

ME-420: Introduction to Microcontrollers

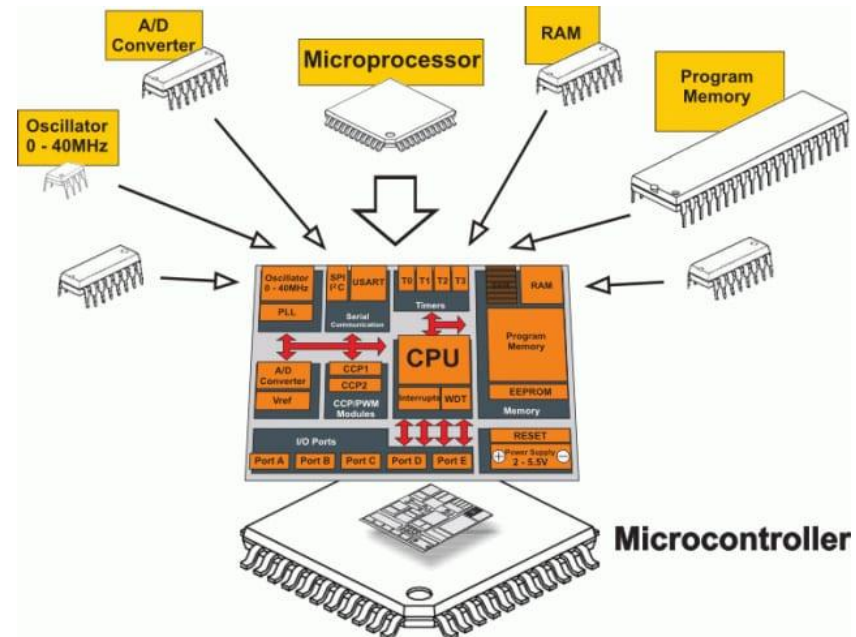
Ziqiao Wang, Prof. Jamie Paik
Reconfigurable Robotics Laboratory
EPFL, Switzerland

What is Microcontroller?

- **Definition:** A compact integrated circuit designed to govern a specific operation in an embedded system.
- **Components:**
 - Processor
 - memory
 - input/output pins
 - peripheral

Why we need microcontroller?

- Size and portability
- Power consumption
- Real-time Operation



Source: Max Embedded

Embedded Systems and Sustainable Development

SUSTAINABLE DEVELOPMENT GOALS



Source: United Nations, 2015

Microcontroller in ME-420

Inputs

Outputs

User



Actuators

Sensors



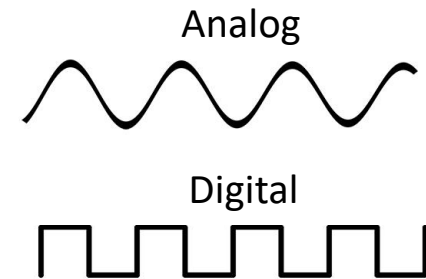
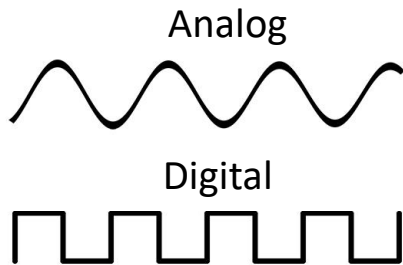
Programming

Reading

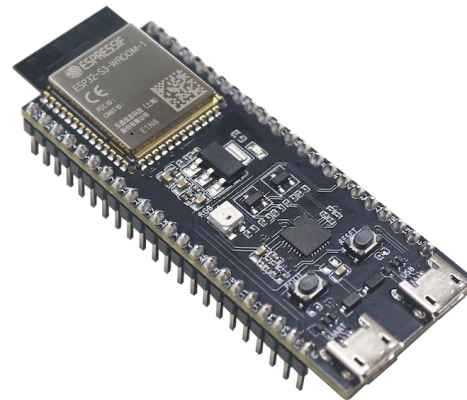


Indicators

Signals



Signals



Microcontroller can interact with electronic components to control your robot

Choose your board

Features	Specs	Integration	Sensors
<ul style="list-style-type: none">• ADCs• DACs• GPIO• Wifi• Bluetooth• ...	<ul style="list-style-type: none">• Clock speed• Memory• Interrupts• Power consumption• ...	<ul style="list-style-type: none">• Size• Weight• Input voltage• Output voltage• Output current• ...	<ul style="list-style-type: none">• Gyro• Accel• Magnetometers• Gesture• Humidity• Sound• ...

