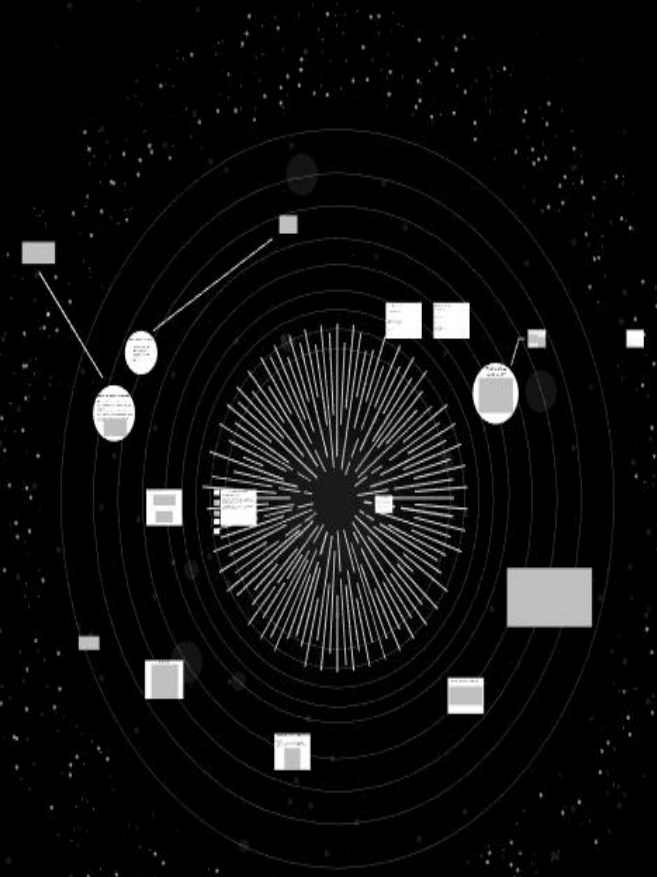


MICROCONTROLLERS AND MICRO PROCESSORS

by Hari Krishna

What are Microcontrollers ?

- A Microcontroller is a small computer on a single integrated circuit made through VLSI fabrication.
- It is basically cheap and small computer on a single chip that comprises a processor, a small memory, and programmable input-output peripherals.

