



Muhammad Novian 20210120004 ▾

M2



Dashboard > My courses > EE4405

General

FORUM

Announcements

Introduction and Kontrak Belajar

ATTENDANCE

Presensi Pertemuan 01

Installing Thonny and Micropython

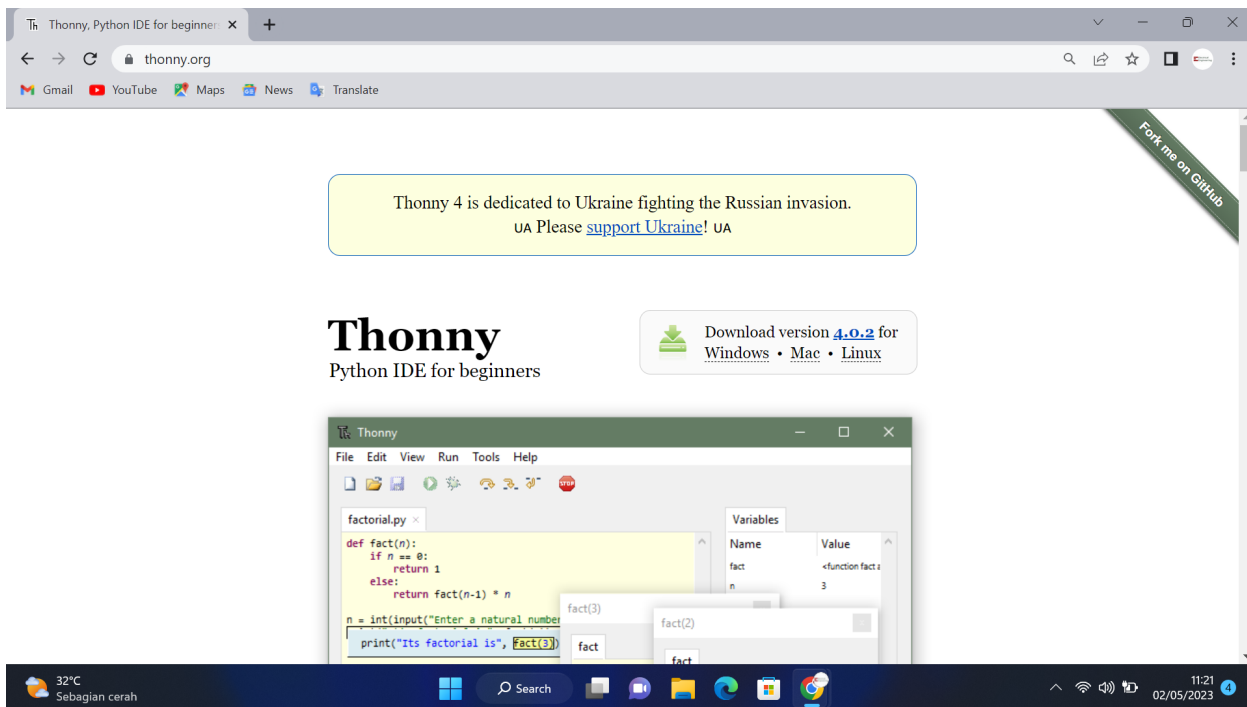
ATTENDANCE

Presensi Pertemuan 02

Tutorial Instalasi Thonny IDE

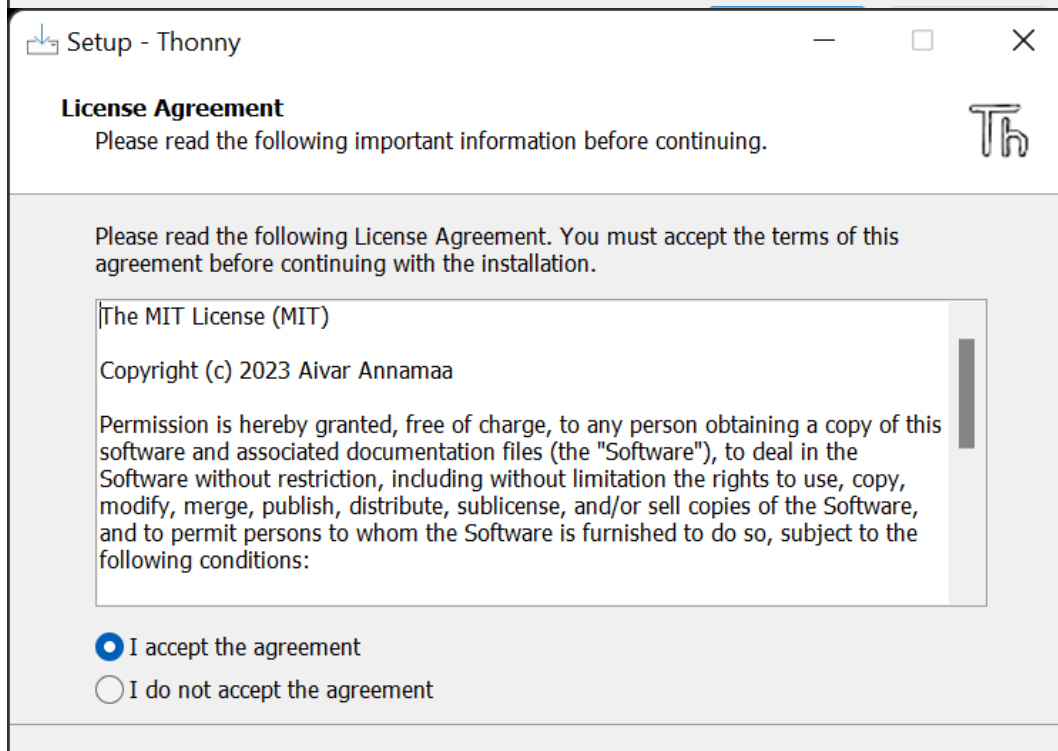