[Official arduino board](#)

Variety of Features,
Specs and Integrations
capabilities

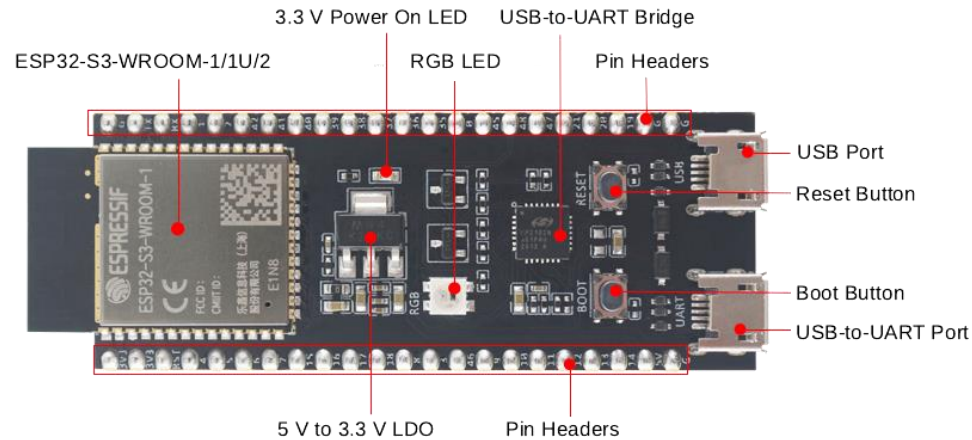
[Adafruit feather](#)

Small and powerful
arduino-like arduino, with
a lot of additional sensors
integrated

[Raspberry Pi](#)

Not an arduino, more like a small
computer = great processing power
and GUI, but only GPIO is available

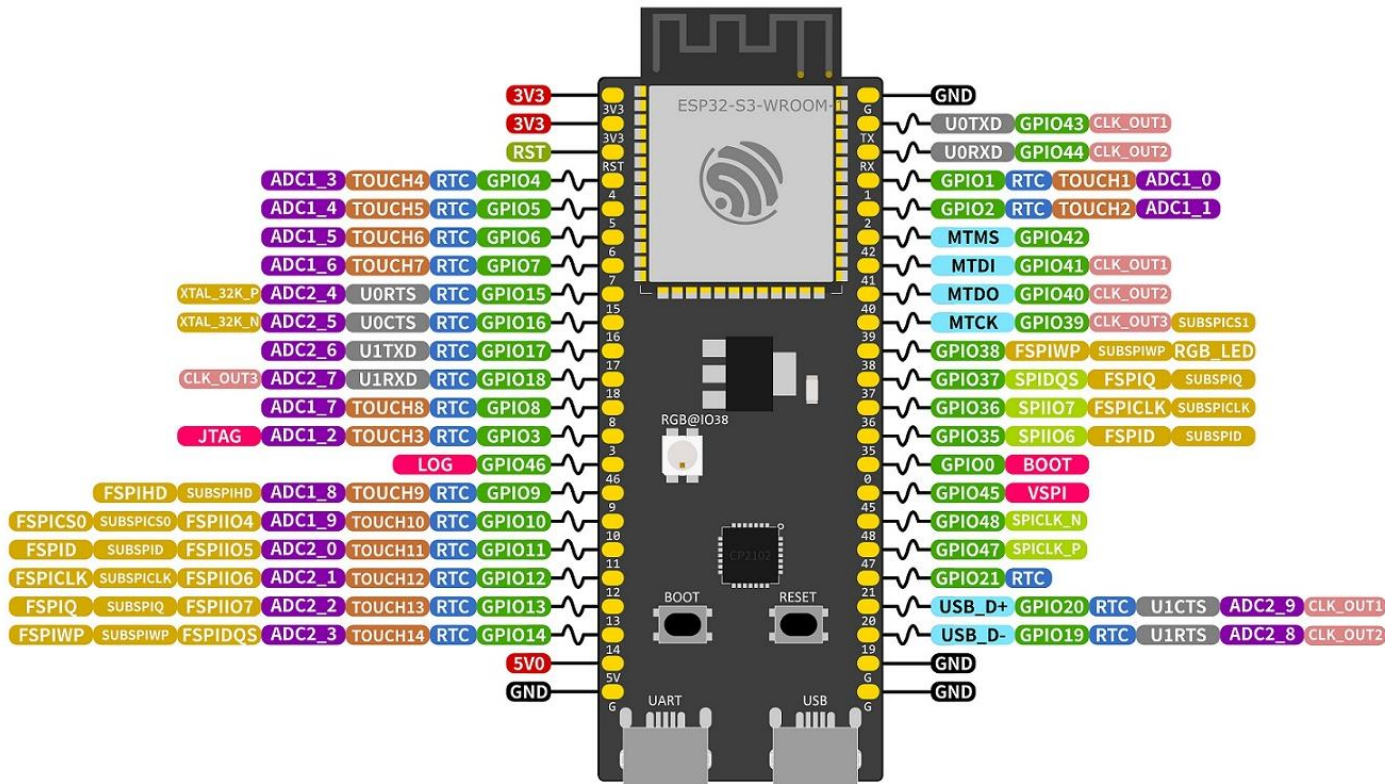
Case Study: ESP32-S3-DevKitC-1 Development Board