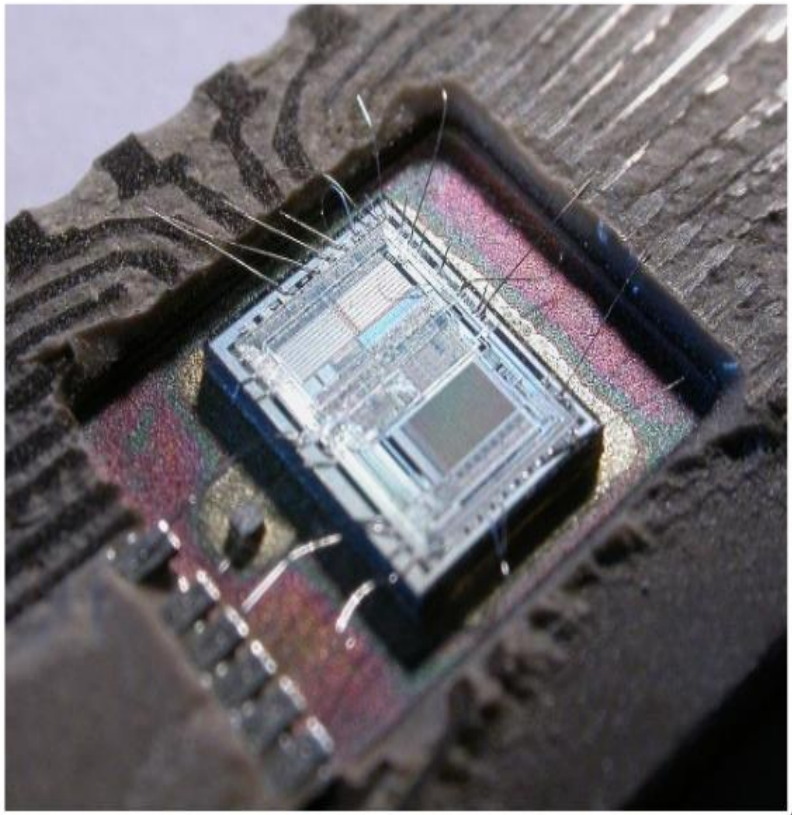
Microcontroller

They are designed to perform specific task

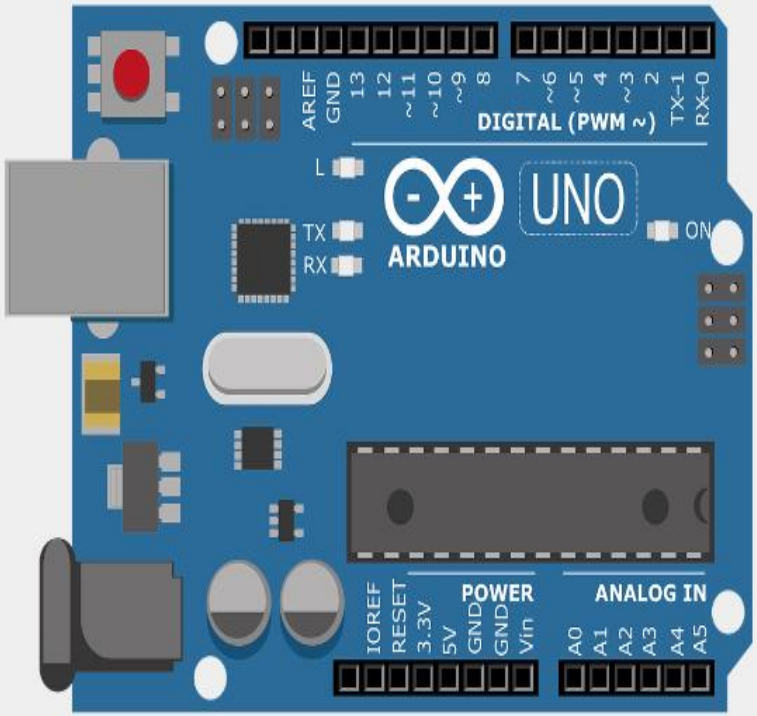
Where do we use them?

- Washing Machine
- Microwave Oven
- Laptops, Desktops
- Cloths Iron
- AC

What are they made up of ?



ARDUINO VS MICROCONTROLLER



Protocols and unknown Pins

- **ICPS** - **I**n **C**ircuit **S**erial **P**rograming - Any arrangement that allows you to program a microcontroller while it is in a circuit using a serial protocol can be called ISP or ICSP. [2 x 3 pins]

- **Aref** - **A**nalog **Ref**erence

  Arduino comes with a 10bit ADC (Analog-Digital-Converter), which converts incoming voltages between 0V and 5V to integer values between 0 and 1023. This results in a resolution of roughly 4.8 mV.

  If a sensor only delivers a lower maximum voltage, it is resonable to apply this voltage to the AREF pin, just in order to obtain a higher resolution.

- **IOref -** Input **O**uput **R**eference

  Pin2 is the IOREF pin (you can see it's wired to +5vdc) and is used to tell any shield attached what voltage the host arduino board is running at.

- **ADC -** **A**nalog to **D**igital **C**onverters

  An Analog to Digital Converter (ADC) is a very useful feature that converts an analog voltage on a pin to a digital number.

  The ADC on the Arduino is a 10-bit ADC meaning it has the ability to detect **1,024 (2^10)** discrete analog levels. Some microcontrollers have **8-bit ADCs (2^8 = 256 discrete levels)** and some have **16-bit ADCs i.e (2^16 = 65,536 discrete levels).**

Relating ADC Value to Voltage

The ADC reports a *ratiometric value*. This means that the ADC assumes 5V is 1023 and anything less than 5V will be ratio between 5V and 1023.

$$\frac{Resolution\ of\ the\ ADC}{System\ Voltage} = \frac{ADC\ Reading}{Analog\ Voltage\ Measured}$$

Analog to digital conversions are dependant on the system voltage. Because we predominantly use the 10-bit ADC o Arduino on a 5V system, we can simplify this equation slightly:

$$\frac{1023}{5} = \frac{ADC\ Reading}{Analog\ Voltage\ Measured}$$

If your system is 3.3V, you simply change 5V out with 3.3V in the equation. If your system is 3.3V and your ADC is reporting 512, what is the voltage measured? It is approximately 1.65V.

