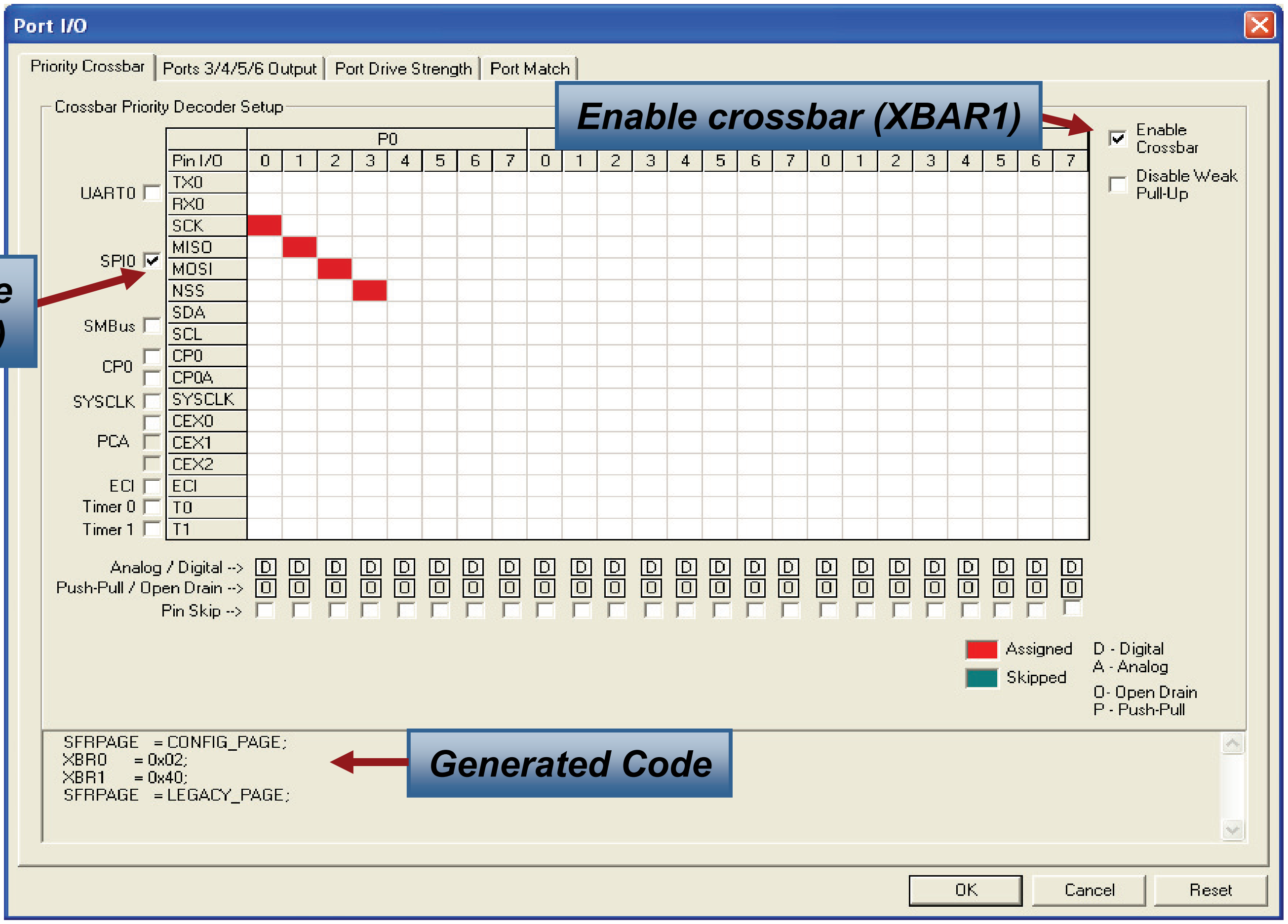
Digital I/O Pins

- All port pins can be used for digital I/O
- Port pin configured as digital using PxMDIN register bits set to a ‘1’
- The output mode is selected to be push-pull using the PxMDOUT bits set to a ‘1’

PxMDIN	PxMDOUT	Px	Description
1	0	0	Open drain low
1	0	1	Open drain high/digital Input
1	1	0	Push Pull: pin driven Low
1	1	1	Push Pull: pin driven High