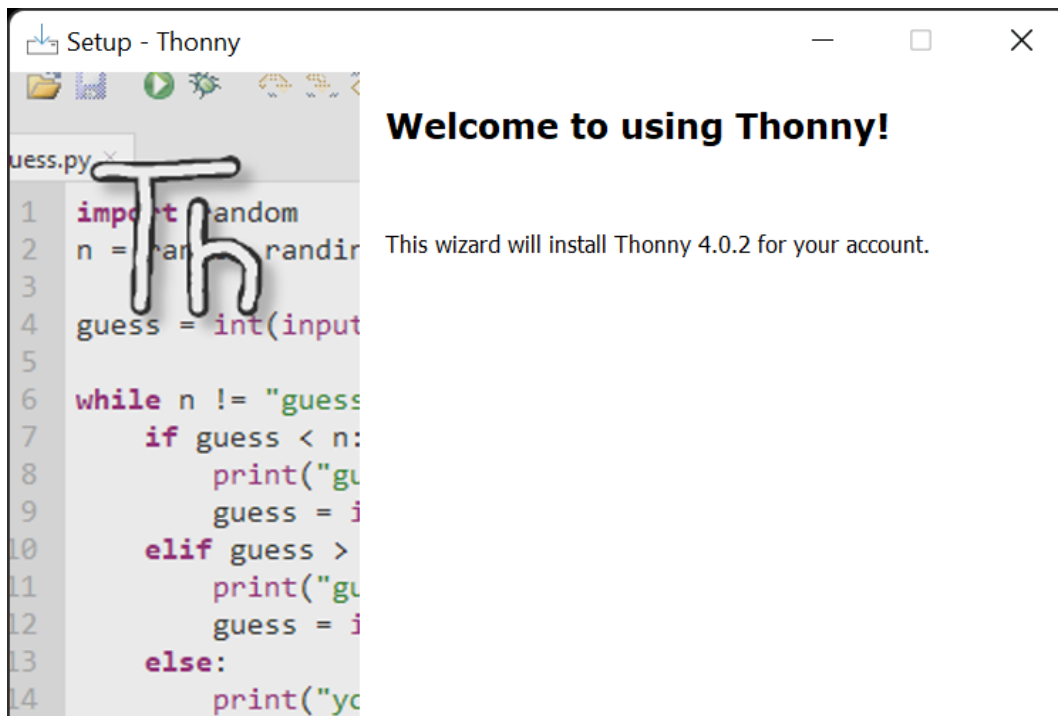
1. Buka website <https://thonny.org/> dan pilih download version di bagian kanan atas (sesuaikan device yang digunakan)





2. Setelah selesai proses download, kemudian buka file tersebut dan lakukan instalasi seperti pada umumnya





Select Destination Location

Where should Thonny be installed?



