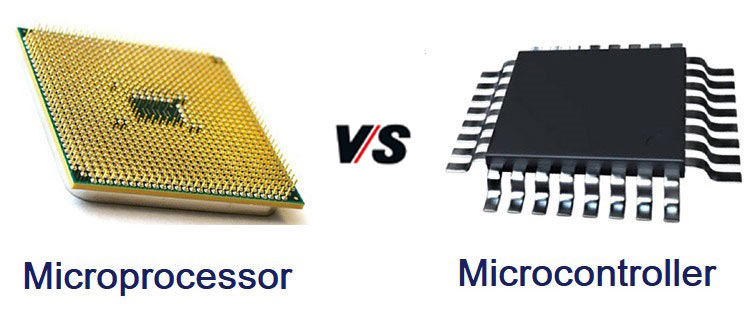
Task 2 – Report on Microprocessors, Microcontrollers and Arduinos

Srihari Srinivasan

1. Microprocessors and Microcontrollers 1
2. Types of Arduinos 2
3. Parts of an Arduino Uno 3
4. Arduino Nano 4
5. Arduino Uno 7
6. Arduino Mega 9
7. Programming 12
8. References 13

# Microprocessors:

* Data processing logic and control included in a single integrated or small number of integrated circuits.
* Contains arithmetic, logic and control circuitry.
* Integrated circuit can interpret and execute program instructions and perform arithmetic operations.
* **Used in personal computers and other general purpose applications**
* **Multi-purpose**
* Accepts binary input, processed according to instructions stored in the memory and provides binary output.
* Combinational Logic (output purely depends on given input at that time like Boolean algebra) and Sequential Digital Logic (output depends on the past inputs passed like FSMs which depend on order of inputs).

# Microcontrollers:

* Contains a CPU and RAM(Random Access Memory), ROM(Read Only Memory), I/O, etc.
* **Used in embedded applications** (software permanently present inside the device)
* **Particular functionality**
* Similar to System on a Chip (SoC has external MCU but usually integrates GPU and Wi-Fi).
* Reduce the size, power consumption and cost by making it perform only specified tasks, making it more efficient over multipurpose microcontrollers.

# Types of Arduinos:

1. Wireless Boards
   * 1. ESP32CAM [camera module, needs FTDI due to lack of usb]
     2. ESP8366 (nodemcu) [Lua] {Amica, Lolin}
     3. ethernet shield [can connect to uno]
     4. SIM900A [can connect to leonardo and mega]
2. Non-wireless Boards
   * 1. arduino pro-micro (Micro) [5 pwm, 9 analog]
     2. arduino pro-mini [needs programmer board FTDI or USB to TTL, 6 pwm, 8 analog] {(3.3/5)V, ATm(328/168)}

iii) arduino nano {ATmega328 (\*plain\*, Old Bootloader (OB), CH340G, OB+CH340G), ATmega168} [6 pwm, 8 analog]

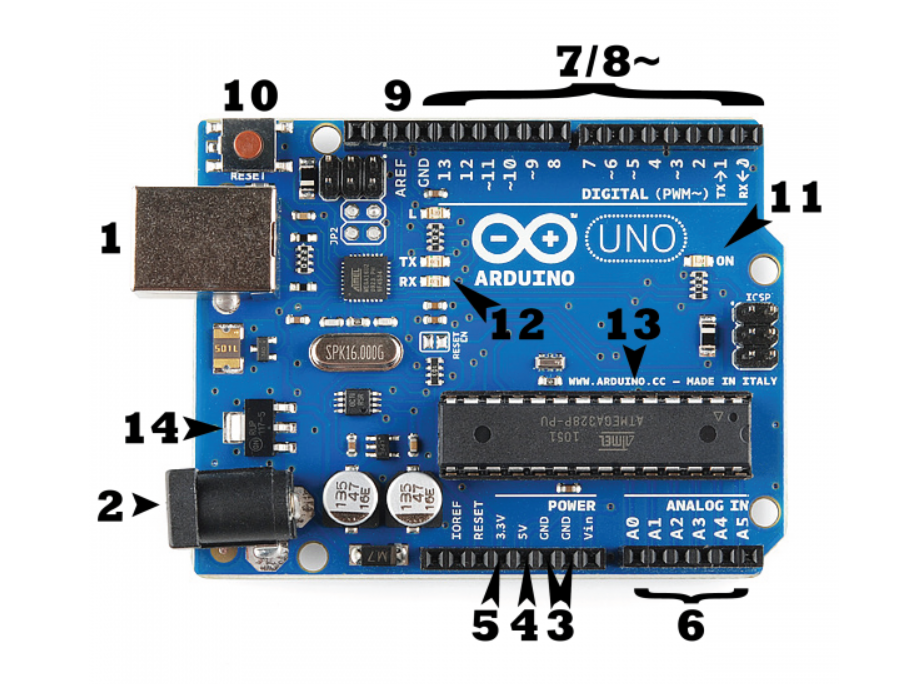
iv) arduino uno {smd (CH340G), non smd (i.e. can remove the IC (microcontroller))} [6 pwm, 6 analog]

