CSE3103 Microprocessor and Microcontroller: SPI – Serial Peripheral Interface

Dr. Mosaddek Tushar, Professor

Computer Science and Engineering, University of Dhaka

March 29, 2022

Contents

Documents and Video on SPI

Must See these videos (Click on the text below)

1. SPI: The serial peripheral interface – Ben Eater
2. Serial Peripheral Interface (SPI) Basics
3. Texas Instrument – SPI Communication mode
4. Electronic Basics #36: SPI and how to use it – GreatScott
5. BME280 I2C or SPI Temperature Humidity Pressure Sensor ARDUINO
6. BME280 with STM32
7. BME280 Datasheet
8. Calibrate a BME280/680 Pressure Sensor or Barometer
9. BME280 – I2C

SPI – Standard

- Serial Communication Interface
- Synchronous Serial communication between MCUs and Devices
- I/O pins are dedicated to SPI communication
  - MISO – Master In Slave Out data
  - MOSI – Master Out and Slave In
  - SCK – Serial clock out pin for master and Clock in for slave
  - NSS – Select an individual slave (like chip select)
  - One master and one or more slave devices
  - At least two wire – Clock and synchronous Data transmission (depending on communication mode)

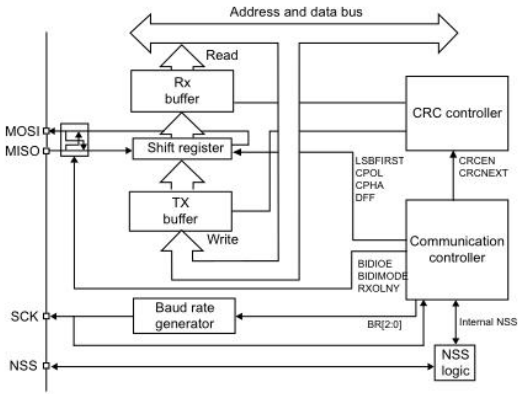


Figure 1: SPI block diagram

Communication between Master and Slave

Communication configuration

- 3 or 4-wire communication (with hardware NSS management)
- 2 or three wires (with software NSS management)
- Communication mode
  - Full-duplex
  - Half-duplex
  - Simplex Communication

Full-Duplex Mode

- Shift registers of both master and slave are connected two unidirectional line (MISO and MOSI)
- During the communication, data is shifted synchronously on the SCK clock edge and sent via MOSI to slave MISO

Data Flow

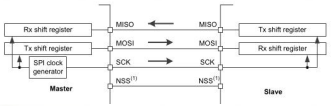


Figure 2: Full duplex

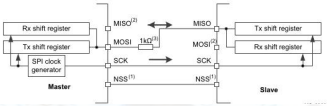


Figure 3: Half-Duplex

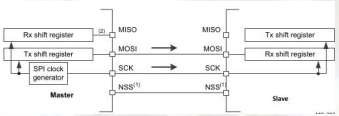


Figure 4: Simplex SPI

Half-Duplex and Simplex Mode

Bidirectional communication with help of BIDIMODE in SPIx_CR1 register

- Only one transmission at a time (Half Duplex)
  - Transmission master to slave or
  - Slave to master
  - transfer direction selected reciprocally by both master and slave with the BDIOE bit in their SPIx_CR1 registers
- when communication finished, the pins (are free) can be used for GPIO

Simplex mode

- Transmit only using RXONLY=0 in the SPIx_CR2 register
- Receive only mode by using RXONLY=1 in the SPIx_CR2 register

