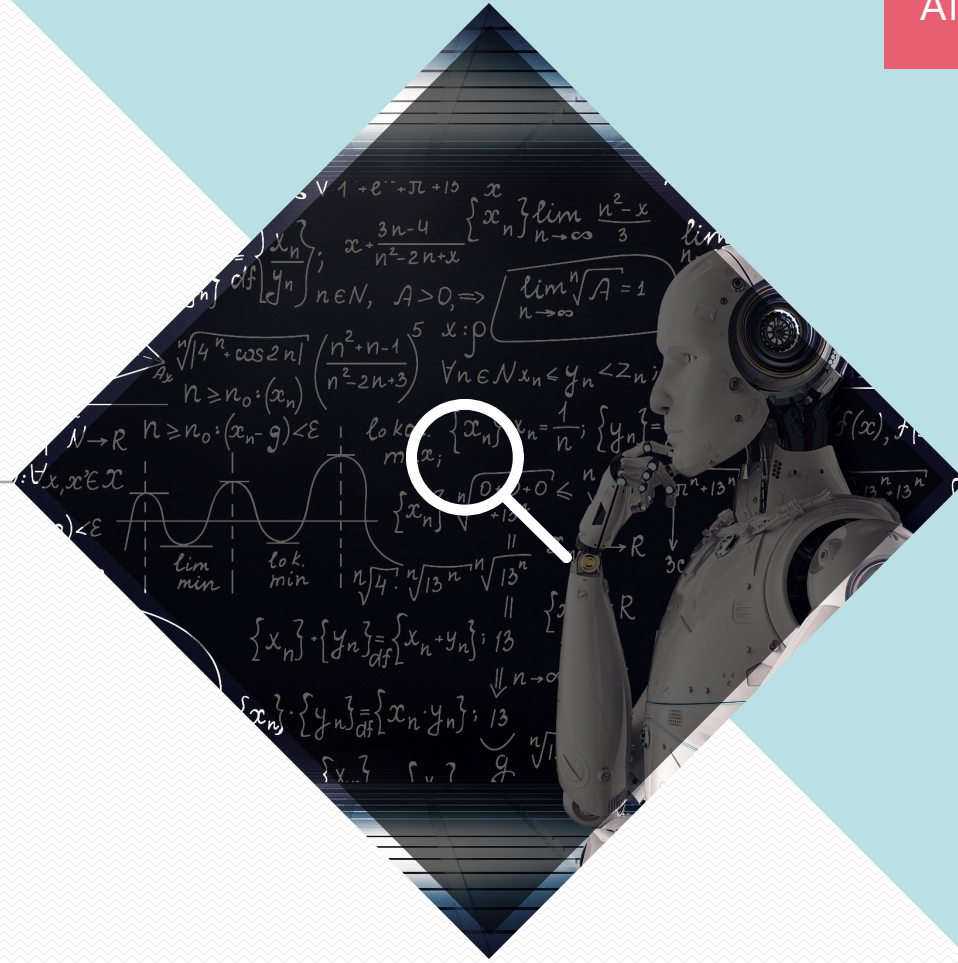


아두이노 실습

Using the Arduino Uno

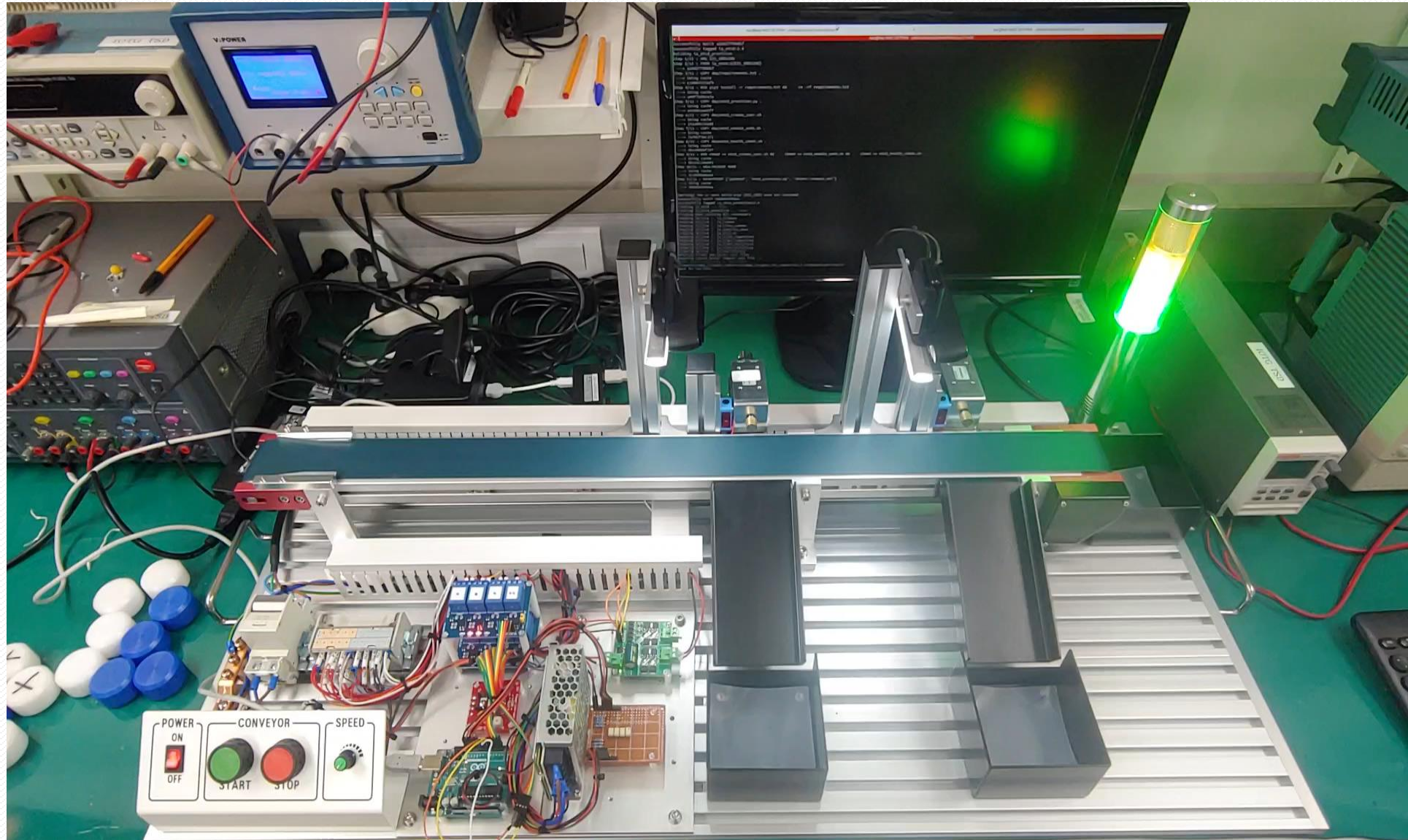


Agenda

- *Hands-on System Overview*
- *Basic Hands-on*
- *Smart Factory Hands-on*
- *AI Hands-on*
- *Demo Implementation*
- *Why Arduino?*
- *Arduino Environment Overview*
- *Arduino Directory Analyze*
- *Arduino Control*

