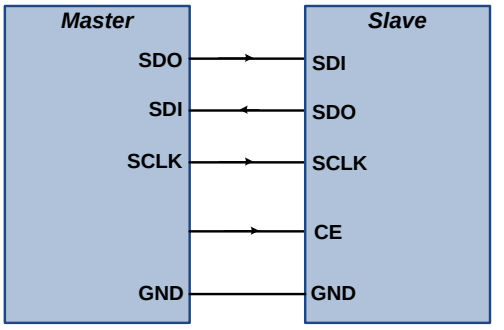
Mechatronics Engineering

SPI Communication Protocol

Prof. Ayman A. El-Badawy
Department of Mechatronics Engineering
Faculty of Engineering and Materials Science
German University in Cairo

SPI Protocol

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



GUC German University in Cairo
Prof. Ayman A. El-Badawy
Department of Mechatronics Engineering
Faculty of Engineering and Material Science

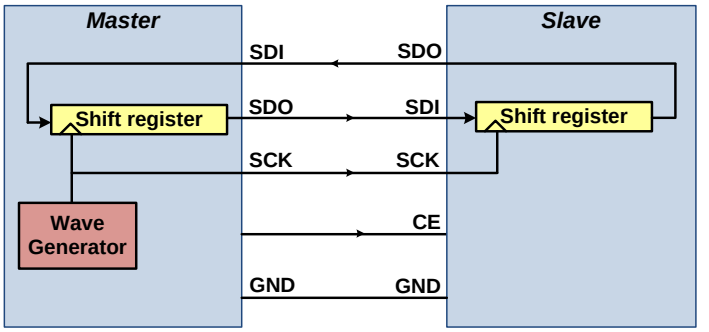
Master vs. Slave

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

Prof. Ayman A. El-Badawy
Department of Mechatronics Engineering
Faculty of Engineering and Material Science

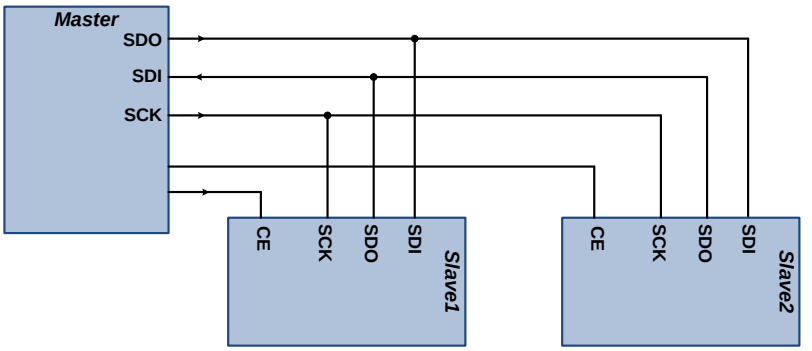
SPI internal circuit

- 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.



Prof. Ayman A. El-Badawy
Department of Mechatronics Engineering
Faculty of Engineering and Material Science

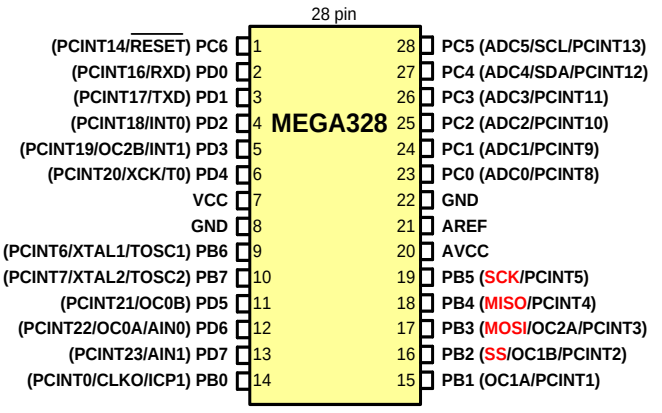
Multi-slave communication



Prof. Ayman A. El-Badawy
Department of Mechatronics Engineering
Faculty of Engineering and Material Science