v) arduino leonardo [7 pwm, 6 analog]

vi) arduino mega [ATmega2560, 12 pwm, 16 analog] {2560 aka ADK, CH340G}

1. Non-Arduino Boards
   * 1. lilypad [only computer/laptop programmable]
     2. tiny lilypad [only computer/laptop programmable]
     3. digispark {\*plain\* 16.5 mhz, Pro (16mhz, neither/32/64 byte)}
2. Programmers
   * 1. FTDI driver (Future Technology Devices International) [to register FTDI USB (they probably lack a USB jack so this is required) to TTL serial converter module to the system to use them as COM ports]
     2. USB to TTL converter board

# Parts of an Arduino Uno:



1. USB JACK

2. BARREL JACK (5V input MAX: 20V)

3. GROUND (Earthing)

4. 5V POWER SUPPLY

5. 3.3V POWER SUPPLY

6. ANALOG [read signals from analog sensor(0-5) and convert to digital value(0-1023), measure voltage (due to high resistance, they draw minimal current), can be used for digital input and output]

7. DIGITAL [read signals when input and act as power supply when output (40mA at 5V)]

8~. PULSE WIDTH MODULATION (PWM) [digital pins simulating analog output]

9. ANALOG REFERENCE (AREF) [reduces max analog voltage to a value below 5V (operating voltage)]

10. RESET BUTTON [sends a logical pulse to reset microcontroller pin, restarting the program]

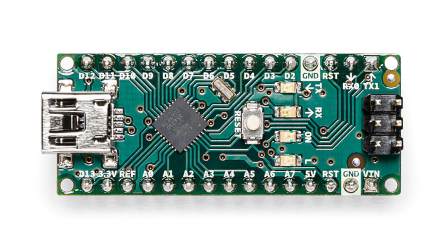
11. POWER LED INDICATOR

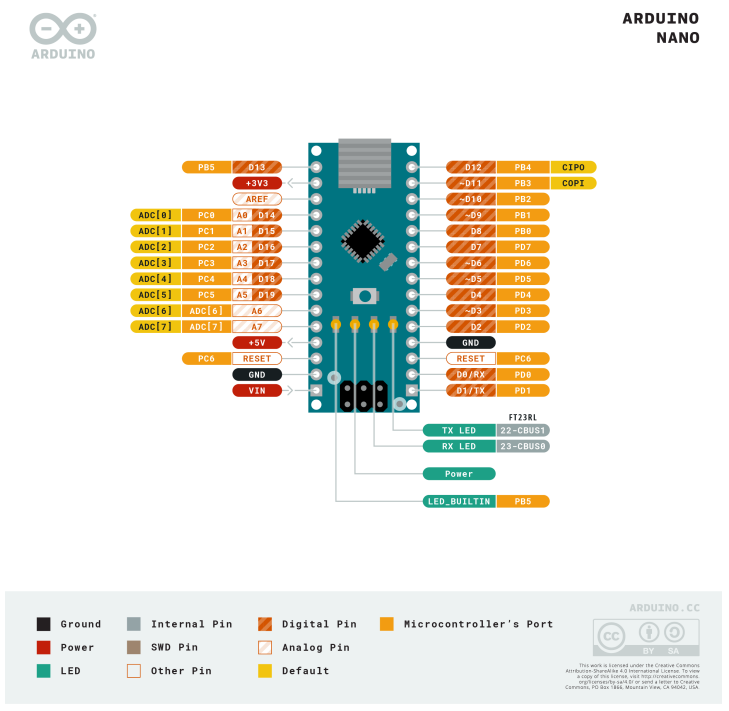
12. TRANSMIT AND RECEIVE [send and receive signals from Wi-Fi, Bluetooth, GPS modules, etc]