Crossbar Pin Assignment Using Config Wizard


Port I/O
Priority Crossbar | Ports 3/4/5/6 Output | Port Drive Strength | Port Match
OK
Cancel
Reset

Crossbar Priority Decoder Setup

		P0																					
	Pin I/O	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
UART0 ☐	TX0																								
	RX0																								
SPI0 ✓	SCK	🟥																							
	MISO		🟥																						
	MOSI			🟥																					
	NSS				🟥																				
SMBus ☐	SDA																								
	SCL																								
CP0 ☐	CP0																								
☐	CP0A																								
SYSCLK ☐	SYSCLK																								
☐	CEX0																								
PCA ☐	CEX1																								
☐	CEX2																								
ECI ☐	ECI																								
Timer 0 ☐	T0																								
Timer 1 ☐	T1																								
	Analog / Digital -->	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
	Push-Pull / Open Drain -->	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O
	Pin Skip -->	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

✓ Enable Crossbar

☐ Disable Weak Pull-Up

🟥	Assigned	D - Digital	A - Analog
🟩	Skipped	O- Open Drain	P - Push-Pull

```
SFRPAGE   = CONFIG_PAGE;
XBR0      = 0x02;
XBR1      = 0x40;
SFRPAGE   = LEGACY_PAGE;
```

Enable crossbar (XBAR1)

Enable SPI0 to the crossbar (XBAR0)

Generated Code

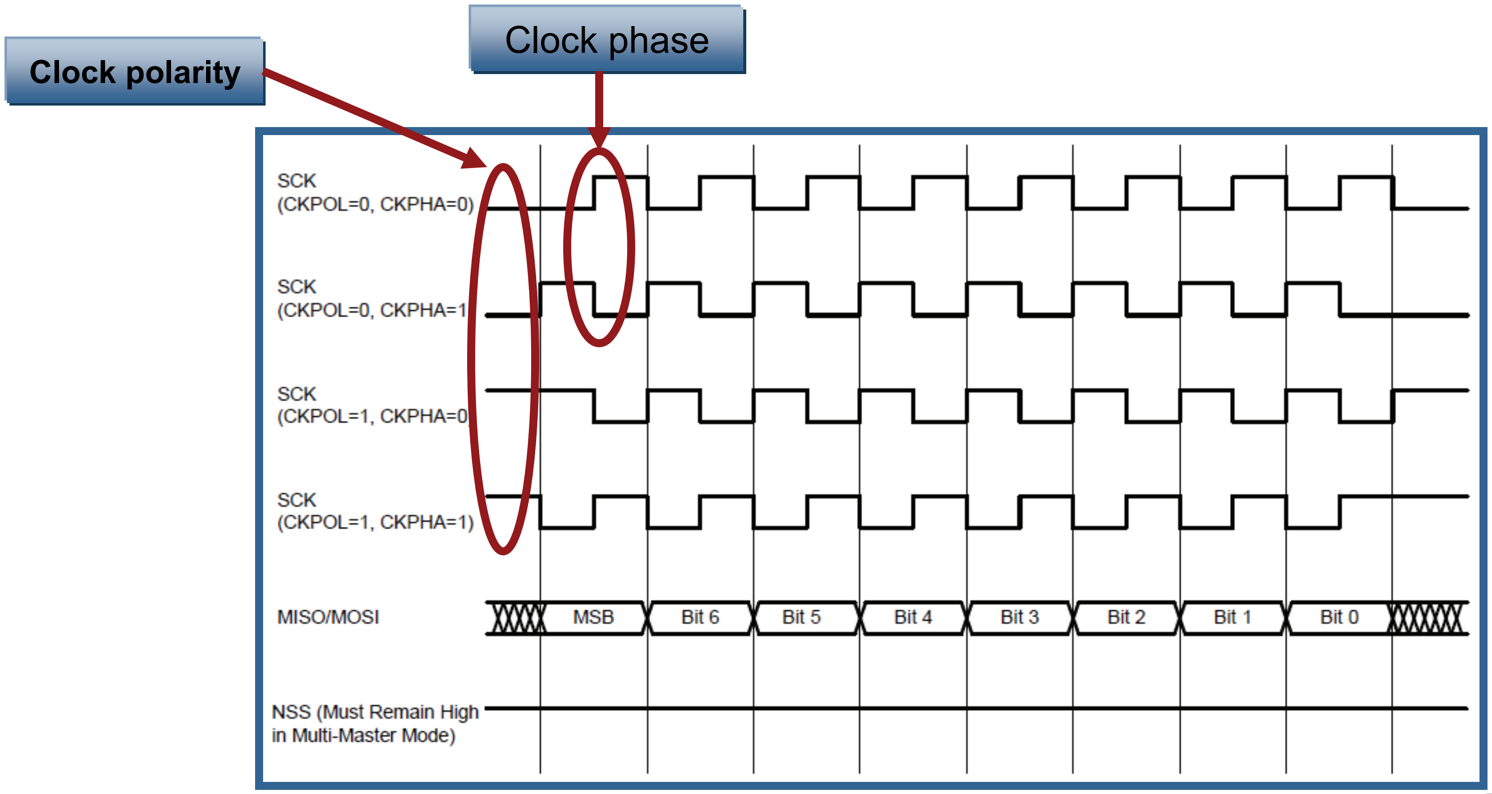
- SPI0 has the second highest crossbar priority and is assigned to P0.0 through P0.3 when enabled
- XBARE: enable the crossbar

