MODUL 1

Pengenalan Editor dan Diagram Alir Pemrograman

1. TUJUAN

1.1 Mahasiswa mampu mengenal dan menggunakan Arduino IDE serta Thonny IDE untuk pemrograman mikrokontroler (Arduino UNO dan ESP32).

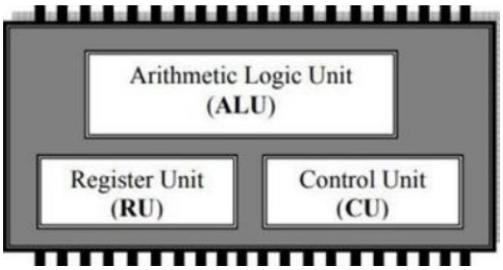
1.2 Mahasiswa mampu mengimplementasikan program sederhana menggunakan Arduino IDE dan Thonny IDE, serta memahami alur kerjanya melalui flowchart.

2. ALAT DAN BAHAN PRAKTIKUM

- Arduino IDE
- Thonny
- Arduino Uno
- ESP32
- Kabel USB-B
- Kabel Micro USB

3. DASAR TEORI

3.1 Mikroprosesor



Mikroprosesor, sering disebut sebagai *Central Processing Unit* (CPU), adalah unit utama yang mengendalikan proses komputasi. Beberapa contoh mikroprosesor yang umum di pasaran meliputi:

1) Intel Core (i3, i5, i7)
2) AMD Ryzen, Athlon, Phenom
3) Qualcomm Snapdragon

Seiring dengan perkembangan teknologi, mikroprosesor juga dikenal sebagai MPU (Microprocessor Unit). Komponen utama dalam mikroprosesor meliputi:

1) ALU (Arithmetic Logic Unit): Melakukan operasi aritmetika (seperti penjumlahan) dan operasi logika (seperti AND, OR).

2) Register Unit (RU): Menyimpan data sementara dan hasil operasi ALU. Register utama disebut akumulator.

3) Control Unit (CU): Mengendalikan aliran data melalui bus data dan bus Alamat.

Arduino

3.2 Arduino IDE

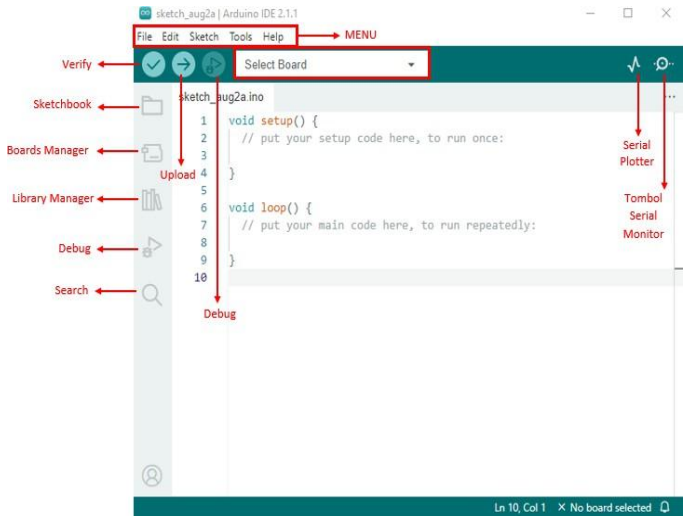
Arduino Integrated Development Environment (IDE) adalah perangkat lunak yang dirancang khusus untuk pemrograman dan pengembangan proyek dengan platform Arduino. Berikut merupakan fitur yang ada pada Arduino IDE.

a. Sintaks dasar

Sintaks/Fungsi	Deskripsi	Contoh Penggunaan
`setup()`	Fungsi ini digunakan untuk inisialisasi. Kode di dalam `setup()` hanya dijalankan sekali saat perangkat dimulai.	`void setup() { pinMode(13, OUTPUT); }`
`loop()`	Fungsi yang dijalankan terus-menerus setelah `setup()` selesai. Ideal untuk menjalankan program utama.	`void loop() { digitalWrite(13, HIGH); delay(1000); }`
`pinMode(pin, mode)`	Mengatur fungsi dari pin Arduino (input atau output).	`pinMode(13, OUTPUT);`
`digitalWrite(pin, val)`	Mengatur nilai pin digital (HIGH atau LOW).	`digitalWrite(13, HIGH);`
`digitalRead(pin)`	Membaca nilai pin digital (HIGH atau LOW).	`int val = digitalRead(2);`
`analogWrite(pin, val)`	Menulis nilai analog (PWM) pada pin output tertentu (0 hingga 255).	`analogWrite(9, 128);`
`analogRead(pin)`	Membaca nilai analog dari pin input tertentu (0 hingga 1023).	`int sensorValue = analogRead(A0);`
`delay(ms)`	Memberikan jeda selama beberapa milidetik.	`delay(1000);`
`Serial.begin(baudrate)`	Menginisialisasi komunikasi serial pada kecepatan baud tertentu.	`Serial.begin(9600);`
`Serial.print(val)`	Mencetak data ke monitor serial.	`Serial.print("Hello World");`
`Serial.println(val)`	Mencetak data ke monitor serial dengan pindah baris baru.	`Serial.println("Data Received");`

