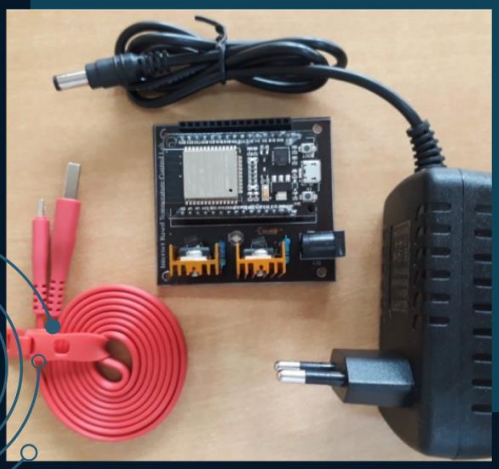
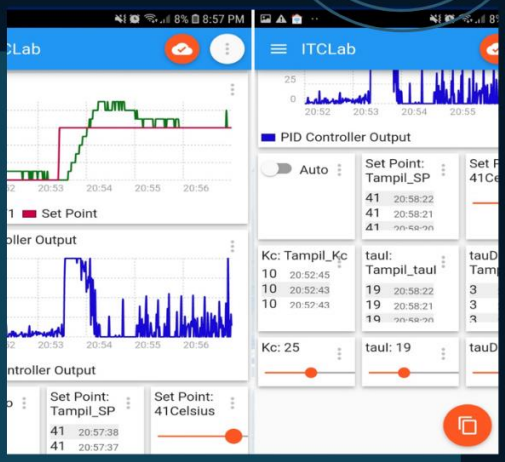
ITCLAB

INTERNET-BASED TEMPERATUE CONTROL LAB



iTCLab Kits for:
- Internet of Things (IoT)
- System Dynamics
- Control System



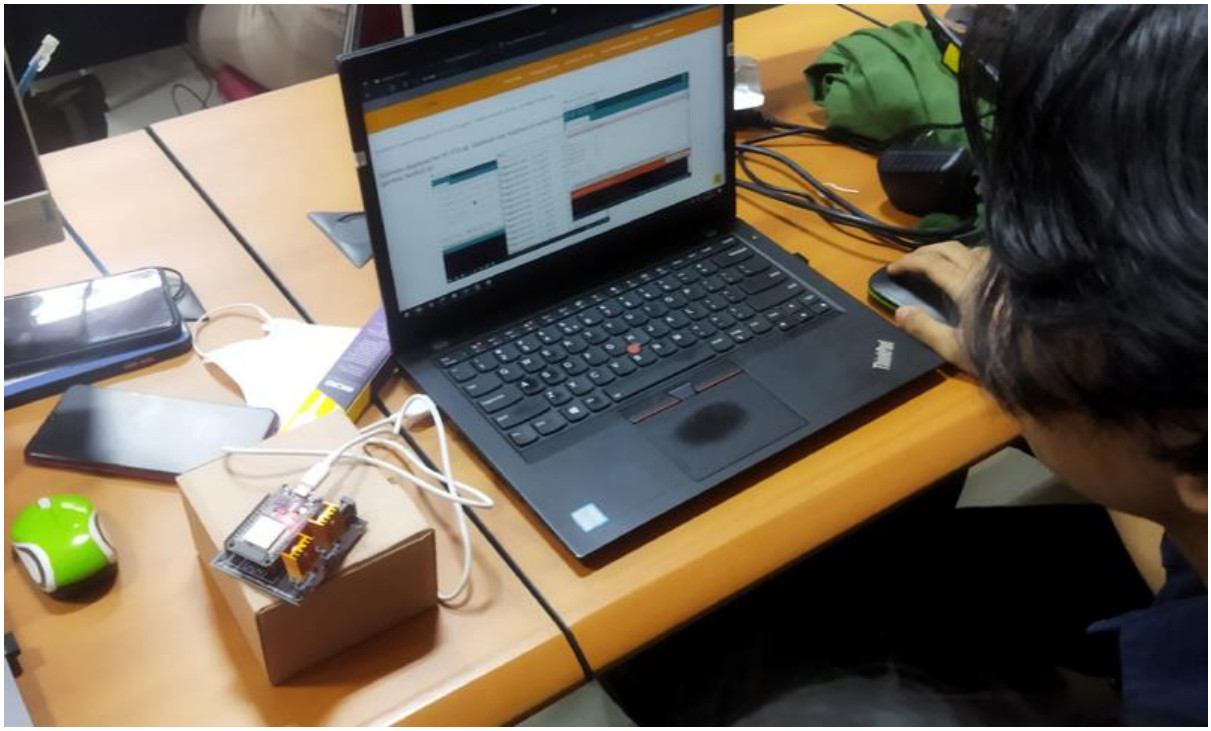
iTCLab Kits for:
- Machine Learning
- Arduino and Python programming
- System Modeling

https://io-t.net/itclab
https://github.com/bsrahmat
https://shopee.co.id/product/78709625/11589970517/

https://io-t.net/itclab
https://github.com/bsrahmat
https://shopee.co.id/product/78709625/11589970517