[Arduino-ESP32 Online Documentation](#)

- Comprehensive Feature Set
 - Combines standard microcontroller components with advanced features like wireless connectivity
- Strong Community and Resources
 - Extensive documentation, tutorials, and community support ease the learning curve
- Versatile Development Options
 - Compatible with various programming environments (e.g., Arduino IDE, MicroPython)

ESP32-S3 Pin Layout



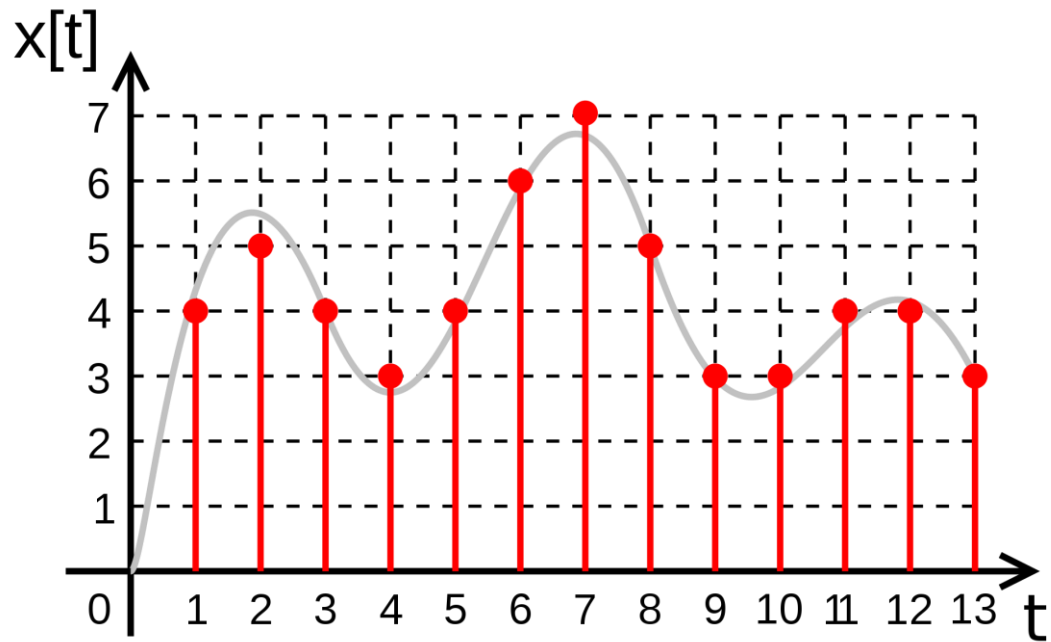
[Specs are available](#)

Features:

- ADC
- PWM
- UART (serial interface)
- Interrupts
- GPIO
- Bluetooth
- Wifi
- ...