b. Tools

Nama Tool	Deskripsi	Fungsi
Verify (Centang)	Memeriksa apakah ada kesalahan dalam kode (kompilasi).	`Memeriksa kesalahan sintaks dalam program.`
Upload (Panah)	Mengunggah kode ke mikrokontroler Arduino setelah kompilasi berhasil.	`Mengunggah kode ke perangkat Arduino.`
New (Kertas)	Membuat sketch (proyek) baru.	`Membuat file proyek baru.`
Open (Folder)	Membuka sketch yang telah disimpan sebelumnya.	`Membuka file proyek yang sudah ada.`
Save (Disk)	Menyimpan sketch yang sedang dikerjakan.	`Menyimpan proyek yang sedang dibuat.`
Serial Monitor (Layar)	Menampilkan data yang dikirim dari dan ke Arduino melalui komunikasi serial.	`Melihat hasil atau data dari komunikasi serial.`
Board Manager	Mengelola jenis board yang digunakan.	`Memilih board yang sesuai dengan perangkat.`
Port	Menentukan port serial tempat Arduino terhubung ke komputer.	`Memilih port tempat perangkat terhubung.`



c. Format angka dalam Arduino IDE

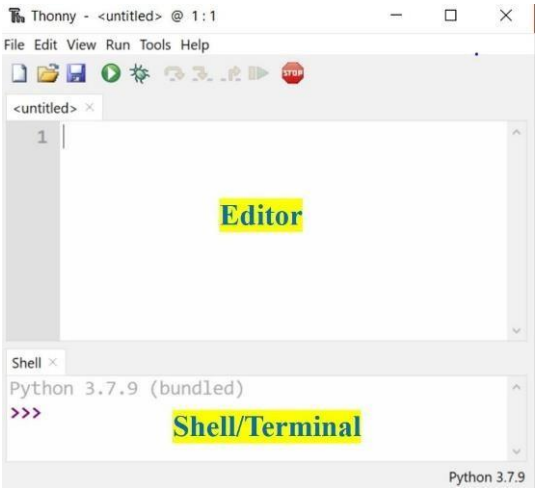
Format	Deskripsi	Contoh Deklarasi	Output
Binary	Angka biner diawali dengan `0b`.	`int varBiner = 0b1010;`	1010 (biner)
Hexadecimal	Angka heksadesimal diawali dengan `0x`.	`int varHex = 0xA;`	A (heksadesimal)
Octal	Angka oktal diawali dengan `0`.	`int varOktal = 012;`	12 (oktal)
Decimal	Angka desimal biasa.	`int var = 10;`	10 (desimal)

d. Baudrate yang sering digunakan

4) 9600
5) 115200
6) 57600
7) 38400

3.3 Thonny

Thonny adalah lingkungan pengembangan terintegrasi (IDE) untuk Python yang dirancang untuk kemudahan belajar, terutama bagi pemula. Thonny juga mendukung pemrograman mikrokontroler dengan MicroPython, seperti ESP32 dan Raspberry Pi Pico.



3.4 Perbandingan Arduino IDE dan Thonny

Aspek	Arduino IDE	Thonny IDE
Bahasa Pemrograman	C/C++ (sederhana)	Python
Dukungan Platform	Arduino Boards	Mikrokontroler dengan MicroPython
Debugging	Tidak ada debugging terintegrasi	Debugging terintegrasi
Pengguna Sasaran	Pemula dan pengguna Arduino	Pemula Python dan MicroPython
Kelebihan	Mudah digunakan, komunitas besar	Mudah dipahami, mendukung MicroPython
Keterbatasan	Fitur terbatas untuk proyek kompleks	Terbatas pada Python

