EE-222: Microprocessor Systems

SPI Protocol

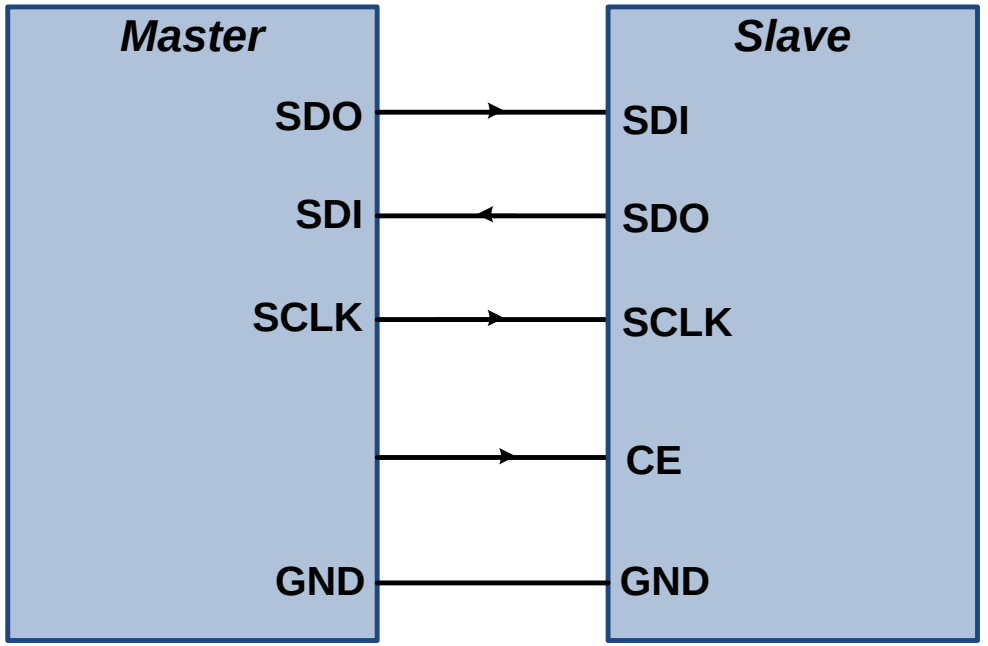
Instructors: Dr. Arbab Latif

NUST

SCHOOL OF ELECTRICAL ENGINEERING &
COMPUTER SCIENCE (SEECS)

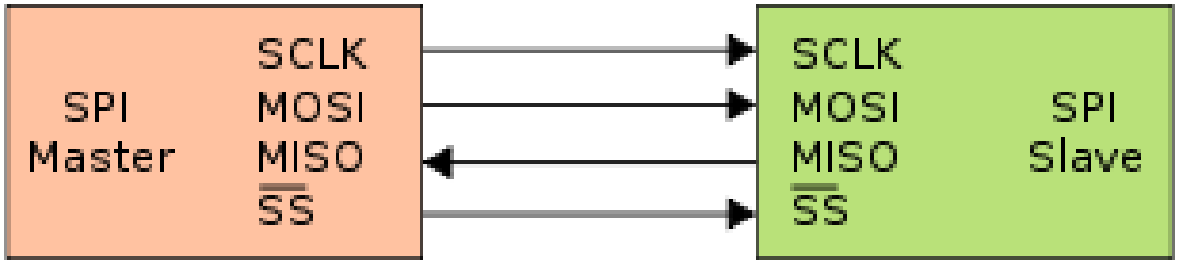
SPI Protocol

- Synchronous
- Full-duplex
- Serial
- Fast communication
- For short distances
- Pins
  - SDO (Data Out)
  - SDI (Data In)
  - SCLK (shift clock)
  - CE (chip enable)



Alternative Pins Name

- Pins are also named as:
  - MOSI (Master Out Slave In)
  - MISO (Master In Slave Out)
  - SCLK (shift clock)
  - SS (Slave Select)