Hands-on System Overview

Smart Factory



Overview

NUC

NUC10i7FN

10th Generation Intel® Core™ i7-10710U

1.1 GHz – 4.7 GHz Turbo, 6 core, 12 thread, 12MB Cache

25W Intel® UHD Graphics, 300 MHz – 1.15 GHz

16GB RAM / 512GB SSD

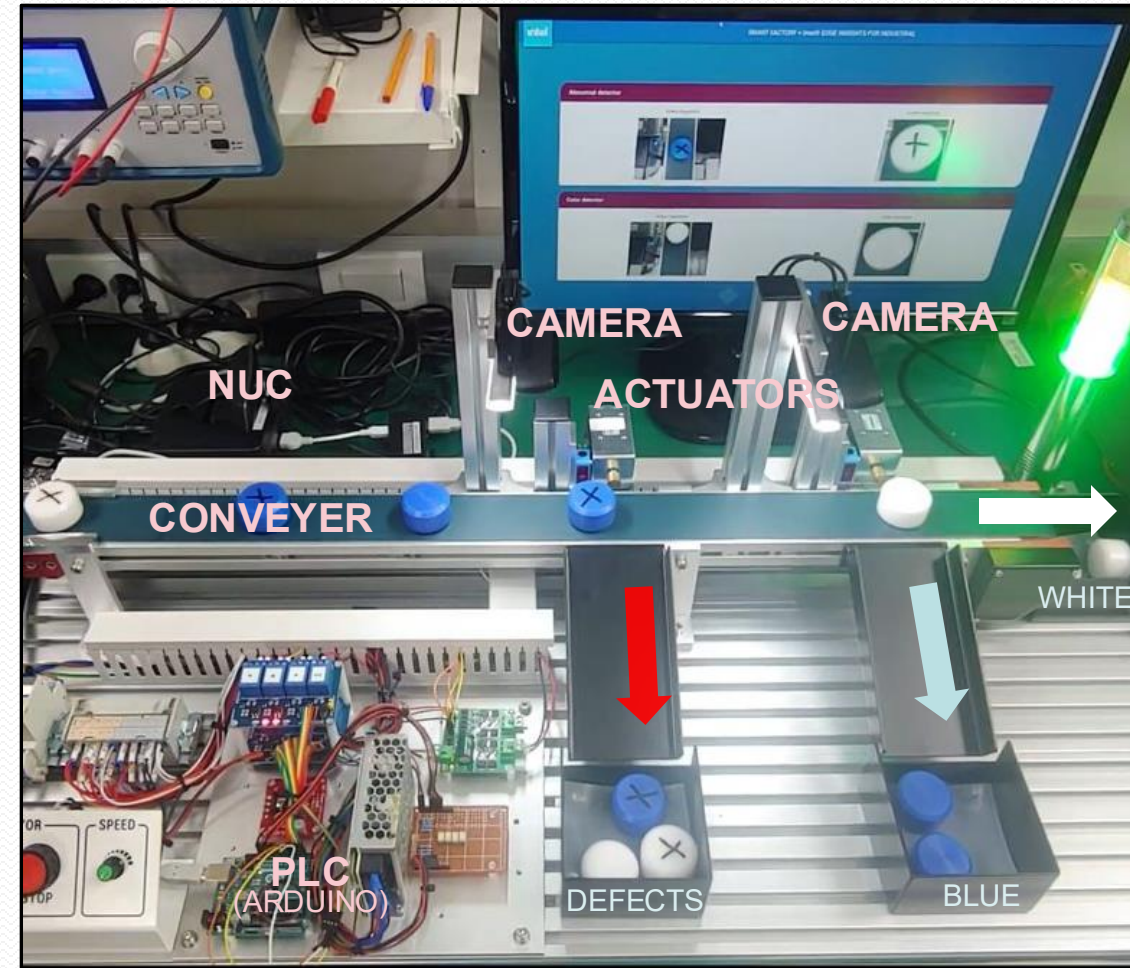
CAMERA

LOGITECH HD PRO WEBCAM C920N

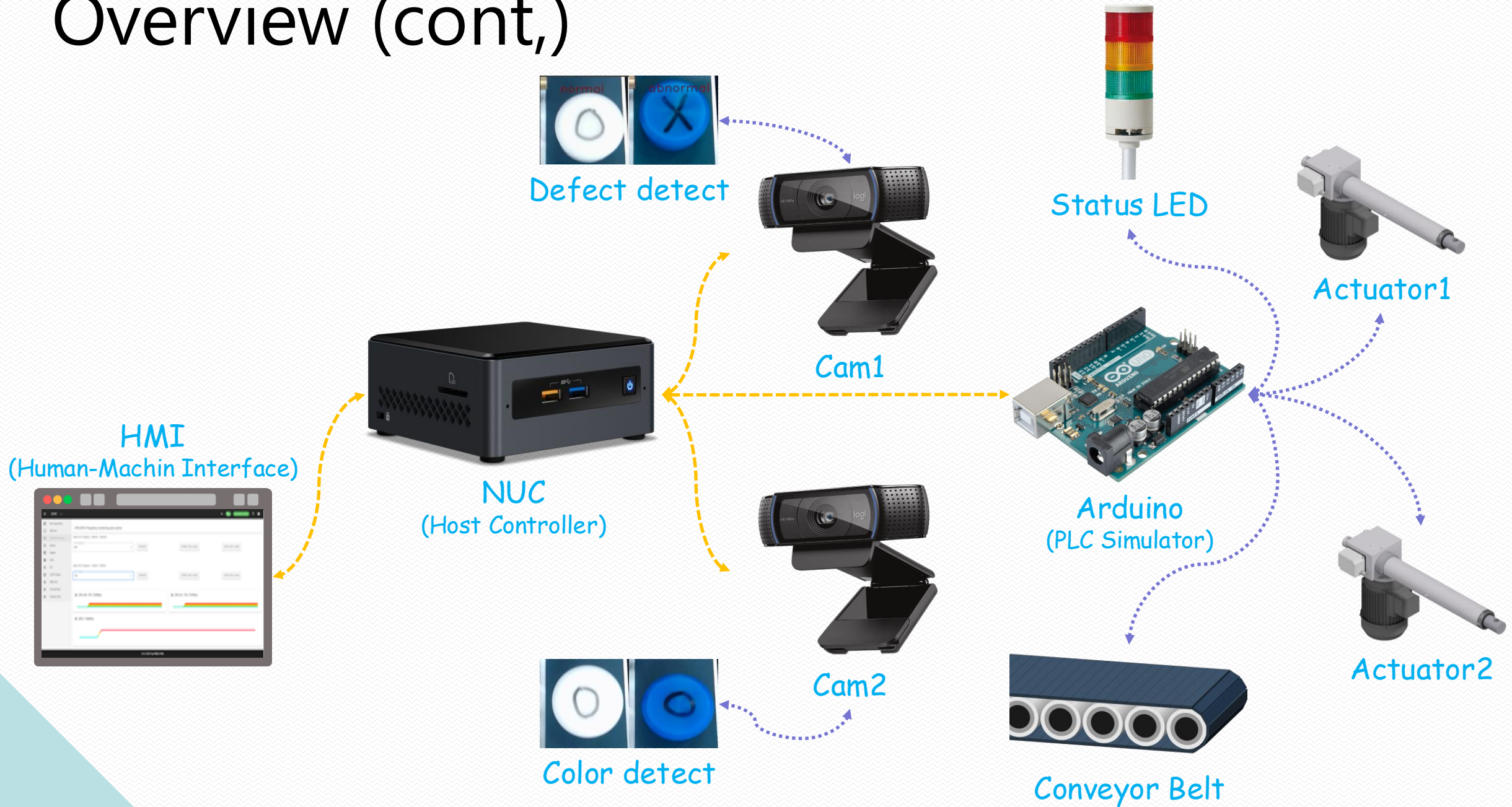
640x480 30 fps / up to 1920x1080 30 fps

ARDUINO

UNO



Overview (cont,)

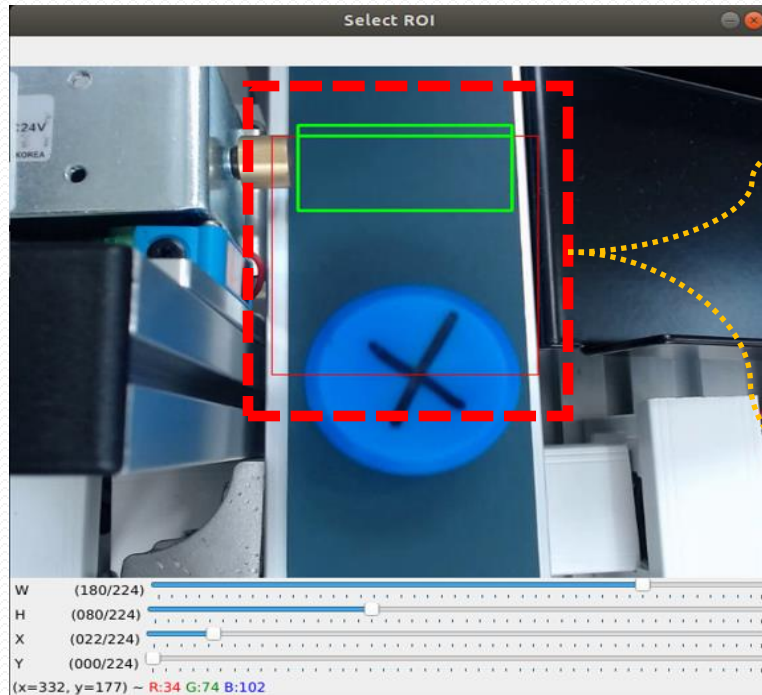


Tool Hands-on

Motion Detect

CALIBRATE CROP AND ROI BOX POSITION

Implement the codes for setting the ROI, motion detection and crop/save the images



PRE-IMPLEMENTED TOOL

```
"crop_box": [[262  83], [486 307]],  
"roi_box":  [[284  73], [464 153]],
```

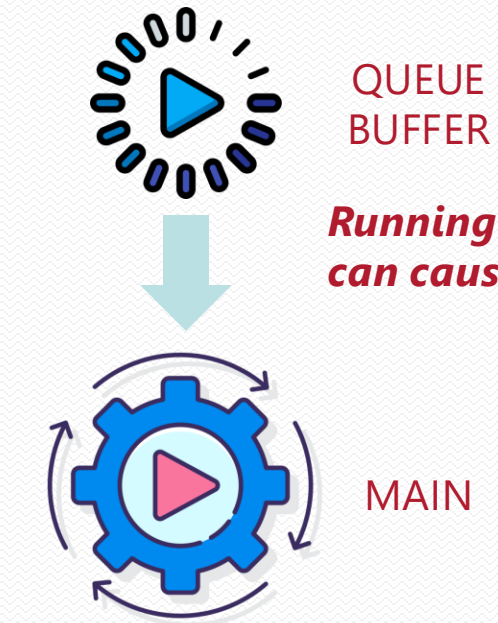
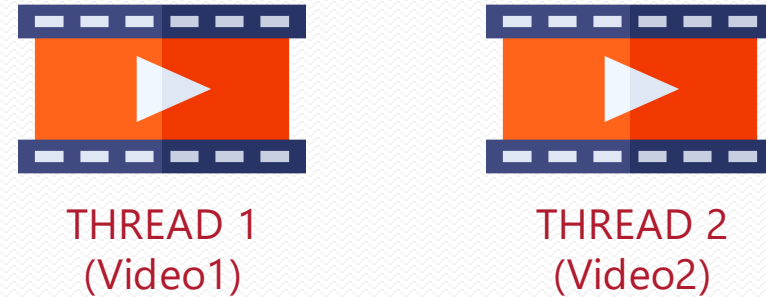
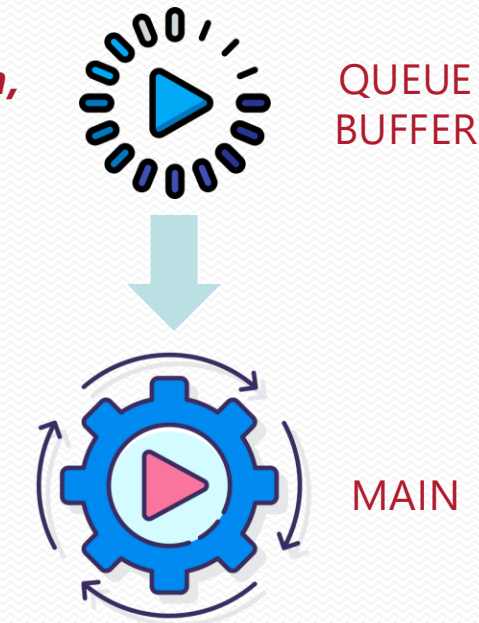


CROPPED IMAGE

Multi-Threading with Queue

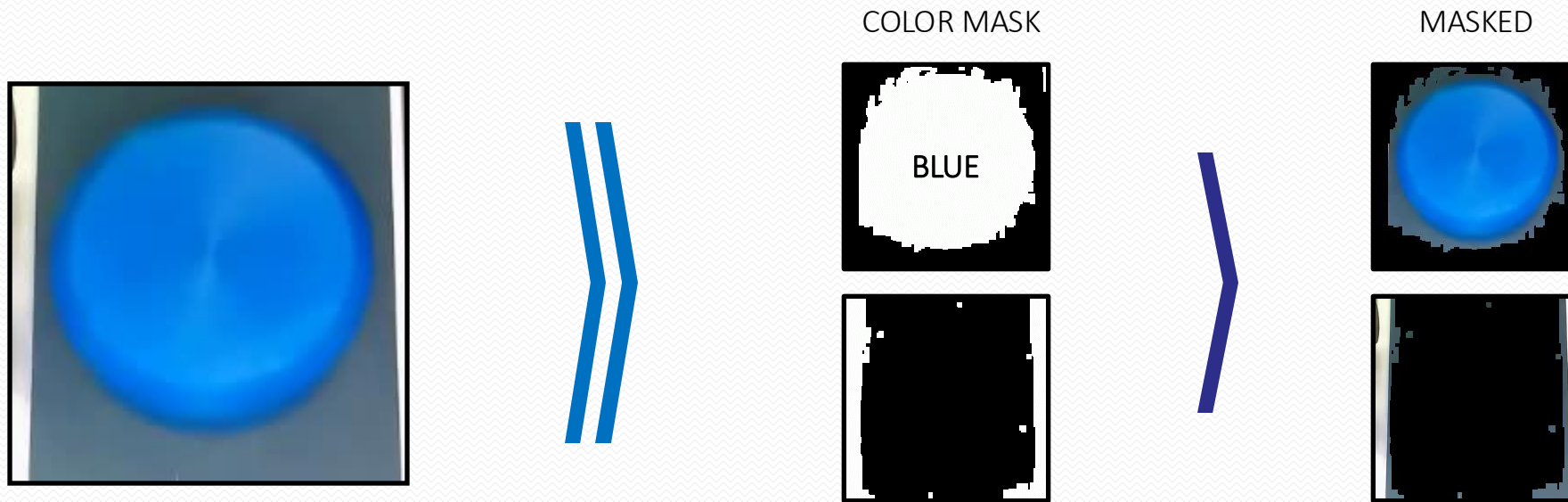


*Queue the camera stream,
receive it from the main
and process it*



*Running 'imshow' on each thread
can cause crash issues!*

Color Detect



APPLY COLOR MASKS TO THE FRAMES

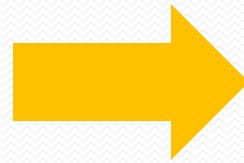
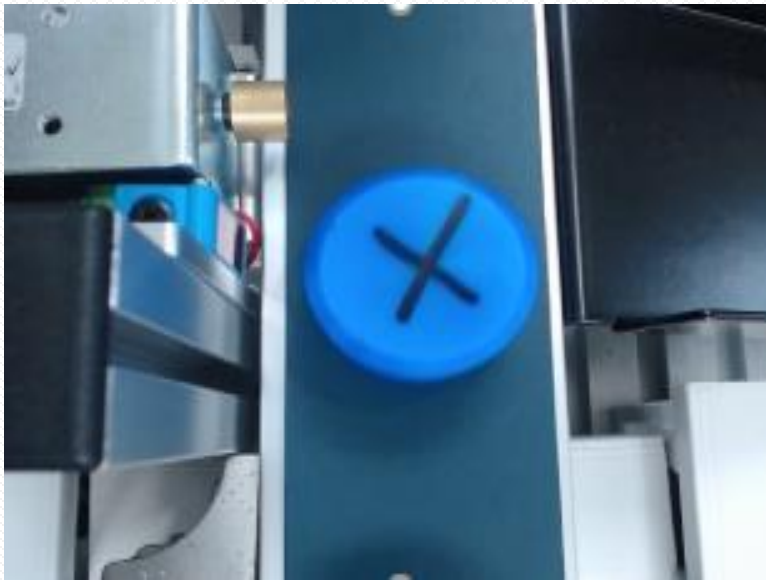
Using the HSV color model to specify the color position and color "purity"