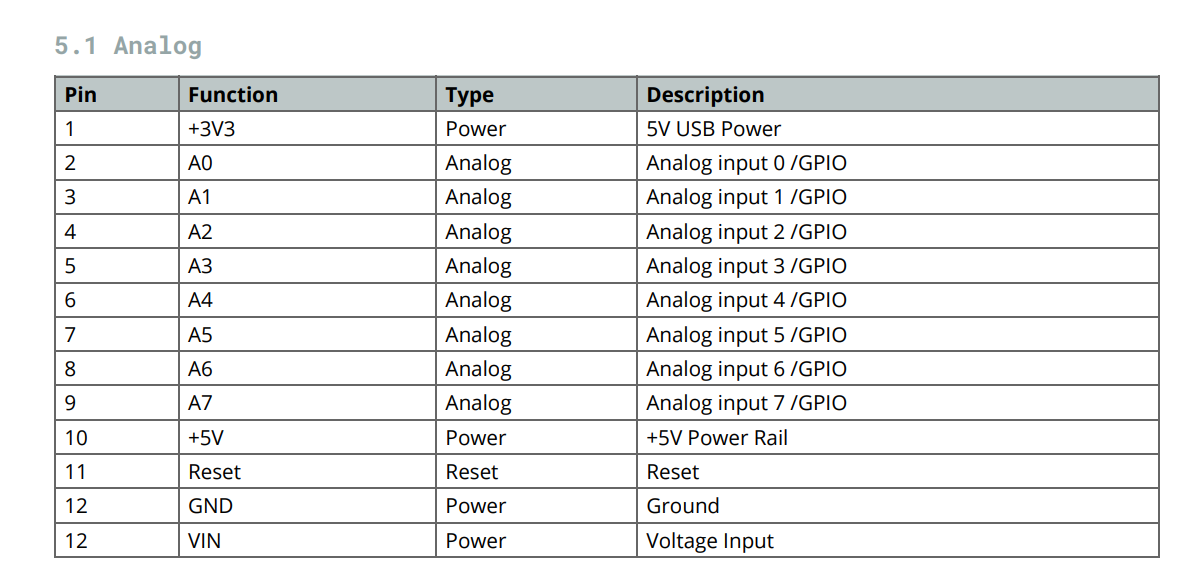
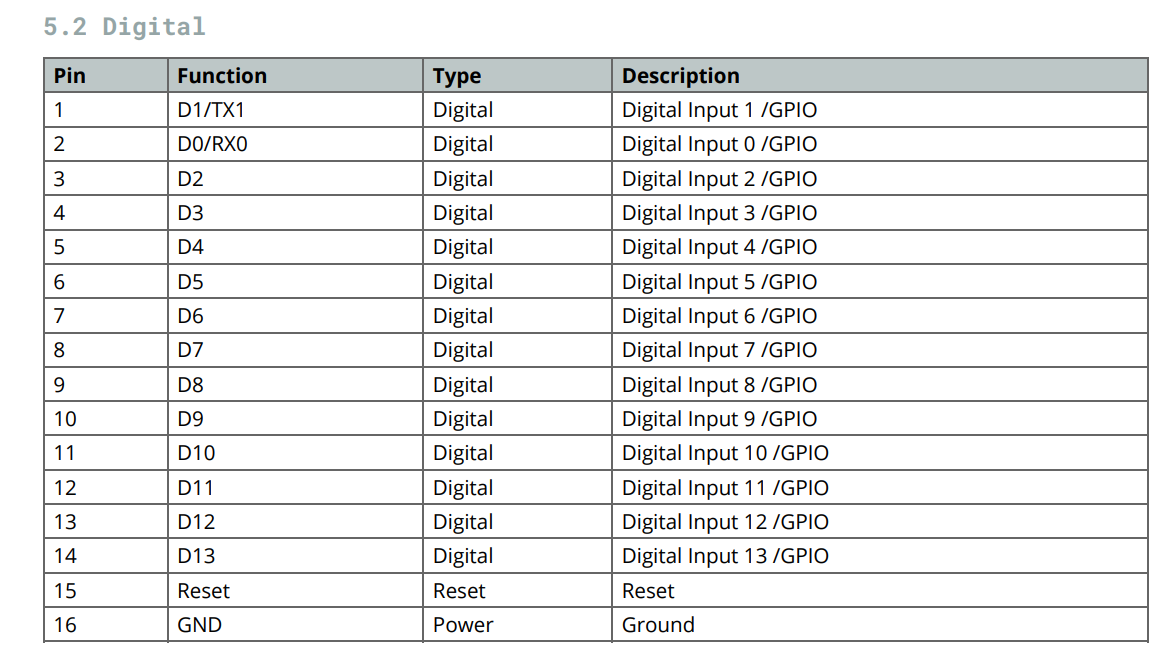
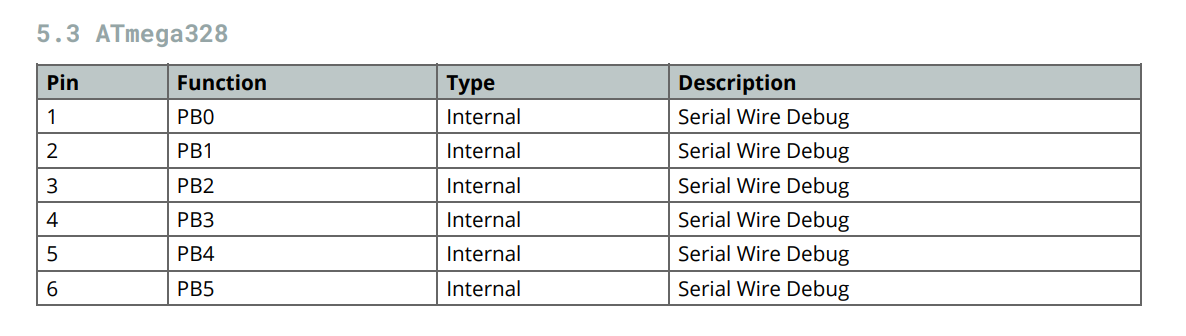
13. IC MICROCONTROLLER (ATmega328P) [32KB flash memory, 2KB RAM, CPU, EEPROM 1KB]