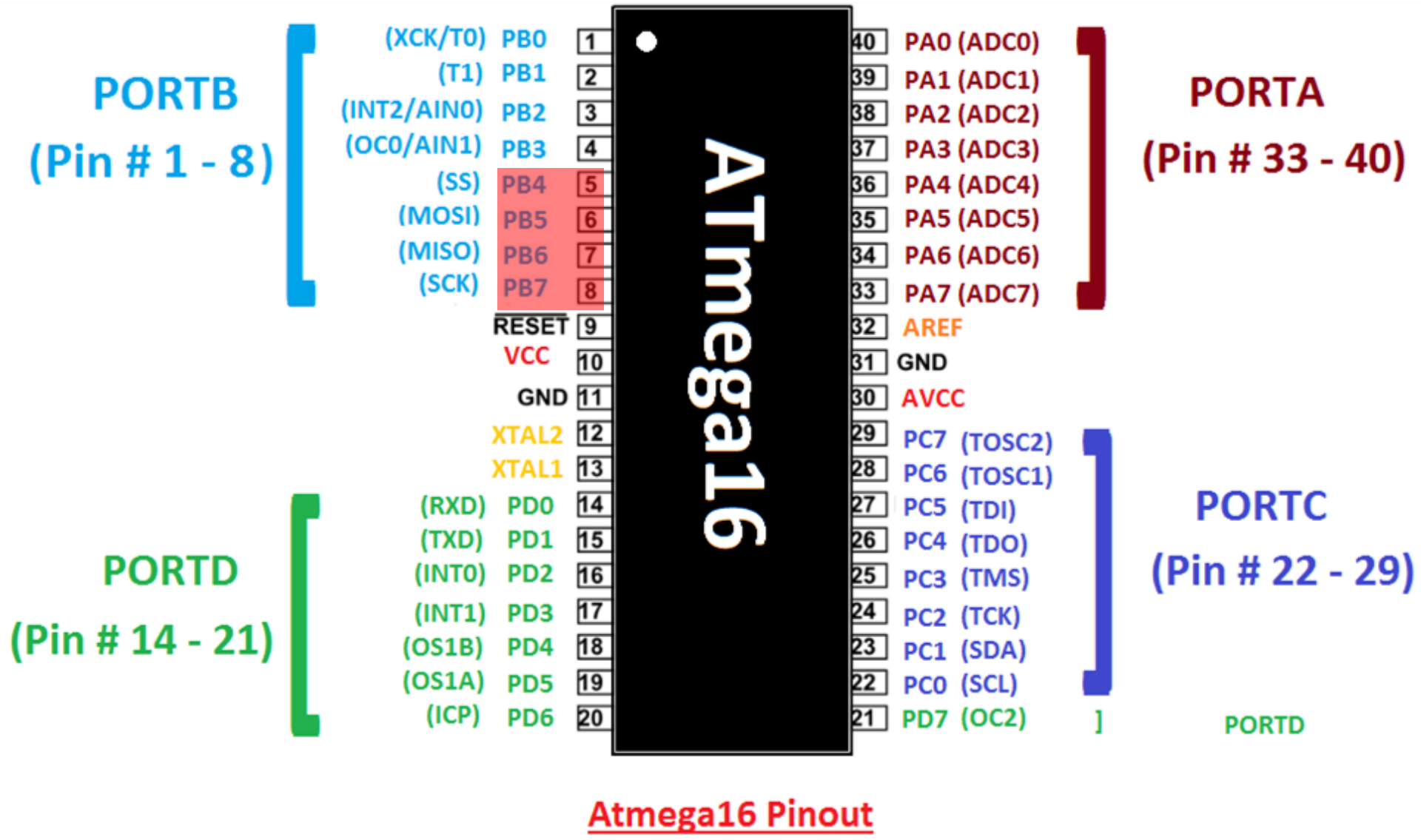


- Master begins the communication by pulling down the CE pin of slave.

