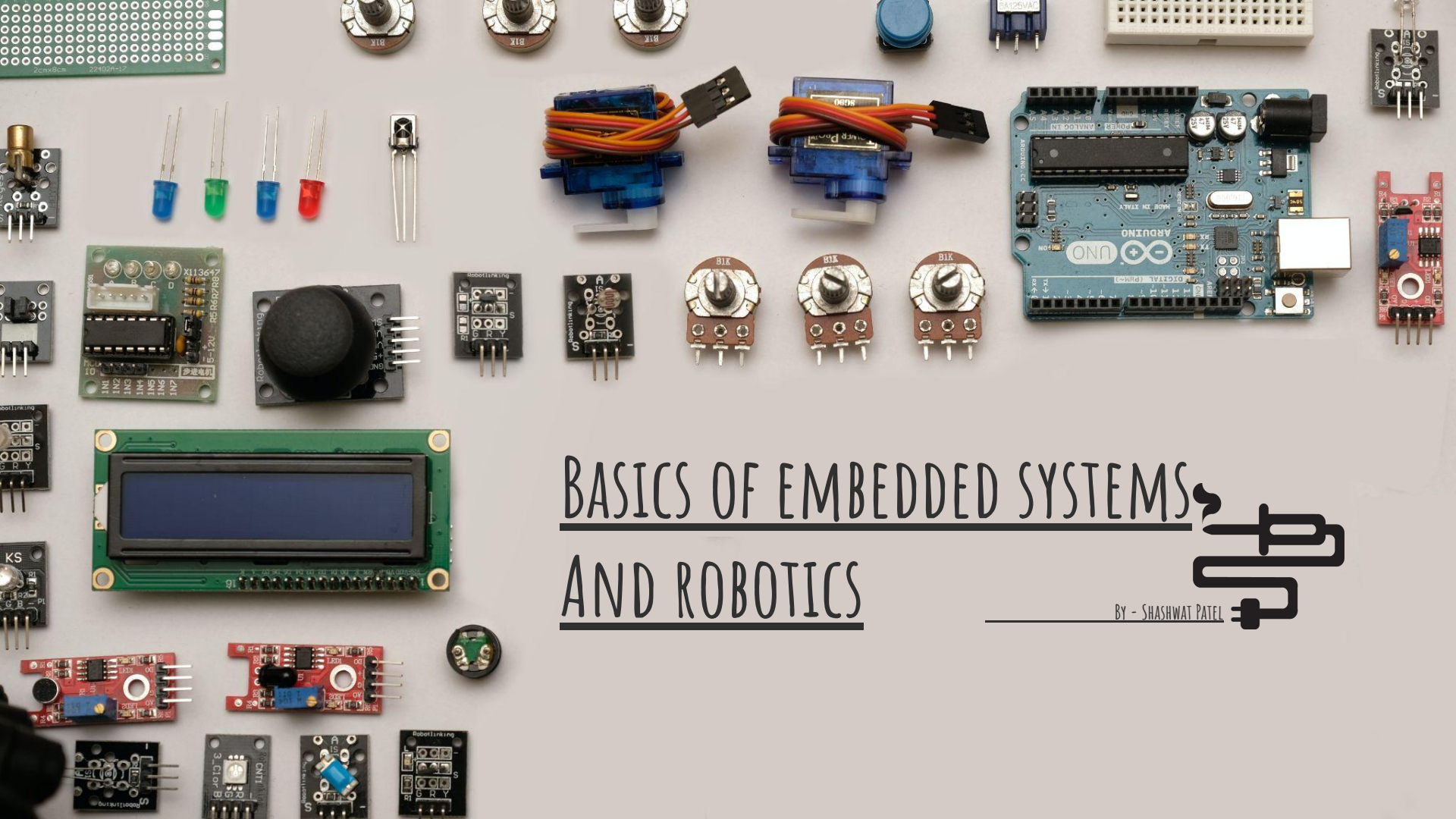


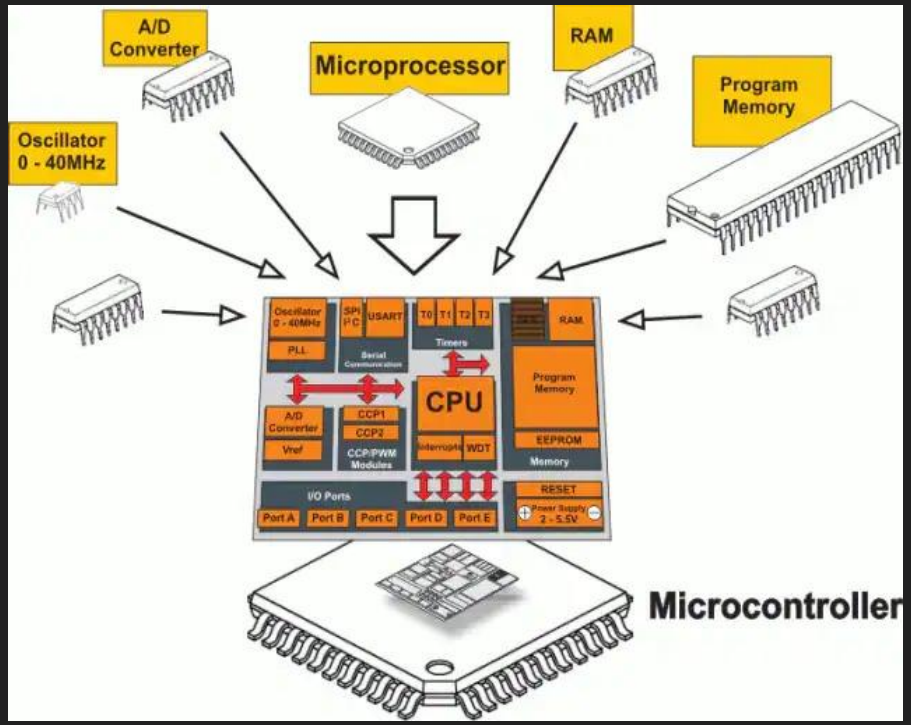
Basics of embedded systems And robotics

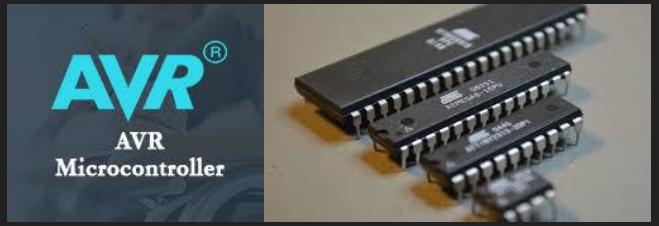
By - Shashwat Patel

Essential Components of a Microcontroller

- A microcontroller can be seen as a small computer, and this is because of the essential components inside of it.

- Random-Access Memory (RAM)
- Flash Memory
- Input/Output Ports (I/O Ports)
- Electrical Erasable Programmable Read-Only Memory (EEPROM)





AVR Microcontrollers

ATtiny series

- Microcontrollers of this subfamily has fewer features, fewer I/O pins, and less memory than other AVR series chips.