If the analog voltage is 2.12V what will the ADC report as a value?

$$\frac{1023}{5.00V} = \frac{x}{2.12V}$$

Rearrange things a bit and we get:

$$\frac{1023}{5.00V} * 2.12V = x$$

$$x = 434$$

- **PWM - P**ulse **W**idth **M**odulation

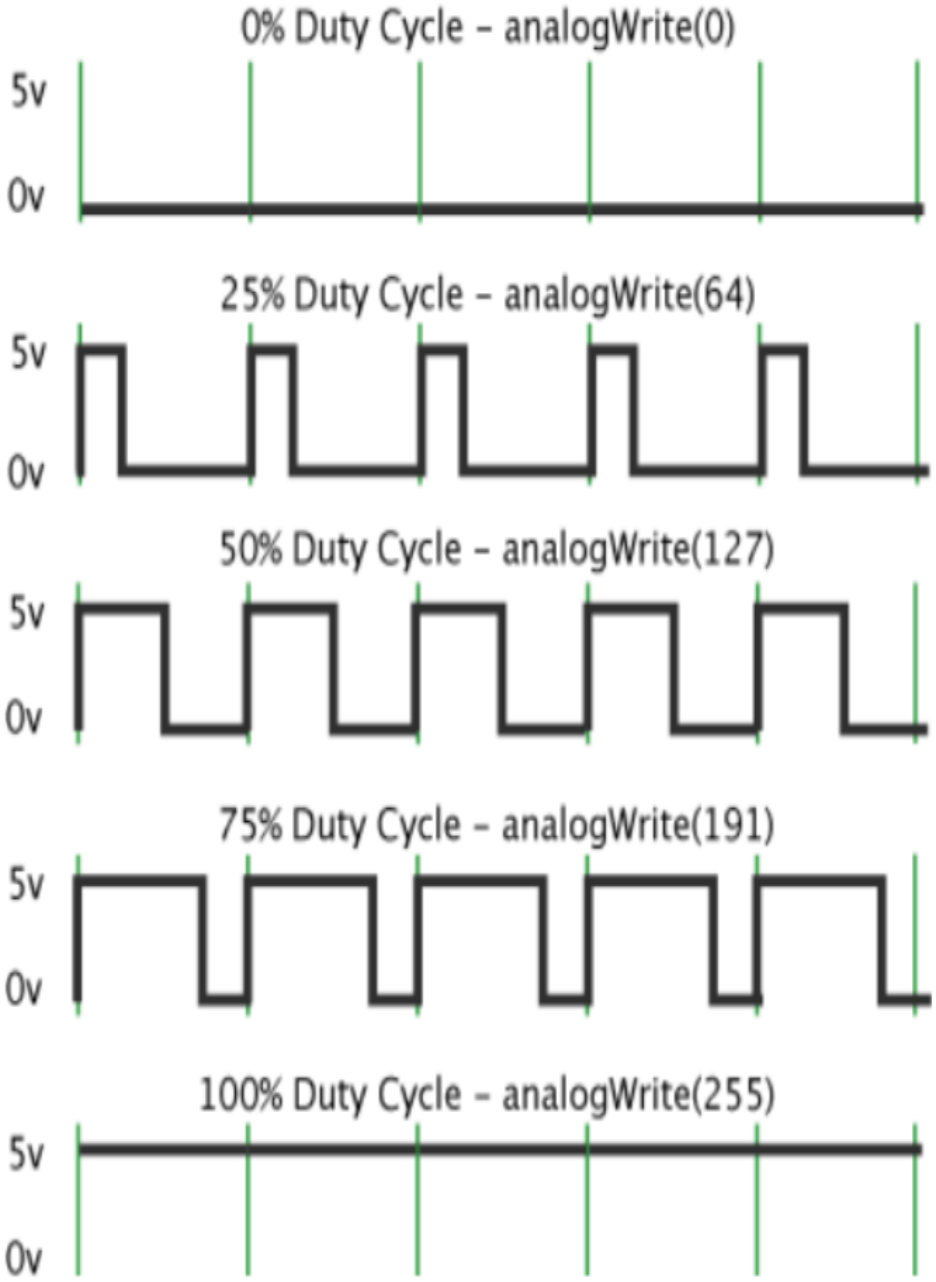
  Pulse Width Modulation, or PWM, is a technique for getting analog results with digital means. Digital control is used to create a square wave, a signal switched between on and off.

- **Duty Cycle**

  A Duty cycle is the fraction of one period in which a signal or system is active.