Standard Multi-Slave mode

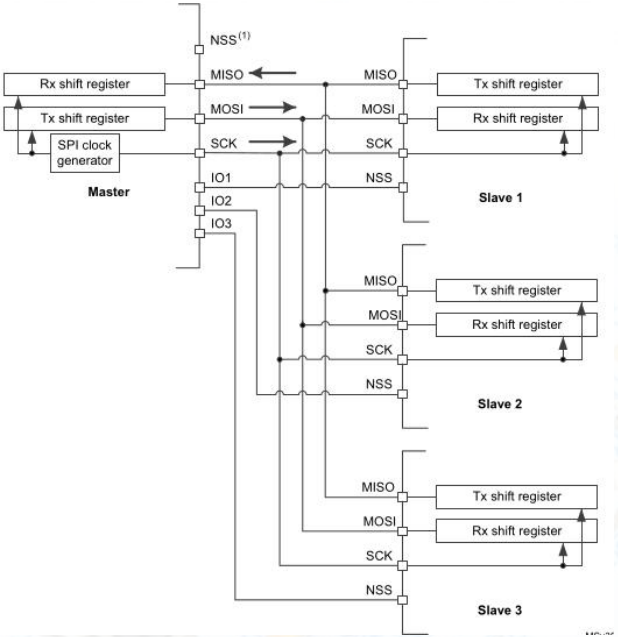


Figure 5: Single master and Multi-Slave

Multi-Master mode

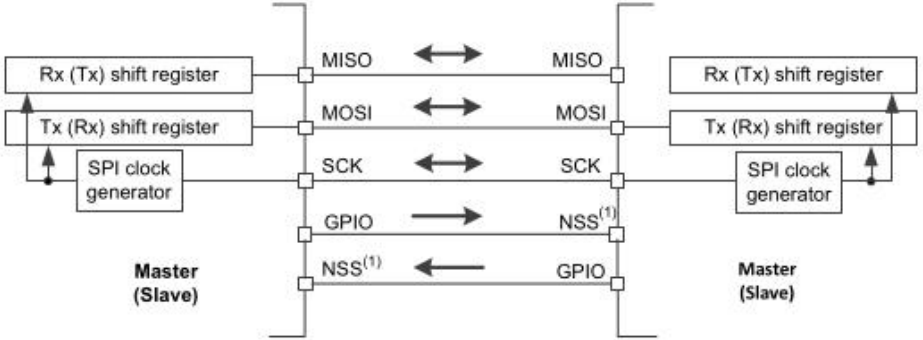


Figure 6: Multi-Master Mode

- Potential conflict to select both as master (MODF event) – mode fault
- In idle state both remains slave
- NSS pins are configured at hardware input mode

Slave Select Management (NSS pin)

SSM & SSOE in SPIx_CR1

- Software NSS pin management (SSM=1)
  - Slave select information is driven by SSI bit value in register SPIx_CR1
- Hardware NSS pin Management (SSM=0), there are two possible configuration. NSS output configuration depends on SSOE bit in register SPIx_CR1
  - NSS output enable (SSM=0, SSOE=1), when MCU act as only master
  - NSS output disable (SSM=0, SSOE=0), MCU act as master on the bus, this configuration allow multimaster capability.
    - If NSS pin pull low, SPI enters master mode fault state. Device automatically reconfigured in slave mode. NSS works as a chip select

Timing Diagram

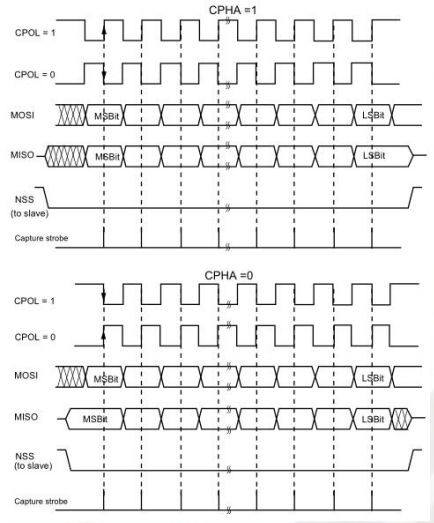
Clock Pulse and Polarity control

- When bit CPOL=0, idle state of the clock set as low for master and salve
- For bit CPOL=1, idle state is high
- CPHA = 1, second edge on the clock capture the first bit of the data (falling edge if CPOL=0)
- FOR CPHA = 0, first edge on the clock capture the first bit of the data. Falling edge if CPOL=0
- Combination of CPHA and CPOL bits selects the data capture clock edge
- Prior to changing the CPOL/CPHA bits the SPI must be disabled by resetting the SPE bit

Data Format

- SPI setup to shift LSB or MSB first depending on value LSBFIRST - bit in SPIx_CR1 bit 7
- Data frame 8 (0) or 16 (1) bit long dependent on DFF bit in SPIx_CR1 (bit-11)

SPI Data format and Timing (clock) Diagram



SPI Configuration

1. Write proper GPIO registers: Configure GPIO for MOSI, MISO and SCK pins.
2. Write to the SPI_CR1 register:
   a. Configure the serial clock baud rate using the BR[2:0] bits
   b. Configure the CPOL and CPHA bits combination to define one of the four relationships between the data transfer and the serial clock.
   c. Select simplex or half-duplex mode by configuring RXONLY or BIDIMODE and BIDIOE (RXONLY and BIDIMODE can't be set at the same time).
   d. Configure the LSBFIRST bit to define the frame format
   e. Configure the CRCEN and CRCEN bits if CRC is needed (while SCK clock signal is at idle state).
   f. Configure SSM and SSI
   g. Configure the MSTR bit (in multimaster NSS configuration, avoid conflict state on NSS if master is configured to prevent MODF error).
   h. Set the DFF bit to configure the data frame format (8 or 16 bits). Write to SPI_CR2 register:
   i. Configure SSOE
   j. Set the FRF bit if the TI protocol is required.
3. Write to SPI_CRCPR register: Configure the CRC polynomial if needed.
4. Write proper DMA registers: Configure DMA streams dedicated for SPI Tx and Rx in DMA registers if the DMA streams are used