Analog to Digital Converter (ADC)

ADC, or Analog-to-Digital Converter, translates analog signals into digital values for microcontrollers like ESP32 to read.



$$\text{Voltage} = \text{ADC reading} \times \frac{\text{reference voltage}}{\text{ADC Resolution}}$$

$2^n - 1$ with n the ADC bits

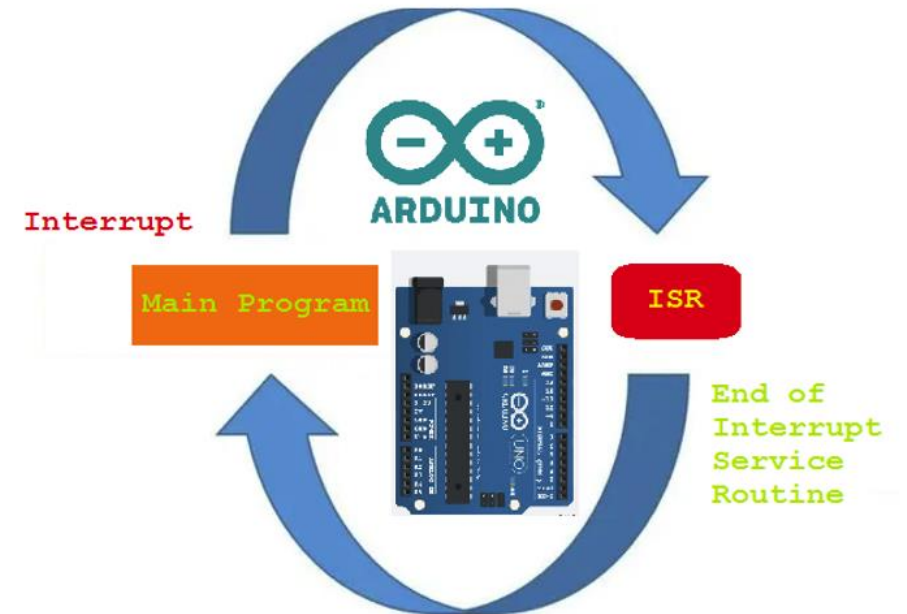
Interrupts

- **What is an Interrupt?**

- A signal to the CPU to pause and handle external events, enhancing real-time responsiveness and computational efficiency.