COUNT THE MASKED PIXELS AND PREDICT COLORS

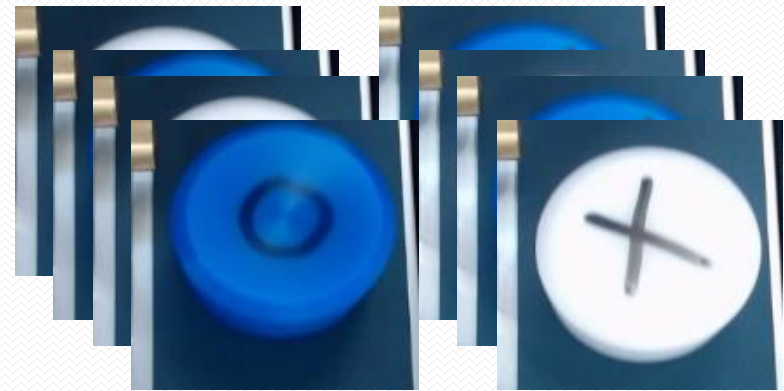
AI Hands-on

Data Preparation

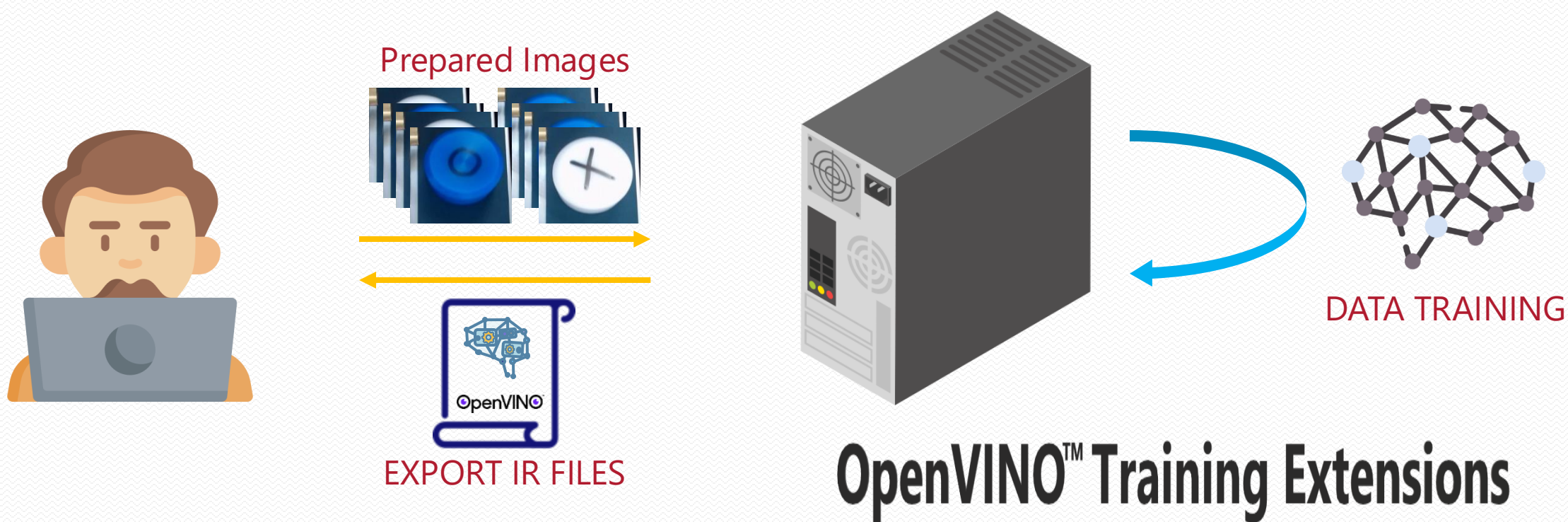
Motion Detect



saved



Training



Why Arduino?

Smart Factory Overview

NUC

NUC10i7FN

10th Generation Intel® Core™ i7-10710U

1.1 GHz – 4.7 GHz Turbo, 6 core, 12 thread, 12MB Cache

25W Intel® UHD Graphics, 300 MHz – 1.15 GHz

16GB RAM / 512GB SSD

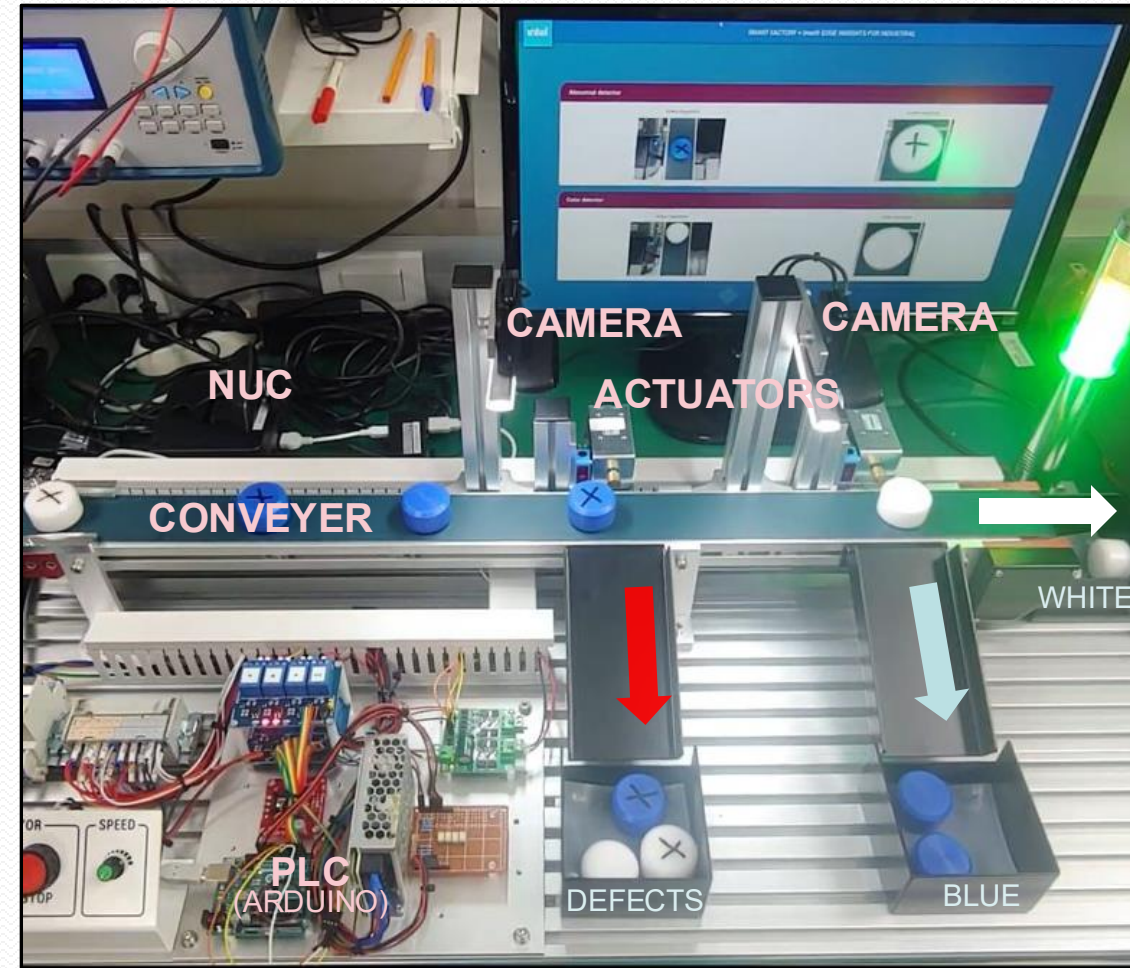
CAMERA

LOGITECH HD PRO WEBCAM C920N

640x480 30 fps / up to 1920x1080 30 fps

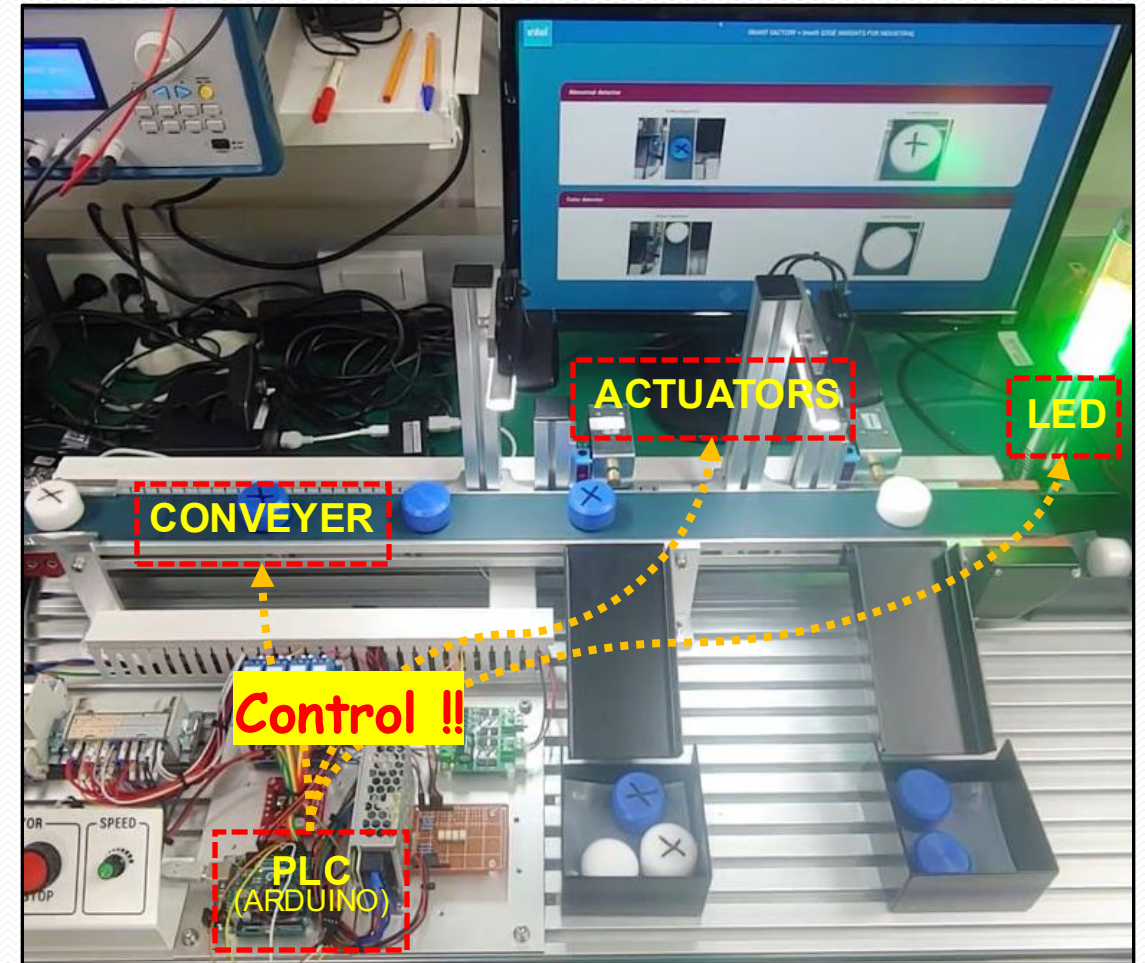
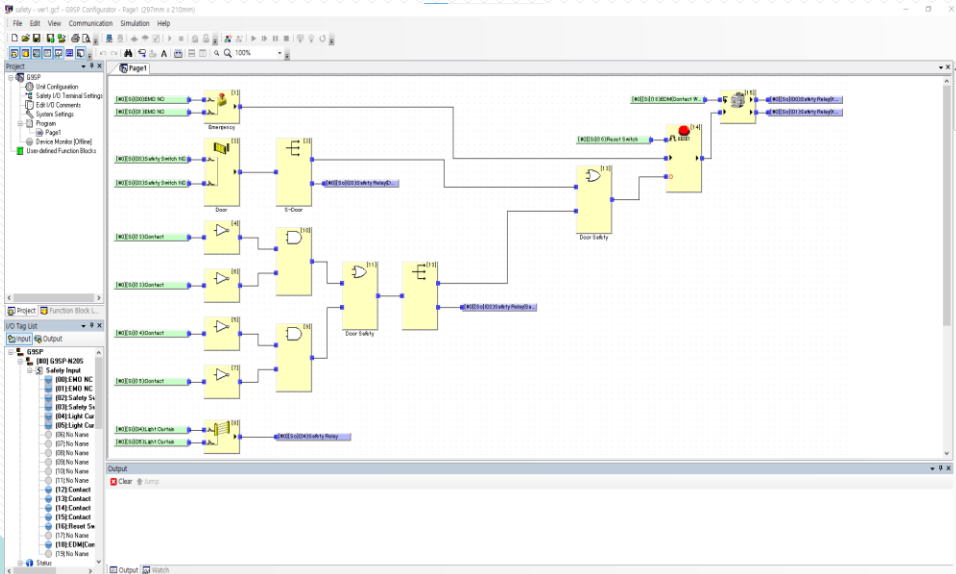
ARDUINO

UNO



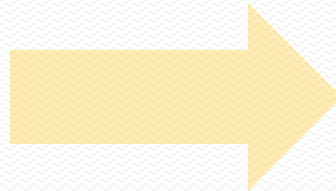
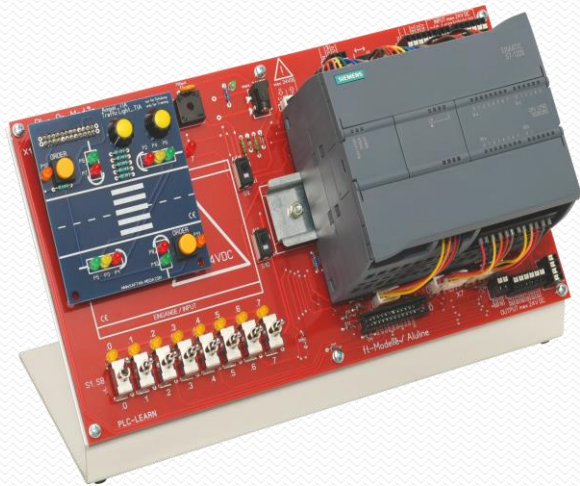
PLC

- Programmable Logic Controller
- Programming using Ladder Diagram



Arduino

- PLC Simulator



- HW ON/OFF Control
- *Using Ladder Diagram*
- *Communication with Host PC*

- GPIO Control only
 - *Conveyor Belt Control (PWM)*
 - *LED Control*
 - *Actuator Control*
- *Using C language*
- *Communication with Host PC also*



Arduino Environment Overview

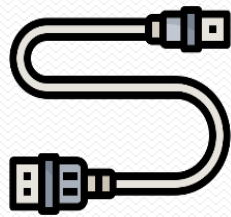
Where is to use the Arduino?