Setup will install Thonny into the following folder.

To continue, click Next. If you would like to select a different folder, click Browse.

C:\Users\Asus\AppData\Local\Programs\Thonny

Browse...

At least 113,8 MB of free disk space is required.

Select Additional Tasks

Which additional tasks should be performed?

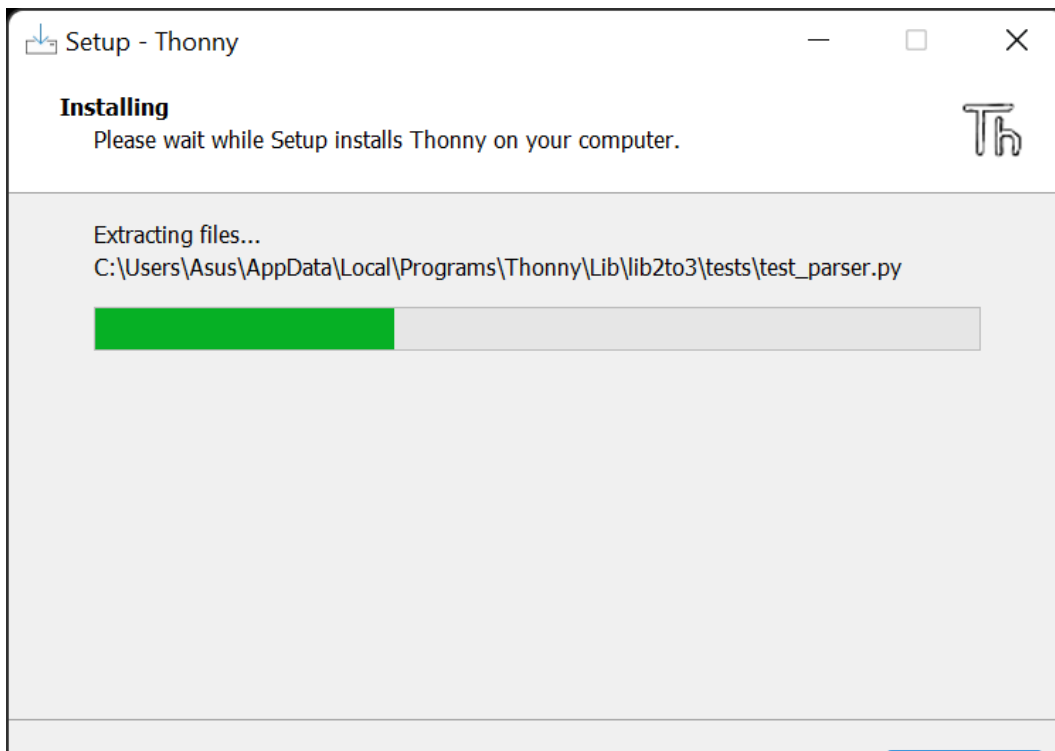


Select the additional tasks you would like Setup to perform while installing Thonny, then click Next.