PWM Testing

Assoc. Prof. Dr. Basuki Rahmat, S.Si, MT, ITS-AI, et al
Intelligent Control, Robotics and Automation Systems Research Group
Universitas Pembangunan Nasional "Veteran" Jawa Timur, Indonesia
i-ot.net, io-t.net
https://github.com/bsrahmat



PWM Testing

PWM_Testing is a simple program for testing Pulse Width Modulation (PWM) of the iTCLab Kit. This PWM setting will later be used to increase and decrease the heat on the iTCLab heater.

Pulse Width Modulation (PWM) is a modulation technique that changes the pulse width with a constant frequency and amplitude value. PWM can be considered as the opposite of ADC (Analog to Digital Converter) which converts Analog signals to Digital, PWM is used to generate analog signals from Digital devices (iTCLab Kit).

The PWM signal will remain ON for a certain period of time and then stop or OFF for the rest of the period. What makes PWM special and more useful is that we can determine how long the ON condition should last by controlling the duty cycle of the PWM.

About these iTCLab kits :

iTCLab - Internet-Based Temperatue Control Lab. Temperature control kit for feedback control applications with ESP32 microcontroller, LEDs, two heaters, and two temperature sensors. The heating power output is adjusted to maintain the desired temperature setpoint. Thermal energy from the heater is transferred by conduction, convection, and radiation to the temperature sensor. Heat is also transferred from the device to the environment.

More about iTCLab:

- Inspired by BYU (Brigham Young University) TCLab Product. A private university in Provo, Utah, United States.
- Miniature Control System in Our Pocket.
- Practical IoT Learning Package.
- Introduction to IoT Systems.
- IoT Programming.
- Practice of IoT-Based Control Systems.
- Can be used to learn System Dynamics and Control Systems.
- Can be used to learn Arduino and Python Programming.
- Can be used to learn Machine Learning Programming.
- And others.

The fundamental difference between iTCLab and BYU's TCLab product is the replacement of the Arduino Uno microcontroller with the ESP32. By using the ESP32, iTCLab has the ability to connect to the Internet of Things (IoT).

iTCLab Upper Temperature Limit Description:

The upper temperature limit of the iTCLab Kit is 60 degrees Celsius. Therefore, when experimenting with this Kit, this Upper Temperature Limit must not be exceeded. Violation of this provision could cause damage (burning) to the components.

Although the upper limit is 60 degrees Celsius, it is still sufficient for experimenting with this Kit. And it is sufficient to see the performance of a control method. For example, control using Proportional Integral and Derivative (PID). Or to see the effect of tuning the PID parameters using the Machine Learning method. An illustration of the capabilities of this iTCLab Kit can be seen from the illustration of the performance of the BYU TCLab, as seen in the following simulation.