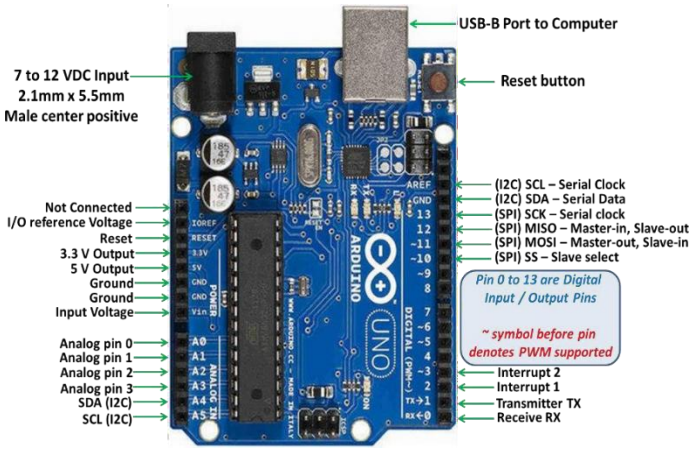
3.5 Micropython

MicroPython adalah implementasi bahasa pemrograman Python yang dirancang untuk mikrokontroler. Memiliki sintaks yang hampir sama dengan Python 3 dan banyak digunakan dalam pengembangan sistem tertanam. Berikut merupakan sintaks dasar pada Thonny.

Sintaks	Deskripsi	Contoh Penggunaan
`print()`	Mencetak teks atau nilai ke terminal.	`print("Hello, MicroPython!")`
`time.sleep(seconds)`	Menunda program dalam detik.	`time.sleep(1)`
`time.sleep_ms(milliseconds)`	Menunda program dalam milidetik.	`time.sleep_ms(500)`
`machine.Pin(pin, mode)`	Mengatur pin sebagai input atau output.	`led = machine.Pin(2, machine.Pin.OUT)`
`led.value(1)`	Menyalakan pin output (misalnya LED).	`led.value(1)`
`led.value(0)`	Mematikan pin output.	`led.value(0)`

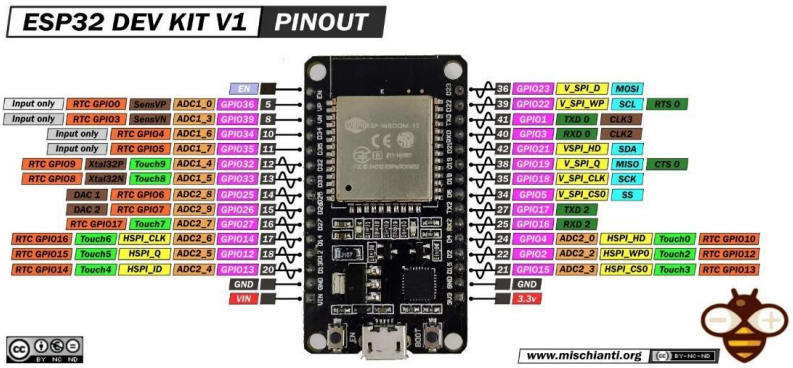
3.6 Arduino UNO

Arduino UNO adalah papan mikrokontroler berbasis sistem open hardware yang menggunakan mikrokontroler Atmel AVR. Arduino UNO telah dilengkapi dengan prosesor, memori, dan input/output (I/O) yang memungkinkan pengguna untuk mengendalikan berbagai perangkat elektronik.

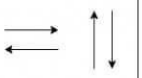
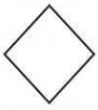


3.7 ESP32

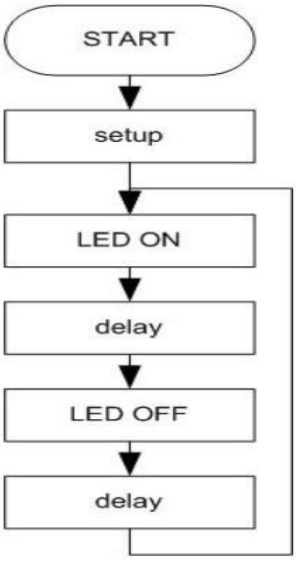
ESP32 adalah mikrokontroler yang dikembangkan oleh Espressif System, sebagai penerus dari ESP8266. Mikrokontroler ini kompatibel dengan Arduino IDE dan dilengkapi dengan modul WiFi serta Bluetooth Low Energy (BLE) dalam satu chip. Hal ini menjadikannya pilihan yang ideal untuk pengembangan aplikasi IoT