- Master makes the clock for communication

Figure from: https://en.wikipedia.org/wiki/Serial_Peripheral_Interface

SPI Pins in AVR

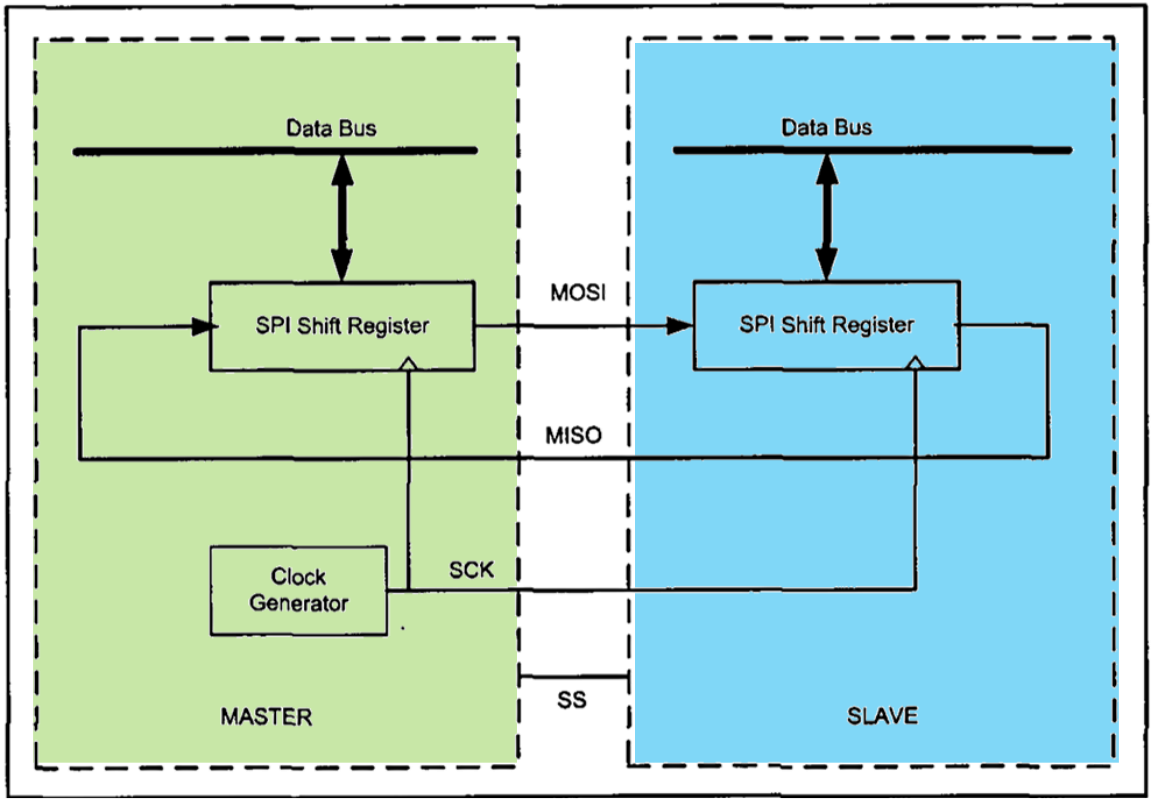


Atmega16 Pinout

SPI Architecture

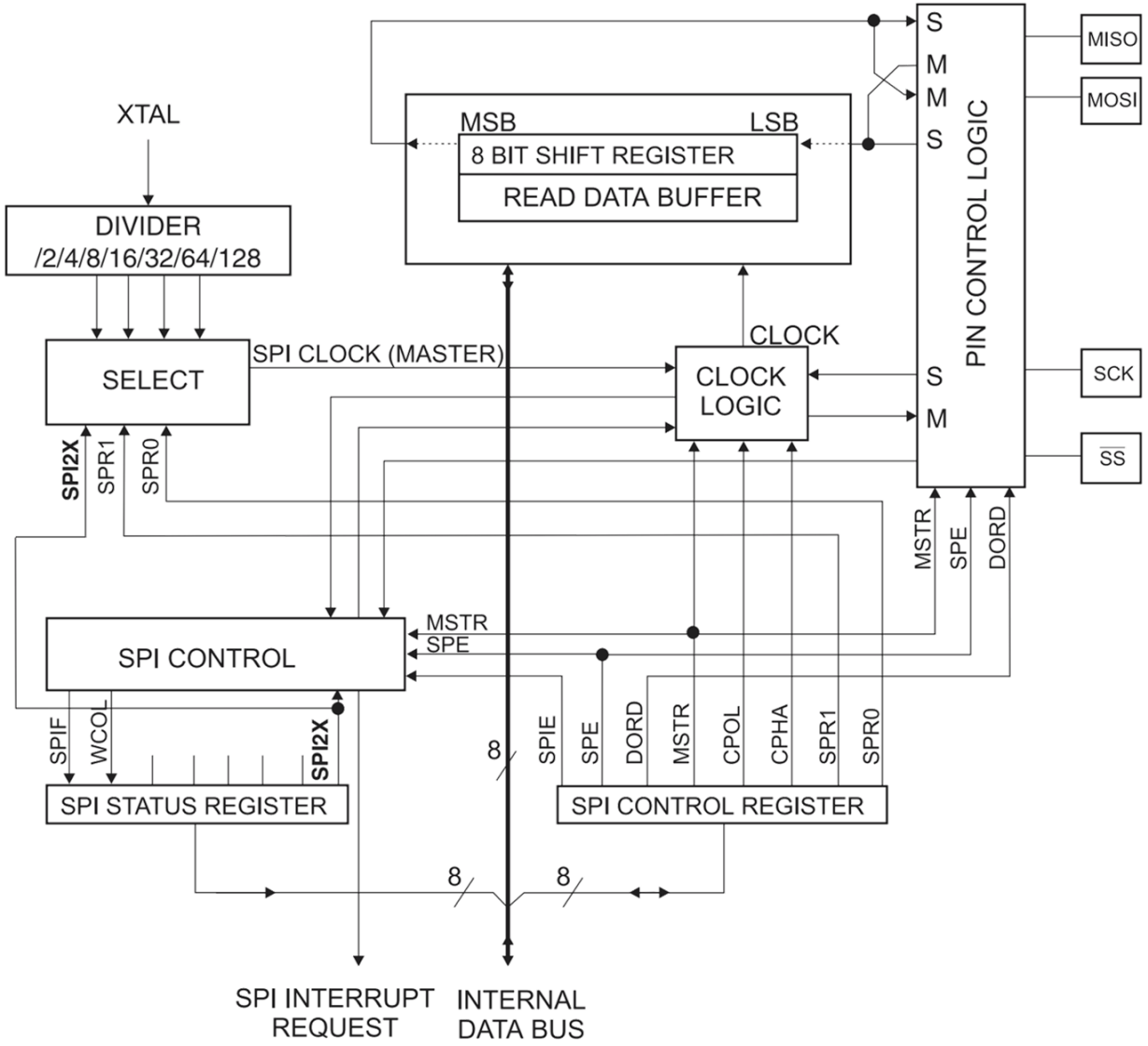
- A shift register in the master and another in the slave

- By each clock, a bit is shifted out from the master's shift register into the slave shift register and a bit is shifted from slave to master.