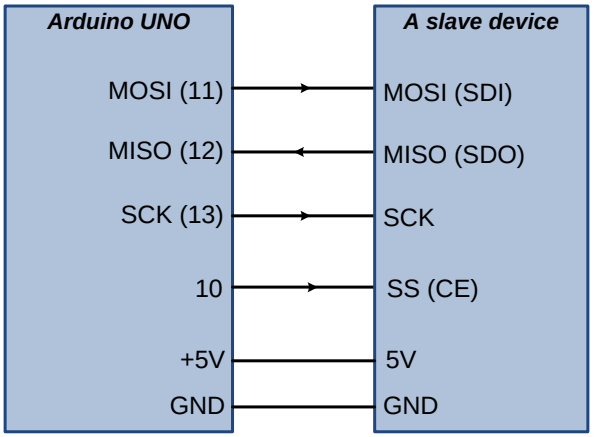
SPI pins in AVR

- MOSI (Master Out Slave In)
- MISO (Master In Slave Out)
- SCK
- SS



Prof. Ayman A. El-Badawy
Department of Mechatronics Engineering
Faculty of Engineering and Material Science

Connecting a slave device to an AVR



Prof. Ayman A. El-Badawy
Department of Mechatronics Engineering
Faculty of Engineering and Material Science

AVR registers

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

Prof. Ayman A. El-Badawy
Department of Mechatronics Engineering
Faculty of Engineering and Material Science

SPSR (SPI Status Register)

SPSR:	SPIF	WCOL	-	-	-	-	-	SPI2X

- 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)

GUC German University in Cairo
Prof. Ayman A. El-Badawy
Department of Mechatronics Engineering
Faculty of Engineering and Material Science

SPCR

SPCR:	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0

- SPIE (SPI Interrupt Enable)
- SPE (SPI Enable)
- DORD (Data Order)
- MSTR (Master)
- CPOL (Clock Polarity)
- CPHA (Clock Phase)
- SPR1, SPR0 :SPI Clock Rate

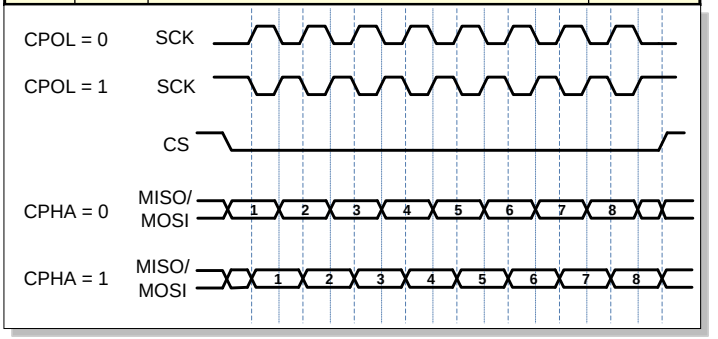
SPI2X	SPR1	SPR0	SCK Freq.
0	0	0	Fosc/4
0	0	1	Fosc/16
0	1	0	Fosc/64
0	1	1	Fosc/128
1	0	0	Fosc/2 (not recommended)
1	0	1	Fosc/8
1	1	0	Fosc/32
1	1	1	Fosc/64

GUC German University in Cairo
Prof. Ayman A. El-Badawy
Department of Mechatronics Engineering
Faculty of Engineering and Material Science

SPCR

SPCR:	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0

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



Prof. Ayman A. El-Badawy
Department of Mechatronics Engineering
Faculty of Engineering and Material Science

Program 1: Sending 'G' through SPI as a master

```
#include <avr/io.h>

#define MOSI 3
#define SCK 5
#define SS 2

int main (void)
{
 DDRB = (1<<MOSI)|(1<<SCK)|(1<<SS); //MOSI and SCK are output
 DDRD = 0xFF; //Port D is output
 SPCR = (1<<SPE)|(1<<MSTR)|(1<<SPR0); //enable SPI as master
 while(1) //do for ever
 {
  PORTB &= ~(1<<SS); //enable slave device
  SPDR = 'G'; //start transmission
  while(!(SPSR & (1<<SPIF))); //wait transfer finish
  PORTD = SPDR; //move received data to PORTD
  PORTB |= (1<<SS); //disable slave device
 }
 return 0;
}
```


Prof. Ayman A. El-Badawy
Department of Mechatronics Engineering
Faculty of Engineering and Material Science

Program 2: Sending 'G' through SPI as a slave

```
#include <avr/io.h>

#define MISO 4

int main (void)
{
 DDRD = 0xFF; //Port D is output
 DDRB = (1<<MISO); //MISO is output
 SPCR = (1<<SPE); //enable SPI as slave
 while(1)
 {
  SPDR = 'G';
  while(!(SPSR &(1<<SPIF))); //wait for transfer finish
  PORTD = SPDR; //move received data to PORTD
 }
 return 0;
}
```