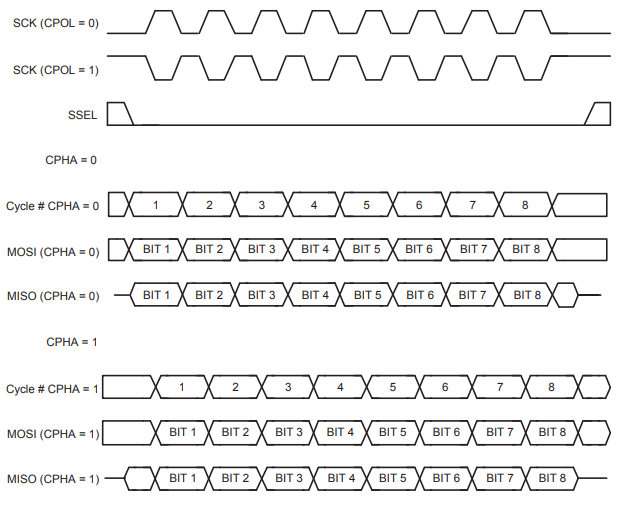
**Features**

* Synchronous Serial Full Duplex
* 8 to 16 bits per transfer
* maximum data rate of 1/8th of input clock rate
* full duplex
* multiple master and slaves
* standard CMOS I/O and not open drain

**Data Transfer**



* Four different types of data transfer based on CPOL and CPHA bits.
* When CPHA = 0, the SSEL signal is low, but when CPHA = 1, the SSEL signal can be high also.

|  |  |  |  |
| --- | --- | --- | --- |
| **CPOL:CPHA** | **When the first data bit is driven** | **When all other data bits are driven** | **When data is sampled** |
| **00** | Prior to first SCK rising edge | SCK falling edge | SCK rising edge |
| **01** | First SCK rising edge | SCK rising edge | SCK falling edge |
| **10** | Prior to first SCK falling edge | SCK rising edge | SCK falling edge |
| **11** | First SCK falling edge | SCK falling edge | SCK rising edge |

**Master Operation**

1. Set the SPI clock counter register to desired clock rate.
2. Set the SPI control register.
3. Write the data to be transmitted to the SPI data register. –to start the SPI data transfer.
4. Check the SPIF bit in SPI status register to be 1 – to indicate the completion of SPI data transfer.
5. Read the SPI Status register.
6. read the received data from SPI data register

**Slave Operation**

**Note: Required that the system clock is 8 times faster than SPI clock from master**

1. Set the SPI control register.
2. Write the data to be transmitted to the SPI data register (optional). Note that this can only be done when a slave SPI transfer is not in progress.
3. Wait for the SPIF bit in the SPI status register to be set to 1. The SPIF bit will be set
4. After the last sampling clock edge of the SPI data transfer.
5. Read the SPI status register.
6. Read the received data from the SPI data register (optional).

**Register Description**

**SPI Control Register (S0SPCR - 0xE002 0000)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **15:12 – Reserved** | **11:8 – BITS** | **7 – SPIE** | **6 – LSBF** | **5 – MSTR** | **4 – CPOL** | **3 – CPHA** | **2 – BitEnable** | **1:0 - Reserved** |

* **BITS –** active only when BitEnable is set to 1 – selects the number of bits per transfer

|  |  |
| --- | --- |
| **BITS** | **No. of bits per transfer** |
| **1000** | 8 bits per transfer |
| **1001** | 9 bits per transfer |
| **1010** | 10 bits per transfer |
| **1011** | 11 bits per transfer |
| **1100** | 12 bits per transfer |
| **1101** | 13 bits per transfer |
| **1110** | 14 bits per transfer |
| **1111** | 15 bits per transfer |
| **0000** | 16 bits per transfer |

**SPIE**

* 1 - to enable SPIF interrupt

**LSBF**

* **0 –** MSB first
* **1 –** LSB first

**MSTR**

* **0 –** Slave Mode
* **1 –** Master Mode

**BitEnable**

* **0 –** SPI controller send and receives 8 bits per transfer
* **1 –** SPI controller send and receives bits specified by BITS field

**SPI Status Register (S0SPSR - 0xE002 0004)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **7 – SPIF** | **6 – WCOL** | **5 – ROVR** | **4 – MODF** | **3 – ABRT** | **2:0 -Reserved** |

* **SPIF –** indicates the SPI transfer complete flag – not the SPI interrupt flag

**SPI Data Register (S0SPDR - 0xE002 0008)**

* **16** bit register

**SPI Clock Counter Register (S0SPCCR - 0xE002 000C)**

* the clock for SPI is ***P\_CLK / S0SPCCR***

**SPI Interrupt register (S0SPINT - 0xE002 001C)**

* **SPI Interrupt Flag – set by the SPI interface to generate an interrupt**

**Steps:**

1. Initialize functions for SPI\_SCK, SPI\_MOSI and SPI\_MISO using **PINSE0.**
2. **Important: SPI\_SSEL** as output and not its corresponding function.
3. Configure the SPI using **S0SPCR** register.
4. Set the SPI clock u sing the formula in **S0SPCCR** register.
5. Transfer can be done by writing into **S0SPDR** register.
6. Transfer complete can be checked by checking the 7th bit of **S0SPSR** register.