SPI Programming Steps

- ***Step 1***: Set port pins to digital inputs and push-pull outputs using the port I/O registers
- ***Step 2***: Enable the SPI module in the crossbar using the XBAR register
- ***Step 3***: Set the master or slave operating mode of the SPI peripheral using the SPI0CFG register as well as the clock phase and polarity.
- ***Step 4***: Select the SPI interface as 3 or 4 wire slave or master and enable the SPI peripheral
- ***Step 5***: Set the SPI clock speed using the SPI0CKR register
- ***Step 6***: Enable the interrupts

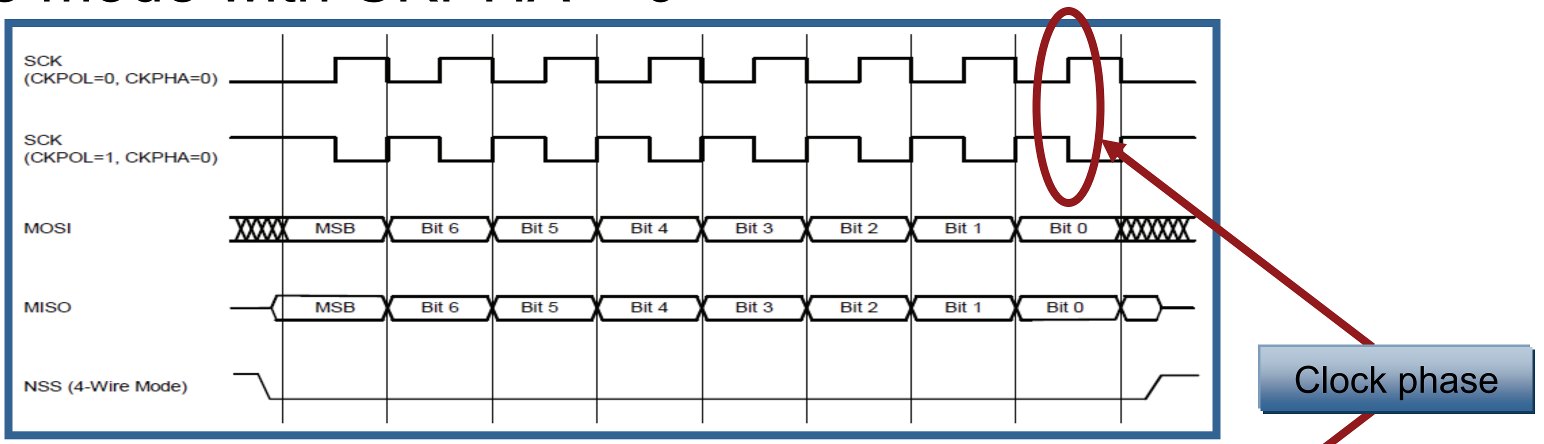
Clock Polarity and Phase

- Four configurations of clock polarity and phase controlled by the SPI0CFG register
  - Clock phase determines the clock edge used to latch the data
  - Clock polarity selects between an active high or active low clock
- Master and slave devices must be configured to use the same clock polarity and phase settings