14. VOLTAGE REGULATOR

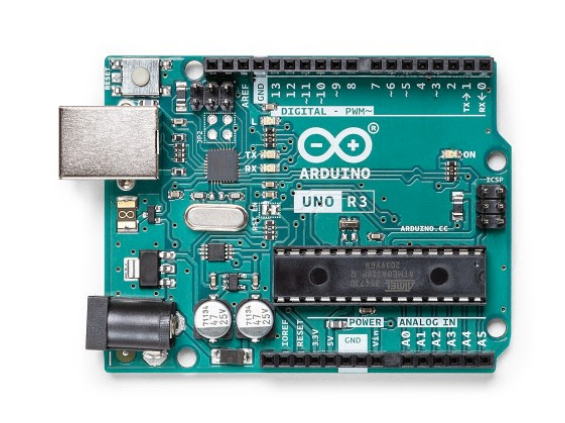
# Arduino Nano:

* ATmega328 Microcontroller
* High-performance low-power 8-bit processor
* Achieve up to 16 MIPS for 16 MHz clock frequency
* 32 kB of which 2 KB used by bootloader
* 2 kB internal SRAM
* 1 kB EEPROM
* Master/Slave SPI Serial Interface
* 22 Digital
* 8 Analog
* 6 PWM Output



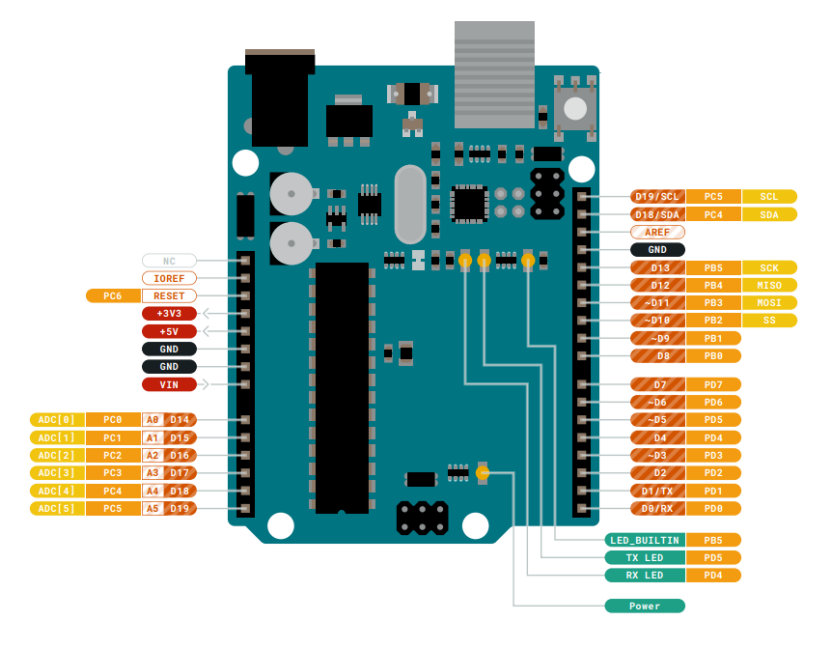


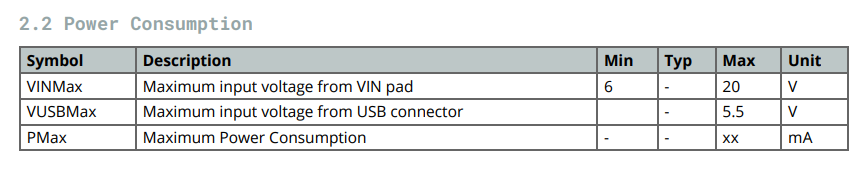
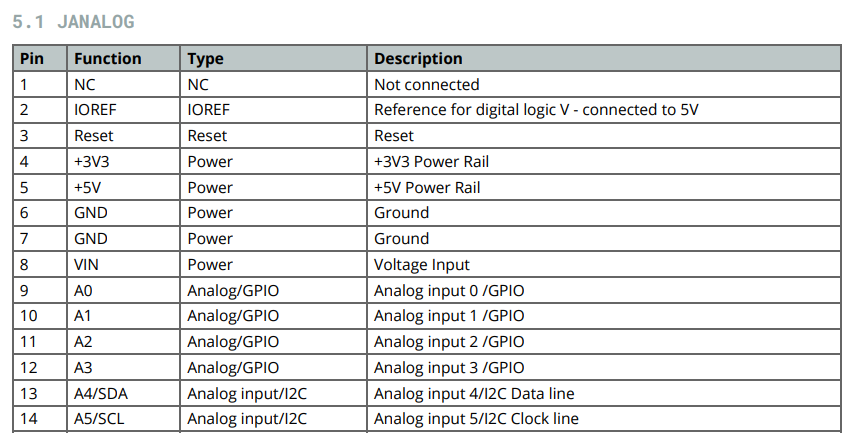
# Arduino Uno:

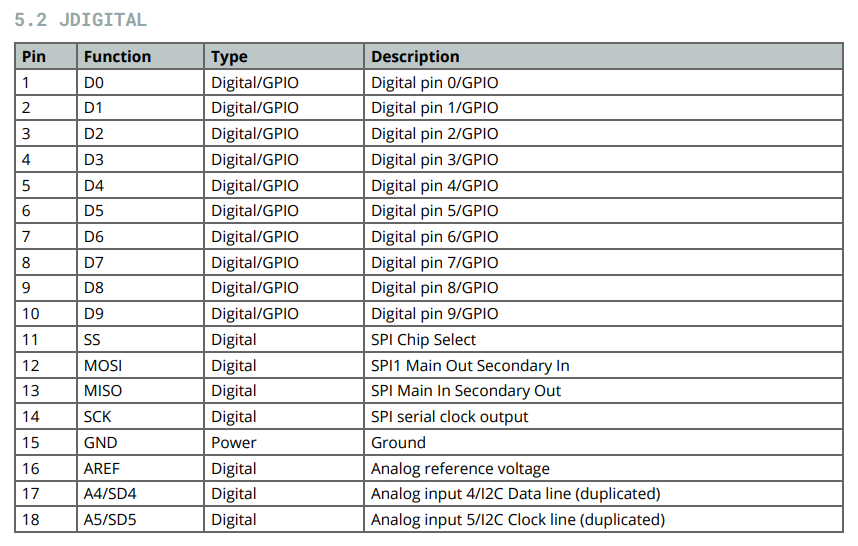


* ATMega328P Processor
* AVR CPU at up to 16 MHz
* 32KB Flash
* 2KB SRAM
* 1KB EEPROM
* 2x 8-bit Timer/Counter with a dedicated period register and compare channels
* 1x 16-bit Timer/Counter with a dedicated period register, input capture and compare channels
* 2.7-5.5V
* **ATMega16U2 Processor**

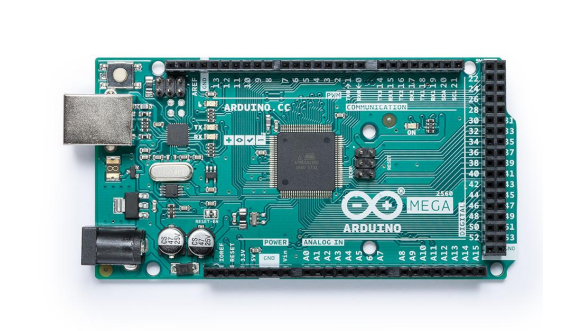
1. 8-bit AVR® RISC-based microcontroller
2. 16 KB ISP Flash
3. 512B EEPROM
4. 512B SRAM
5. debugWIRE interface for on-chip debugging and programming