CONTROL the HW through ARDUINO / UART

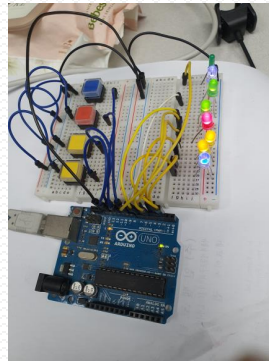
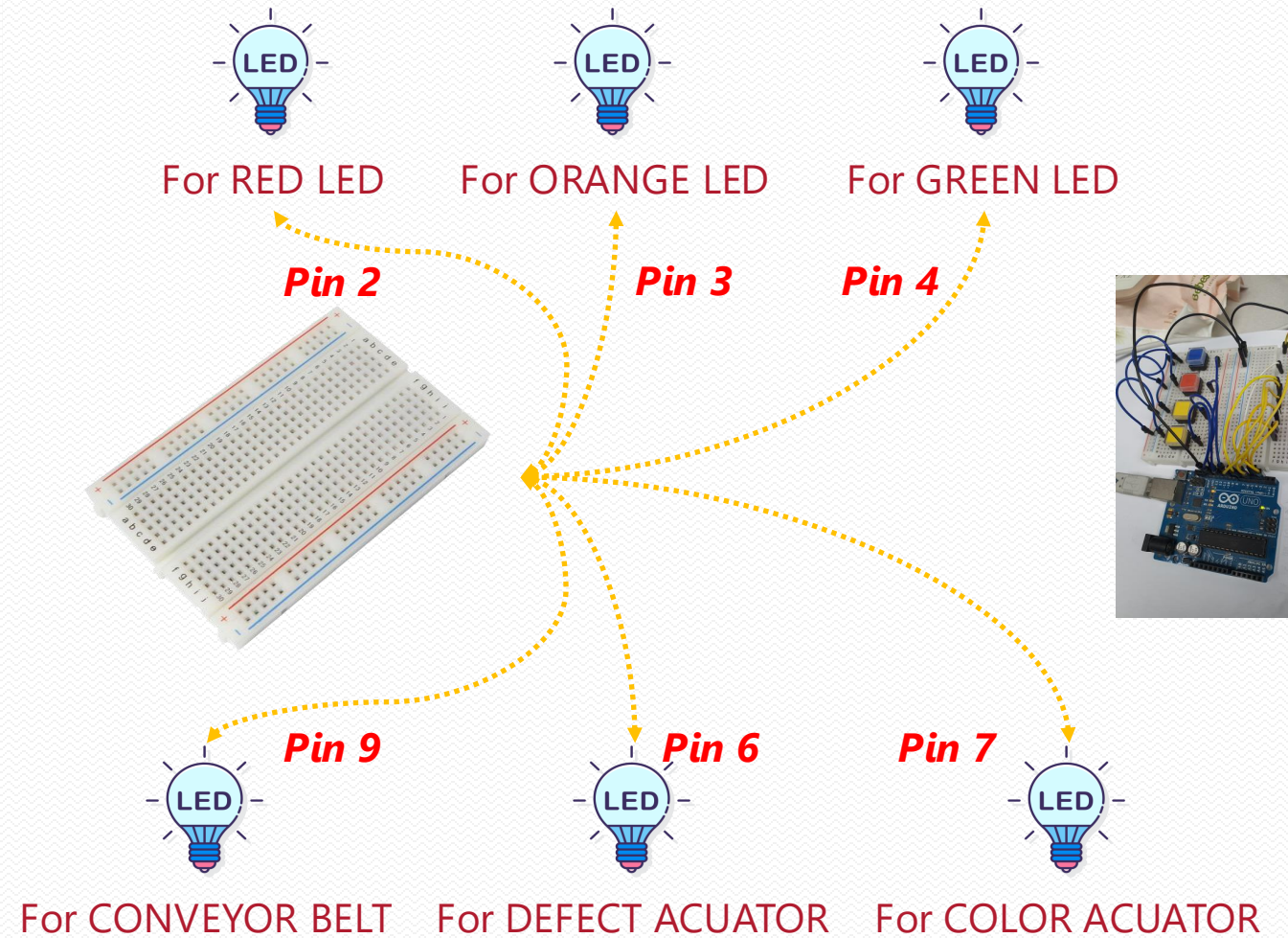
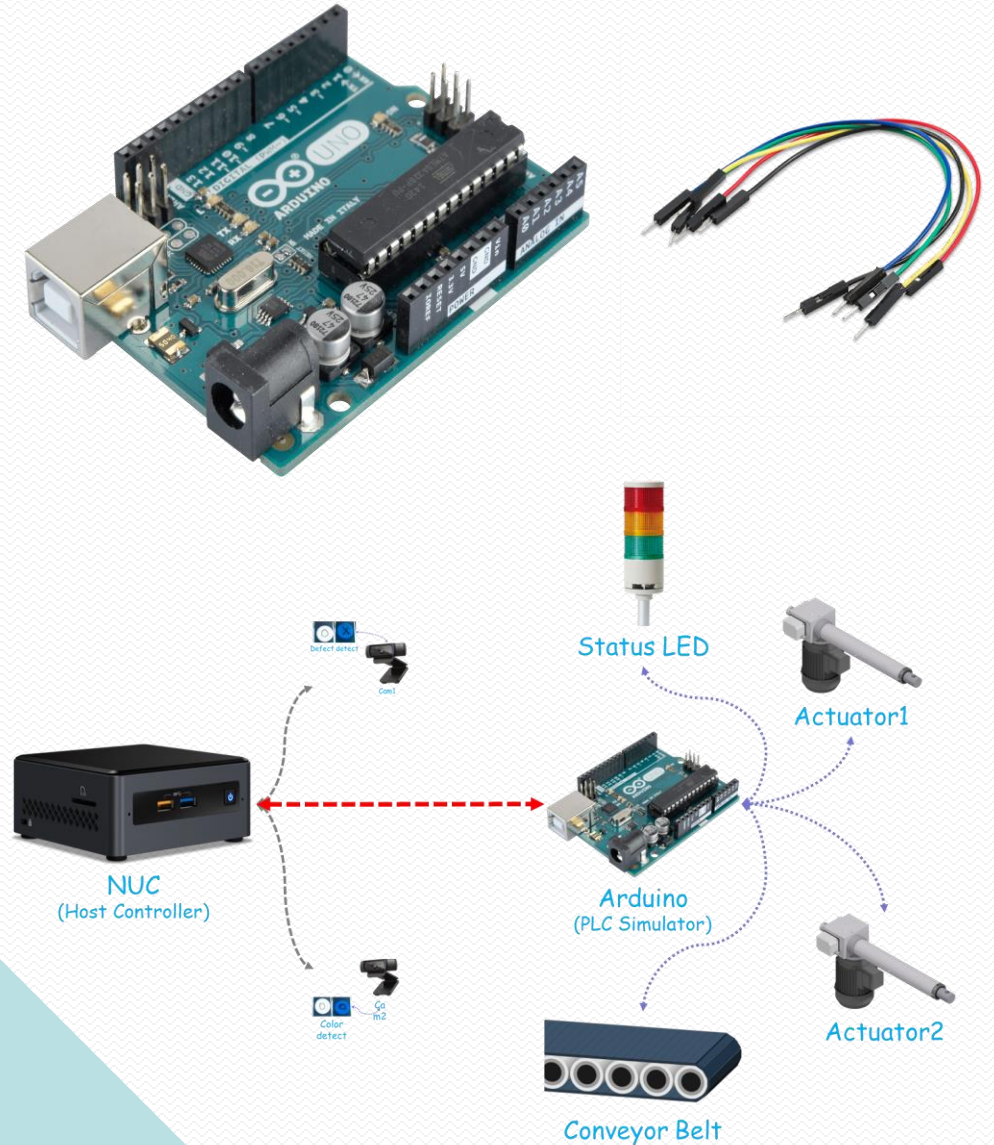
Conveyer Belt On/Off

Actuator Motor Push/Release

LED(Red/Orange/Green) Control



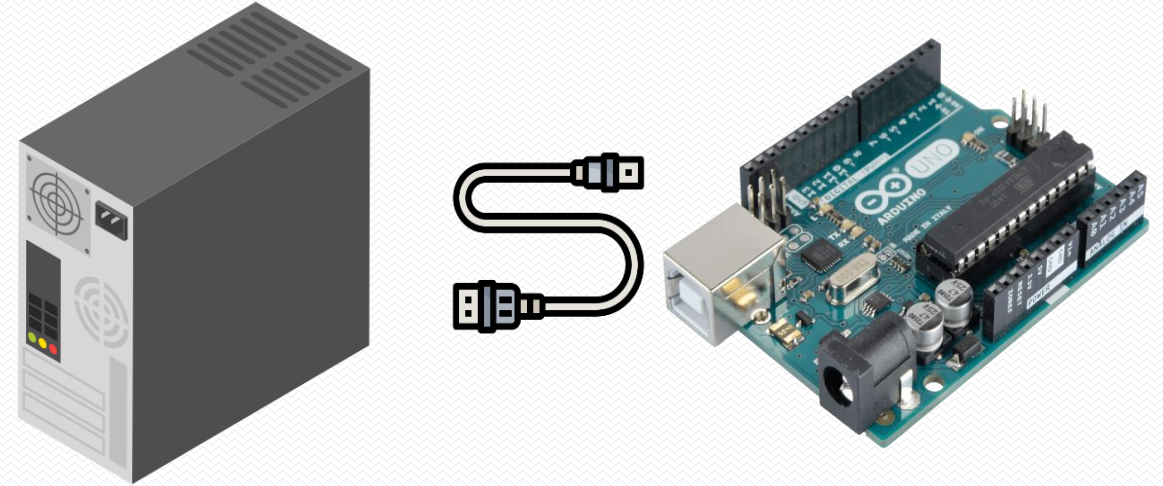
Arduino Setup for Simulation



Arduino Directory Analyze

Setup the Permission

- In arduino directory,
`$ cd <root dir>/arduino`
- Connect the Arduino with the PC
- Check the permission
`$ ls -al /dev/ttyACM0`
- Change the permission(Temporarily)
`$ sudo chmod a+rw /dev/ttyACM0`
- Re-check the permission
`$ ls -al /dev/ttyACM0`



- Add the user to group
`$ sudo usermod -aG dialout $USER`

Arduino Connection & Flash

- Download Arduino CLI Application

```
$ curl -fsSL https://raw.githubusercontent.com/arduino/arduino-cli/master/install.sh | sh  
$ cd bin && ls -al
```

- Setup the Environment

```
$ arduino-cli config init --overwrite  
$ arduino-cli core install arduino:avr
```




- Build and Flash the Arduino

```
$ arduino-cli compile -b arduino:avr:uno . -e -v  
$ arduino-cli upload -p /dev/ttyACM0 -b arduino:avr:uno -t -v -i  
build/arduino.avr.uno/arduino.ino.hex
```

Make

- Make
 - A build automation tool that builds executable programs and libraries from source code by reading files called makefiles which specify how to derive the target program.



