Serial Peripheral Interface : SPI

Microcontroller Application and Development 2565

Sorayut Glomglome

Content

- What is it?
- Basic Serial Peripheral Interface (SPI)
- Capabilities
- Protocol
- Pro / Cons
- Uses
- STM32F767 SPI
- Conclusion

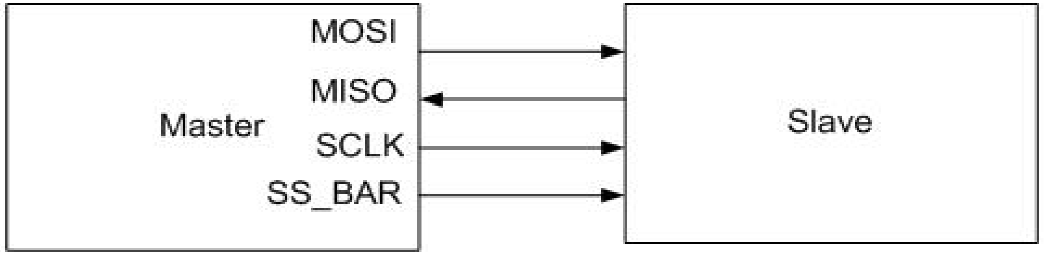
SPI Basics

- A communication protocol using 4 wires
  - ❑ Also known as a 4-wire bus
- Used to communicate across small distances
- Multiple Slaves, Single Master
- Synchronized

Capabilities of SPI

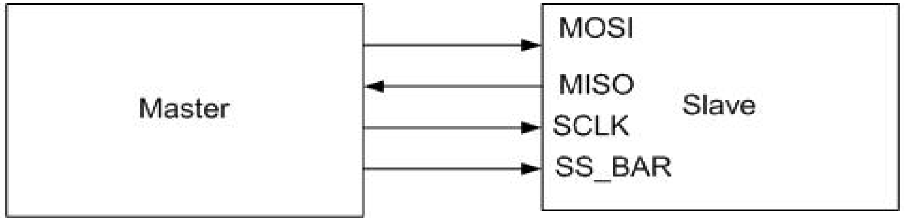
- Always Full Duplex
  - Communicating in two directions at the same time
- Multiple Mbps transmission speed
- Transfers data in 4 to 16-bit characters
- Multiple slaves
  - Daisy-chaining possible

Protocol



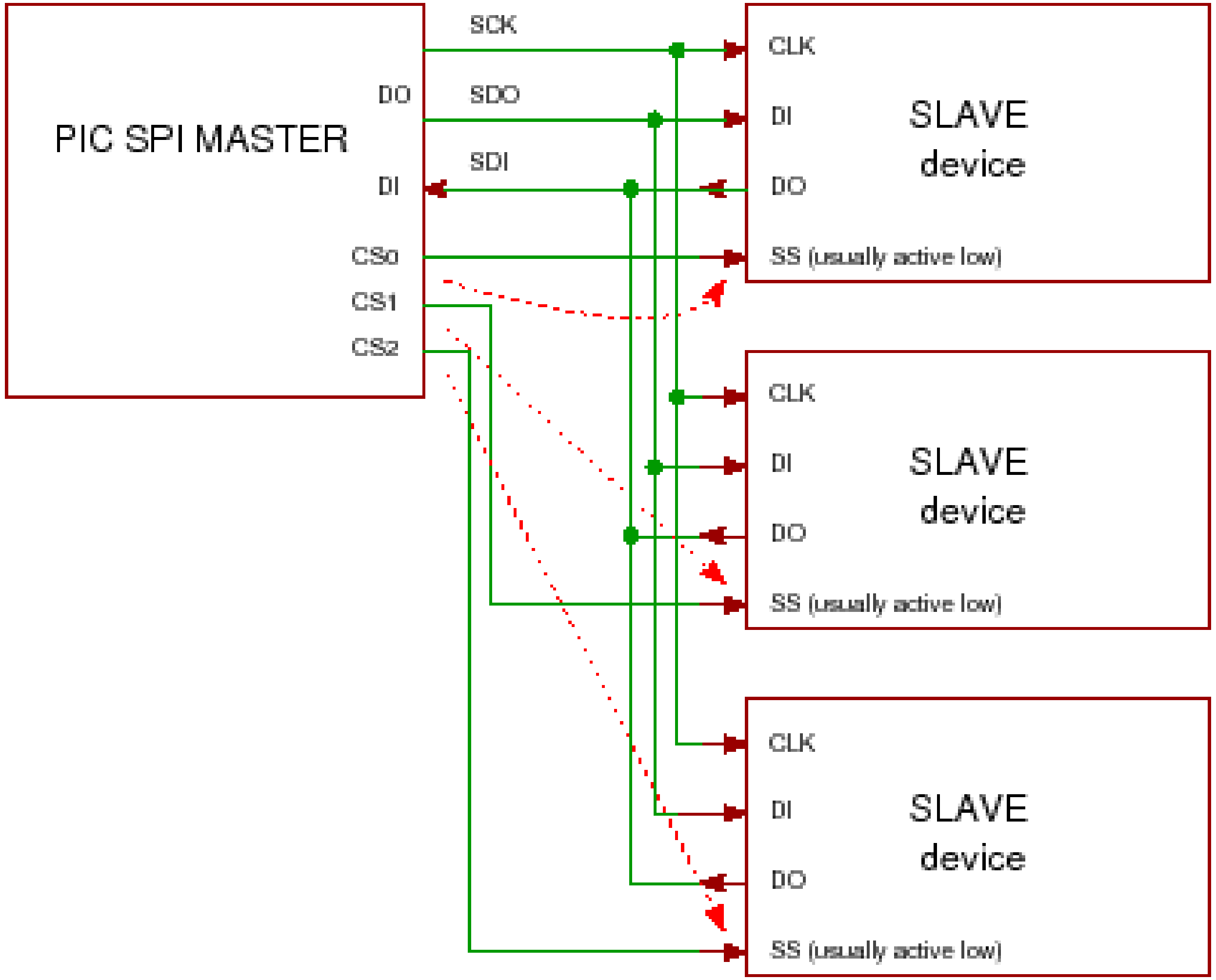
- Wires:
  - Master Out Slave In (MOSI)
  - Master In Slave Out (MISO)
  - System Clock (SCLK)
  - Slave Select 1…N
- Master Set Slave Select low
- Master Generates Clock
- Shift registers shift in and out data