3.8 Flowchart

	**Flow** Simbol yang digunakan untuk menggabungkan antara simbol yang satu dengan simbol yang lain. Simbol ini disebut juga dengan Connecting Line.		**Input/output** Simbol yang menyatakan proses input atau output tanpa tergantung peralatan.
	**On-Page Reference** Simbol untuk keluar - masuk atau penyambungan proses dalam lembar kerja yang sama.		**Manual Operation** Simbol yang menyatakan suatu proses yang tidak dilakukan oleh komputer.
	**Off-Page Reference** Simbol untuk keluar - masuk atau penyambungan proses dalam lembar kerja yang berbeda.		**Document** Simbol yang menyatakan bahwa input berasal dari dokumen dalam bentuk fisik, atau output yang perlu dicetak.
	**Terminator** Simbol yang menyatakan awal atau akhir suatu program.		**Predefine Proses** Simbol untuk pelaksanaan suatu bagian (sub-program) atau prosedure.
	**Process** Simbol yang menyatakan suatu proses yang dilakukan komputer.		**Display** Simbol yang menyatakan peralatan output yang digunakan.
	**Decision** Simbol yang menunjukan kondisi tertentu yang akan menghasilkan dua kemungkinan jawaban, yaitu ya dan tidak.		**Preparation** Simbol yang menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberikan nilai awal.

3.9 Flowchart Program Blink



Pada Flowchart tersebut menggambarkan alur kerja program **blink** untuk mikrokontroler, seperti pada Arduino, yang bertujuan menyalakan dan mematikan LED secara bergantian dengan jeda waktu tertentu. Proses dimulai dengan inisialisasi di bagian **setup**, di mana pin yang terhubung ke LED diatur sebagai output.

Setelah inisialisasi, LED dinyalakan dengan memberikan sinyal HIGH pada pin tersebut, diikuti oleh periode tunggu atau **delay** selama satu detik untuk menjaga LED tetap menyala. Setelah periode ini, LED dimatikan dengan

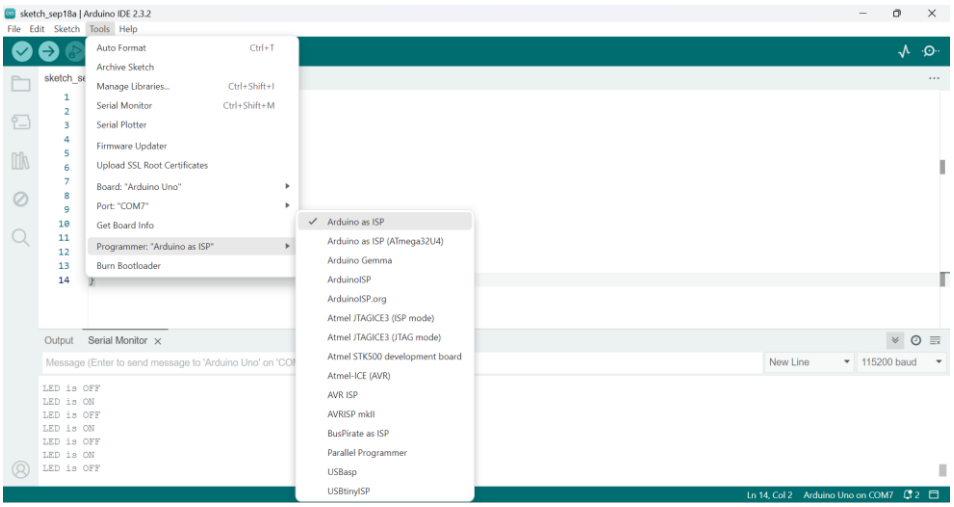
mengirimkan sinyal LOW ke pin yang sama, dan program kembali menunggu selama satu detik lagi.Siklus ini kemudian diulang secara terus-menerus, memastikan bahwa LED menyala dan mati secara bergantian dalam interval waktu yang sama, menciptakan efek kedip (blink) pada LED.