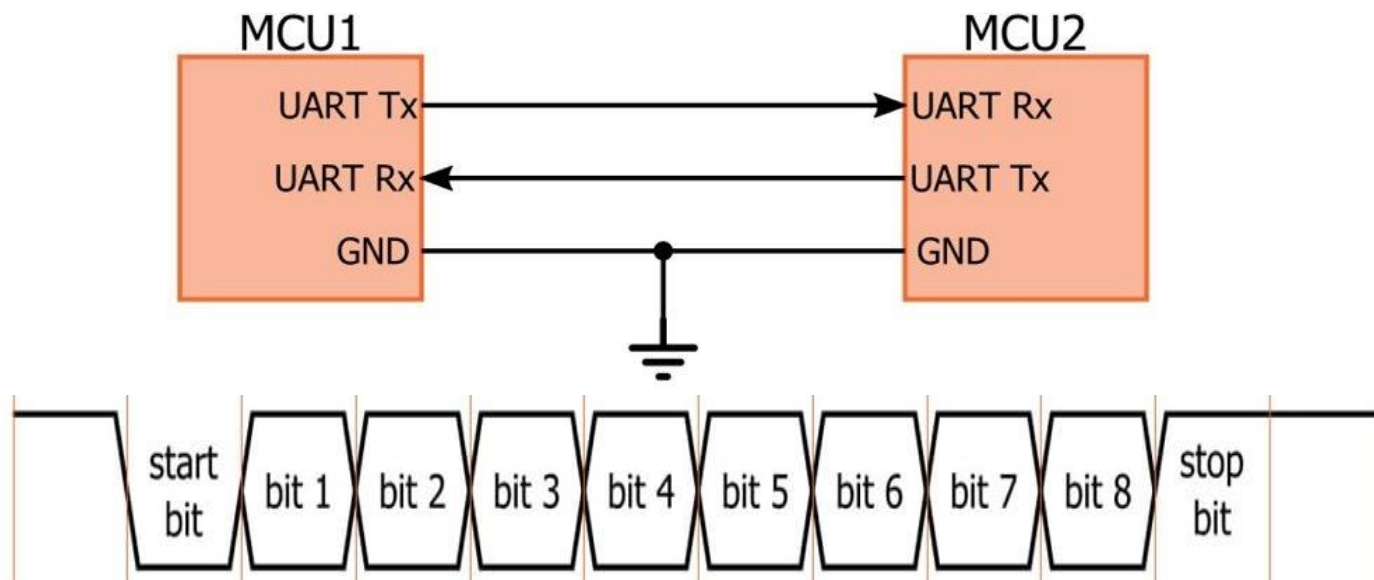
- **Key Concepts**

- **Types:** Hardware and Software Interrupts.
- **Purpose:** Ensure timely system responses and allow multitasking during I/O operations.
- **Usage:** Employed to handle tasks like responding to a button press, receiving data, or managing timed events in embedded systems.



Universal Asynchronous Receiver and Transmitter (UART)

UART (Universal Asynchronous Receiver/Transmitter) is a hardware communication protocol used for asynchronous serial communication between devices.



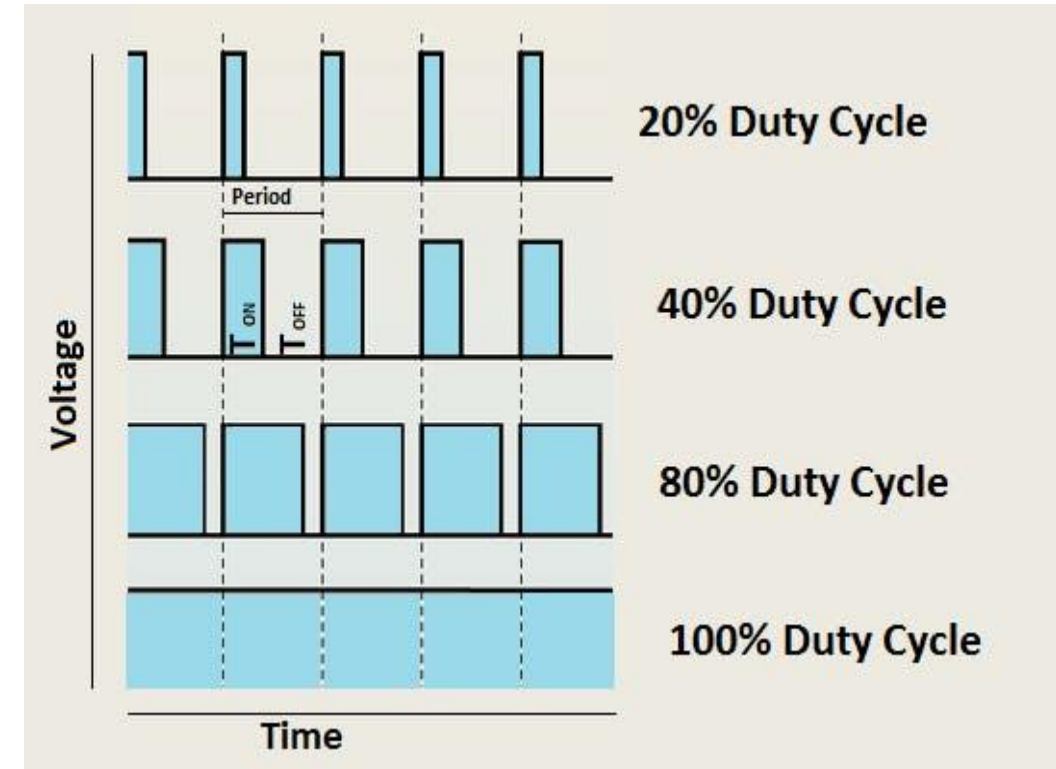
Key Terms

- **Start bit:** The first bit of a one-byte UART
- **Stop bit:** The last bit of a one-byte UART transmission.
- **Baud rate:** The approximate rate (in bits per second, or bps) at which data can be transferred.