Wires in Detail



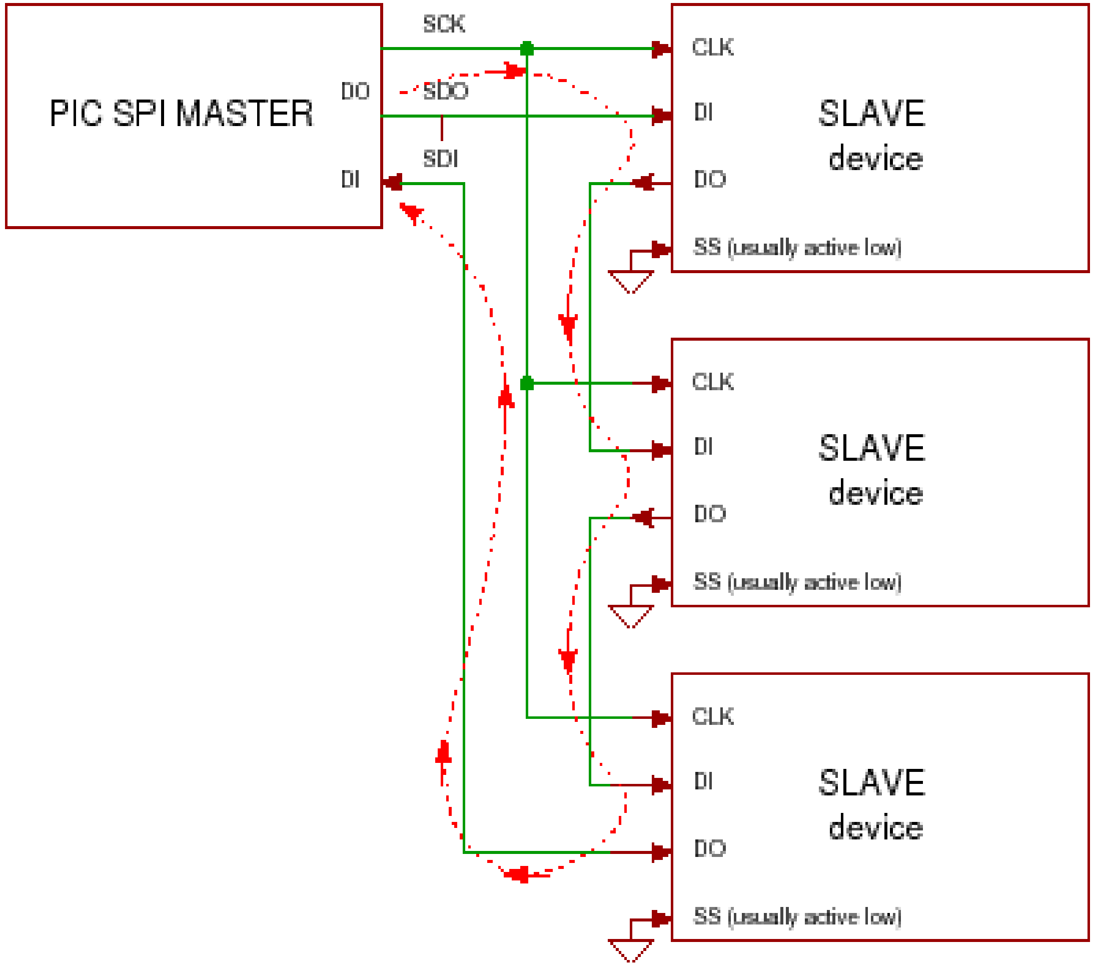
- MOSI – Carries data out of Master to Slave
- MISO – Carries data from Slave to Master
  - Both signals happen for every transmission
- SS_BAR – Unique line to select a slave
- SCLK – Master produced clock to synchronize data transfer

Master with Independent Slaves

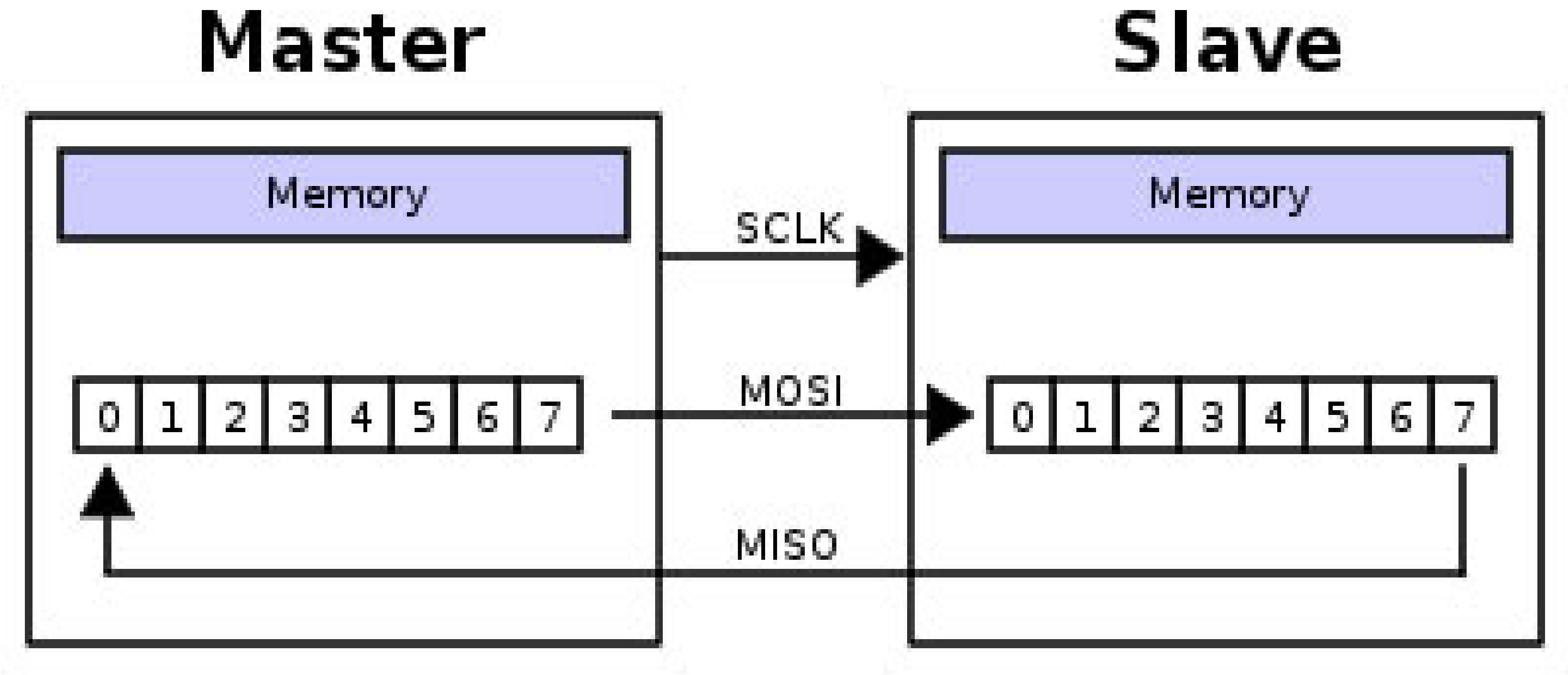


Master-Slaves/Daisy Chain

- Less wiring
- Slow data transfer
- Broadcast is also possible.