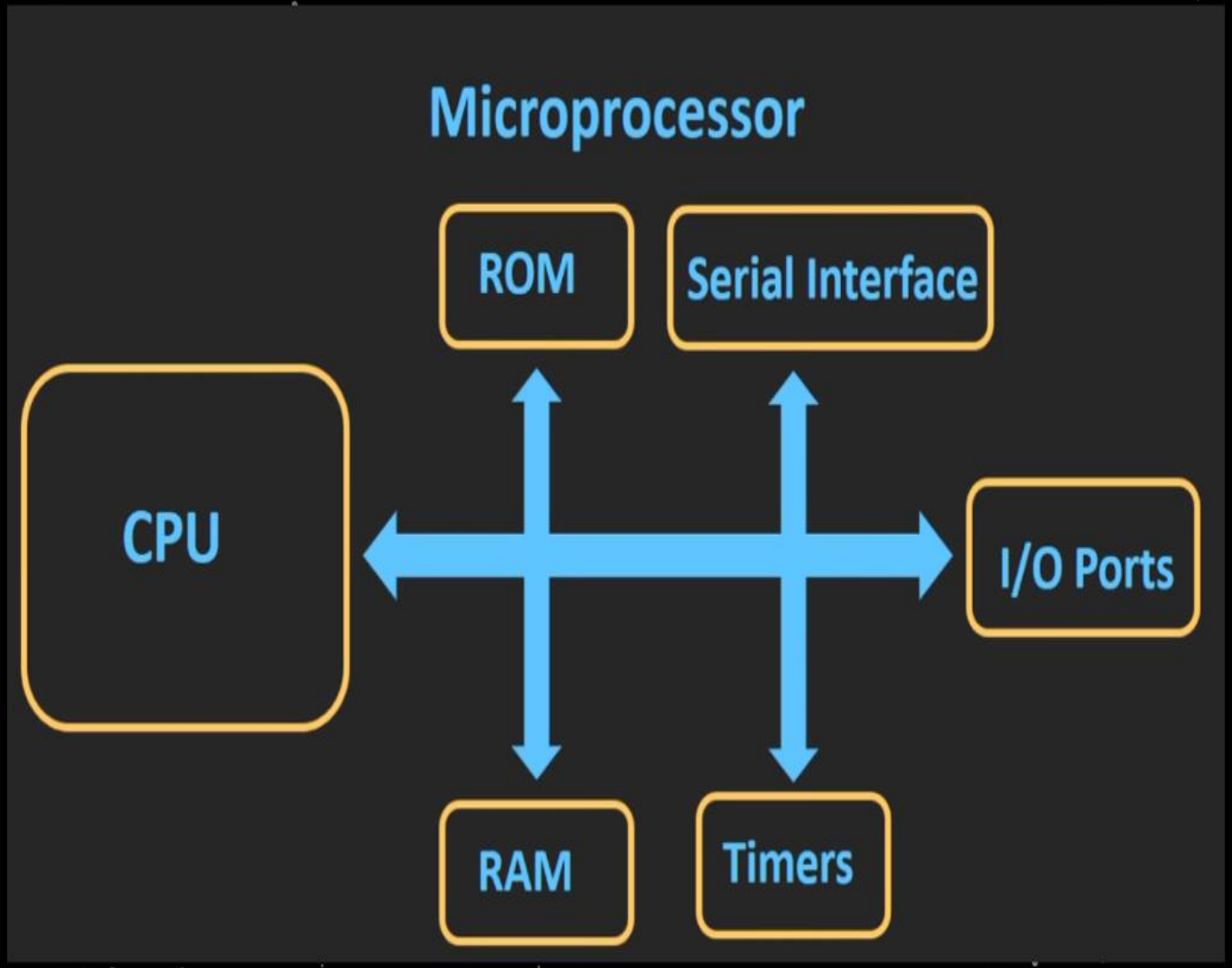
  Duty cycle is commonly expressed as a **percentage** or a **ratio**.