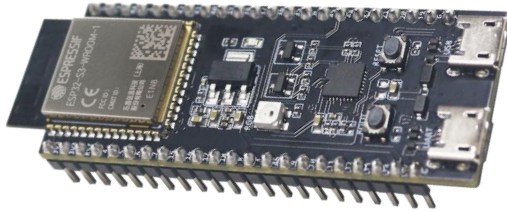
Pulse Width Modulation (PWM)

PWM stands for Pulse Width Modulation and it is a technique used in controlling the brightness of LED, speed control of DC motor, controlling a servo motor or where you have to get analog output with digital means.

- **TON (On Time):** It is the time when the signal is high.
- **TOFF (Off Time):** It is the time when the signal is low.
- **Period:** It is the sum of on time and off time.
- **Duty Cycle:** It is the percentage of time when the signal was high during the time of period.



Select the peripherals



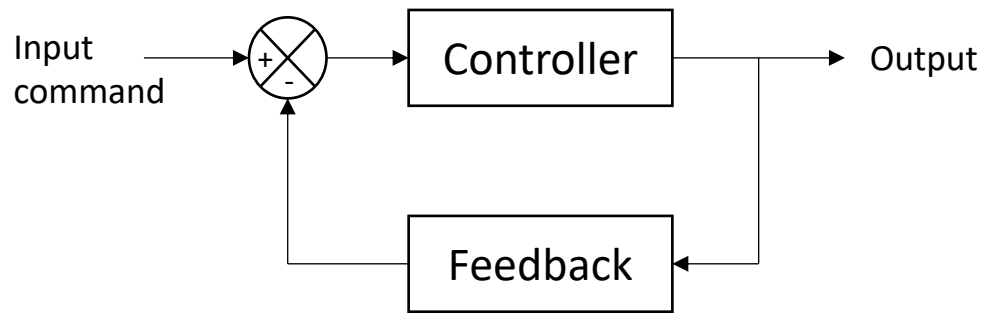
ECON Servo-180°

ESP32 S3 specs

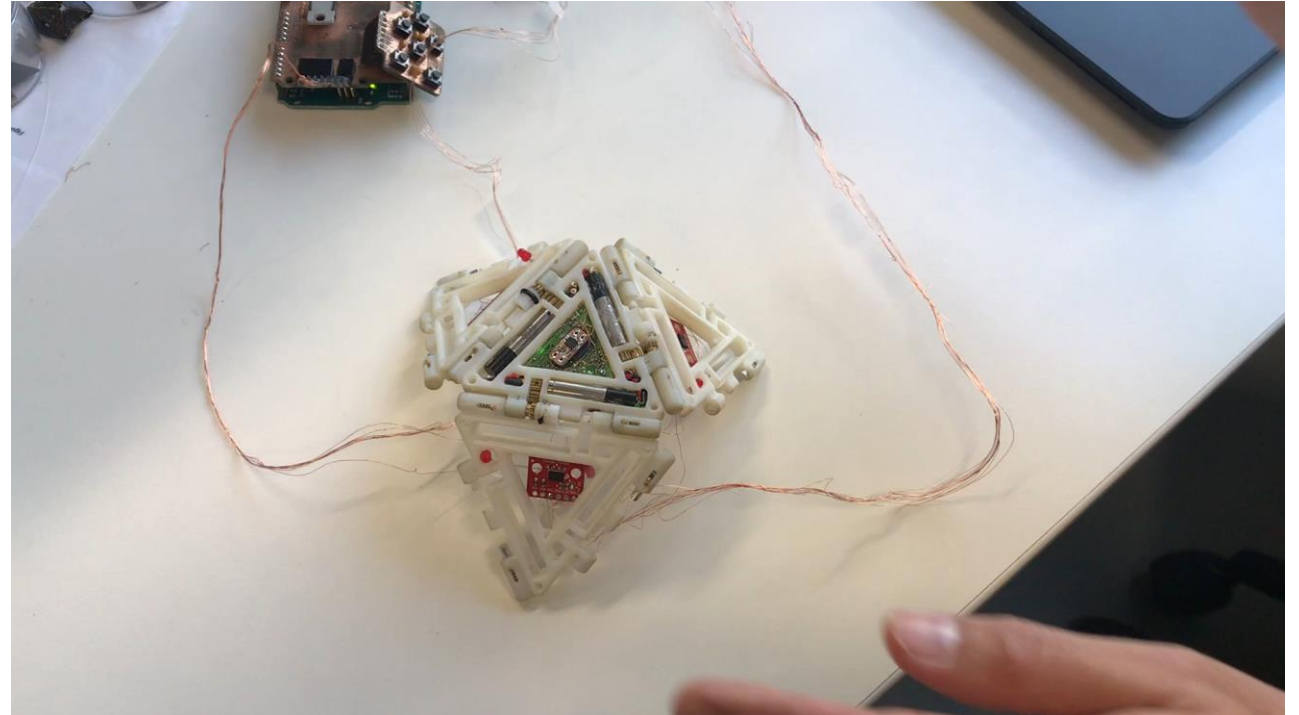
Operating Voltage	5V
Processor Architecture	32-bit
ROM	384KB
SRAM	512KB
GPIOs	45
UART	3x
Dc current per I/O Pin	40mA

Dimensions	40.8*20.1*38mm
Product Weight	44g
Output Shaft Style	25T
Angle Rotation	180 degree
Voltage Range	4.8~7.2 Volts
No-Load Speed(6.0V)	0.19sec/60°
No-Load Speed(4.8V)	0.23sec/60°
Stall Torque(6.0V)	4.1kg.cm
Stall Torque(4.8V)	3.2kg.cm
Max PWM Signal Range(Standard)	500~2500usec