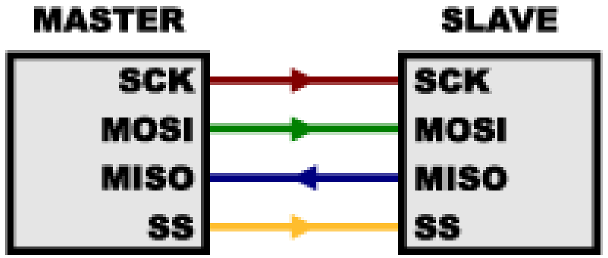
Shifting Protocol

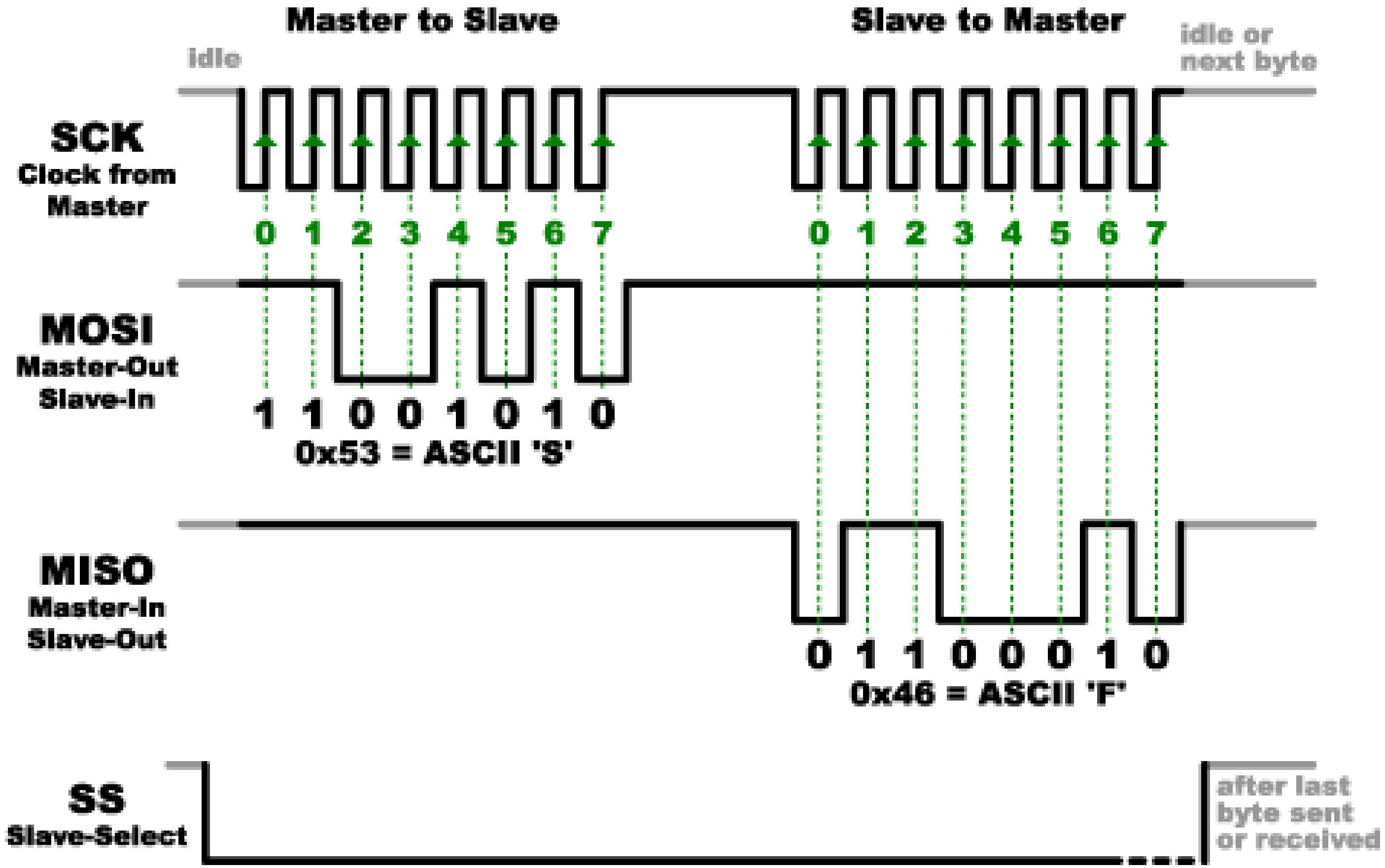


Master shifts out data to Slave, and shift in data from Slave

http://upload.wikimedia.org/wikipedia/commons/thumb/b/bb/SPI_8-bit_circular_transfer.svg/400px-SPI_8-bit_circular_transfer.svg.png

Data Transfer





Clock Modes

SPI-mode	CPOL	CPHA	Sampling
0	0	0	Leading (Rising) Edge
1	0	1	Trailing (Falling) Edge
2	1	0	Leading (Falling) Edge
3	1	1	Trailing (Rising) Edge

- Two phases and two polarities of clock => Four modes
- Master and selected slave must be in the same mode
- Master must change polarity and phase to communicate with slaves of different numbers

Clock & Sampling

SPI-mode	CPOL	CPHA	Sampling
0	0	0	Leading (Rising) Edge
1	0	1	Trailing (Falling) Edge
2	1	0	Leading (Falling) Edge
3	1	1	Trailing (Rising) Edge