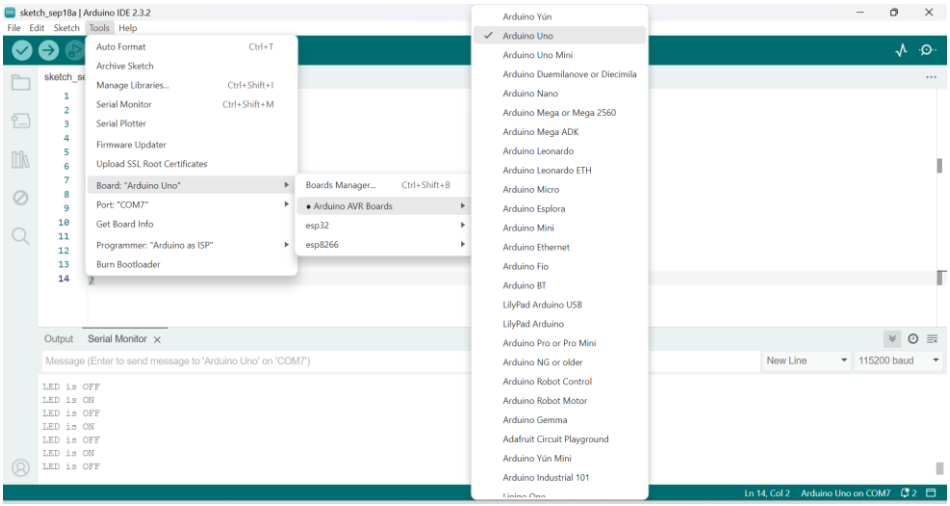
4. PERCOBAAN

4.1 Menampilkan Blink ON/OFF pada Arduino

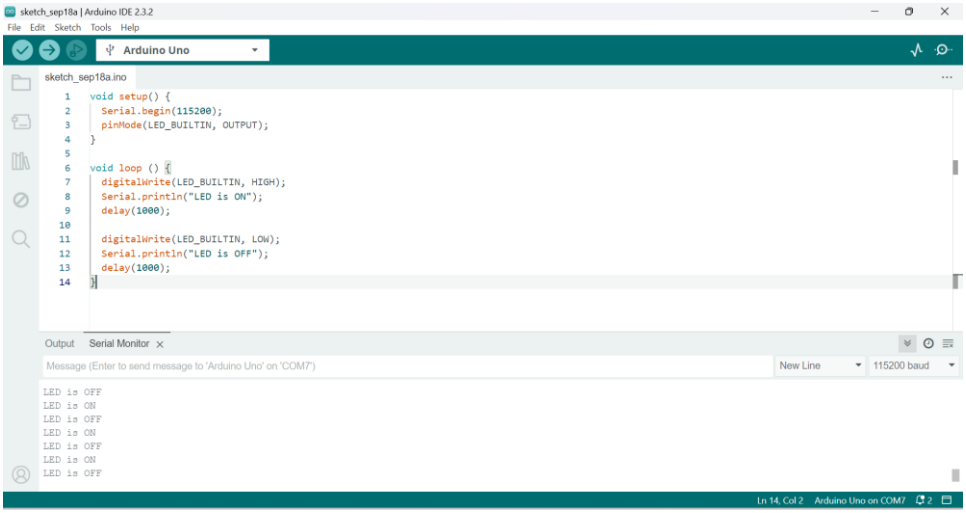
1. Buka Software Arduino IDE pada laptop yang sudah diinstall
2. Sambungkan USB Arduino UNO ke Laptop kalian
3. Buat pemrograman seperti dibawah ini:

4. Pada tools, ubah board menjadi “Arduino UNO”, lalu pilih port yang tersambung pada Arduino IDE. (sesuai port yang ada di laptop kalian).

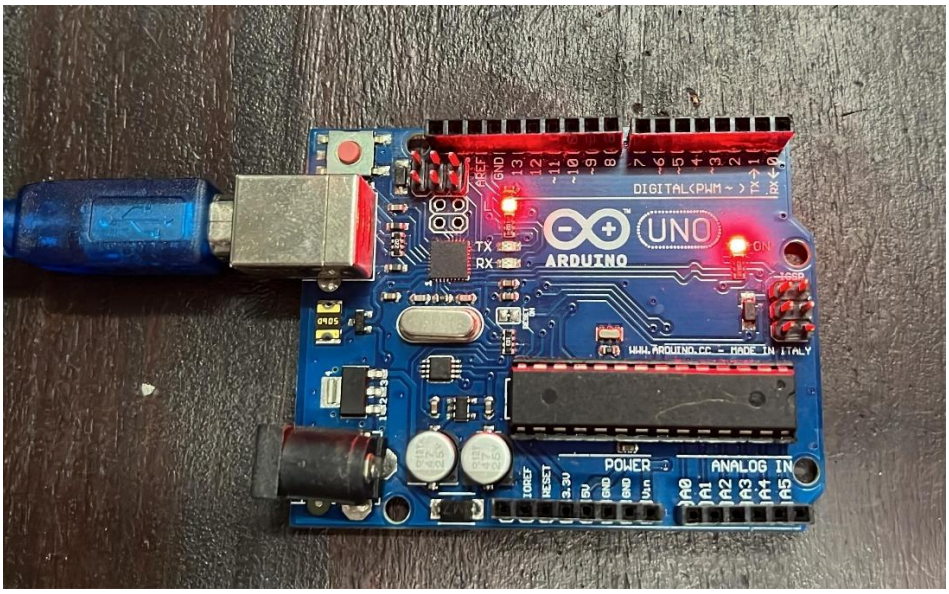




5. Lalu compile dan upload program tersebut. Tunggu hingga proses upload program selesai hingga bertuliskan “Done Uploading”. Selanjutnya buka Serial Monitor dan akan ditampilkan hasil seperti dibawah ini.