Pulse Width Modulation



Microprocessors

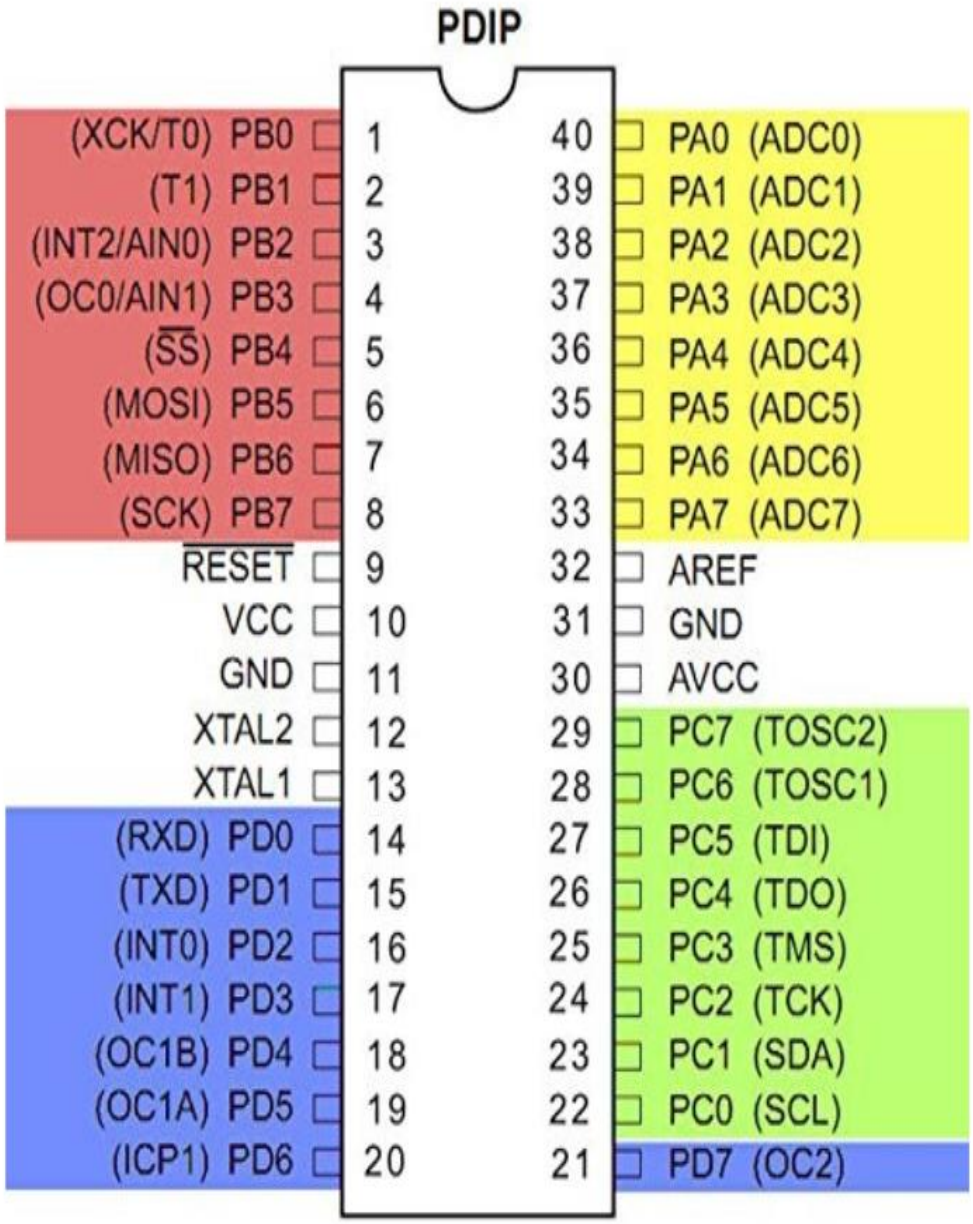
- Task is not Predefined
- It is used in applications where intensive processing is required
- For example, Gaming, calculation, Photo editing, simulations etc
- It only contains CPU and memory elements and I/O interfaces are connected Externally


Microprocessor
ROM
Serial Interface
CPU
I/O Ports
RAM
Timers

Summery

Parameters	Microprocessors	Microcontroller
Applications	Gaming, web browsing, document creation etc.	Dedicated for specific tasks (Camera, washing machine etc.)
Internal Structure	Memory and I/O devices connected externally	CPU, memory and I/O are present internally
Cost	High	Low
Power Consumption	High	Low
Memory (RAM)	512 MB to 32 GB	2KB to up to 256 KB
Storage	Hard Disk (128 GB to up to 2 TB)	Flash memory (32 KB to 2 MB)
Peripheral Interfaces	USB, UART, High speed Ethernet	UART, I2C, SPI