- Ex- ATtiny11,ATtiny12, ATtiny13,ATtiny85 etc

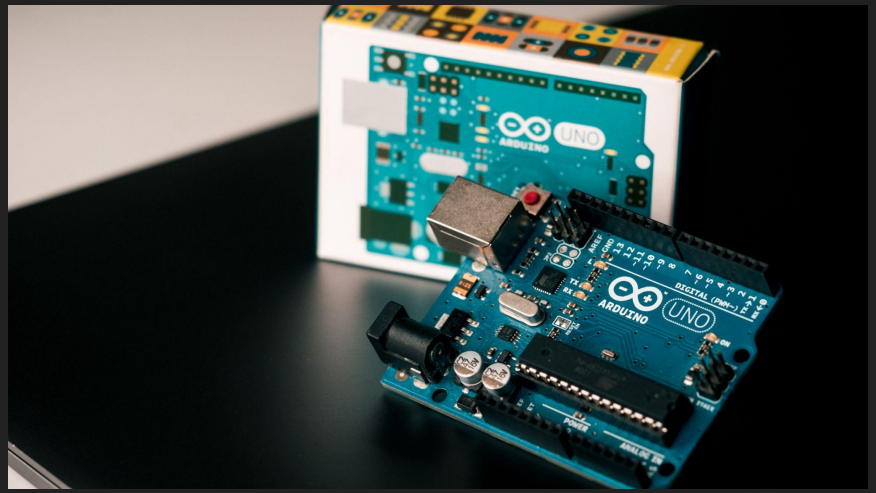
ATmega series

- microcontrollers of this subfamily typically has advance features with average number of I/O pins, and sufficient memory space.
- Ex- ATmega16, ATmega8, ATmega328(Arduino uno), ATmega168, ATmega644 etc.

ATxmega series

- Microcontrollers of this family provides combination of real time performance, high integration and low power consumption and simultaneously offers plenty of GPIO and huge amount of memory space.
- Ex- ATxmega162, ATxmega256 etc

Introduction to Arduino

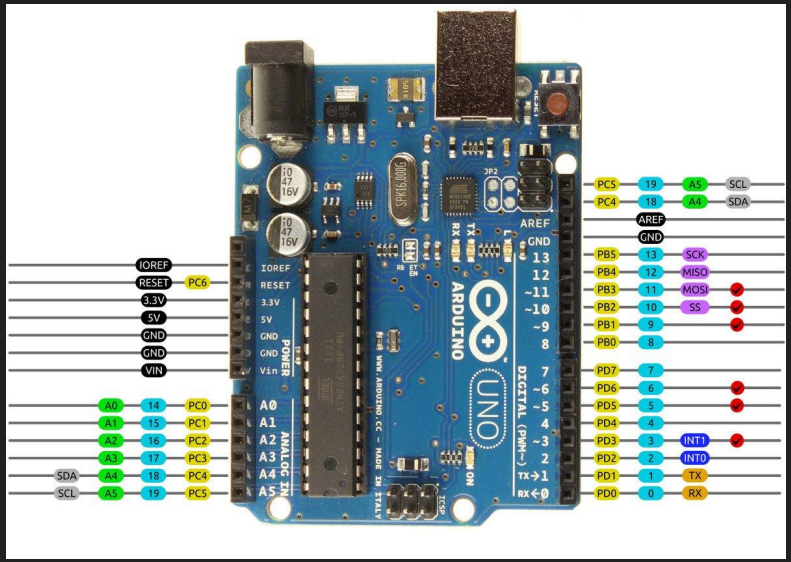


- Arduino is the development board of AVR microcontrollers.