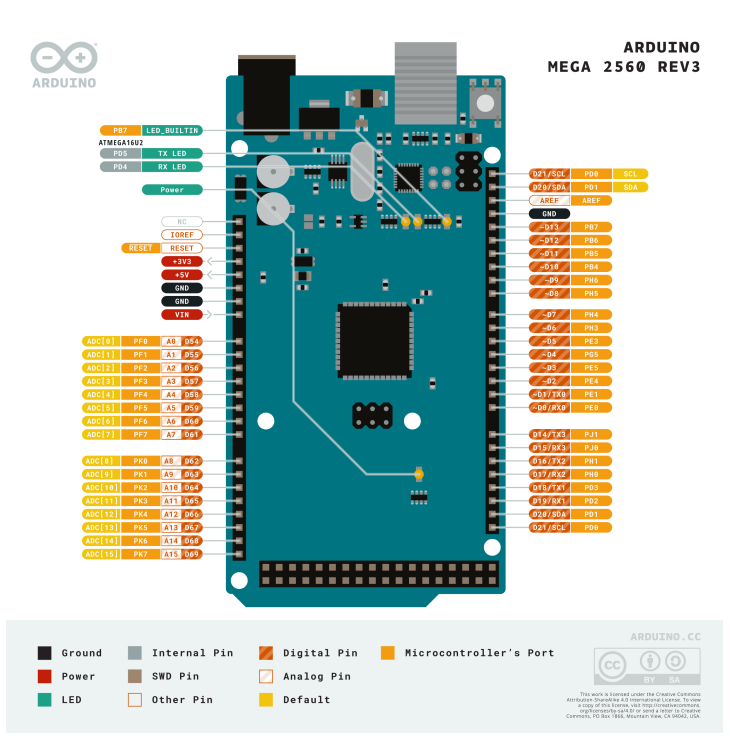


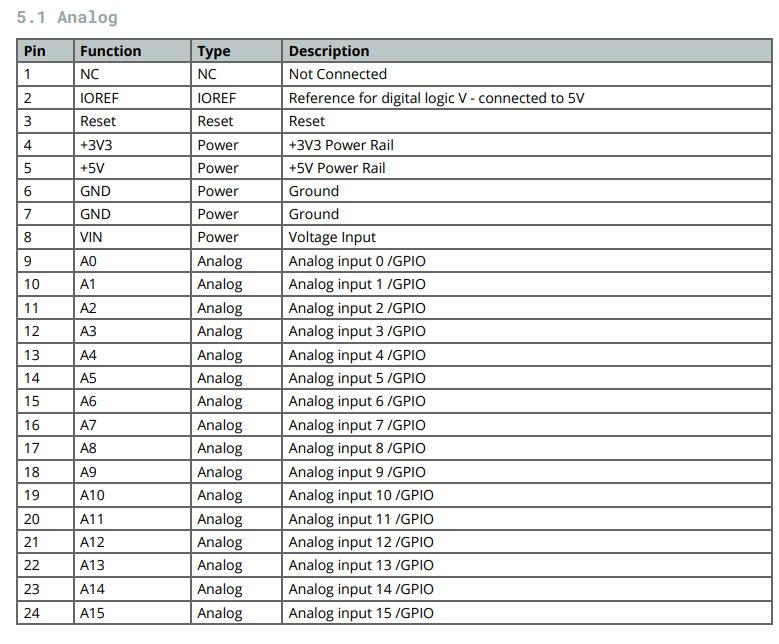
# Arduino Mega:

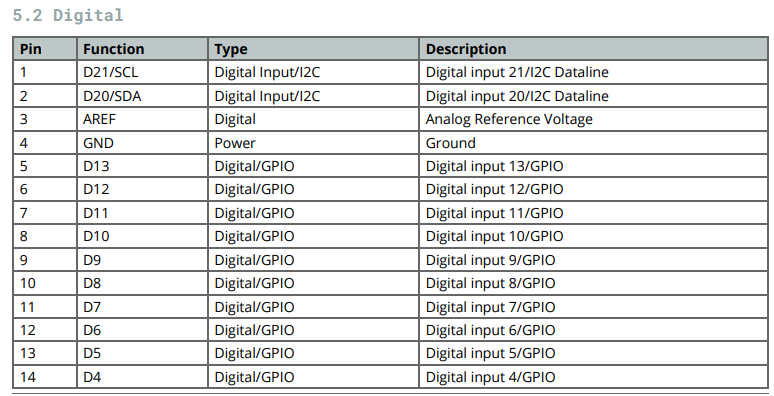
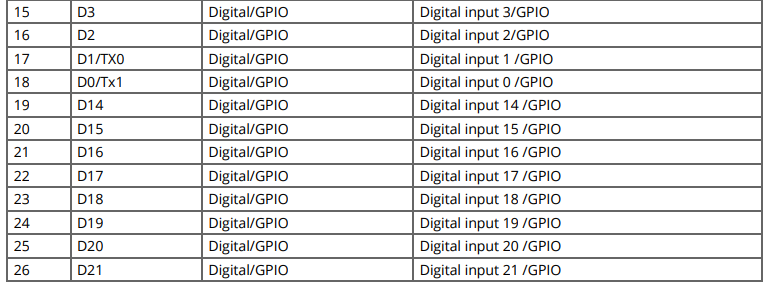


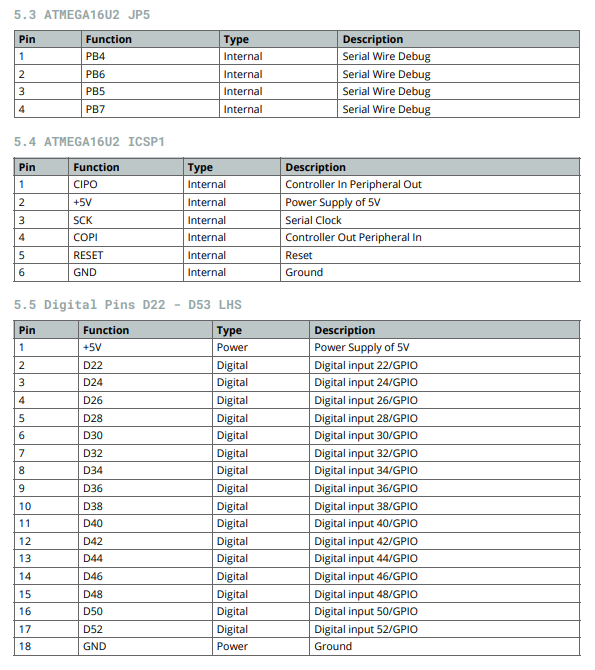
* ATmega2560 Processor
* Up to 16 MIPS Throughput at 16MHz
* 256k bytes (of which 8k is used for the bootloader)
* 4k bytes EEPROM
* 8k bytes Internal SRAM
* Four 8-bit PWM Channels
* 54 Digital
* 16 Analog
* 15 PWM Output
* **ATmega16U2**