Travel per μs(out of box)	/
Max Travel(out of box)	/
Pulse Amplitude	3~5V
Operating Temperature	(-)10 to +50 degree C
Current Drain -idle(6.0V)	20mA
Current Drain - no-load(6.0V)	200mA

Application example



- Inertial Measurement Unit give the robot's configuration
- Motors' relative rotation angles with encoders
- Interaction with human through distance sensors on the legs



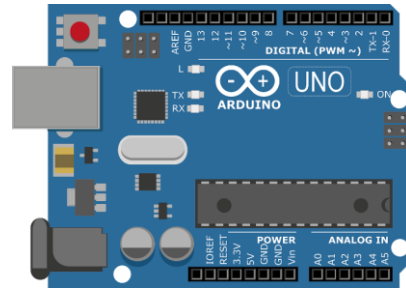
Huang, J. L. et al. (2018). A reconfigurable interactive interface for controlling robotic origami in virtual environments. *The International Journal of Robotics Research*, 37(6), 629-647.

What is Arduino?

Open-source hardware and software company that designs and manufactures **single-board microcontrollers**

Code and designs files are accessible by anyone!

- No “black box” = more control
- Completely free
- Don't rely on a company
- Active and large community



Provides all of the circuitry necessary for a useful control task: a microprocessor, I/O circuits, a clock generator, RAM, stored program memory and any necessary support ICs.