SPI Architecture

Figure 65. SPI Block Diagram[1]



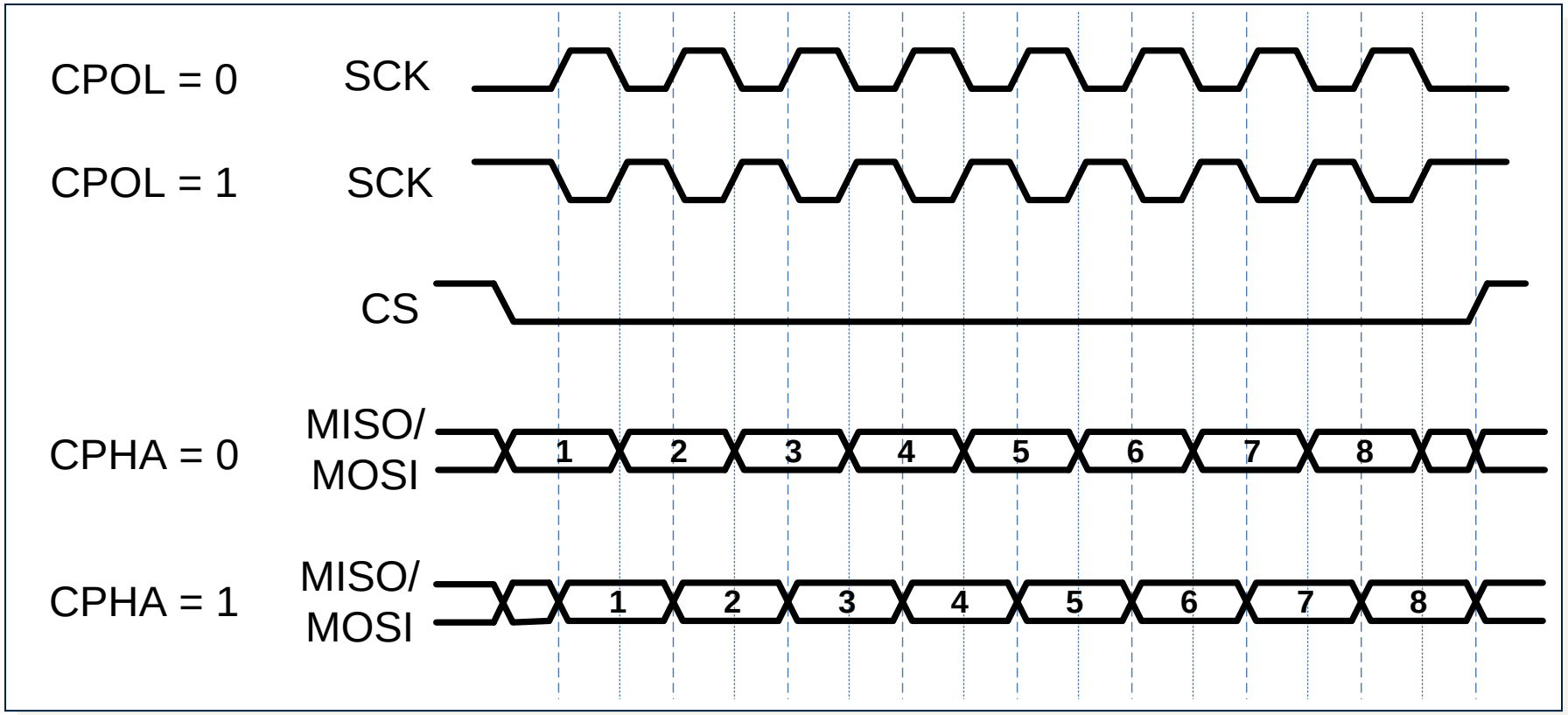
SPI Mode: Clock Polarity and Phase

- In SPI, the master and slave must agree on the clock polarity and phase (which together define the SPI mode): i.e
  - CPOL = 0 means, the base of clock is zero and
  - CPHA = 0 means sample on leading clock edge

CPOL	CPHA	Data Read and Change Time	SPI Mode
0	0	Read on rising edge, changed on a falling edge	0
0	1	Read on falling edge, changed on a rising edge	1
1	0	Read on falling edge, changed on a rising edge	2
1	1	Read on rising edge, changed on a falling edge	3

SPI Mode: Clock Polarity and Phase

CPOL	CPHA	Data Read and Change Time	SPI Mode
0	0	Read on rising edge, changed on a falling edge	0
0	1	Read on falling edge, changed on a rising edge	1
1	0	Read on falling edge, changed on a rising edge	2
1	1	Read on rising edge, changed on a falling edge	3



SPI Write Operation (single byte)

1. Make CE = 0 to begin writing.
2. The 8-bit address is shifted in, one bit at a time, with each edge of SCLK. Notice that A7 = 1 for the write operation, and the A7 bit goes in first.
3. After all 8 bits of the address are sent in, the SPI device expects to receive the data belonging to that address location immediately.
4. The 8-bit data is shifted in one bit at a time, with each edge of the SCLK.
5. Make CE = 1 to indicate the end of the write cycle.