Create desktop icon

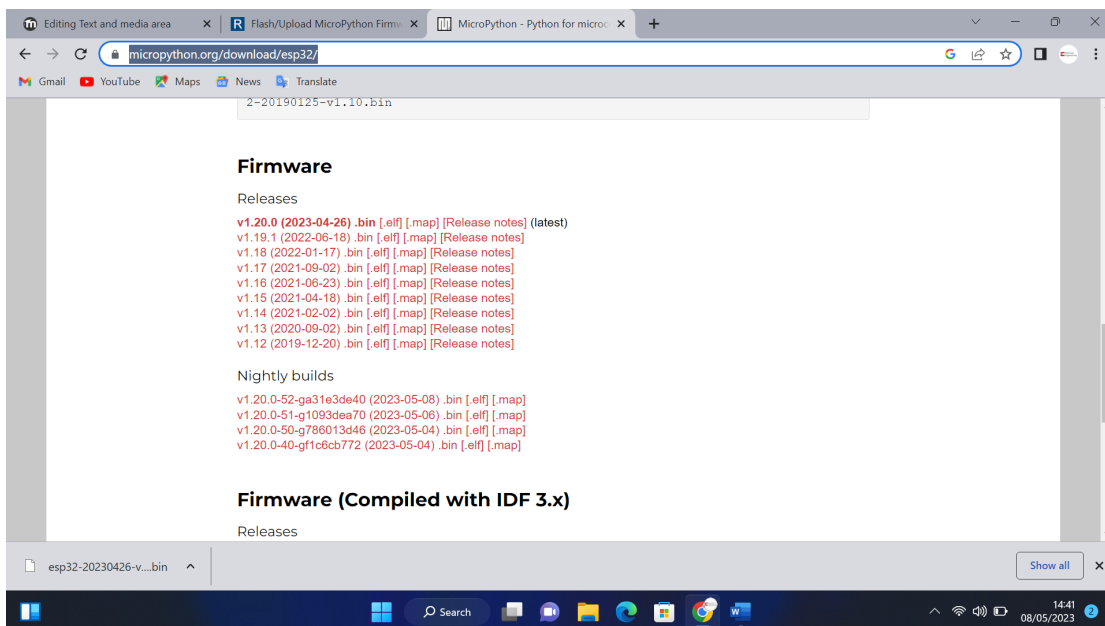




Cara Install Firmware Micropython ke ESP32

1. buka website <https://micropython.org/download/esp32/>

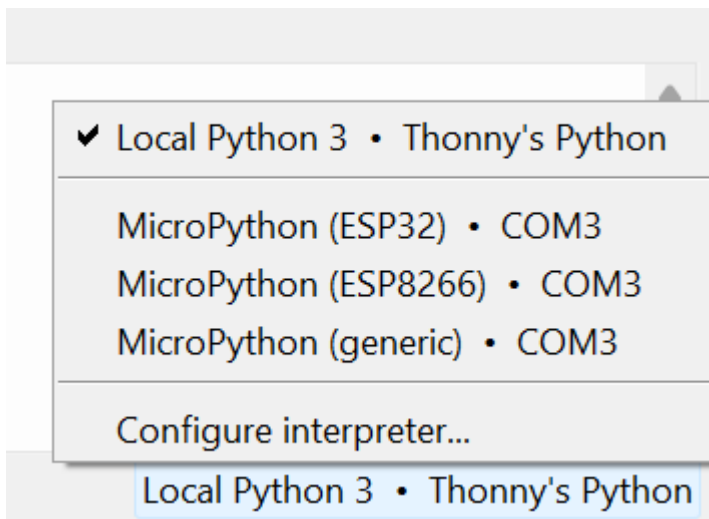