- In Makefile
 - make init  Setup the basic environment using *arduino-cli*
 - make build  Source code build(*arduino.ino*)
 - make flash  Flash to the Real HW Arduino
(*build/arduino.avr.uno/arduino.ino.hex*)

Arduino Control

Arduino Control using Python

- Change directory for Python API

```
$ cd <root dir>
```

```
$ cd iotdemo/factory_controller
```

```
$ ls -al
```

- factory_controller.py
 - Refer to the Public Properties/methods

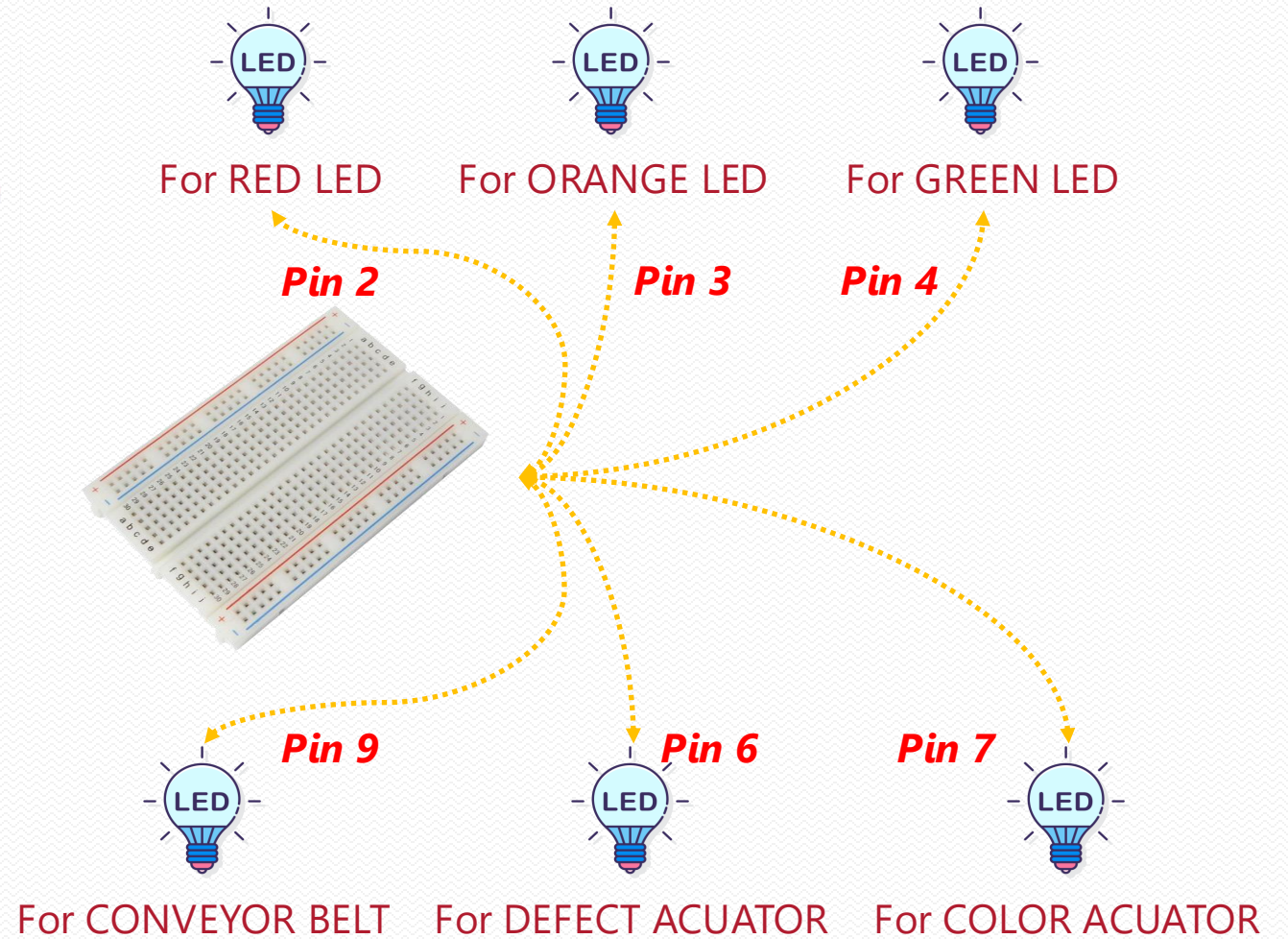
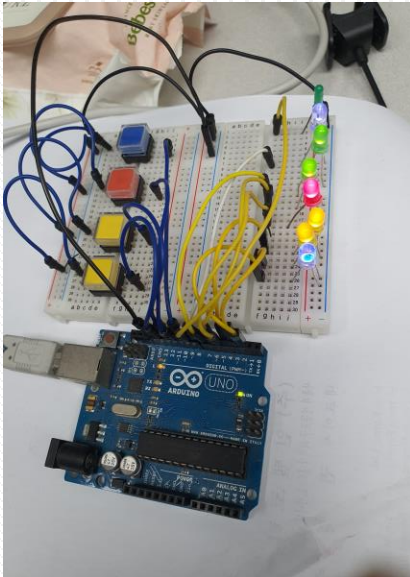
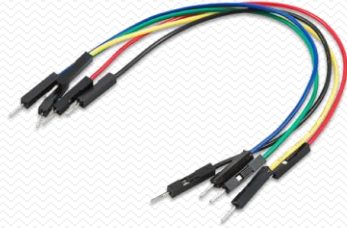
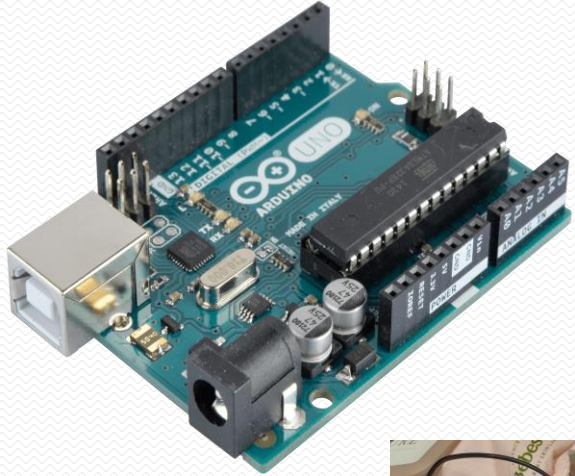
Hands-on

- Change the arduino directory and delete “*arduino-cli*”

```
$ cd <root_dir>  
$ cd arduino  
$ rm ./bin/*
```

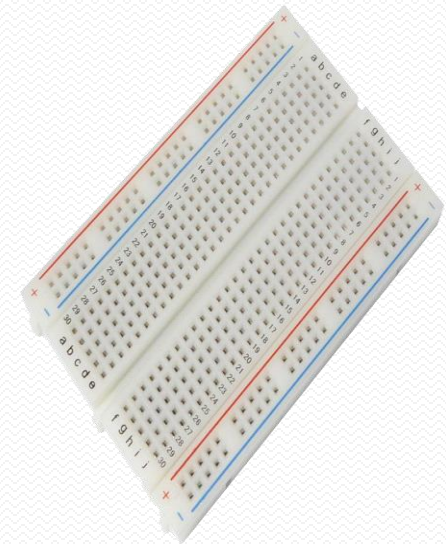
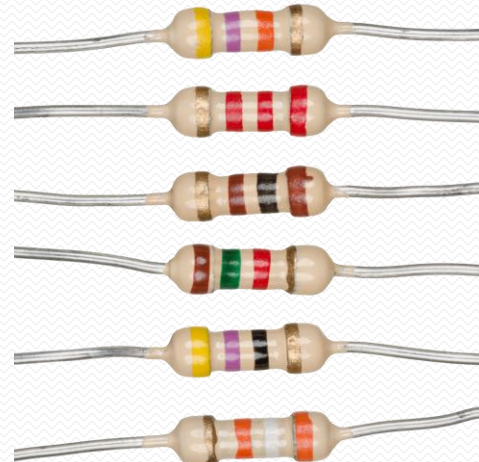
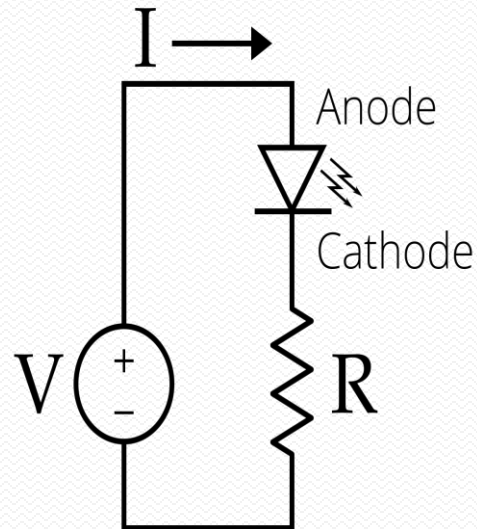
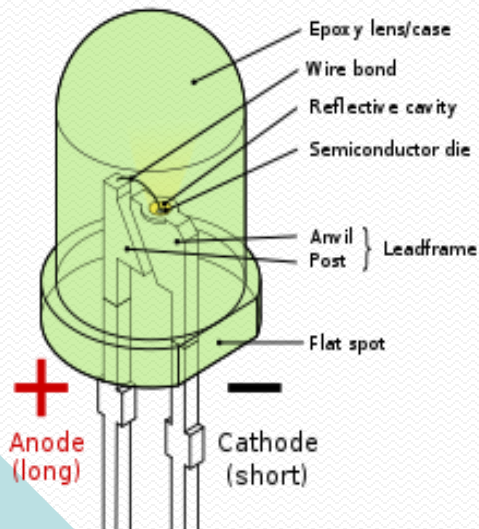
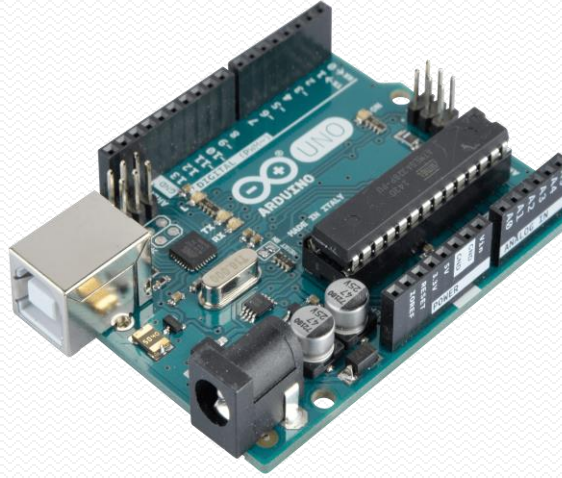
- Test what happens when executing the command below.
 - make init
 - make build
 - make flash
 - make clean

Hands-on with Real HW



Hands-on with Real HW (cont,)

- How to connect wires, LEDs, resistors with Arduino?



Arduino Control using Python (cont,)

PYTHON MODULE APIs

(from iotdemo import FactoryController)

Init the system

- with *FactoryController('/dev/ttyACM0')* as *ctrl*:
- or *ctrl = FactoryController('/dev/ttyACM0')*

Close the system

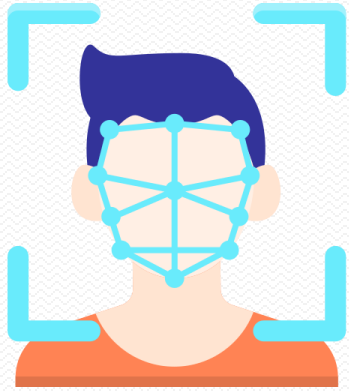
- *ctrl.close()*

For Micro Control the HW Module

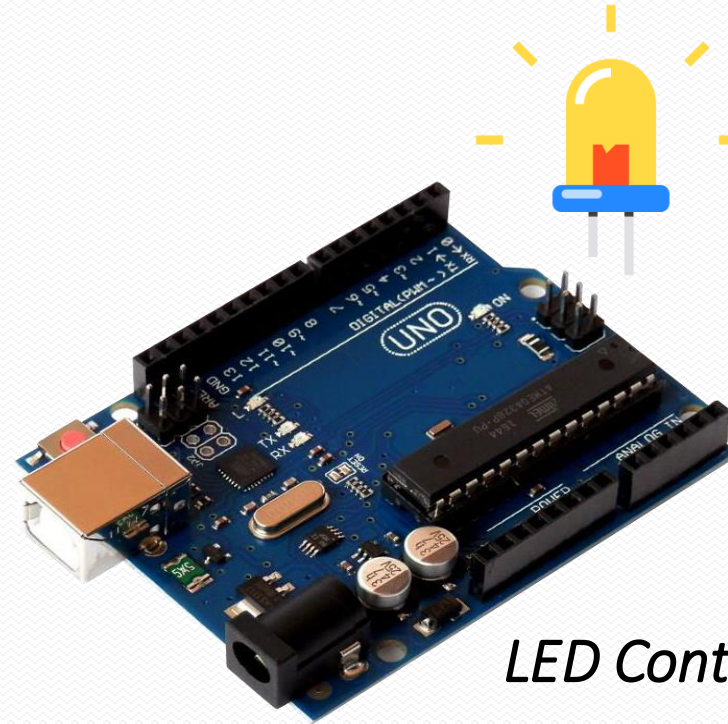
Input	API
1	system_start()
2	system_stop()
3	red
4	orange
5	green
6	conveyor
7	push_actuator(1)
8	push_actuator(2)

Basic Hands-on

HW Control



Inferencing Result



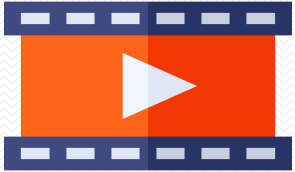
LED Control with Arduino

Utilize Arduino LED with Inferencing Result

Demo Implementation

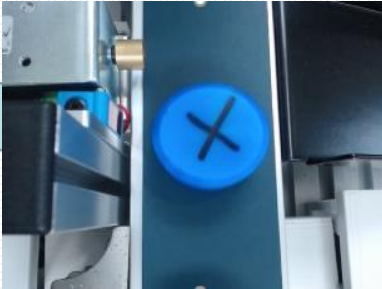
Demo on Localhost

VIDEO INPUT

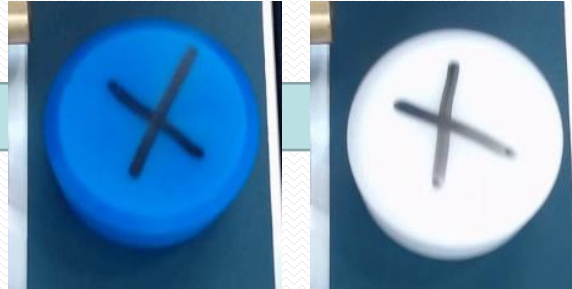


[./resources/factory/conveyor.mp4](#)

MOTION DETECT



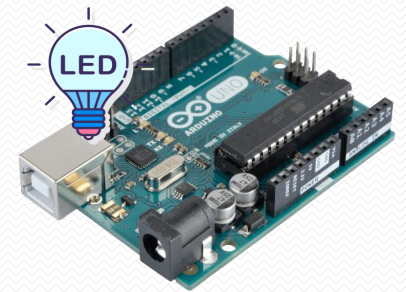
COLOR DETECT



DEFECT DETECT



ARDUINO SIMULATION



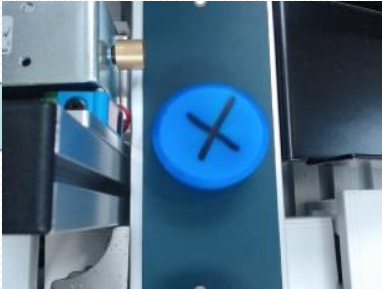
Let's try Real HW World!