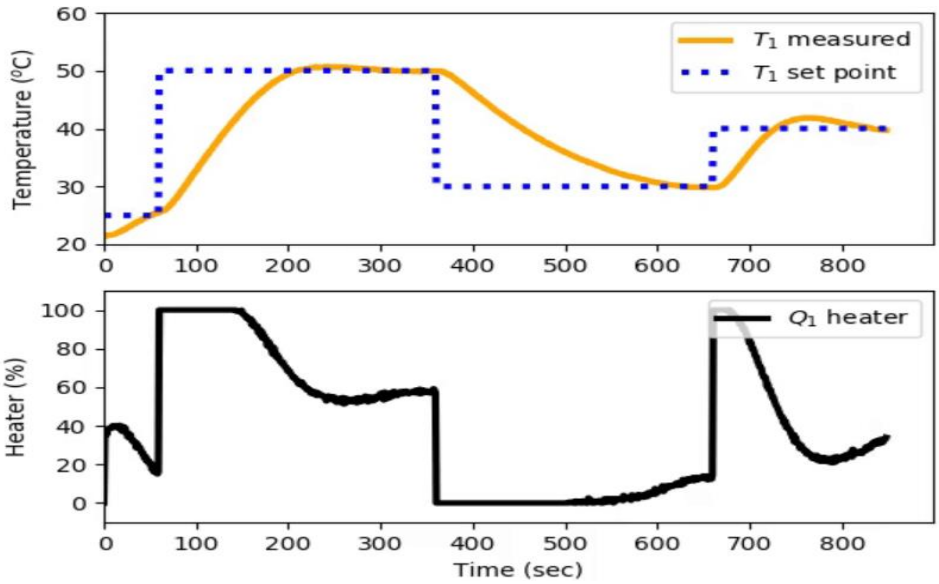


Figure 1. BYU TCLab Kit performance overview

Coding Upper Temperature Limit

It is necessary to set a limit so that the iTCLab Kit always operates in a safe area. It must not exceed the upper limit of 60 degrees Celsius. The following is an example of an Arduino program script that must be added every time you experiment with this Kit. In the Loop, it is added that if it reaches the specified upper limit (it can be lowered slightly, for example 55 degrees Celsius), then the heater must be turned off.

```
void loop() {
  // put your main code here, to run repeatedly:
  cektemp();
  if (cel > upper_temperature_limit){
    Q1off();
    ledon();
  }
  else {
    Q1on();
    ledoff();
  }
  if (cel1 > upper_temperature_limit){
    Q2off();
    ledon();
  }
  else {
    Q2on();
    ledoff();
  }
  delay (100);
}
```

PWM Testing Program

PWM_Testing is a simple program for testing Pulse Width Modulation (PWM) of the iTCLab Kit. This PWM setting will later be used to increase and decrease the heat on the iTCLab heater. In this simple PWM Testing program example, the result of the PWM setting is shown by turning on the LED from dim to bright, and from bright to dim. Repeatedly.

Required Equipment:

- iTCLab Kit
- PWM_Testing.ino Program
- PWM_Testing.pdf Tutorial

Another alternative is to download the tutorial:

- https://www.academia.edu/116155979
- https://www.researchgate.net/publication/378904388

Required Settings:

File Settings - Preferences:

The iTCLab Kit uses an ESP32 microcontroller. Please copy and paste the following address in the File - Preferences section:

Address: https://dl.espressif.com/dl/package_esp32_index.json

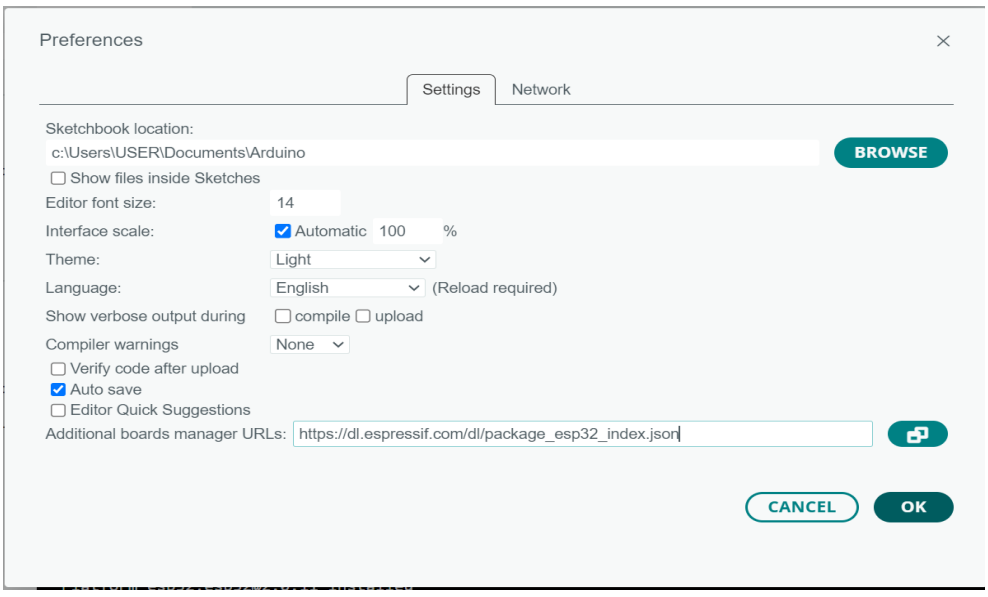


Figure 2. File Settings – Preferences

Board Settings for DOIT ESP32 DEVKIT V1

ESP32 Microcontroller Board Settings, in the Menu Tools - Board - Boards Manager. Please select DOIT ESP32 DEVKIT V1.