kemudian download firmware yang terbaru



2. Sambungkan ESP32 ke laptop kemudian buka Thonny IDE

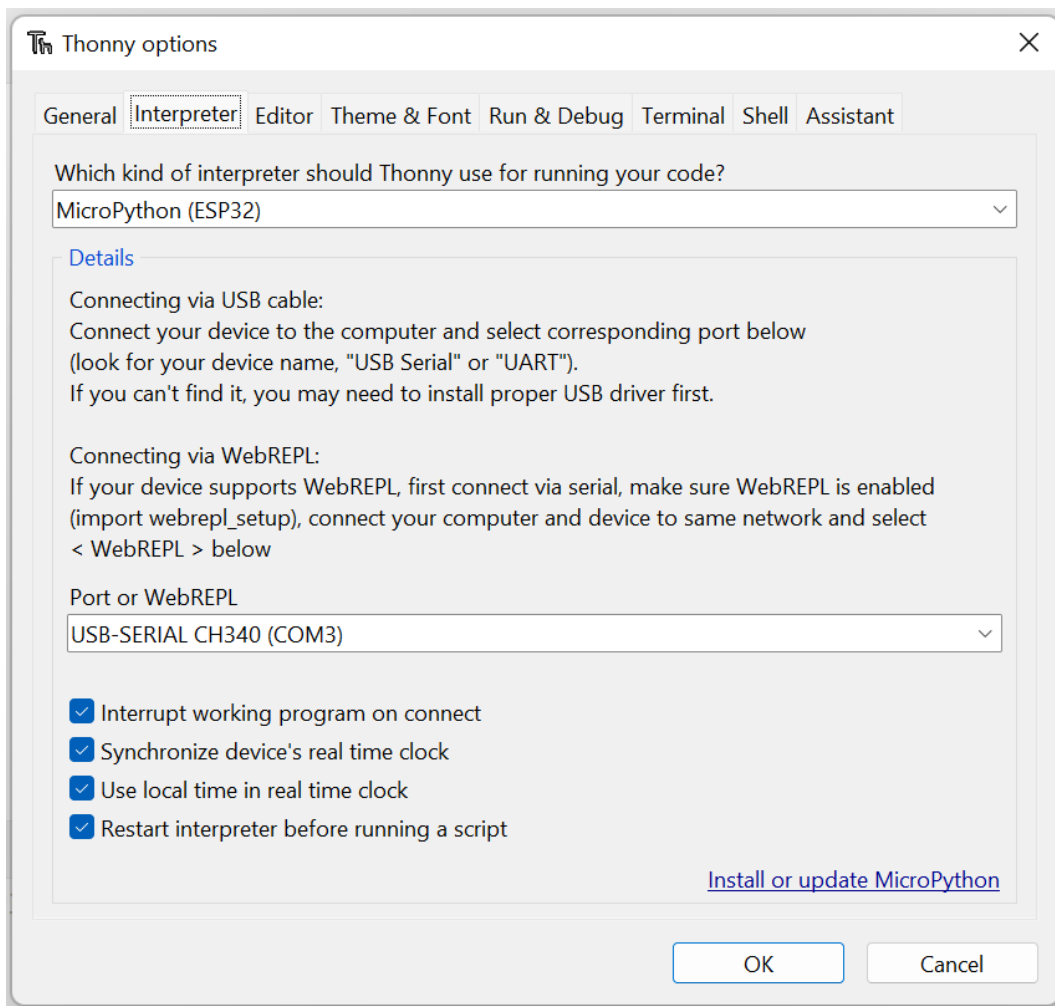
3. Klik pada pojok kanan bawah, pilih MicroPython (ESP32)