CAMERA INPUT

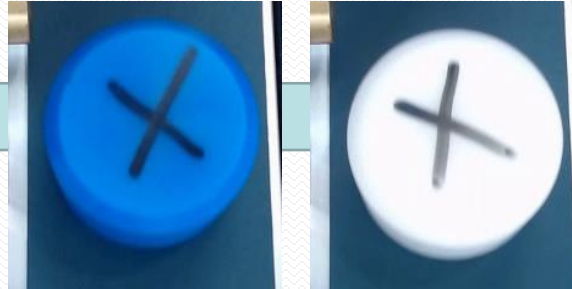


*/dev/video**

MOTION DETECT



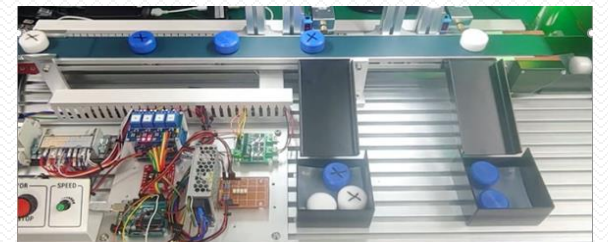
COLOR DETECT



DEFECT DETECT



SMART FACTORY

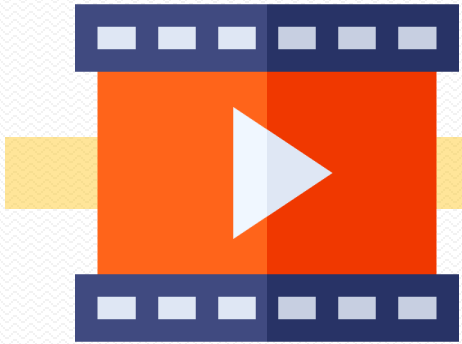




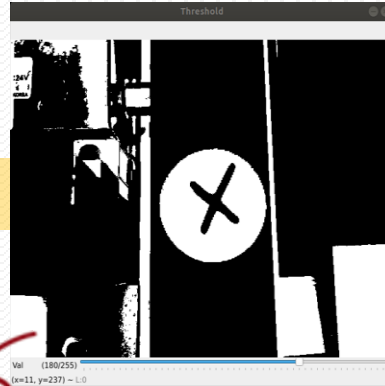
How to use tool?

Motion Detect

Camera Input



Binarization

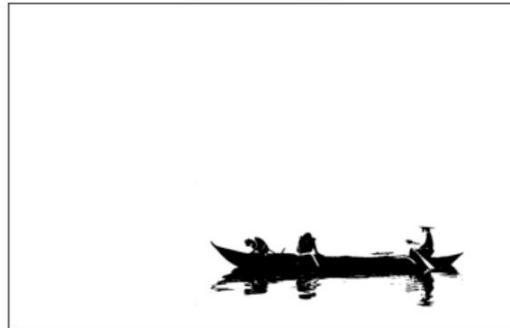


To drastically reduce the amount of information you have to work with

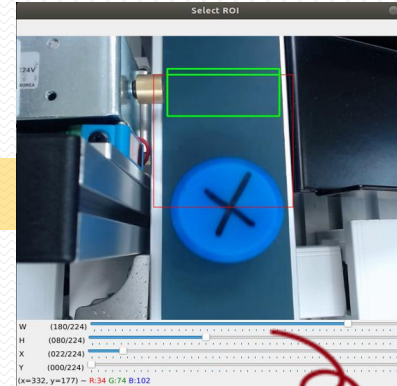
original image



filtered image



Region of Interesting
ROI Setting



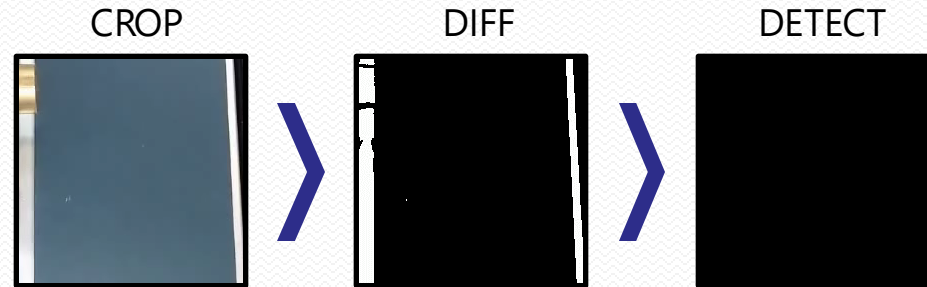
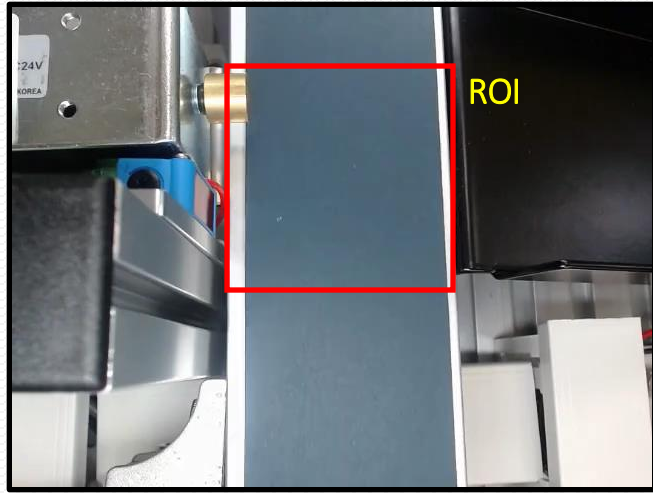
Video Stream
with 640x480 resolution

Motion Detect



224x224 image

Detailed Flow

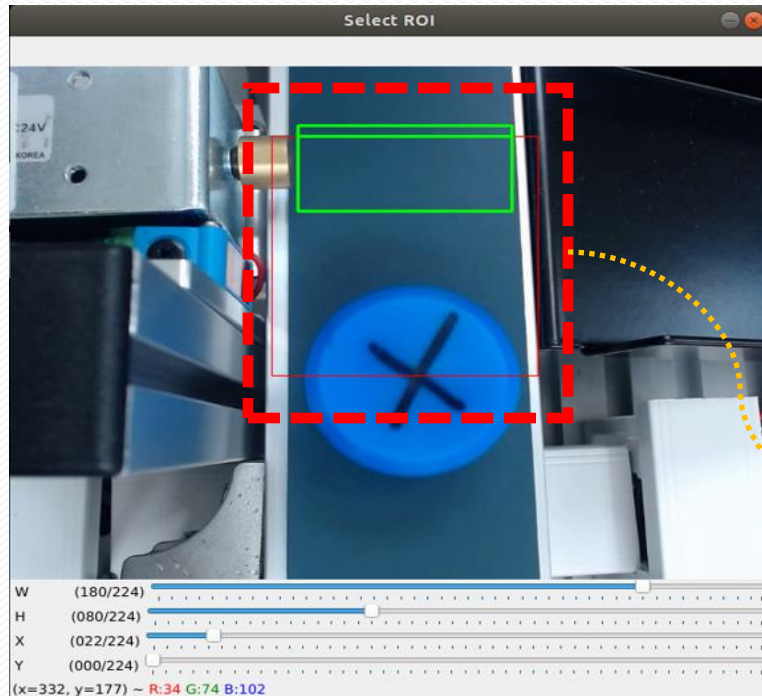


- *Crop the frame per selected ROI*
- *Calculate the frame difference*
 - Apply custom threshold with brightness value
- *Choose the best frames to pass*

Select ROI to Detect Motion

Python Tool (*iotdemo-motion-detector*)

```
iotdemo-motion-detector -l  
./resource/factory/conveyor.mp4
```



Pre-Implemented Tool

Python Main (*factory.py*)

Detect the Frame (each Thread)

```
- detected = det.detect(frame)
```

```
if detected is None:  
    continue
```

Enqueue (each Thread)

```
- q.put(('VIDEO: Cam1 detected', detected))
```

```
- q.put(('VIDEO: Cam2 detected', detected))
```

```
[default]  
inverted = False  
flipped = False  
top_margin = 10  
threshold = 127  
top_ratio = 0.6  
mid_ratio = 0.4  
skip_time = 0.099  
roi_top_left = (12, 0)  
roi_bottom_right = (212, 90)  
crop_top_left = (0, 10)  
crop_bottom_right = (224, 234)
```

motion.cfg



Cropped Image

Homework Submission

- There'll be several homework
- Submission
 - Fork the class GitHub <https://github.com/kccistc/smart-factory/>
 - Create your homework submission PR under **Class02/homework/<your_name>/hw##**

```
./Class02
├── homework
│   └── <your_name>
│       ├── hw1
│       ├── hw2
│       └── hw3
```

- Deadline is by next day morning



THANK YOU

Back Up

Prerequisite

Terminal Command

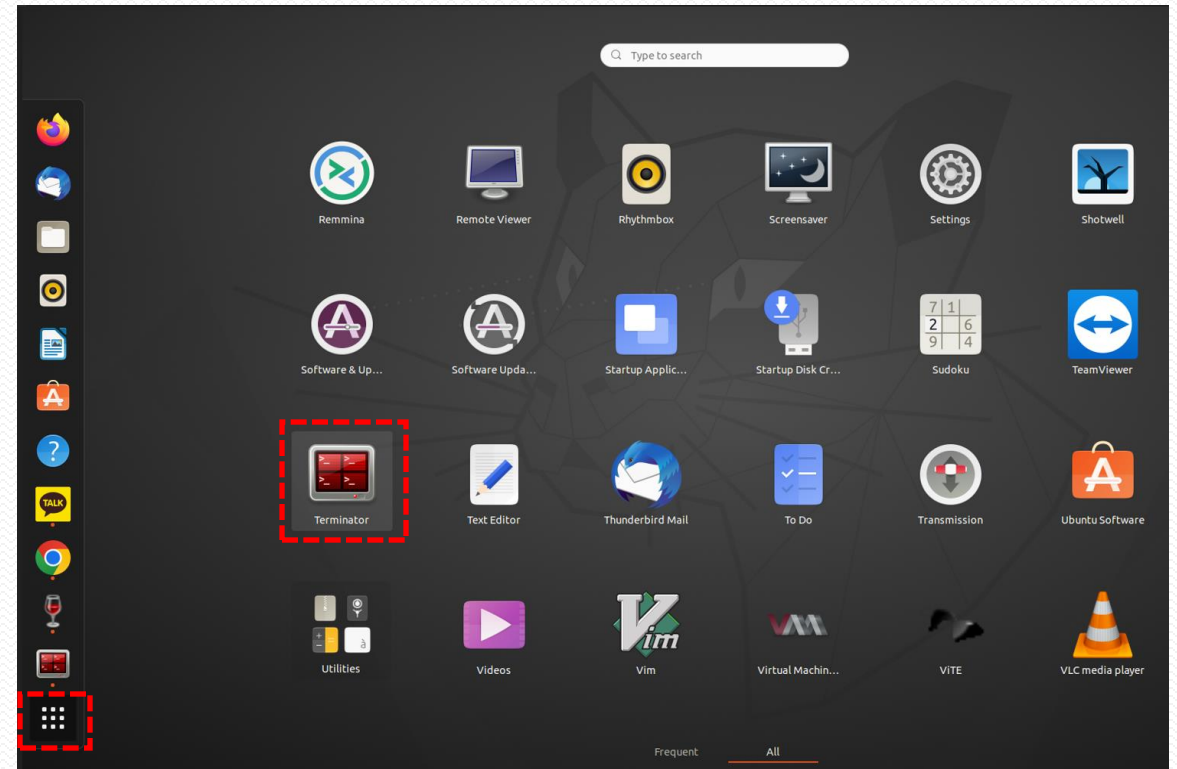
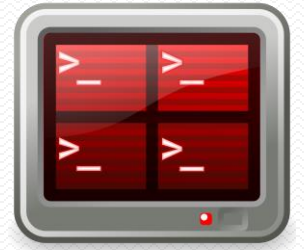
- We use Terminator(Terminal) and Visual Studio Code(VSC)
 - Next page, explain how to install the Terminator and VSC
- Command on Terminator
 - `$ ls -al`
- `<root dir>`
 - It means the root directory from the Github code