Figure 76. SPI Transfer Format with CPHA = 0

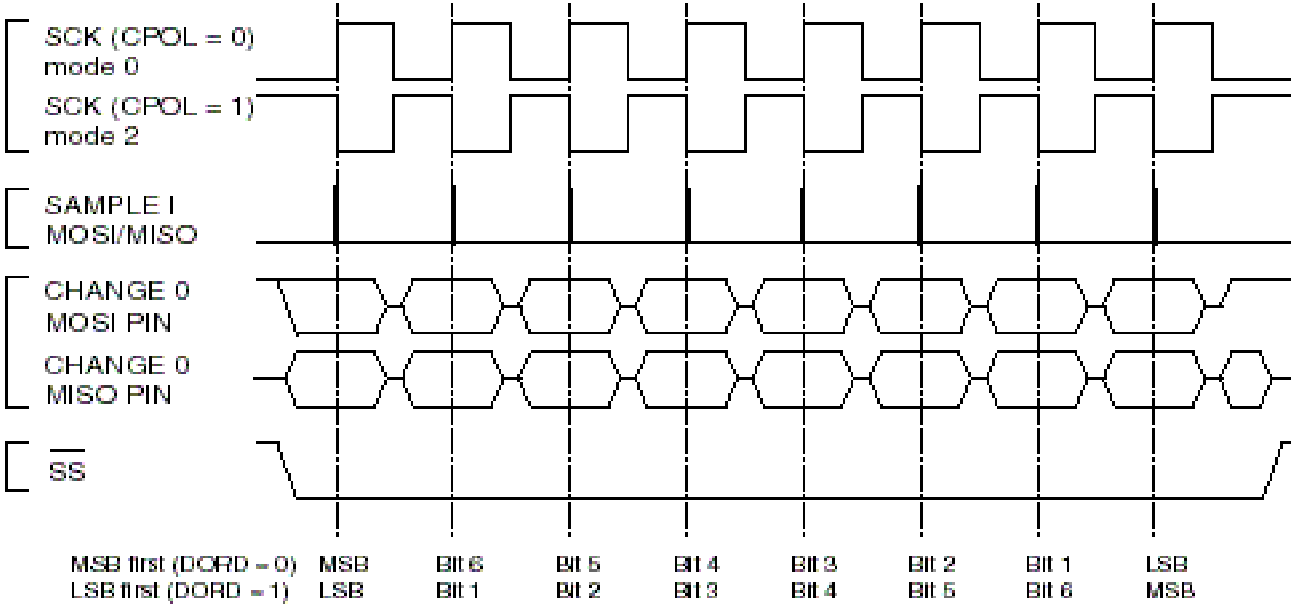
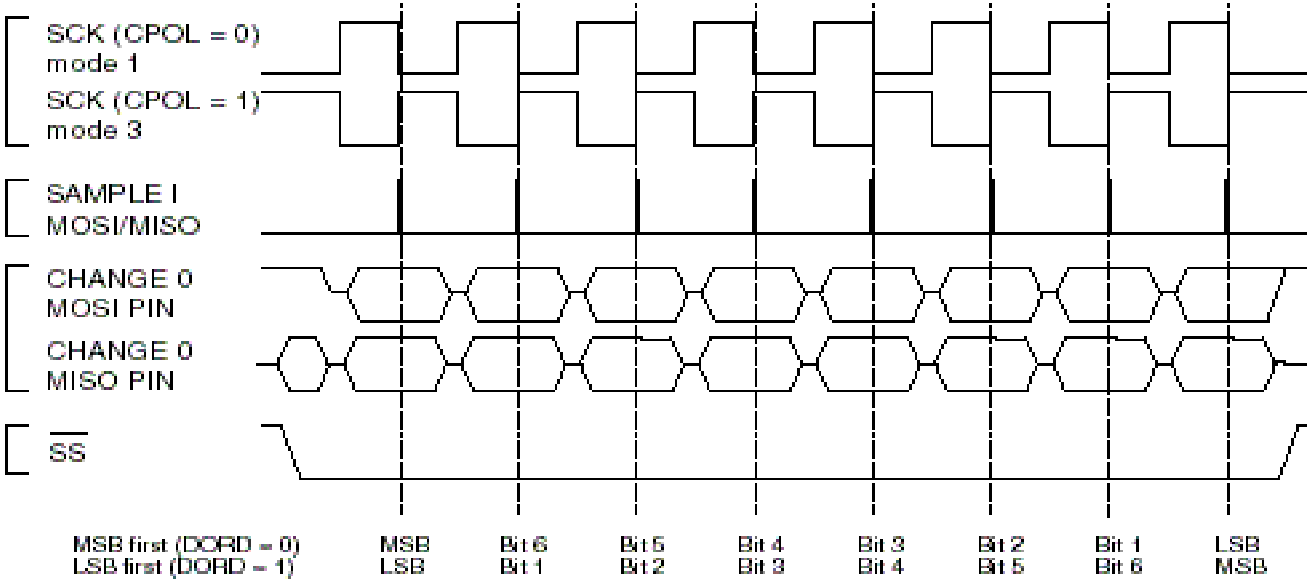


Figure 77. SPI Transfer Format with CPHA = 1



SPI Data Transfer Modes

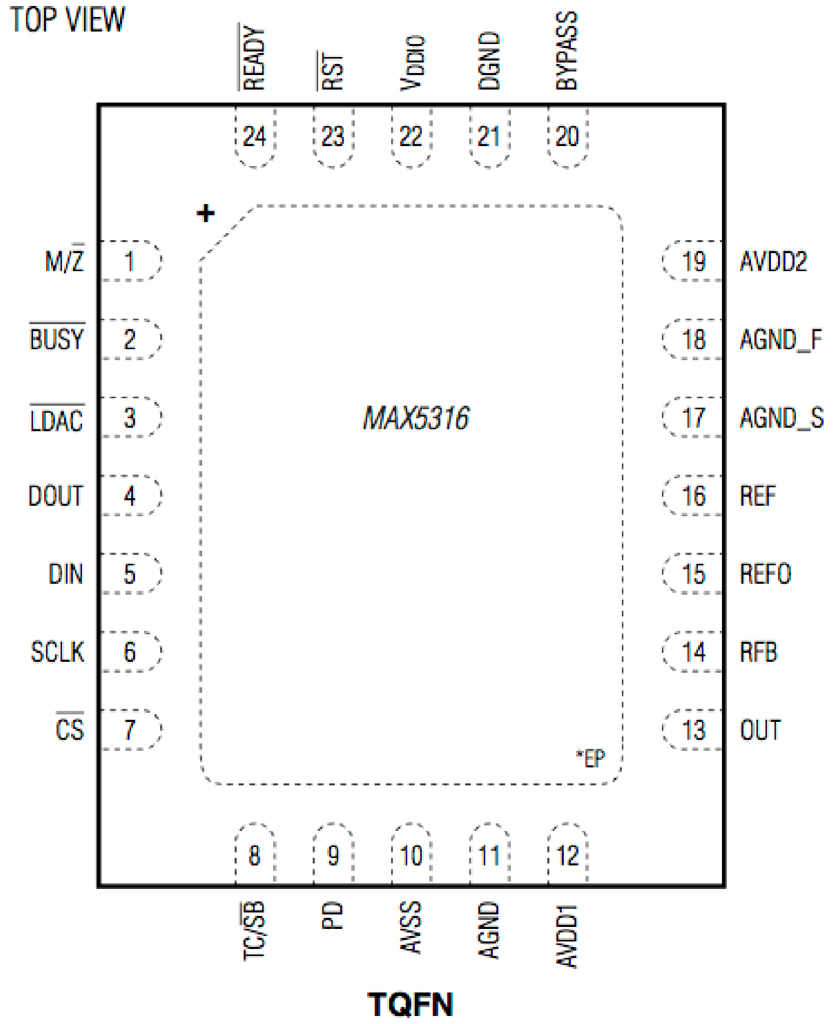
- These four modes are the combinations of CPOL and CPHA.
- Modes 0 and 3 are the most common.
- With SPI modes 0 and 3, data is always latched in on the rising edge of SCK and always output on the falling edge of SCK.