ATMEGA 32



Peripherals and Protocols

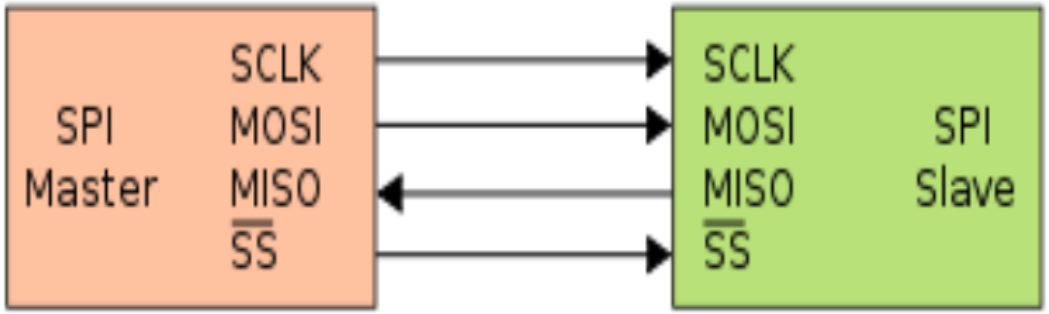
- **SPI - Serial Peripheral Interface**

It is an interface bus commonly used to send data between microcontrollers and small peripherals such as shift registers, sensors, and SD cards.

It uses separate clock and data lines, along with a select line to choose the device you wish to talk to.

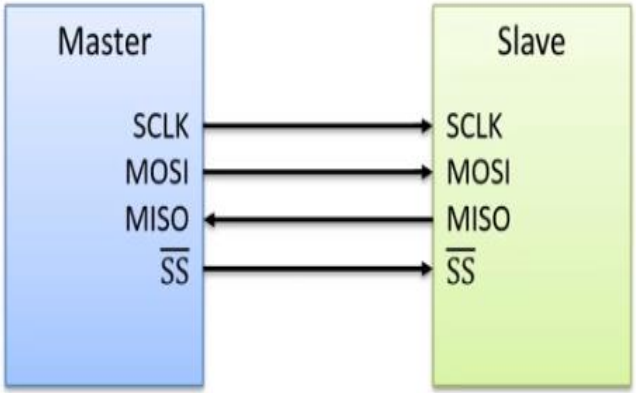
MOSI - Master Output Slave Input
MISO - Master Input Slave Output
SCK - Serial Clock
SS - Slave Select

SPI includes >

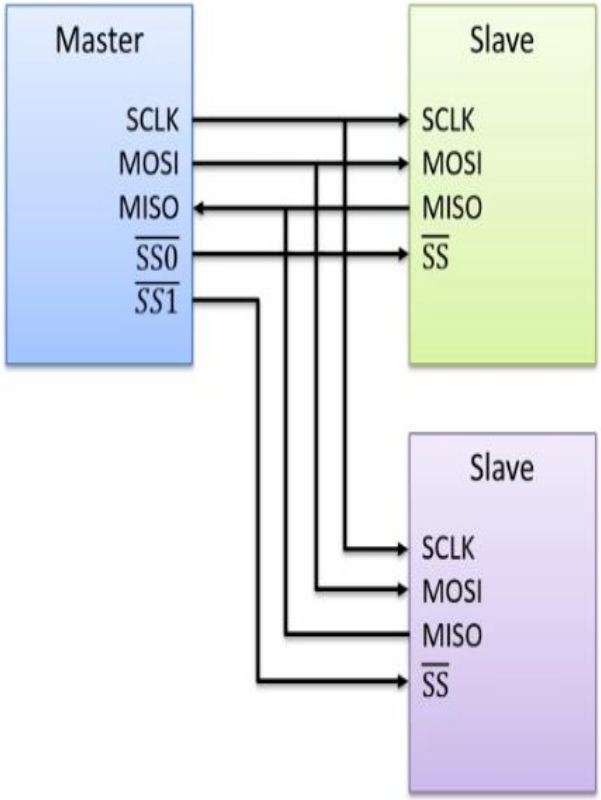


SPI Signaling

- Four wire communication
  - SCLK/SCK - Serial clock
  - MOSI - Master out, Slave in
  - MISO - Master in, Slave out
  - SS – Slave Select (or Chip Select)
    - Tells slave to prepare for data
    - Active low signal



Multiple Slave Setup



- **Watchdog timer**

- **SDA (Serial Data)** – The line for the master and slave to send and receive data.

- **SCL (Serial Clock)** – The line that carries the clock signal.