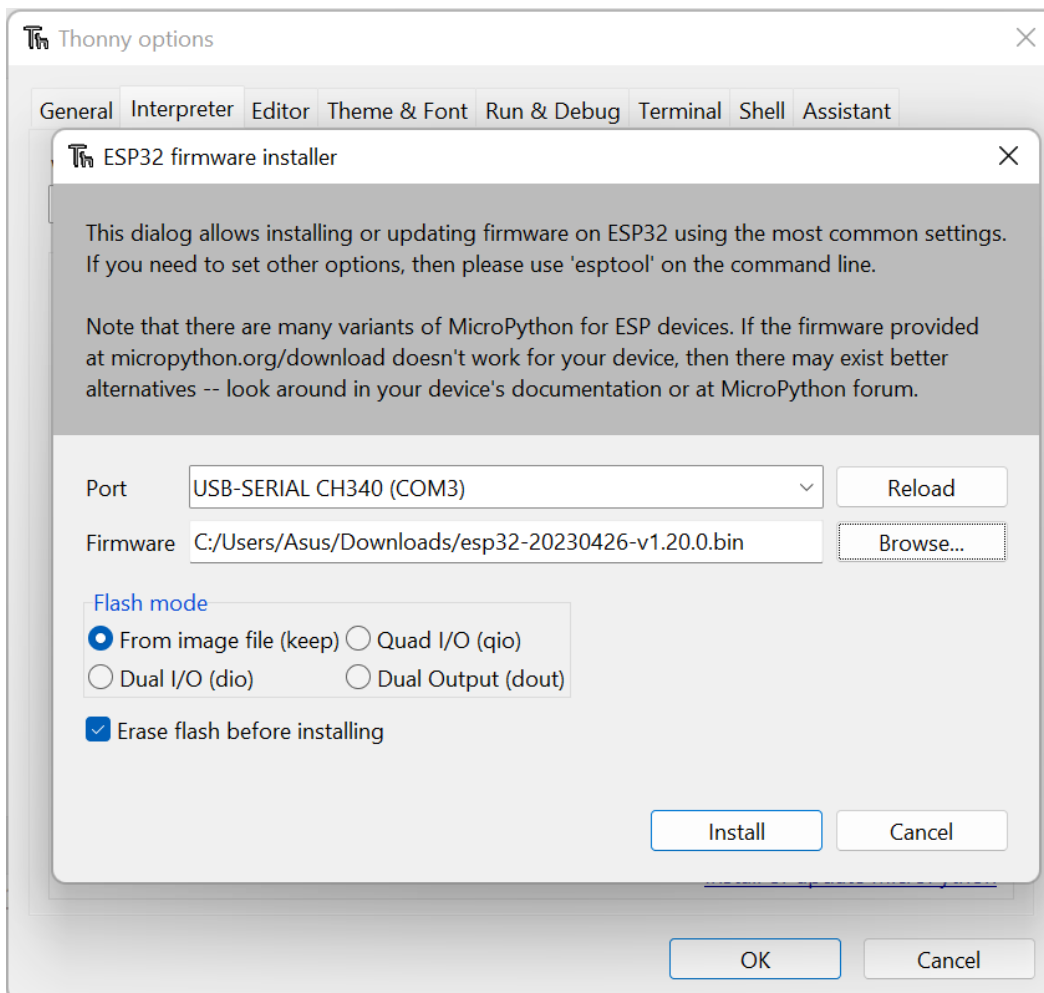
4. Klik kembali pada pojok kanan bawah, dan klik configure interpreter