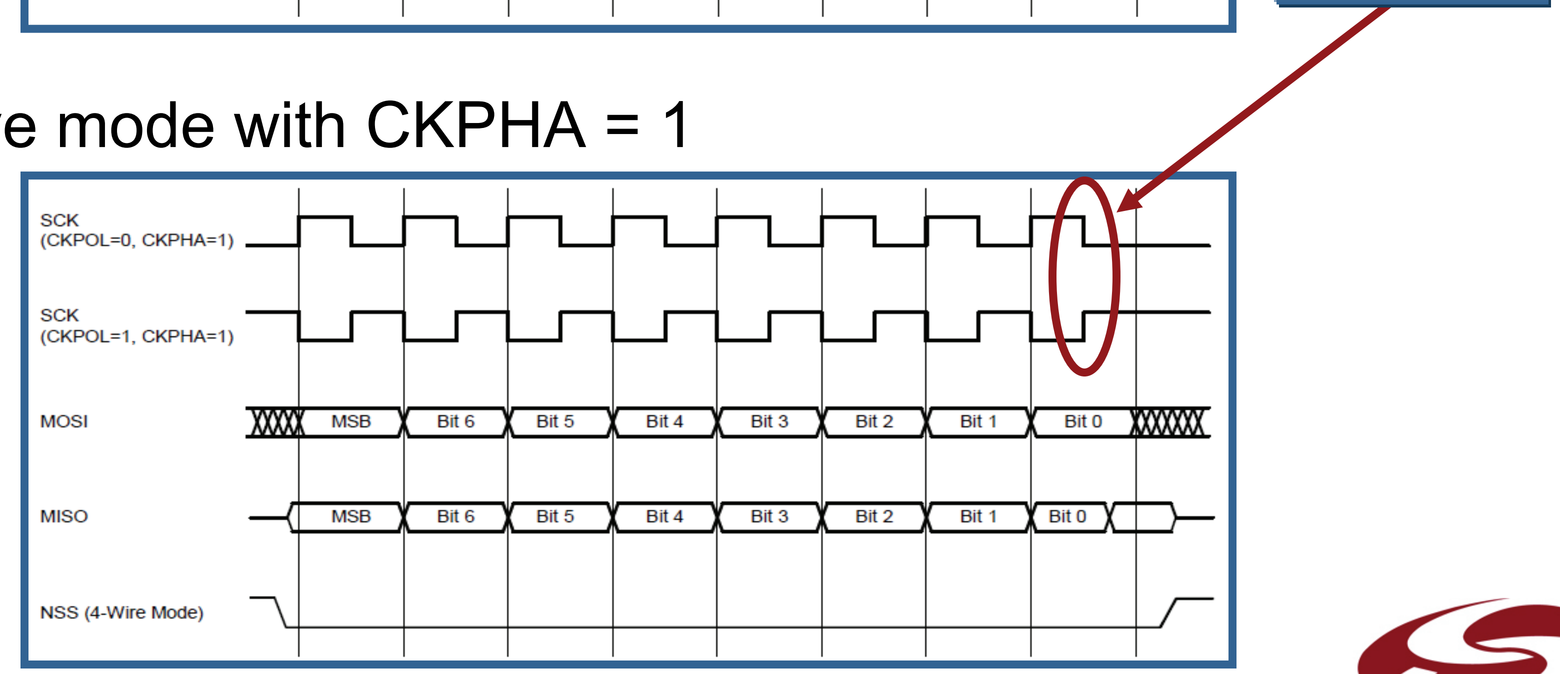


Slave Mode Clock Polarity and Phase

- Slave mode with CKPHA = 0



- Slave mode with CKPHA = 1



SPI0 Configuration: SPI0CFG Register

Bits	Name	Function
7	SPIBSY	**SPI Busy** This bit is set to logic 1 when a SPI transfer is in progress (master or slave mode)
6	MSTEN	**Master Mode Enable** 0: Disable master mode. Operate in slave mode 1: Enable master mode. Operate as a master
5	CKPHA	**SPI0 Clock Phase** 0: Data centered on first edge of SCK period 1: Data centered on second edge of SCK period
4	CKPOL	**SPI0 Clock Polarity** 0: SCK line low in idle state 1: SCK line high in idle state
3	SLVSEL	**Slave Selected Flag** This bit is set to logic 1 whenever the NSS pin is low indicating SPI0 is the selected slave. It is cleared to logic 0 when NSS is high (slave not selected). This bit does not indicate the instantaneous value at the NSS pin, but rather a de-glitched version of the pin input.
2	NSSIN	**NSS Instantaneous Pin Input** This bit mimics the instantaneous value that is present on the NSS port pin at the time that the register is read. This input is not de-glitched.
1	SRMT	**Shift Register Empty (valid in slave mode only)** This bit will be set to logic 1 when all data has been transferred in/out of the shift register, and there is no new information available to read from the transmit buffer or write to the receive buffer. It returns to logic 0 when a data byte is transferred to the shift register from the transmit buffer or by a transition on SCK. SRMT = 1 when in Master Mode.
0	RXBMT	**Receive Buffer Empty (valid in slave mode only)** This bit will be set to logic 1 when the receive buffer has been read and contains no new information. If there is new information available in the receive buffer that has not been read, this bit will return to logic 0. RXBMT = 1 when in Master Mode.