`~/code_test/temp/demo.smart-factory-legacy`

```
jerrylee@jaeseong-mobl2:~/code_test/temp/demo.smart-factory-legacy$  
jerrylee@jaeseong-mobl2:~/code_test/temp/demo.smart-factory-legacy$ ls -al  
total 72  
drwxrwxr-x 9 jerrylee jerrylee 4096 7월 27 16:47 .  
drwxrwxr-x 9 jerrylee jerrylee 4096 7월 27 13:30 ..  
drwxrwxr-x 4 jerrylee jerrylee 4096 7월 25 15:05 arduino  
drwxrwxr-x 4 jerrylee jerrylee 4096 7월 25 14:55 build  
-rwxrwxr-x 1 jerrylee jerrylee 175 7월 25 15:01 build.sh  
-rwxrwxr-x 1 jerrylee jerrylee 638 7월 27 16:56 cam.sh  
drwxrwxr-x 2 jerrylee jerrylee 4096 7월 25 14:55 dist  
-rwxrwxr-x 1 jerrylee jerrylee 3993 7월 25 14:55 factory.py  
drwxrwxr-x 9 jerrylee jerrylee 4096 7월 25 14:55 iotdemo  
drwxrwxr-x 2 jerrylee jerrylee 4096 7월 25 14:55 iotdemo.egg-info  
-rw-rw-r-- 1 jerrylee jerrylee 1078 7월 25 14:55 LICENSE  
-rw-rw-r-- 1 jerrylee jerrylee 37 7월 25 14:55 MANIFEST.in  
-rw-rw-r-- 1 jerrylee jerrylee 482 7월 27 14:47 motion.cfg  
-rw-rw-r-- 1 jerrylee jerrylee 192 7월 25 14:55 README.md  
-rw-rw-r-- 1 jerrylee jerrylee 94 7월 25 15:01 requirements.txt  
drwxrwxr-x 3 jerrylee jerrylee 4096 7월 25 14:59 resources  
drwxrwxr-x 2 jerrylee jerrylee 4096 7월 25 15:07 saved  
-rw-rw-r-- 1 jerrylee jerrylee 962 7월 25 14:55 setup.py  
jerrylee@jaeseong-mobl2:~/code_test/temp/demo.smart-factory-legacy$
```


Terminator

- Install the Terminator
`$ sudo apt install terminator`
- Execute the Terminator
 - Find in the all application on Ubuntu and Click the Terminator icon
 - On the Desktop, input the key below
“ctrl + alt + t”



Visual Studio Code(VSC)



- Pre-packages Install

```
$ sudo apt update
```

```
$ sudo apt install software-properties-common apt-transport-https wget
```

- Get Microsoft GPG key

```
$ wget -q https://packages.microsoft.com/keys/microsoft.asc -O- | sudo apt-key add -
```

- Add the apt repository for VSC

```
$ sudo add-apt-repository "deb [arch=amd64] https://packages.microsoft.com/repos/vscode stable main"
```

- VSC install

```
$ sudo apt install code
```

- Execution the VSC

```
$ code
```

Arduino Basic

What is the Arduino

- A microcontroller board, contains the onboard power supply, USB port to communicate with PC, and an Atmel microcontroller chip
- It simplifies the process of creating any control system by providing the standard board that can be programmed and connected to the system without the need for any sophisticated PCB design and implementation
- It is open-source hardware, anyone can get the details of its design and modify it or make his own one himself

Why Arduino?

- It is Open Source, both in terms of Hardware and Software
- It is cheap, the hardware can be built from components or a prefab board can be purchased
- It can communicate with a computer via a serial connection over USB
- It can be powered from USB or standalone DC power
- It can run standalone from a computer (chip is programmable) and it has memory (a small amount)
- It can work with both Digital and Analog electronic signals. Sensors and Actuators.
- You can make cool stuff!!
Some people are even making simple robots, and we all know robots are just cool.

What can Arduino do?

- Sensors
 - Push button, touch pads, tilt switches.
 - Variable resistors (e.g. volume knob / sliders)
 - Photoresistors (sensing light level)
 - Thermistors (temperature)
 - Ultrasound (proximity range finders)
- Actuators
 - Lights, LEDs
 - Motors
 - Speakers
 - Display (LCD)