Example SPI devices

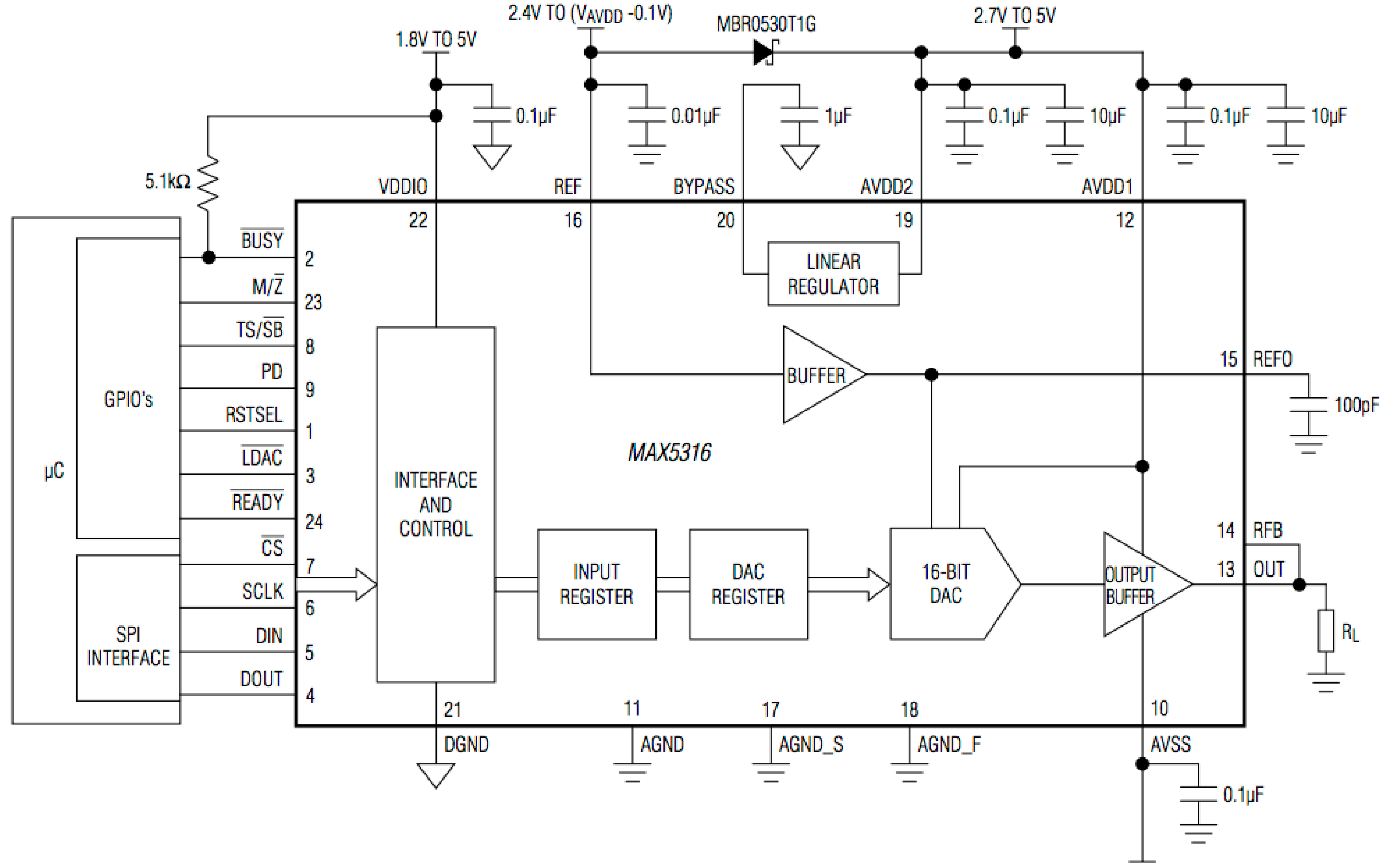
- 25LC020A - 2K SPI Bus Serial EEPROM
- TC77-5.0 - Thermal Sensor with SPI Interface
- MCP3201 - 2.7V 12-Bit A/D Converter with SPI Serial Interface
- MCP4822 - 12-Bit DAC with Internal VREF and SPI Interface
- MCP41010 - Single/Dual Digital Potentiometer with SPI Interface
- MCP6S92 - Single-Ended, Rail-to-Rail I/O, Low-Gain PGA
- MCP23S08 - 8-Bit I/O Expander with Serial Interface

Example: SPI-DAC

- MAXIM MAX5316
- 16-bit +/- 1 LSB Accuracy
- 2.7V to 5.5V supply voltage range
- 50MHz 3-wire SPI interface

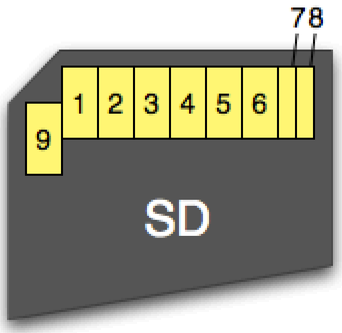


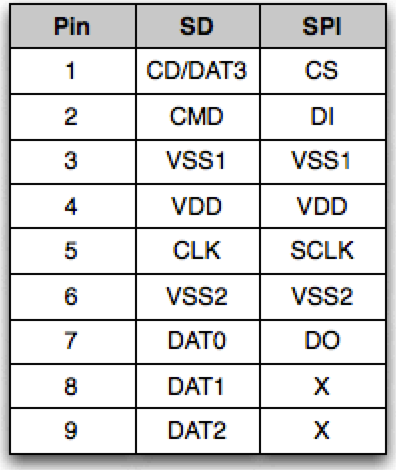
MAX5316 Diagram



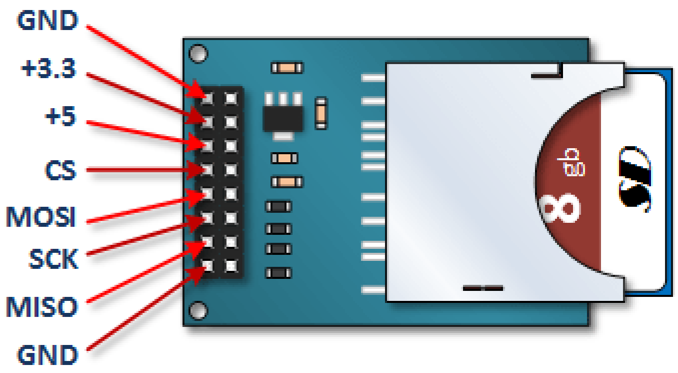
*CONNECT THE EXPOSED PAD TO AGND.