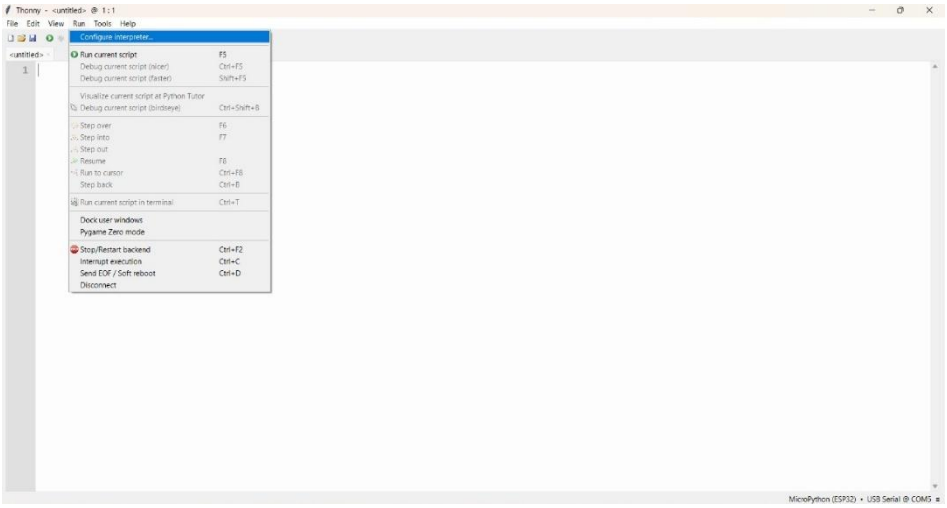


6. Berikut adalah hasil yang didapatkan:

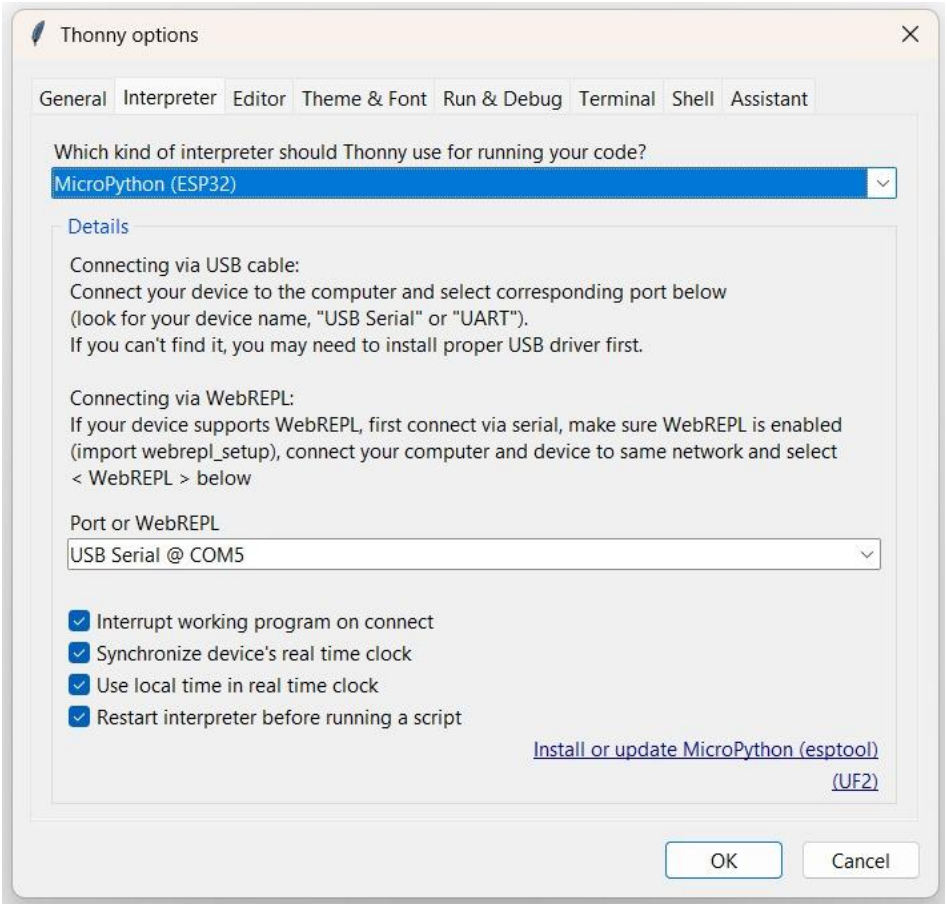


4.2 Menampilkan Blink ON/OFF dengan Micropython

1. Buka software Thonny
2. Sambungkan USB ESP32 ke laptop
3. Klik run, lalu klik Configure interpreter

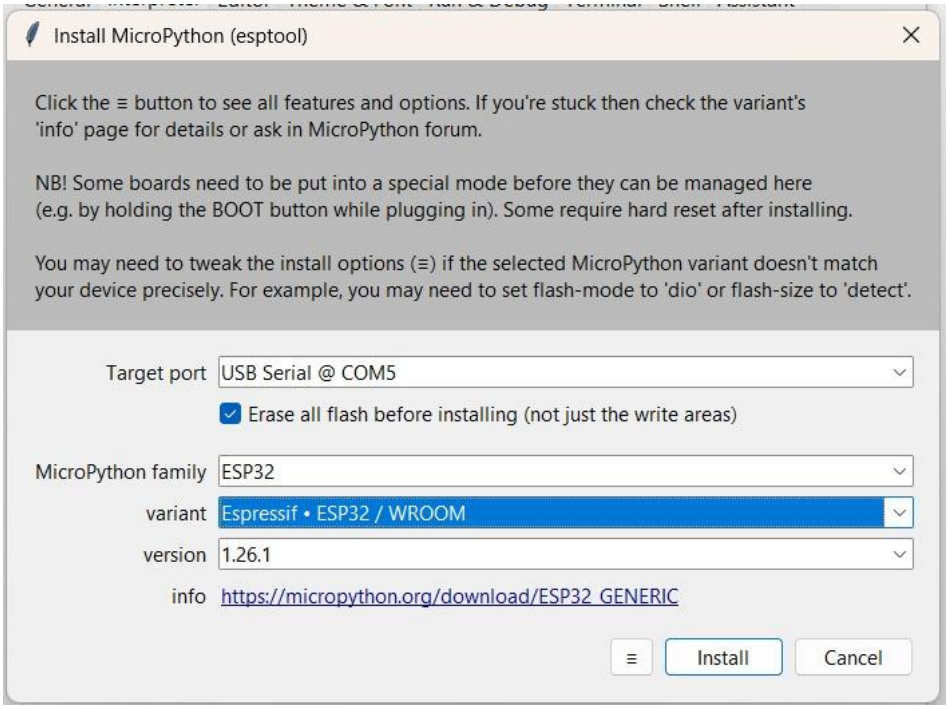