The kinds of the Arduino



UNO R3



MEGA 2560



NANO



DUE



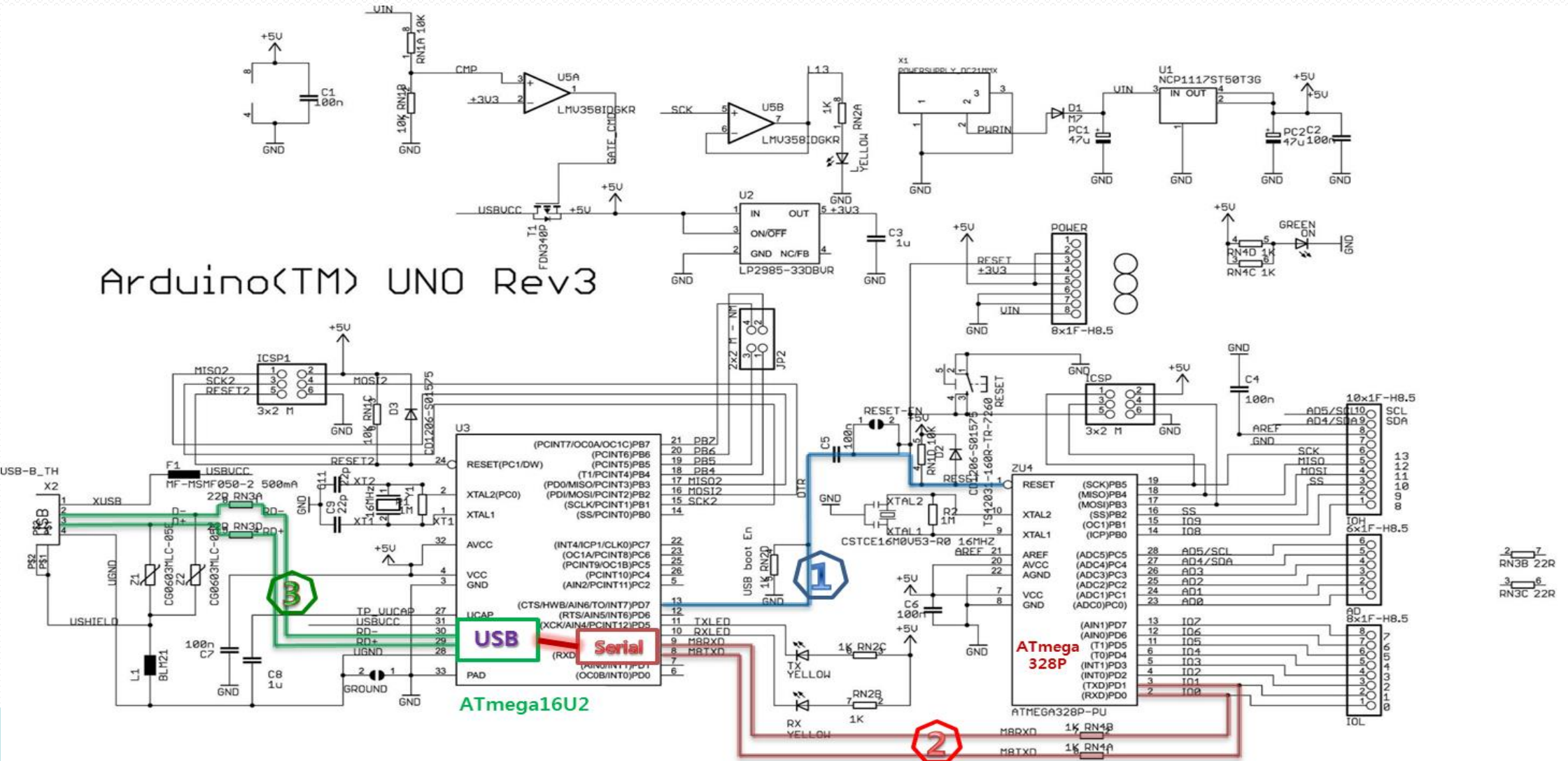
LILYPAD



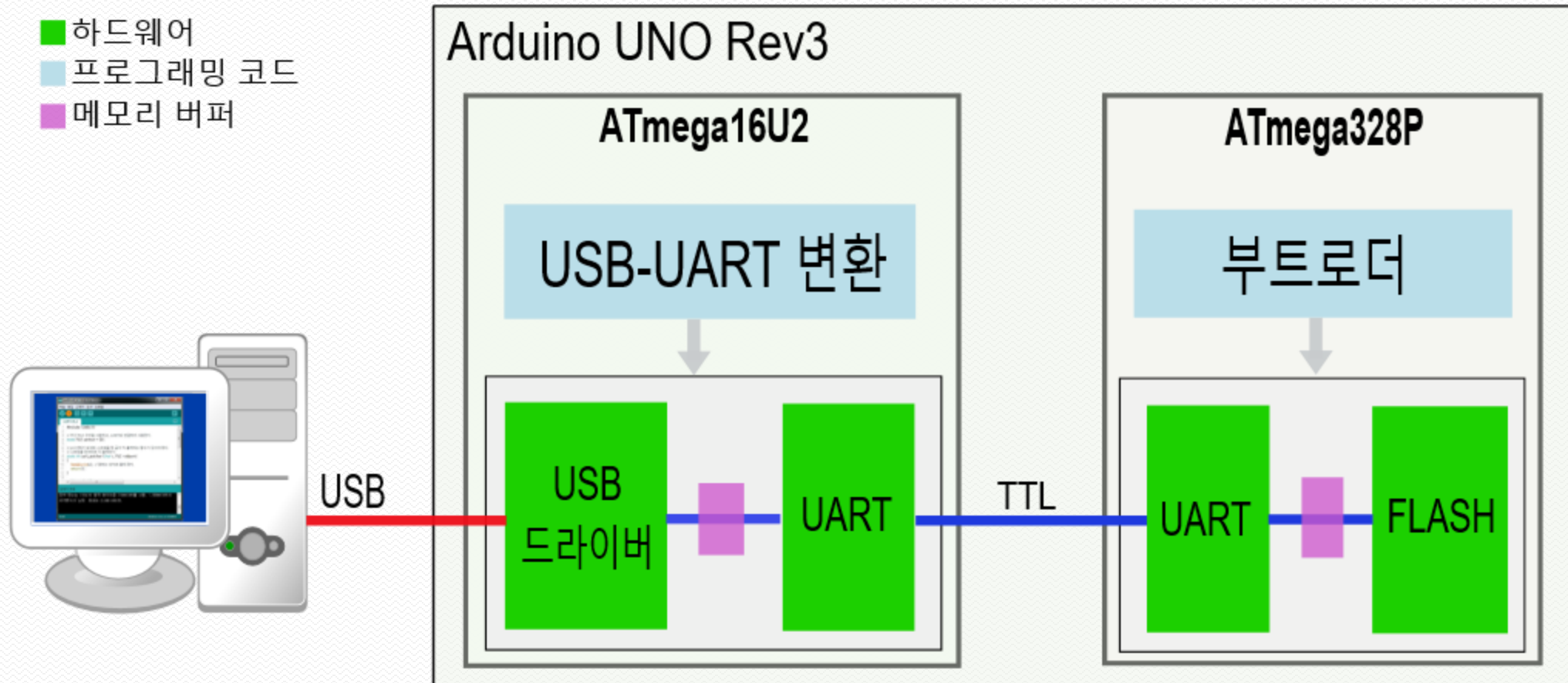
FIO



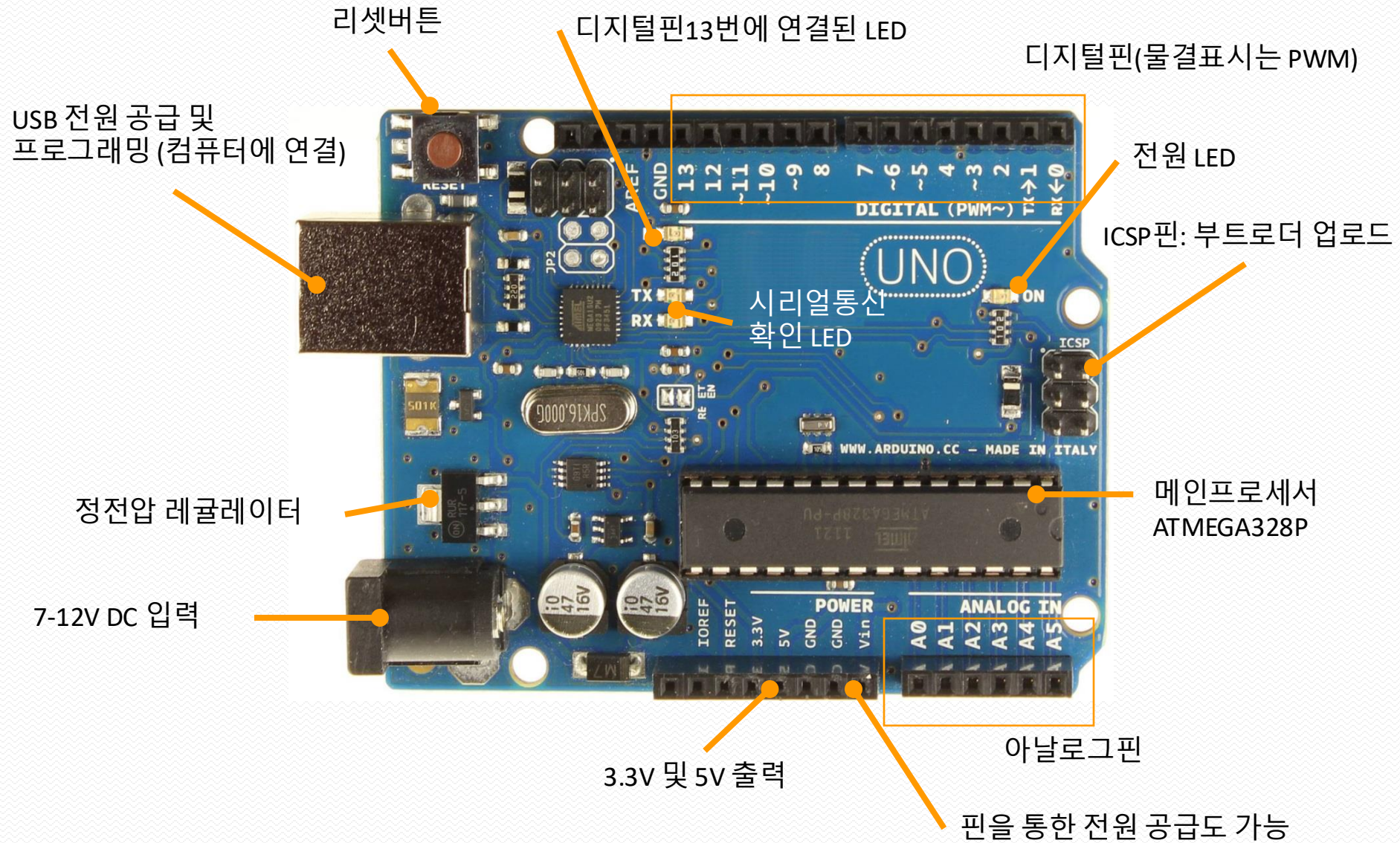
YUN



Environment using Arduino Uno



HW of the Arduino Uno

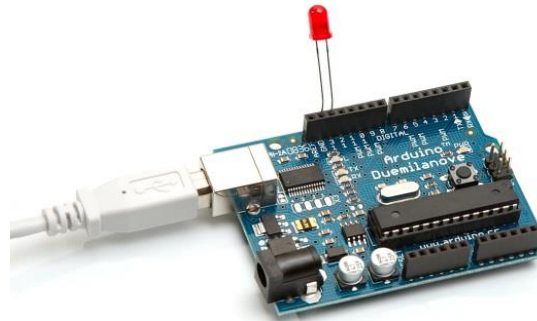


Power for Arduino Uno

1. 배럴잭을 통한 전원 공급 (DC 어댑터를 사용하거나 배터리와 배터리홀더를 사용한 방법)
2. USB 케이블을 통한 전원 공급 (컴퓨터에 USB 케이블을 연결)
3. VIN과 GND를 통한 전원 공급 (배럴잭이 아닌 일반 전선을 사용할 경우)



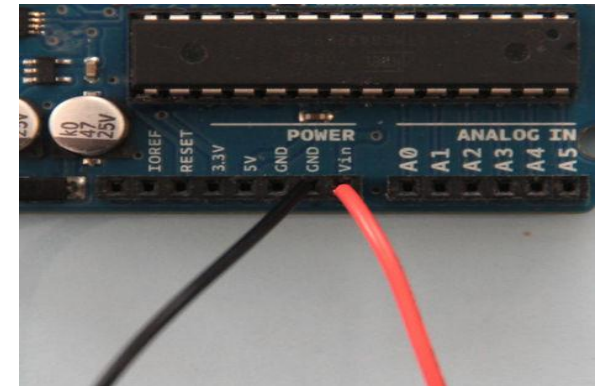
DC 어댑터의 배럴잭을 통한 전원 공급



컴퓨터에 연결된 USB 케이블을 통한 전원 공급



9v 배터리와 배럴잭홀더를 통한 전원 공급



배터리의 전선을 보드의 VIN과 GND에 직접 연결하여 전원 공급

Digital or Analog

- All physical quantities are analog
- Analog means that the quantity can take any value between its minimum value and maximum value
- Digital means that the quantity can take specific levels of values with specific offsets between each other
- Ex - Digital
 - English alpha consists of 26 letters, there is no letter between A and B
- Ex - Analog
 - Temperature can take any value [-1, 12.8, 25.002, ...]
 - Sine waves are analog

Download Arduino IDE

- <https://www.arduino.cc/en/software>

Downloads



Arduino IDE 1.8.19

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the [Getting Started](#) page for Installation instructions.


SOURCE CODE

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

DOWNLOAD OPTIONS

Windows Win 7 and newer

Windows ZIP file

Windows app Win 8.1 or 10 [Get](#) 

Linux 32 bits

Linux 64 bits

Linux ARM 32 bits

Linux ARM 64 bits

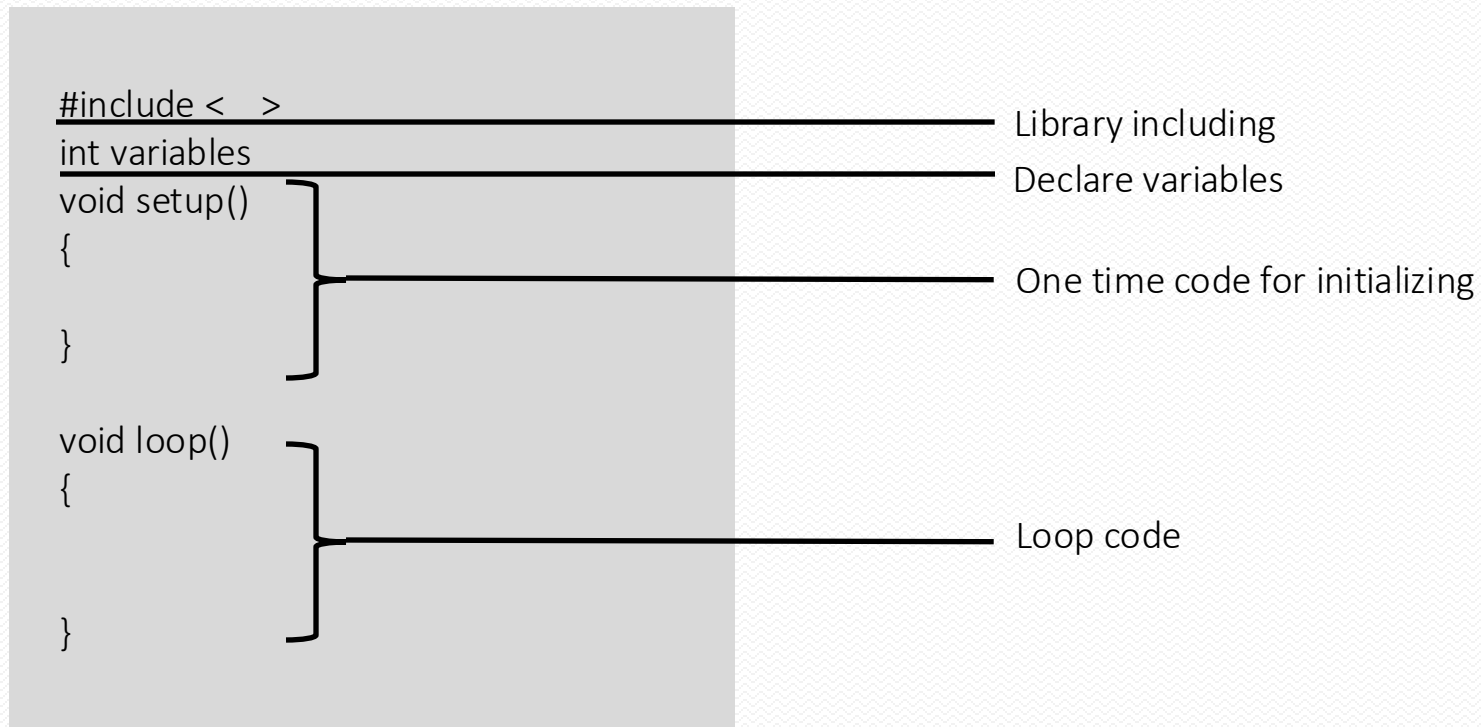
Mac OS X 10.10 or newer

[Release Notes](#)

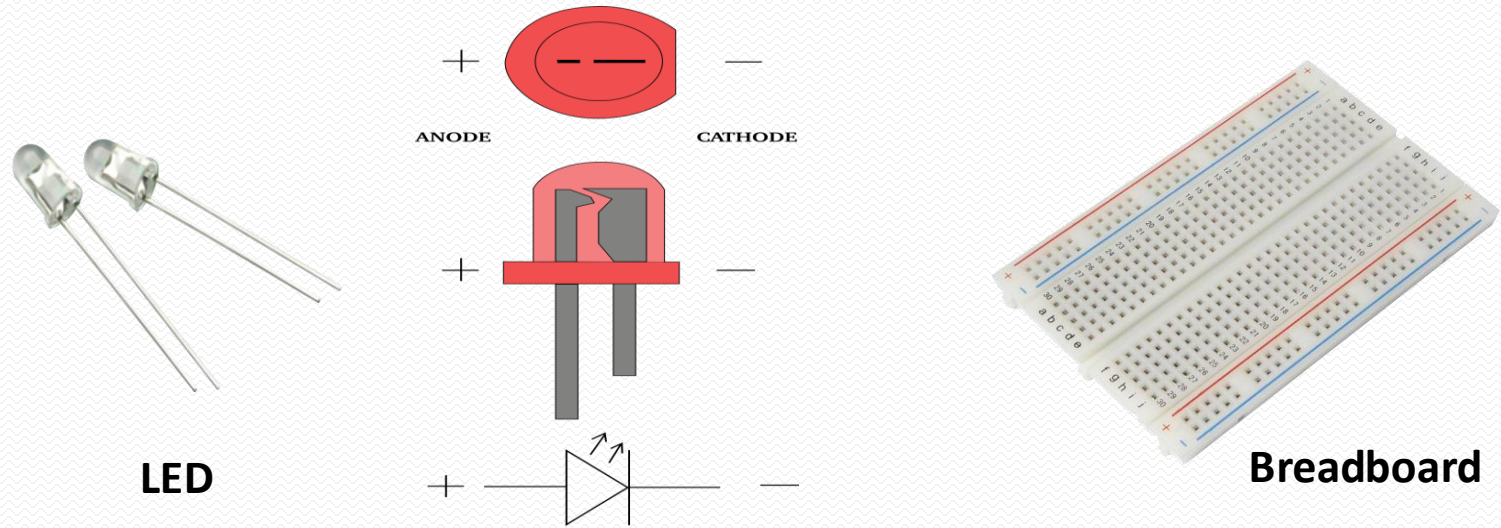
[Checksums \(sha512\)](#)

Code Structure

- void setup() {}
 - Used to indicate the initial values of system on starting
- void loop() {}
 - Contains the statements that will run whenever the system is powered after setup



How to deal with LED and Breadboard



How to connect the LED on the Breadboard



Let's implement

- LED on/off in every 1sec on the Arduino

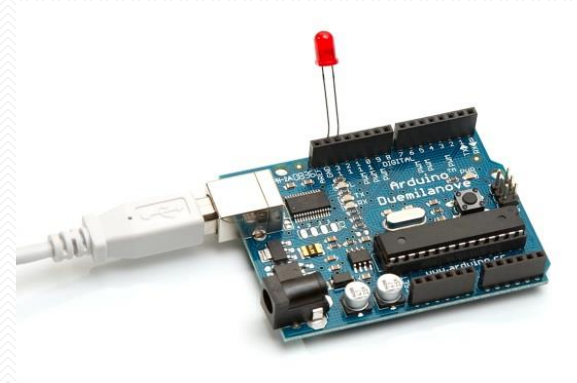
- Used functions :

[Language Reference Link Click](#)

- pinMode()
- digitalWrite()
- digitalWrite()
- delay(time_ms)

Communicate with Arduino

- NUC sends command to Arduino via USB-UART to turn on LEDs.



References

- <https://www.slideshare.net/xxahmedsakrxx/introduction-to-arduino-16634116>
- <https://ko.wikipedia.org/wiki/%EC%95%84%EB%91%90%EC%9D%B4%EB%85%B8>
- <http://mechasolution.com/>
- <https://www.arduino.cc/>
- <https://www.slideshare.net/avikdhupar/intro-to-arduino>