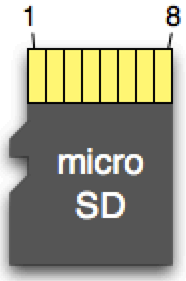
Secure Digital (SD) Media Cards

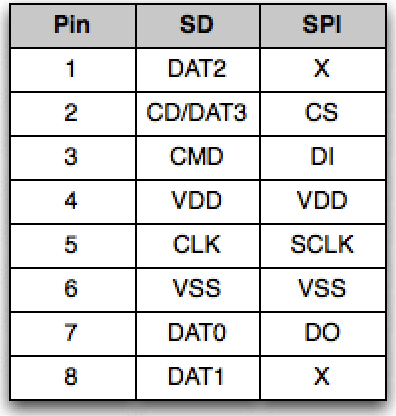




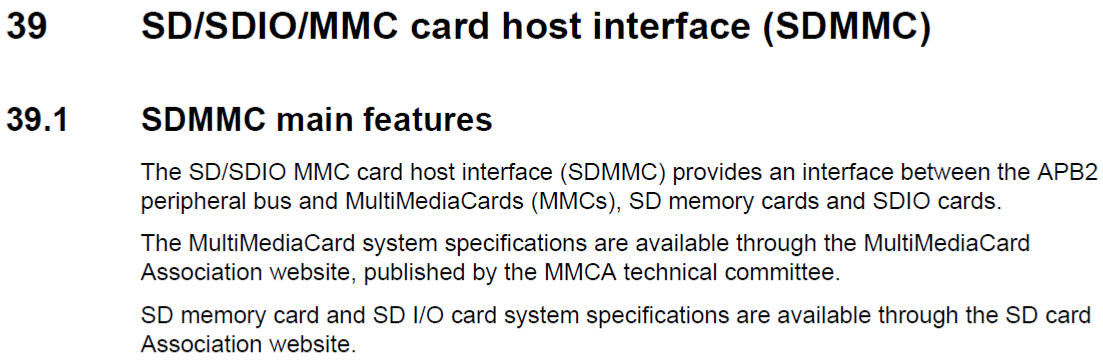
Pin	SD	SPI
1	CD/DAT3	CS
2	CMD	DI
3	VSS1	VSS1
4	VDD	VDD
5	CLK	SCLK
6	VSS2	VSS2
7	DAT0	DO
8	DAT1	X
9	DAT2	X







Pin	SD	SPI
1	DAT2	X
2	CD/DAT3	CS
3	CMD	DI
4	VDD	VDD
5	CLK	SCLK
6	VSS	VSS
7	DAT0	DO
8	DAT1	X



39 SD/SDIO/MMC card host interface (SDMMC)

39.1 SDMMC main features

The SD/SDIO MMC card host interface (SDMMC) provides an interface between the APB2 peripheral bus and MultiMediaCards (MMCs), SD memory cards and SDIO cards.

The MultiMediaCard system specifications are available through the MultiMediaCard Association website, published by the MMCA technical committee.

SD memory card and SD I/O card system specifications are available through the SD card Association website.

SD card commands

- The host (master) sends 48-bit commands (3 16-bit words) to the card (slave) to:
  - ❑ Prepare to read a **block**.
  - ❑ Check if block is ready to read.
  - ❑ Read the block.
  - ❑ Prepare to write a block.
- These **blocks** are not files.
- They’re just linearly addressed chunks of data on the storage device.

File systems

- Might not want to just write and read blocks.
  - ❑ FATFS: a library for reading/writing Microsoft FAT/exFAT filesystem on an SD card.
- https://github.com/kiwih/cubemx-mmc-sd-card
  - ❑ An example of a **STM32CubeMX**-generated system with **FatFs** middleware controlling an **SPI-connected MMC/SD** memory card.
  - ❑ Initially created in CubeMX
  - ❑ then code written by ChaN was **ported** to the CubeMX HAL.
- Example projects on Repository
  - ❑ C:\Users\xxxxxxxxxxx\STM32Cube\Repository

Pros and Cons

Pros:

- Fast and easy
  - Fast for point-to-point connections
  - Easily allows streaming/Constant data inflow
  - No addressing/Simple to implement
- Everyone supports it

Cons:

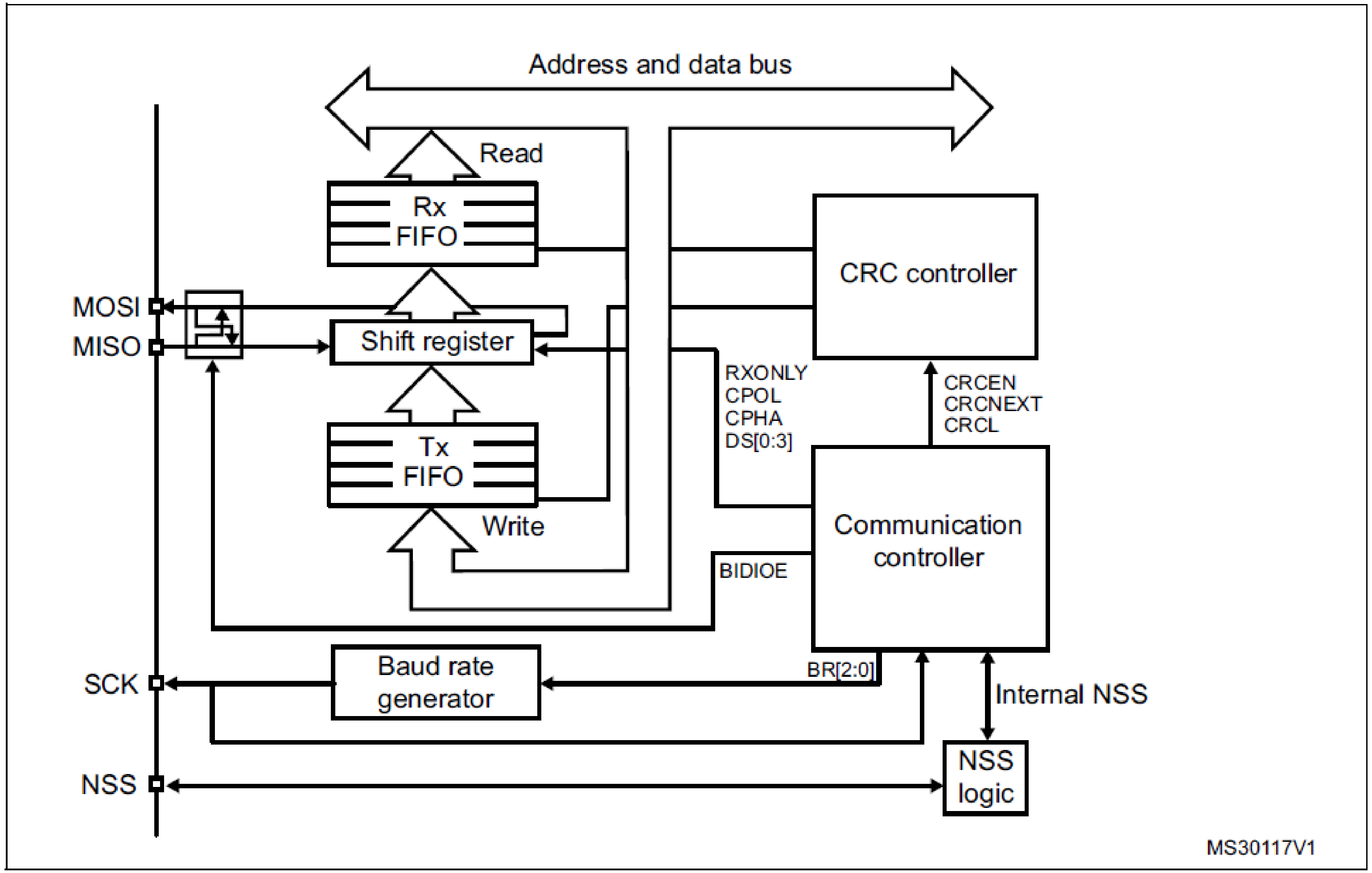
- SS makes multiple slaves very complicated
- No acknowledgement ability
- No inherent arbitration
- No flow control