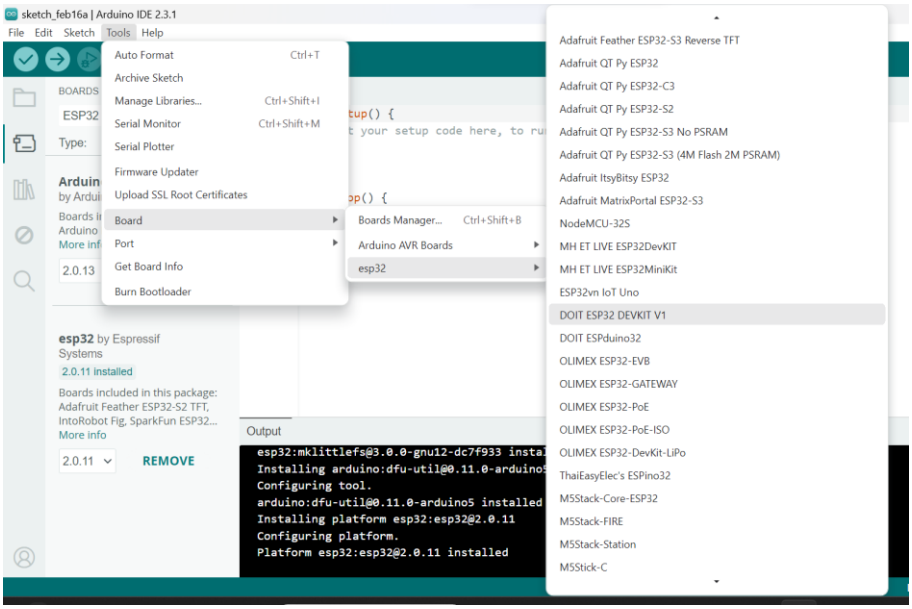


Figure 3. Selecting the DOIT ESP32 DEVKIT V1 Board

iTCLab Port Selection:

Please select the port that corresponds to the iTCLab Kit when it is connected to the computer (laptop).

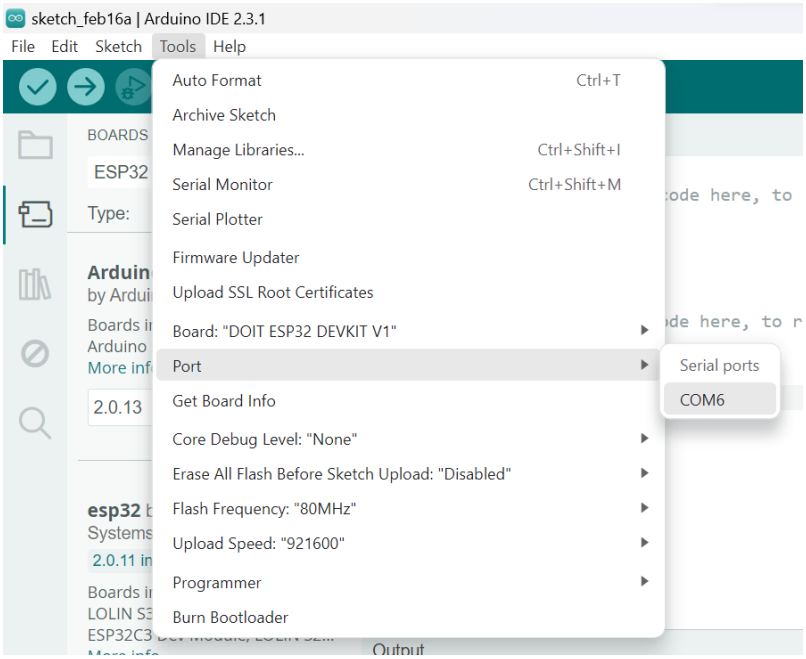


Figure 4. iTCLab Port Selection

Upload speed setting

Please select the upload speed, at 115200.