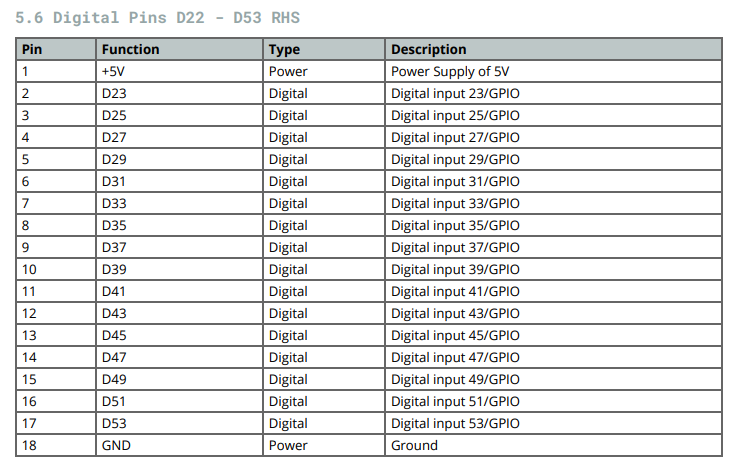
1. Up to 16 MIPS Throughput at 16 MHz
2. 16k bytes ISP Flash Memory
3. 512 bytes EEPROM
4. 512 bytes SRAM

****









### Analog vs Digital signals

An analog signal can take on any number of values. A digital signal, on the other hand, has only two values: HIGH and LOW. The Arduino has a built-in analog-to-digital converter (ADC,10-bit) that measures the value of analog signals. The ADC converts the analog voltage into a digital value.

# Programming:

The Arduino programming language is a **modified version of C/C++.** Usually, we program in C++ with the addition of methods and functions. A program written in Arduino programming language is called **sketch** and saved with .ino extension.

# Basic Functions:

#### Structure

**void setup() { <statements>; } -** Preparation for the program

**void loop() { <statements>; } -** Code that keeps repeating when arduino is being used

#### Setup()

**pinMode(<pin>, [OUTPUT/INPUT]); -** Declares the pin type.

**Serial.begin(9600); -** Opens serial port and sets the baud rate (usually 9600 bits/second) for serial data transmission.

#### Execution

**digitalWrite(<pin>, [HIGH/LOW]); --** Outputs either logic level HIGH or LOW at (turns on or off) a specified digital pin.

**digitalRead(<pin>);** -- Reads the value from a specified digital pin with the result either HIGH or LOW.

**analogWrite(<pin>, [0-255]);** -- Writes a pseudo-analog value using hardware enabled pulse width modulation (PWM) to an output pin marked PWM.

**analogRead(<pin>);** -- Reads the value from a specified analog pin with a 10-bit resolution. This function only works on the analog in pins. The resulting integer values range from 0 to 1023.

**delay(ms);** -- Pauses a program for the amount of time as specified in milliseconds

# References:

<https://en.wikipedia.org/wiki/Microprocessor>

<https://en.wikipedia.org/wiki/Microcontroller>

<https://www.youtube.com/watch?v=j2reiw1ptgk>

<https://www.youtube.com/watch?v=nL34zDTPkcs>

<https://learn.sparkfun.com/tutorials/what-is-an-arduino/whats-on-the-board#:~:text=Main%20IC,the%20brains%20of%20our%20Arduino>

<https://docs.arduino.cc/static/e7d6ca34b88350545d67282788df0956/A000005-datasheet.pdf>

<https://docs.arduino.cc/static/d1051542567b8387bd68998d1f08b197/A000066-datasheet.pdf>

<https://docs.arduino.cc/static/bd49a88997b4590197ac00ed9393d951/A000067-datasheet.pdf>