SPI0 Control: SPI0CN Register

Bits	Name	Function
7	SPIF	**SPI0 Interrupt Flag** This bit is set to logic 1 by hardware at the end of a data transfer
6	WCOL	**Write Collision Flag** This bit is set to logic 1 if a write to SPI0DAT is attempted when TXBMT is 0. When this occurs, the write to SPI0DAT will be ignored, and the transmit buffer will not be written.
5	MODF	**Mode Fault Flag** This bit is set to logic 1 by hardware when a master mode collision is detected (NSS is low, MSTEN = 1, and NSSMD[1:0] = 01)
4	RXOVRN	**Receive Overrun Flag (valid in slave mode only)** This bit is set to logic 1 by hardware when the receive buffer still holds unread data from a previous transfer and the last bit of the current transfer is shifted into the SPI0 shift register.
3:2	NSSMD[1:0]	**Slave Select Mode** Selects between the following NSS operation modes: 00: 3-Wire Slave or 3-Wire Master Mode. NSS signal is not routed to a port pin. 01: 4-Wire Slave or Multi-Master Mode (Default). NSS is an input to the device 1x: 4-Wire Single-Master Mode. NSS signal is mapped as an output from the device and will assume the value of NSSMD0
1	TXBMT	**Transmit Buffer Empty** This bit will be set to logic 0 when new data has been written to the transmit buffer. When data in the transmit buffer is transferred to the SPI shift register, this bit will be set to logic 1, indicating that it is safe to write a new byte to the transmit buffer.
0	SPIEN	**SPI0 Enable** 0: SPI disabled 1: SPI enabled

SPI Programming Steps

- ***Step 1***: Set port pins to digital inputs and push-pull outputs using the port I/O registers
- ***Step 2***: Enable the SPI module in the crossbar using the XBAR register
- ***Step 3***: Set the master or slave operating mode of the SPI peripheral using the SPI0CFG register as well as the clock phase and polarity
- ***Step 4***: Select the SPI interface as 3 or 4 wire slave or master and enable the SPI peripheral
- ***Step 5***: Set the SPI clock speed using the SPI0CKR register
- ***Step 6***: Enable the interrupts

Clock Rate Settings

- Master mode clock setting derived from the system clock (SYSCLK)