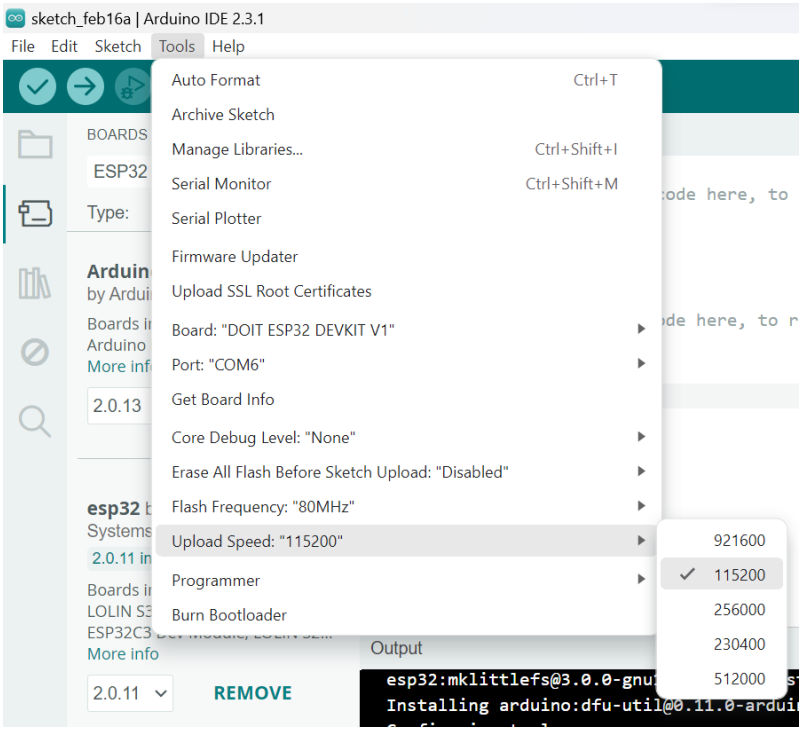


Figure 5. Upload speed setting

Here is the PWM_Testing.ino coding script:

Program : PWM_Testing

```
/*********************************************************************
 * Program    : PWM_Testing
 * By         : Assoc. Prof. Dr. Basuki Rahmat, S.Si, MT, ITS-AI, et al
 * Pro. Team  : i-ot.net, io-t.net
 * R. Group   : Intelligent Control, Robotics and Automation Systems Research Group
 * Univ.      : Universitas Pembangunan Nasional "Veteran" Jawa Timur
 * Country    : Indonesia
 *********************************************************************/

// the number of the LED pin
const int ledPin = 26;

// setting PWM properties
const int freq = 5000;
const int ledChannel = 0;
const int resolution = 8;

void setup(){
```

```
  // configure LED PWM functionalitites
  ledcSetup(ledChannel, freq, resolution);

  // attach the channel to the GPIO to be controlled
  ledcAttachPin(ledPin, ledChannel);
}

void loop(){
  // increase the LED brightness
  for(int dutyCycle = 0; dutyCycle <= 255; dutyCycle++){
    // changing the LED brightness with PWM
    ledcWrite(ledChannel, dutyCycle);
    delay(20);
  }

  // decrease the LED brightness
  for(int dutyCycle = 255; dutyCycle >= 0; dutyCycle--){
    // changing the LED brightness with PWM
    ledcWrite(ledChannel, dutyCycle);
    delay(20);
  }
}
```

Please upload the code above to the iTCLab Kit, wait until the process is finished. After it has been successfully uploaded. Please check the result by looking at the brightness change of the iTCLab LED Kit. The LED added to the iTCLab Kit, not the LED on the ESP32 microcontroller.

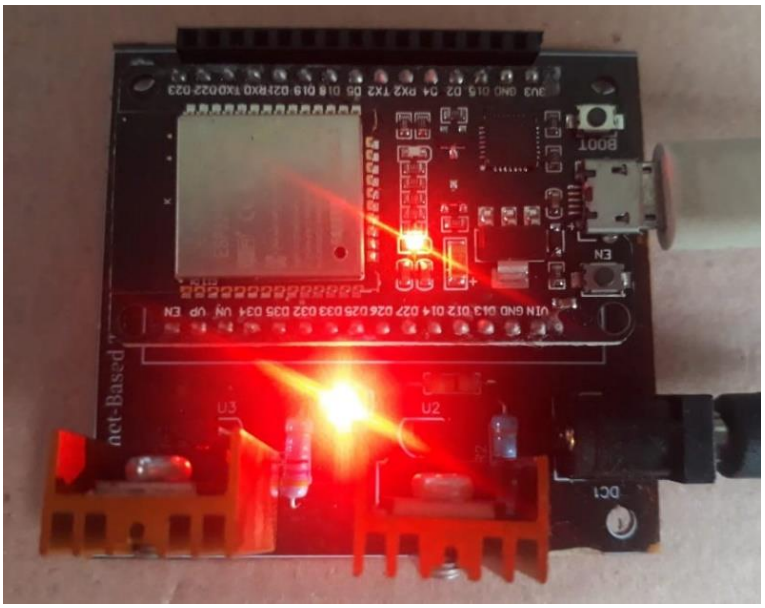


Figure 6. Brightness change of the iTCLab Kit LED according to the PWM duty cycle