4. Pada which kind of interpreter should Thonny use for running your code? pilih MicroPython (ESP32).



5. Klik Install or update Micropython (esptool)

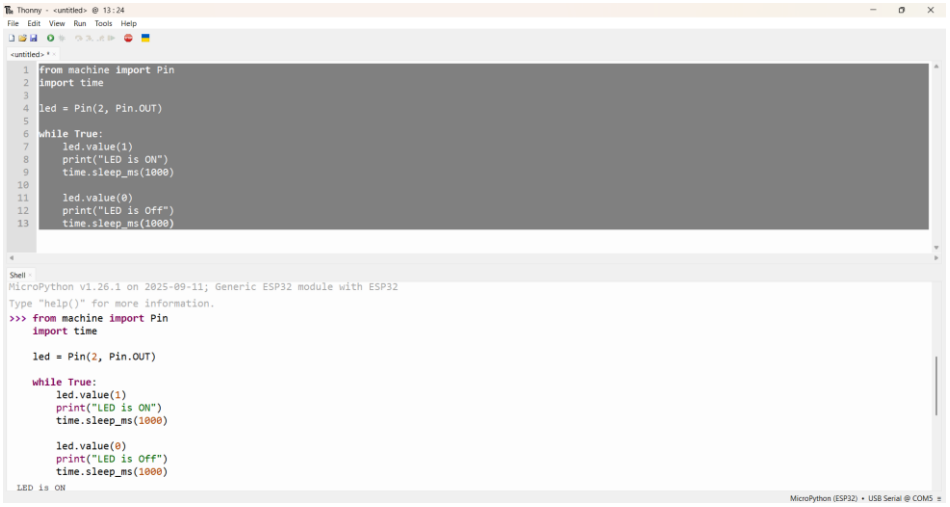
6. Untuk Target port pilih port yang tersedia. Micropython family pilih ESP32. Variant pilih Espressif • ESP32 / WROOM. Lalu klik install.