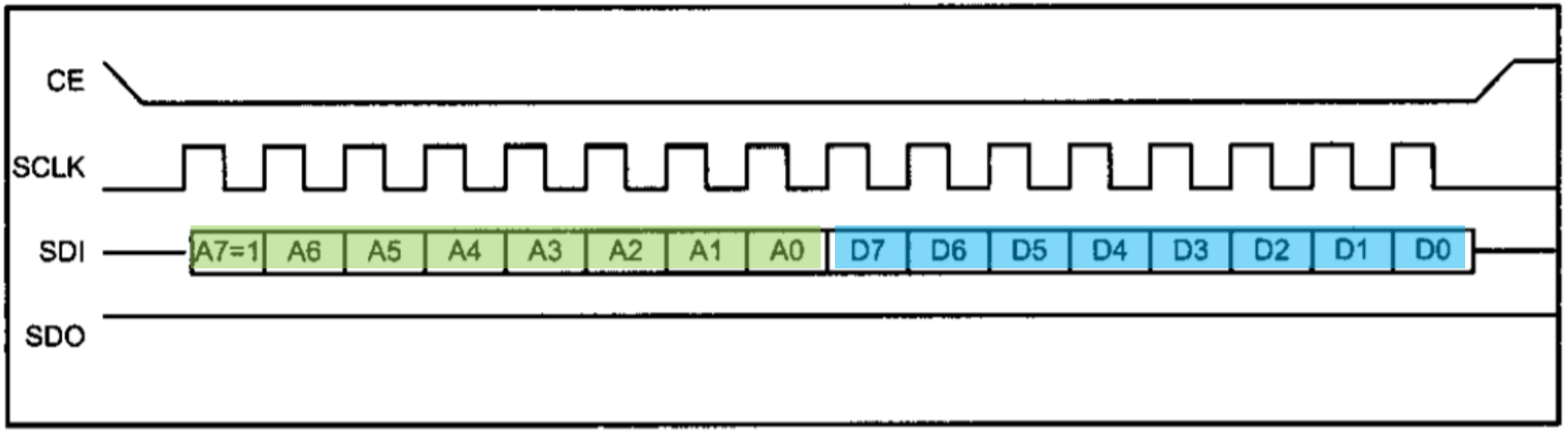


Figure 17-4. SPI Single-Byte Write Timing (Notice A7 = 1)

SPI Write operation (Burst mode)

1. Make CE = 0 to begin writing.
2. The 8-bit address of the first location is provided and shifted in, one bit at a time, with each edge of SCLK. Notice that A7 = 1 for the write operation and the A7 bit goes in first.
3. The 8-bit data for the first location is provided and shifted in, one bit at a time, with each edge of the SCLK. From then on, we simply provide consecutive bytes of data to be placed in consecutive memory locations. In the process, CE must stay low to indicate that this is a burst mode multibyte write operation.
4. Make CE = 1 to end writing.