Basics in Arduino programming: IDE

[Download basic usual IDE](#)

A screenshot of the Arduino IDE interface. The window title is "sensortest | Arduino 1.8.13". The menu bar includes "Fichier", "Édition", "Croquis", "Outils", and "Aide". The toolbar shows icons for opening, saving, and running. The main editor area shows the following code:

```
sensortest
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_ADXL343.h>

/* Assign a unique ID to this sensor at the same time */
/* Uncomment following line for default Wire bus */
Adafruit_ADXL343 accel = Adafruit_ADXL343(12345);

/* NeoTrellis M4, etc. */
/* Uncomment following line for Wire1 bus */
//Adafruit_ADXL343 accel = Adafruit_ADXL343(12345, &Wire1);

void displaySensorDetails(void)
{
  sensor_t sensor;
  accel.getSensor(&sensor);
}
```

The status bar at the bottom indicates the board and library: "1 Adafruit Feather Bluefruit Sense, 0.3.2 SoftDevice s140 0.1.1, Level 0 (Release) sur COM3".

Pros:

- Perfect for beginners
- Easy to use
- Great Serial plotter
- GUI makes it easy to change the board connection parameters

Cons:

- No color highlight
- No autocomplete
- Difficult Multi-file programming
- It's like programming on a notepad

Basics in Arduino programming: IDE

Visual studio code

+

PlatformIO



```

2 'use strict';
3 var express = require('express');
4 var router = express.Router();
5 var notfound_1 = require('./utils/notfound'); // use latest TS 1.5,
  inspired from ES6
6 //import four0four = require('./utils/404');
7 var data = require('./data');
8 router.get('/people', getPeople);
9 router.get('/person/:id', getPerson);
10 router.get('/*', notfound_1.notFoundMiddleware);
11 module.exports = router;
12 ///////////////////////////////////////////////////
13 //EG TODO: find type for next argument
14 function getPeople(req, res, next) {
15   res.status(200).send(data.getPeople());
16 }
17 function getPerson(req, res, next) {
18   var id = +req.params.id;
19   var person = data.getPeople().filter(function (p) {
20     return p.id === id;
21   })[0];
22   if (person) {
23     res.status(200).send(person);
24   }
25   else {
26     notfound_1.send404(req, res, 'person ' + id + ' not found');
27   }
28 }

```

Pros:

- For **real** programmers
- Multi-file organisation
- Programs easily transportable
- DARK MODE
- Works with a lot of boards

Cons:

- Serial communication extremely basic
- Not as straightforward to use than the official IDE

Arduino is C++

(So you can use any C++ IDE)

Basics in Arduino programming

Arduino is embedded C++ made easy

Arduino code must
contains two functions:

Run once at the code
initialization (board startup)

```
void setup(void)  
{  
}
```

Init

- Variables
- GPIO
- Timers
- Interrupts
- All the system parameters
your program needs

Infinite loop, equivalent
to while(true)

```
void loop(void)  
{  
}
```

Run continuously while the
board is power supplied

- Read sensors
- Send control commands
- Communicate with the
computer
- Contains the program logics

Basics in Arduino programming

Example

*Declare variables as global
(variables declared inside a function are
deleted when the function ends)*

```
/*
DigitalReadSerial

Reads a digital input on pin 2, prints the result to the
Serial Monitor

This example code is in the public domain.
http://www.arduino.cc/en/Tutorial/DigitalReadSerial
*/
```

```
// digital pin 2 has a pushbutton attached to it.
Give it a name:
int pushButton = 2;

// the setup routine runs once when you press
reset:
void setup() {
  // initialize serial communication at 9600 bits
per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}
```

```
// the loop routine runs over and over again
forever:
void loop() {
  // read the input pin:
  int buttonState = digitalRead(pushButton);
  // print out the state of the button:
  Serial.println(buttonState);
  delay(1);    // delay in between reads for
stability
}
```

Lots of examples are available under *File* ->
Examples In the Arduino IDE

Basics in Arduino programming

Functions

Please look at the [datasheet](#) before using built-in functions

Declaration → `void customFunction(char inputVariable);`

```
void setup(void)
{
  Serial.begin(9600)
}

void loop(void)
{
  customFunction('a');
}
```

Call →

Function body → `void customFunction(char inputVariable){
 Serial.println(inputVariable);
}`

Take home messages

As much as possible:

- If the electronic is more complex than a voltage divider, buy it
- Use arduino libraries, do not program everything from scratch
- Read carefully the datasheet

EPFL Demo: Programming the ESP32 Using the Arduino IDE