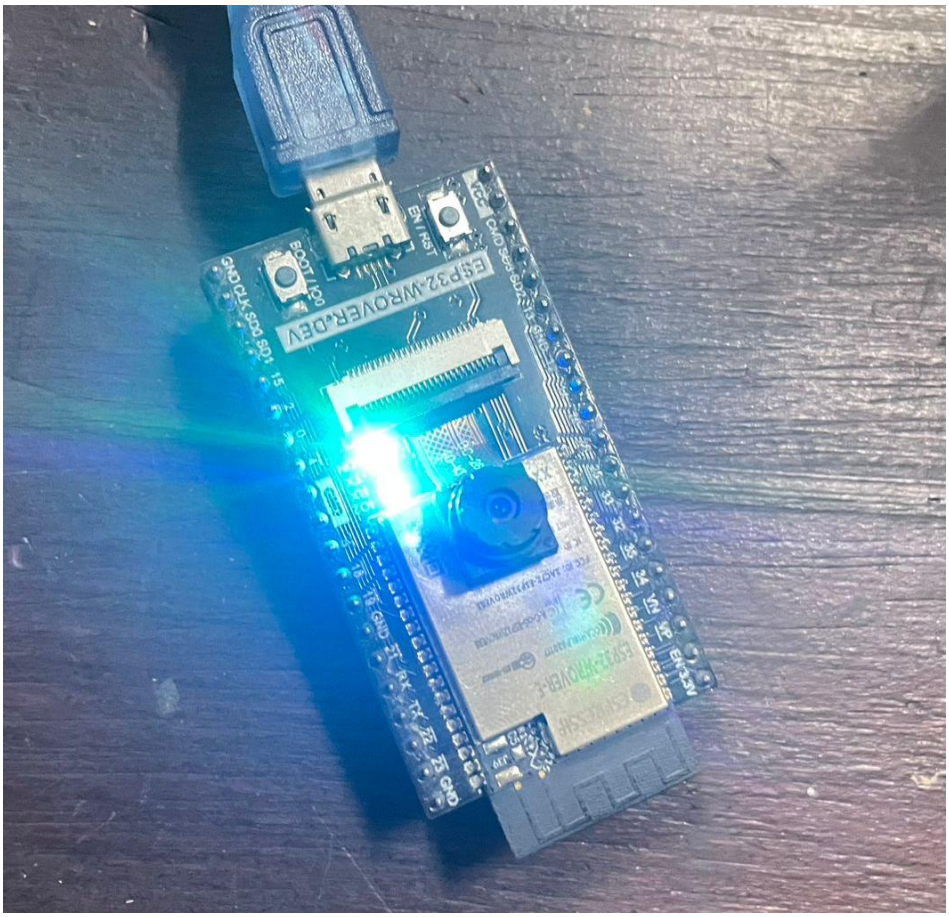
7. Jika instalasi berhasil maka shell akan menampilkan kode sebagai berikut.

8. Ketikkan perintah berikut lalu save as dengan format .py

9. *Copy* lalu *paste* kode ke shell, lalu klik enter.



10. Maka shell akan menampilkan hasil sebagai berikut.



5. KESIMPULAN

Mikrokontroler merupakan otak dari sistem elektronika yang berfungsi untuk mengendalikan berbagai perangkat berdasarkan instruksi yang diberikan. Dalam praktiknya, penggunaan *Arduino IDE* sangat membantu untuk memprogram papan seperti Arduino Uno maupun ESP32 dengan bahasa pemrograman berbasis C/C++,

sehingga mudah dipahami bahkan oleh pemula. Sementara itu, *Thonny IDE* memberikan alternatif lain dengan bahasa Python (MicroPython), yang lebih sederhana dan fleksibel untuk mengembangkan aplikasi IoT pada ESP32. Dengan demikian, pemahaman konsep mikrokontroler serta kemampuan menggunakan kedua IDE tersebut memberikan dasar yang kuat dalam merancang dan mengoperasikan sistem berbasis Arduino Uno maupun ESP32 secara efektif.

6. LATIHAN

1. Jika frekuensi LED sebesar 50Hz, maka berapa delay yang dibutuhkan pada program Blink di Arduino IDE?
2. Jika frekuensi LED sebesar 10Hz, maka berapa time.sleep_ms yang dibutuhkan pada program MicroPython di Thonny?
3. Tuliskan kode program yang dapat menampilkan teks ”LED is ON” di Shell Thonny!
4. Apa yang dimaksud dengan **led.value(1)** dan **led.value(0)**?

Terimakasih (★‿★)