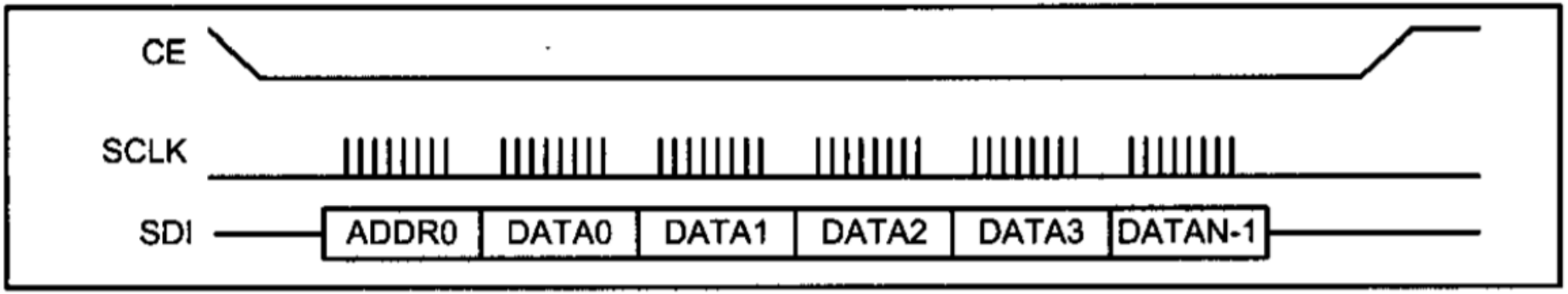


Figure 17-5. SPI Burst (Multibyte) Mode Writing

SPI Read Operation

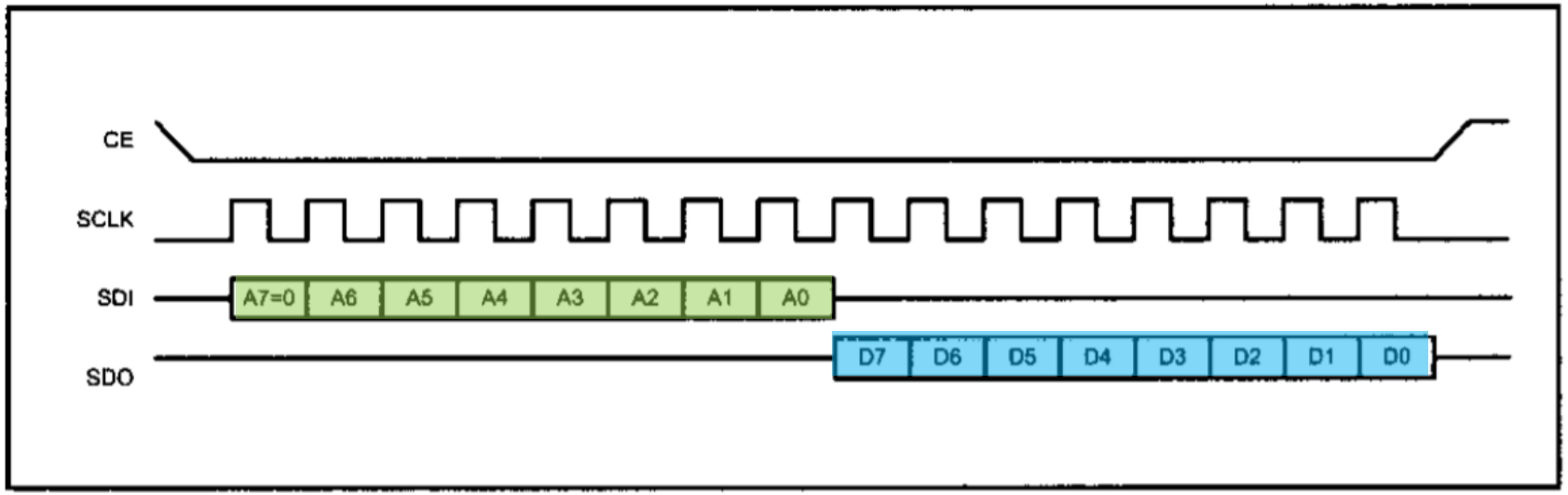


Figure 17-6. SPI Single-Byte Read Timing (Notice A7 = 0)

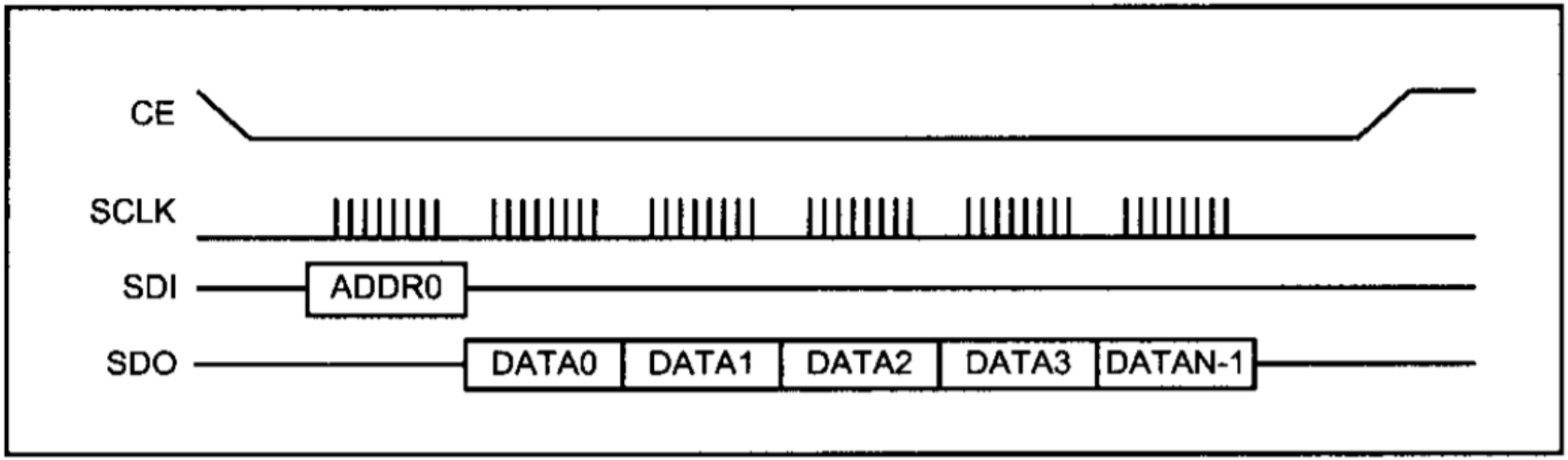


Figure 17-7. SPI Burst (Multibyte) Mode Reading

SPI Programming in AVR

SPI AVR Registers

- Control register:
  - SPCR (SPI Control Register)
- Status Register:
  - SPSR (SPI Status Register)
- Data Register:
  - SPDR (SPI Data Register)

SPSR (SPI Status Register)