- The first Arduino board was introduced in 2005 in italy to help students who had no previous experience in electronics or mcu programming to create working prototypes connecting the physical world to the digital world.

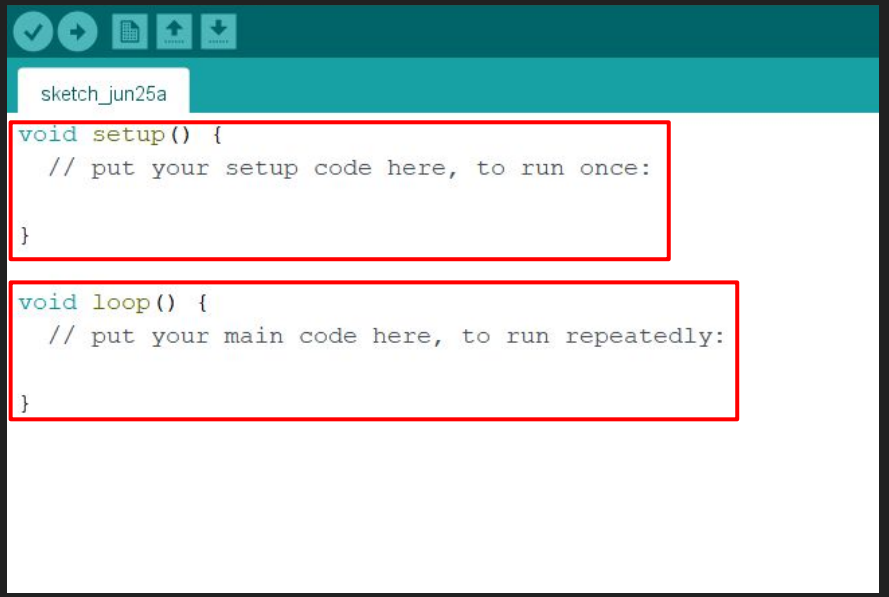
Hardware overview of Arduino

- Arduino UNO is the development board of ATmega328 controller of AVR family.
- It is an 8-bit device, which means that its data-bus architecture and internal registers are designed to handle 8 parallel data signals.
- It has nonvolatile Flash memory of 32KB volatile RAM of 2KB and nonvolatile EEPROM of 1KB.
- It has 23 GPIO pins out of which 6 pins can be used for analog and 6 pins can be used for PWM signals.



Structure of Arduino programming

- Any Arduino program must have at least 2 default functions.
- One is called setup(), the other is called loop(). The first is called once, when the program starts, the second is repeatedly called while your program is running.



Handling I/O

1. Digital I/O

- **pinMode()** - sets a pin to be an input, or an output. You pass the pin number and the INPUT or OUTPUT value as parameters.
  ex- pinMode(9, OUTPUT);

- **digitalWrite()** - writes a HIGH or LOW value to a digital output pin. You pass the pin number and HIGH or LOW as parameters.
  ex- digitalWrite(9, HIGH);

- **digitalRead()** - reads the value from a digital pin. Accepts a pin number as a parameter, and returns the HIGH or LOW constant.
  ex- int value = digitalRead(9);

2. Analog I/O

- **analogRead() -** reads the value from an analog pin**.**
  Ex- int value = analogRead(A0);

- **analogWrite() -** writes an analog value to a pin.
  Ex- analogWrite(A0, 200);

Time functions

- **delay() -** pauses the program for a number of milliseconds specified as parameter
  Ex- delay(1000); // pause the program for 1 second

- **delayMicroseconds() -** pauses the program for a number of microseconds specified as parameter
  Ex- delayMicroseconds(500); // pause the program for 500 microseconds

Math functions

- **map()** - Re-maps a number from one range to another. That is, a value of fromLow would get mapped to toLow, a value of fromHigh to toHigh, values in-between to values in-between, etc.
  Syntax- map(value, fromLow, fromHigh, toLow, toHigh);
  Ex- int y = map(x, 50, 100, 0, 255); // if x = 50 then y = 0

- **max() -** returns the maximum of two numbers
  Ex- int y = max(-1, 80); // y = 80

- **min() -** returns the minimum of two numbers
  Ex- int y = min(-1, 80); // y = -1

Write your first code

LED blink

```
#define ledPin 9
void setup() {
  pinMode(ledPin, OUTPUT);
   // configuring pin 9 as output
  digitalWrite(ledPin, LOW); // initially turning it off
}

void loop() {
  digitalWrite(ledPin, HIGH); // turn on the led
  delay(1000); // wait for 1 sec
  digitalWrite(ledPin, LOW); // turn off the led
  delay(1000); // wait for 1 sec
}
```