Data Transmission and Reception Procedure

Data Transmission and Reception

- Rx and Tx buffer
  - Data received and stored in Rx buffer while transmitting Data – a read operation of SPix_DR register returns the Rx buffer data
  - Data stored in Tx buffer before transmission – write access to SPIx_DR register store data for transmission
- Transmission Handling
  - Shift register is loaded data from Tx buffer and sent to the MOSI line LSB or MSB first depending on LSBFIRST bit setting
  - TXE bit in status register is set when data copied from Tx (TDR) buffer to shift register
  - and interrupt can be generated is TXEIE bit of SPIx_CR2 is set
  - continuous transmission of byte can be done by checking TXE bit and copied data into DR register
  - when data writing to DR register, TXE bit is not set, however after writing TXE is set until data is not loaded to shift register
- Reception Handling
  - RXNE bit is set on the last clock sampling edge when data shifted to receiving shift register to Rx buffer
  - Read from DR register clear the RXNE bit
  - An interrupt can be generated if RXNEIE bit set into SPIx_CR2 register
  - An overrun condition occur when previous data byte is not cleared and results OVR-bit set.
  - Interrupt can be generated to handle the error by setting ERRIE bit of CR2 register

Data Transmission and Reception Procedure

Frame Sequence handling

- BSY-bit is set when data transmission is ongoing
- BSY flag remains set between frame (transmitted) on the master side
- In slave side it is low for a minimum duration of a SPI clock cycle between frames
- BSY flag can be used to wait until last byte transmission is complete
- for receive (half-duplex or simple) only mode, as soon as the half-duplex (BIDIMODE=1 and BIDIOE=0) or simplex (BIDIMODE=0 and RXONLY=1) is configured
- the clock signal is provided by master until receive only mode is disabled by the master
- However, master respects slave's capability, either slow down the clock or separate data session with sufficient delay
- In parallel slave transmission can be control by NSS pulse for multislave communication. For single slave it is not necessary.

Disabling SPI

When SPI is disable, Mandatory procedure needs to follow (Standard disable procedure)

- Pulling BSY and TXE flag to check if transmission is fully completed
- When NSS signal is managed (toggle) by GPIO, master needs to send proper end of NSS signal
- correct disable procedure:
  1. Wait until RXNE=1 to receive the last data.
  2. Wait until TXE=1 and then wait until BSY=0 before disabling the SPI.
  3. Read received data.

For receive only mode

1. Interrupt the receive flow by disabling SPI (SPE=0) in the specific time window while the last data frame is ongoing.
2. Wait until BSY=0 (the last data frame is processed).
3. Read received data.

SPI status register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	FRE	BSY	OVR	MODF	CRC ERR	UDR	CHSIDE	TXE	RXNE
							r	r	r	r	rc_w0	r	r	r	r

Figure 7: SPI Status Register

Status flag (SR) if set

- FRE – frame error. Set by hardware and cleared by software when SPx_SR register is read
- BSY – busy in communication or Tx buffer is not empty. Set and cleared by hardware
- OVR – overrun error. set by hardware and cleared by software sequence
- MODF – mode fault. set by hardware and cleared by software sequence
- CRCERR – set by hardware and cleared by software writing ‘0’
- UDR – under run flag. set by hardware and cleared by software sequence
- and so on . . .

Control Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BIDI MODE	BIDI OE	CRC EN	CRC NEXT	DFF	RX ONLY	SSM	SSI	*LSB FIRST*	SPE	BR [2:0]			MSTR	CPOL	CPHA
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Figure 8: SPI Control Register 1 (CR1)

Comments:
Baud Rate:[5:3] – 3-bits

- 000: fPCLK/2
- 001: fPCLK/4
- 010: fPCLK/8
- 011: fPCLK/16
- 100: fPCLK/32
- 101: fPCLK/64
- 110: fPCLK/128
- 111: fPCLK/256

Control Register 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXEIE	RXNEIE	ERRIE	FRF	Res.	SSOE	TXDMAEN	RXDMAEN
								rw	rw	rw	rw		rw	rw	rw

Figure 9: SPI control register 2 (CR2)

- FRF – frame format Motorola (0) or TI mode (1)
- RXDMAEN – Rx buffer DMA enable
- TXDMAEN – Tx buffer DMA enable
- SSOE – ‘0’ – SS output disable for multimaster, ‘1’ for master mode