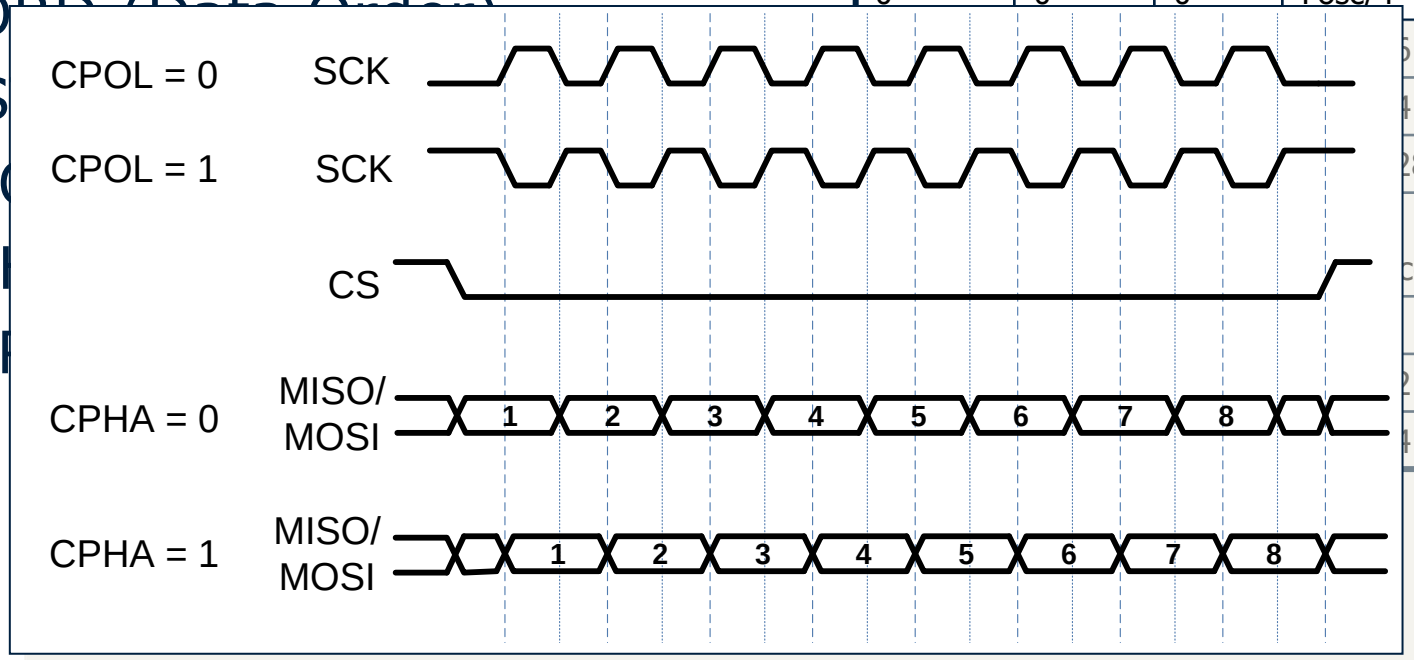
- SPIF (SPI Interrupt Flag)
  - A serial transfer is completed.
  - The SS pin is driven low in slave mode
- WCOL (Write Collision)
- SPI2X (Double SPI Speed)

SPCR

SPCR:	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0

- SPIE
- SPE
- DO
- MS
- CP
- CPI
- SPI

CPOL	CPHA	Data Read and Change Time	SPI Mode
0	0	Read on rising edge, changed on a falling edge	0
0	1	Read on falling edge, changed on a rising edge	1
1	0	Read on falling edge, changed on a rising edge	2
1	1	Read on rising edge, changed on a falling edge	3



Example 17-1

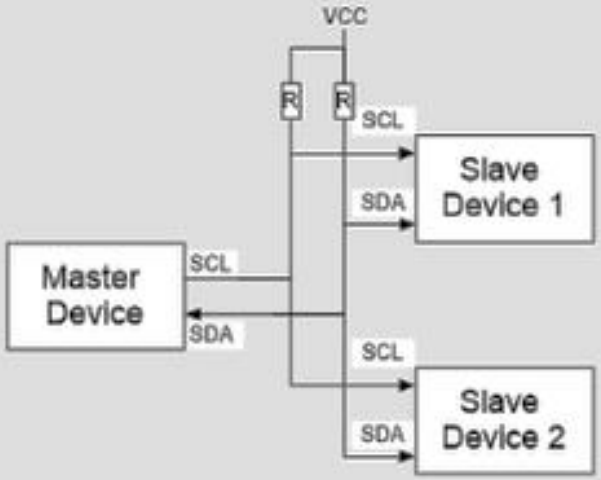
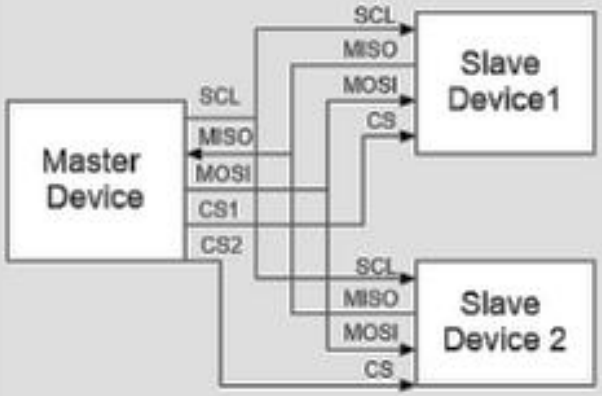
Write an AVR program to initialize the SPI for master, mode 0, with CLCK frequency = Fosc/16, and then transmit 'G' via SPI repeatedly. The received data should be displayed on Port A.

Solution:

```
.equ  MOSI = 5                          ;for ATmega32
.equ  SCK = 7
.equ  SS = 4

      LDI   R17,0xFF                    ;Port A is output
      OUT   DDRA,R17

      LDI   R17,(1<<MOSI)|(1<<SCK)|(1<<SS)
      OUT   DDRB,R17                    ;MOSI,SCK, and SS output

      LDI   R17,(1<<SPE)|(1<<MSTR)|(1<<SPR0) ;enable SPI
      OUT   SPCR,R17                    ;master, CLK = fck/16
Transmit:
      CBI   PORTB,SS                    ;enable slave device
      LDI   R17,'G'                     ;move G letter to R17
      OUT   SPDR,R17                    ;start transmission of G
Wait:
      SBIS  SPSR,SPIF                   ;wait for transmission
      RJMP  Wait                        ;to complete
      IN    R18,SPDR                    ;read received data into R18
      OUT   PORTA,R18                   ;move R18 to PORTA

      SBI   PORTB,SS                    ;disable slave device
      RJMP Transmit                     ;do it again
```