STM32F767 SPI: Features

- 6 SPIs
  - SPI1, SPI4, SPI5, and SPI6 can communicate at up to 54 Mbits/s.
  - SPI2 and SPI3 can communicate at up to 25 Mbit/s.
- Master or slave operation
- Full-duplex synchronous transfers on three lines
- Half-duplex / Simplex
- 4-bit to 16-bit data size selection
- Multimaster mode capability
- Programmable clock polarity and phase
- Programmable data order with MSB-first or LSB-first shifting
- Dedicated transmission and reception flags with interrupt capability

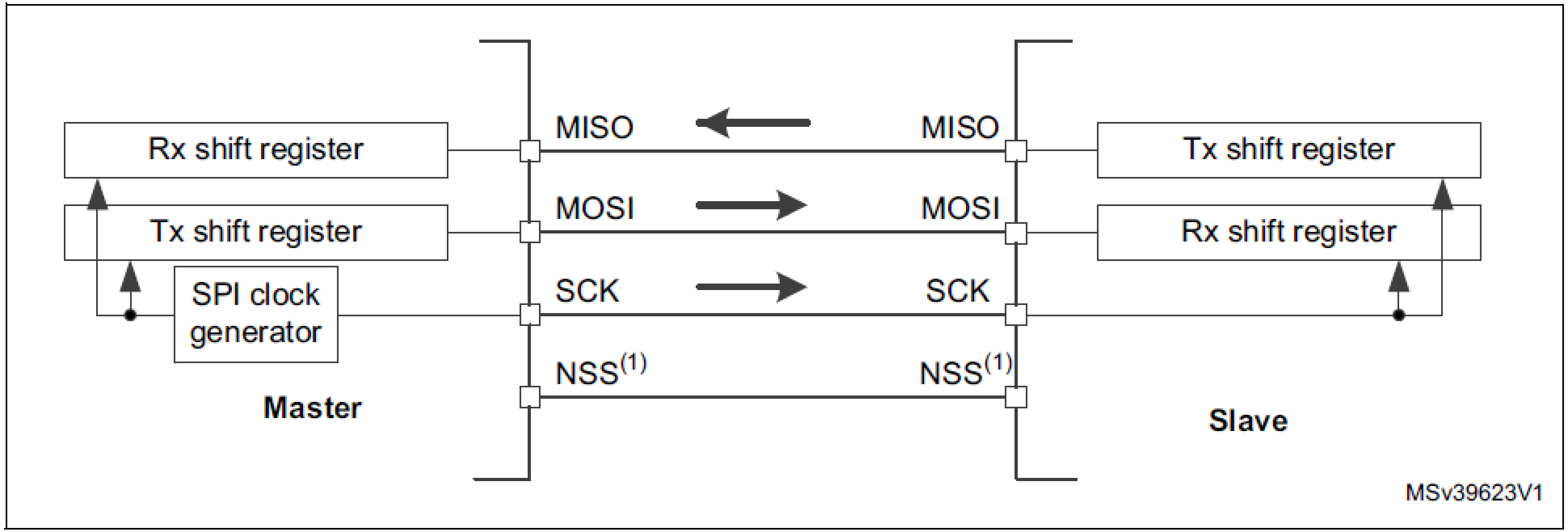
STM32F767 SPI: Block Diagram

Figure 386. SPI block diagram



Full-duplex Communication

Figure 387. Full-duplex single master/ single slave application



1. The NSS pins can be used to provide a hardware control flow between master and slave. Optionally, the pins can be left unused by the peripheral. Then the flow has to be handled internally for both master and slave. For more details see *Section 35.5.5: Slave select (NSS) pin management*.