- **USART -** USART stands for **U**niversal **S**ynchronous and **A**synchronous **T**ransmitter and Receiver. It is a serial communication of two wire protocol.

  It is a **full-duplex** protocol means transmitting and receiving data simultaneously to different board rates. Different devices communicate with microcontroller to this protocol.

- There are three modes of transmission namely: simplex, half duplex, and full duplex. Transmission mode defines the direction of flow of signal between two connected deviceus.

I²C - Inter-Integrated Circuit

- **I²C** is a **multi-master protocol** that uses 2 signal lines.
- The two I²C signals are called 'serial data' (**SDA**) and 'serial clock' (**SCL**).
- Virtually any number of slaves and any number of masters can be connected onto these 2 signal lines and communicate between each other using a protocol

THINGS YOU NEED TO PROGRAM A MICROCONTROLLER

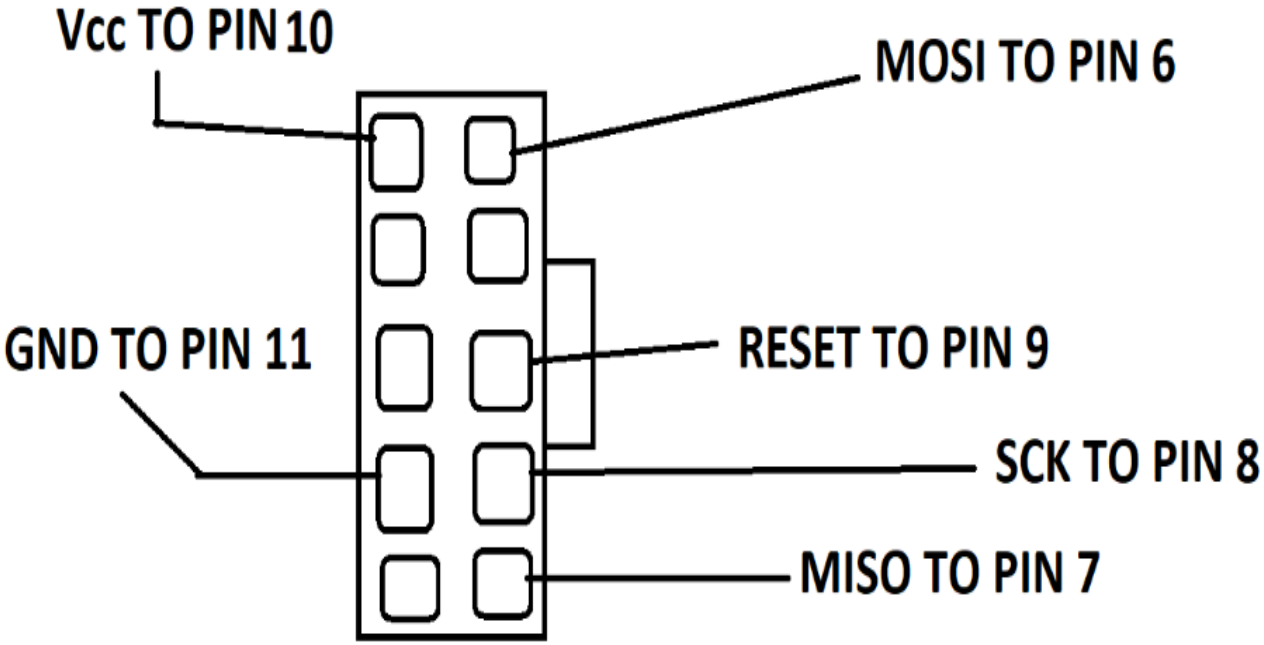
- **USBASP**

  It is a USB in-circuit programmer for Atmel AVR controllers. It simply consists of an ATMega88 or an ATMega8 and a couple of passive components.

- **Atmel Studios 7.0**

- **eXtreme Burner 1.2v**

USBASP Connections to ATMEGA 32



Basic Program for LED ON

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

int main(void)
{
   DDRB = 0b11111111;
   PORTB = 0b00000001;

   while (1)
   {
    }
}
```

LED ON AND OFF WITH DELAY

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

int main(void)
{
   DDRB = 0b11111111;
 PORTB = 0b11111111;


   while (1)
   {
 PORTB = 0b00000001;
 _delay_ms(500);
 PORTB = 0b000000010;
 _delay_ms(500);
 PORTB = 0b00000100;
 _delay_ms(500);
   }
}
```