Difference Between SPI and I2C

Difference between I2C and SPI

Parameter	I2C	SPI
Full Form	I2C stands for Inter integrated circuit	SPI stands for Serial peripheral interface
Number of wires	I2C requires 2 wires (SCL and SDA)	SPI requires 4 wires (MISO, MOSI, CS and CLK)
Connection diagram		
Communication	I2C is half duplex communication means master and slave can talk with each other but not simultaneously.	SPI is full duplex communication means master and slave can talk with each other simultaneously

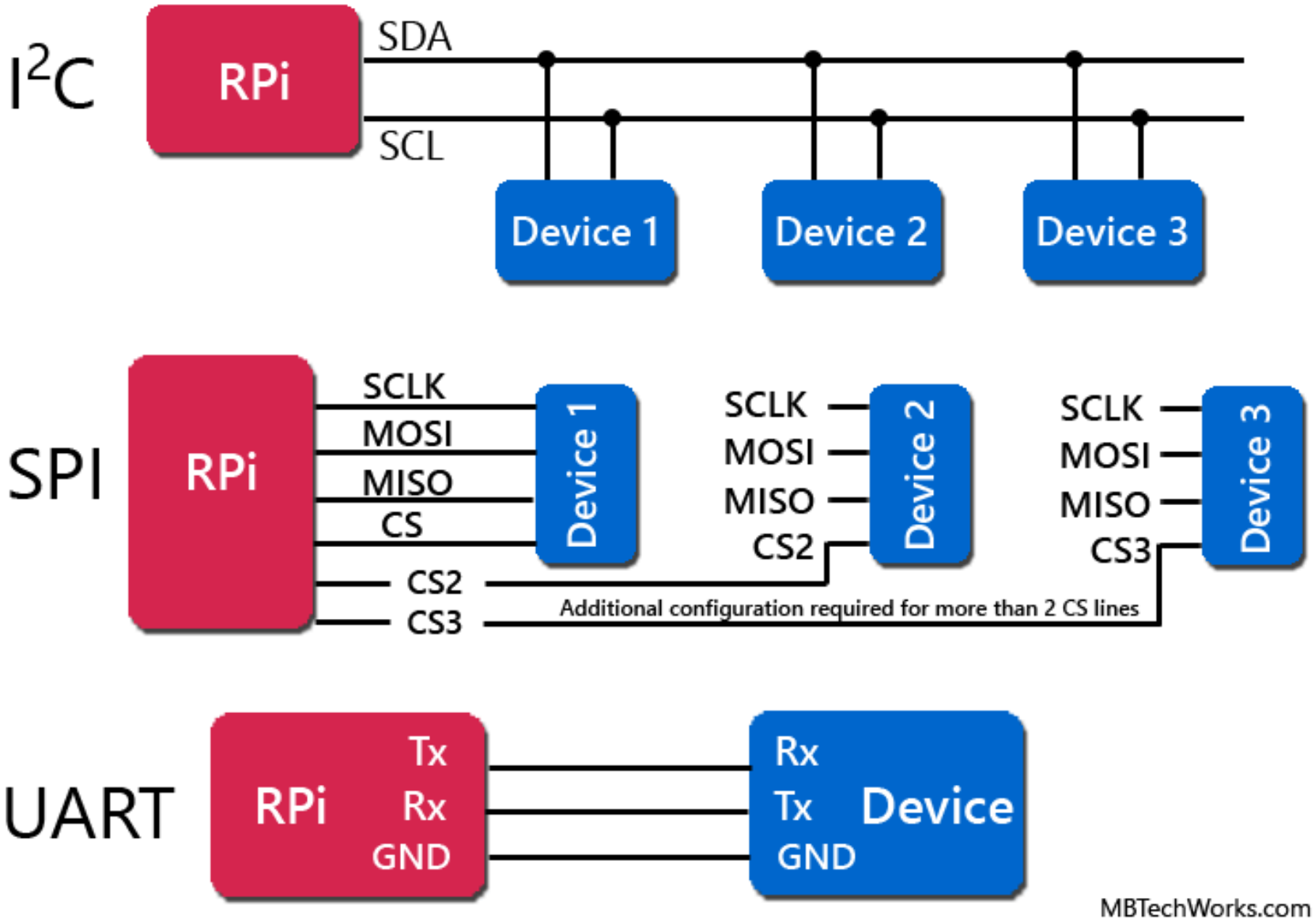
Overall, UART vs I2C vs SPI

I2C Protocol

- UART
  - Requires only two wires for full duplex data transmission
  - No need for clock or any other timing signal
  - Transmitter and receiver must agree to the rules
- SPI
  - multiple slaves to a single master
  - Pins restriction
  - full – duplex communication at very high speeds

- I2C
  - Multiple slaves to a single master (like SPI)
  - **No need of prior agreement** on data transfer rate like in UART communication
  - Only **two common bus lines**
  - Uses **7-bit addressing system** to target a specific device/IC on the I2C bus
  - **Scalable:** New devices can simply be connected to the two common I2C bus lines

Comparison of communication protocol



Reading

- The AVR Microcontroller and Embedded Systems: Using Assembly and C by Mazidi et al., Prentice Hall
  - Chapter 18: SPI Protocol

THANK YOU