Data Alignment

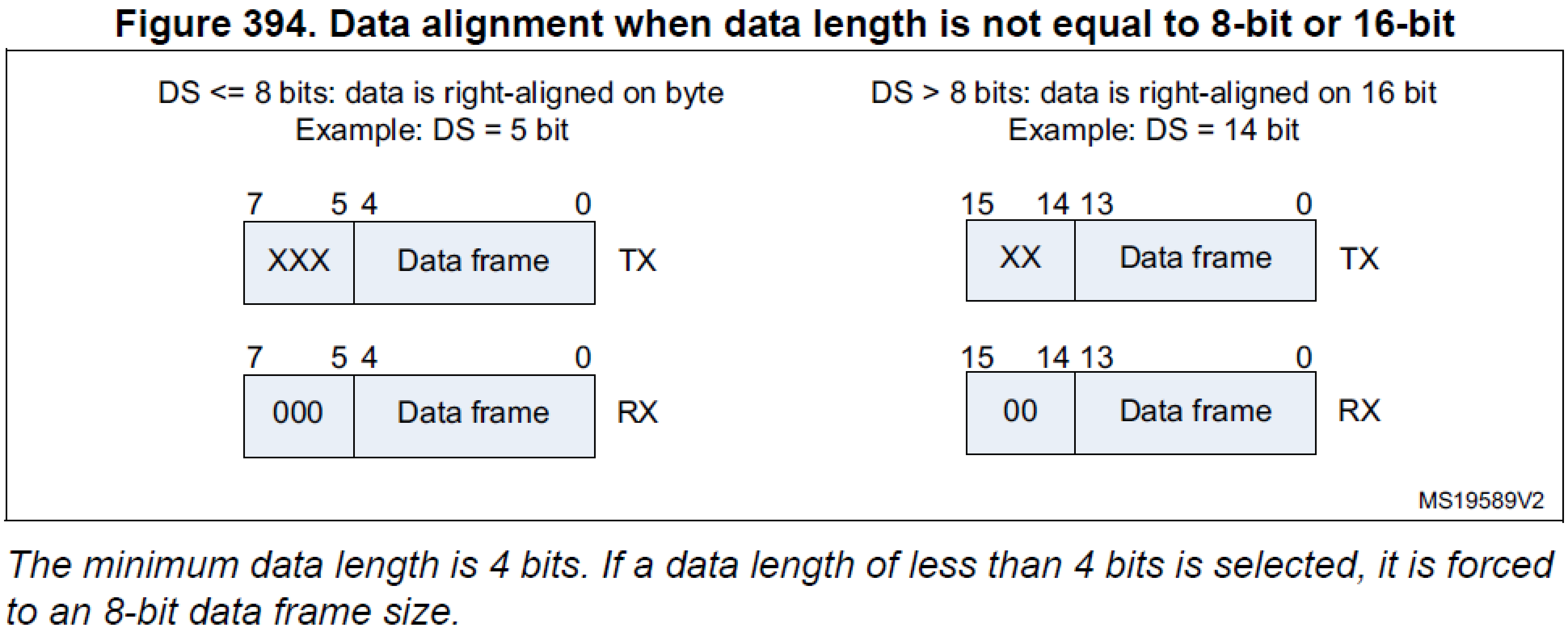


Figure 394. Data alignment when data length is not equal to 8-bit or 16-bit

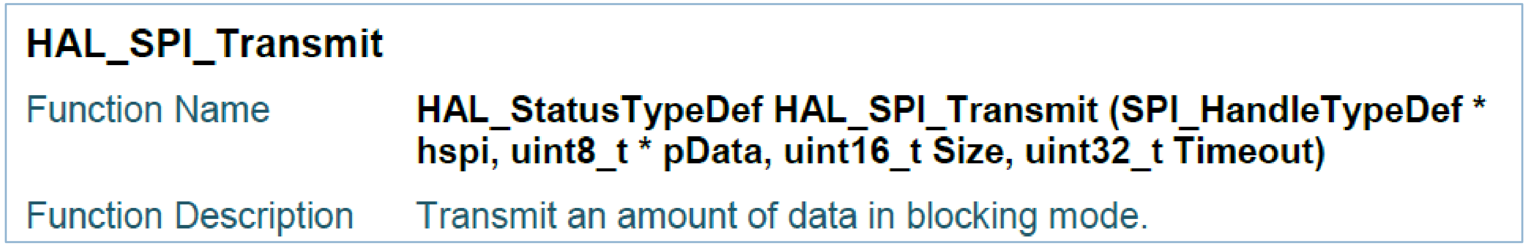
The minimum data length is 4 bits. If a data length of less than 4 bits is selected, it is forced to an 8-bit data frame size.

HAL API

- HAL_SPI_Transmit()
- HAL_SPI_Receive()
- HAL_SPI_TransmitReceive()
- HAL_SPI_Transmit_IT()
- HAL_SPI_Receive_IT()
- HAL_SPI_TransmitReceive_IT()
- HAL_SPI_Transmit_DMA()
- HAL_SPI_Receive_DMA()
- HAL_SPI_TransmitReceive_DMA()
- HAL_SPI_DMAPause()
- HAL_SPI_DMAResume()
- HAL_SPI_DMAStop()

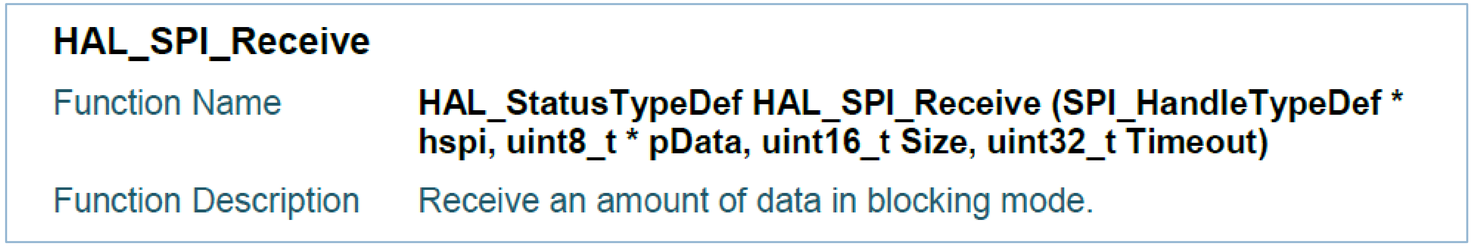
- HAL_SPI_IRQHandler()
- HAL_SPI_TxCpltCallback()
- HAL_SPI_RxCpltCallback()
- HAL_SPI_TxRxCpltCallback()
- HAL_SPI_TxHalfCpltCallback()
- HAL_SPI_RxHalfCpltCallback()
- HAL_SPI_TxRxHalfCpltCallback()
- HAL_SPI_ErrorCallback()

HAL SPI



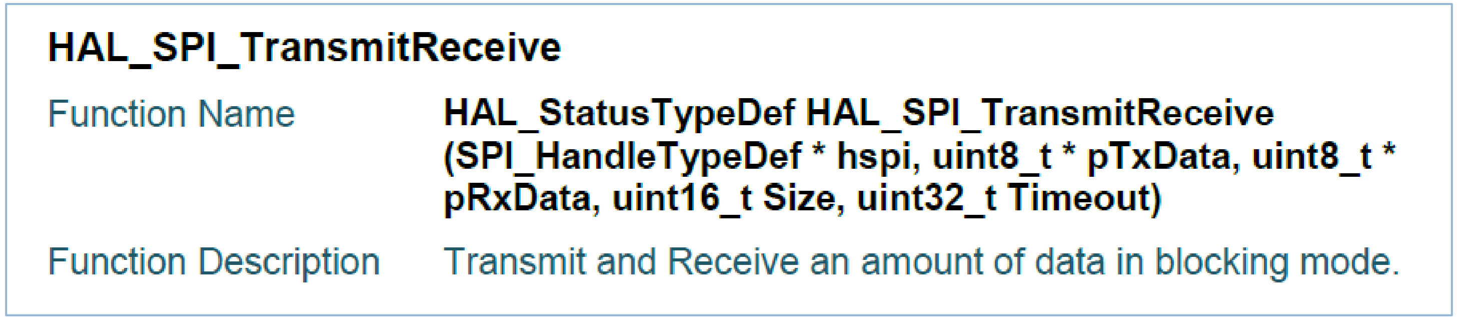
HAL_SPI_Transmit

Function Name	**HAL_StatusTypeDef HAL_SPI_Transmit (SPI_HandleTypeDef * hspi, uint8_t * pData, uint16_t Size, uint32_t Timeout)**
Function Description	Transmit an amount of data in blocking mode.



HAL_SPI_Receive

Function Name	**HAL_StatusTypeDef HAL_SPI_Receive (SPI_HandleTypeDef * hspi, uint8_t * pData, uint16_t Size, uint32_t Timeout)**
Function Description	Receive an amount of data in blocking mode.



HAL_SPI_TransmitReceive

Function Name	**HAL_StatusTypeDef HAL_SPI_TransmitReceive (SPI_HandleTypeDef * hspi, uint8_t * pTxData, uint8_t * pRxData, uint16_t Size, uint32_t Timeout)**
Function Description	Transmit and Receive an amount of data in blocking mode.

https://controllerstech.com/how-to-use-spi-with-stm32/

Conclusion

- SPI – 4 wire serial bus protocol
  - MOSI, MISO, SS, and SCLK wires
- Full duplex
- Multiple slaves, One master
- Best for point-to-point streaming data
- Easily Supported
- STM32F767: 6 SPIs at different max speed
- STM32F767: Full duplex / Half duplex / Simplex