$$f_{SCK} = \frac{SYSCLK}{2x(SPI0CKR[7:0]+1)}$$

$$or$$

$$SPI0CKR[7:0] = \frac{SYSCLK}{2f_{SCK}} - 1$$

- Example:
  - Desired SPI rate is 250 KHz
  - System clock = 24.5 MHz

$$SPI0CKR = \frac{24500000}{2x250000} - 1$$

$$SPI0CKR = 48$$

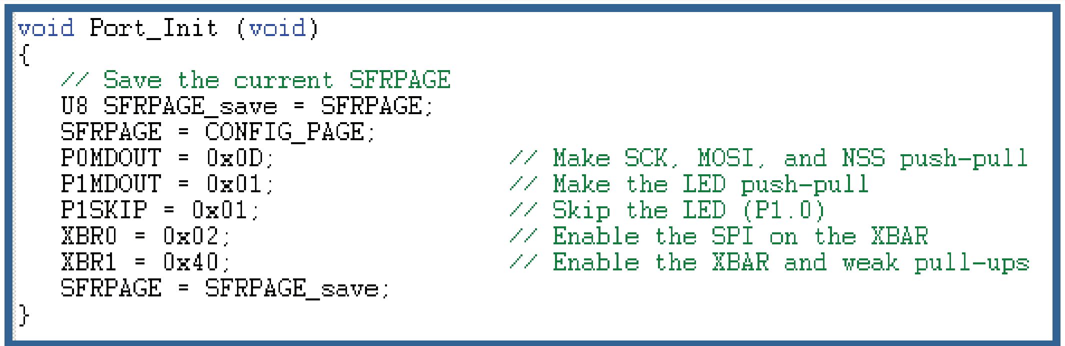
SILICON LABS

SPI0 Code: I/O and SPI0 Configuration

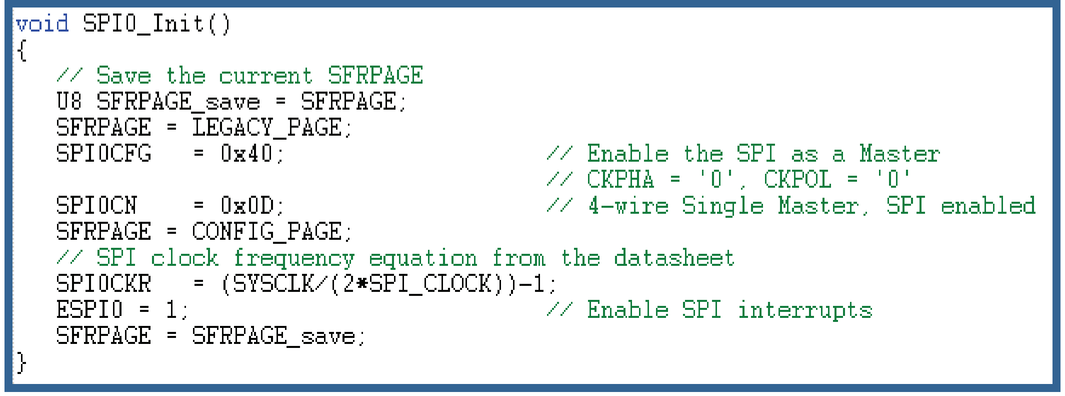
- ***Step 1***: Set port pins to digital inputs and push-pull outputs using the port I/O registers

- ***Step 2***: Enable the SPI module in the crossbar using the XBAR register

- ***Step 3***: Set the master or slave operating mode of the SPI peripheral using the SPI0CFG register as well as the clock pahse and polarity



```
void Port_Init (void)
{
   // Save the current SFRPAGE
   U8 SFRPAGE_save = SFRPAGE;
   SFRPAGE = CONFIG_PAGE;
   P0MDOUT = 0x0D;                    // Make SCK, MOSI, and NSS push-pull
   P1MDOUT = 0x01;                    // Make the LED push-pull
   P1SKIP = 0x01;                     // Skip the LED (P1.0)
   XBR0 = 0x02;                       // Enable the SPI on the XBAR
   XBR1 = 0x40;                       // Enable the XBAR and weak pull-ups
   SFRPAGE = SFRPAGE_save;
}
```



```
void SPI0_Init()
{
   // Save the current SFRPAGE
   U8 SFRPAGE_save = SFRPAGE;
   SFRPAGE = LEGACY_PAGE;
   SPI0CFG   = 0x40;                    // Enable the SPI as a Master
                                        // CKPHA = '0', CKPOL = '0'
   SPI0CN    = 0x0D;                    // 4-wire Single Master, SPI enabled
   SFRPAGE = CONFIG_PAGE;
   // SPI clock frequency equation from the datasheet
   SPI0CKR   = (SYSCLK/(2*SPI_CLOCK))-1;
   ESPI0 = 1;                           // Enable SPI interrupts
   SFRPAGE = SFRPAGE_save;
}
```

- ***Step 4***: Select the SPI interface as 3 or 4 wire slave or master and enable the SPI peripheral

- ***Step 5***: Set the SPI clock speed using the SPI0CKR register

- ***Step 6***: Enable the interrupts

Learn More at the Education Resource Center

- Visit the Silicon Labs website to get more information on Silicon Labs products, technologies and tools
- The Education Resource Center training modules are designed to get designers up and running quickly on the peripherals and tools needed to get the design done
  - http://www.silabs.com/ERC
  - http://www.silabs.com/mcu
- To provide feedback on this or any other training go to:

  http://www.silabs.com/ERC and click the link for feedback

SILICON LABS

SILICON LABS

www.silabs.com/MCU