5. Klik install or update pada bagian kanan bawah

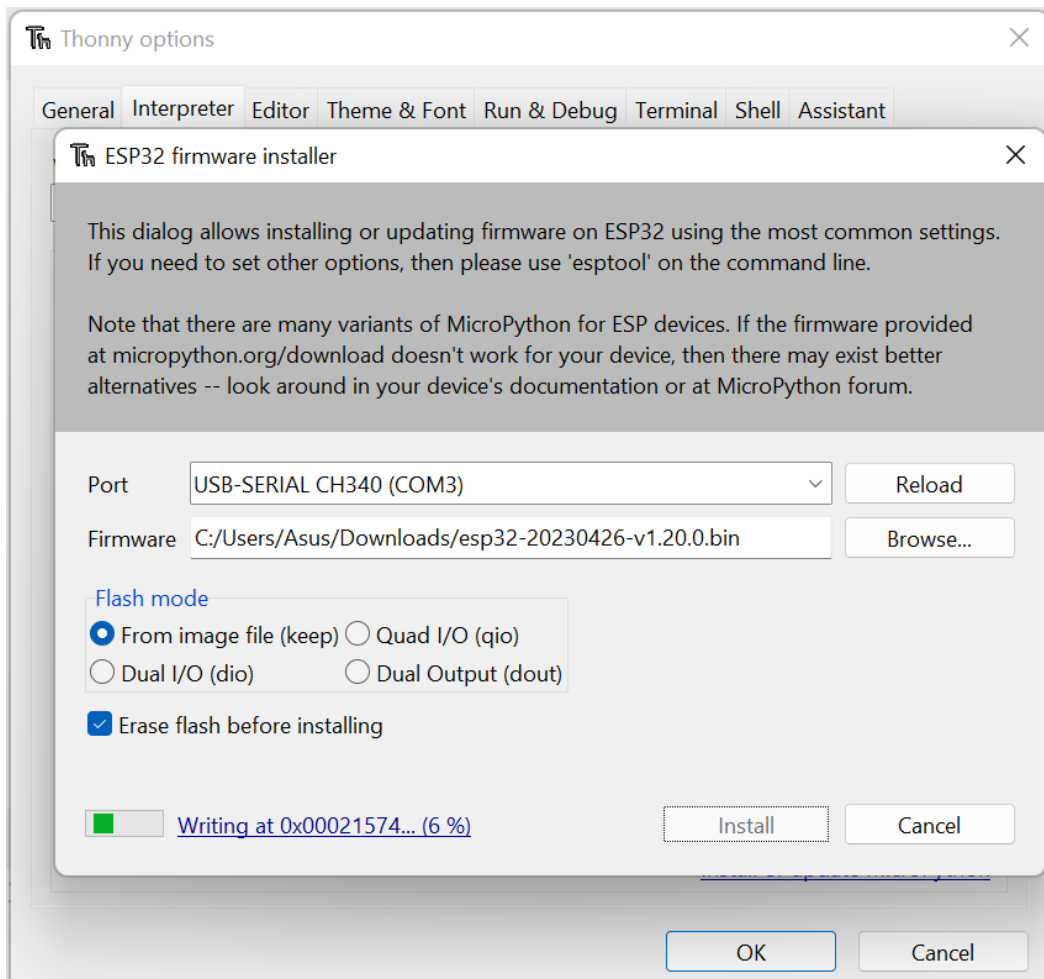


6. Pilih port yang terhubung pada ESP32, dan cari firmware yang telah di download sebelumnya kemudian klik install





7. Tunggu hingga proses instalasi selesai



GPIO

ATTENDANCE

Presensi Pertemuan 03

GPIO mempunyai kepanjangan *General Purpose Input/Output*, dan bertindak sebagai antar muka yang dapat digunakan secara universal. Setiap pin GPIO mempunyai nomor agar memudahkan penggunaanya, dan masing masing pin mempunyai fungsi yang berbeda beda

Ada 3 hal penting yang harus dipahami pada pin GPIO, antara lain

- Voltase tegangan : ada 2 buah pin 5V, dan 2 buah pin 3.3V yang dapat digunakan, dan beberapa pin untuk tujuan ground (0V).
- Output : pin GPIO di desain dengan nilai *high* adalah 3.3V dan *low* adalah 0V.
- Input : semua pin GPIO dapat digunakan untuk input atau output secara bergantian, jika bertindak sebagai pin input, maka pin tersebut dapat membaca nilai *high* adalah 3.3V hingga *low* adalah 0V. Sehingga GPIO dapat membaca nilai pada pull up resistor, atau pull down resistor.

contoh program GPIO

```
from machine import Pin

p0 = Pin(0, Pin.OUT)    # create output pin on GPIO0
p0.on()                 # set pin to "on" (high) level
p0.off()                # set pin to "off" (low) level
p0.value(1)             # set pin to on/high

p2 = Pin(2, Pin.IN)     # create input pin on GPIO2
print(p2.value())       # get value, 0 or 1

p4 = Pin(4, Pin.IN, Pin.PULL_UP) # enable internal pull-up resistor
p5 = Pin(5, Pin.OUT, value=1) # set pin high on creation
p6 = Pin(6, Pin.OUT, drive=Pin.DRIVE_3) # set maximum drive strength
```

catatan:

- Pin 1 dan 3 masing-masing adalah REPL UART TX dan RX
- Pin 6, 7, 8, 11, 16, dan 17 digunakan untuk menghubungkan embedded flash, dan tidak disarankan untuk penggunaan lain
- Pin 34-39 hanya input, dan juga tidak memiliki resistor pull-up internal

Timer

ATTENDANCE

Presensi Pertemuan 04



Pada dasarnya, timer adalah sebuah jam, yang digunakan untuk mengukur dan mengontrol peristiwa waktu. memberikan waktu tunda yang tepat. Sebagian besar mikrokontroler memiliki pengatur waktu bawaan. Timer pada mikrokontroler tidak hanya digunakan untuk menghasilkan waktu tunda tetapi juga digunakan sebagai pencacah. Karakteristik pengatur waktu ini digunakan untuk banyak aplikasi. Timer dalam mikrokontroler dikendalikan oleh register fungsi khusus yang ditugaskan untuk operasi timer.

ESP32 memiliki dua grup pengatur waktu, masing-masing dengan dua pengatur waktu perangkat keras untuk keperluan umum. Semua pengatur waktu didasarkan pada penghitung 64 bit dan anak prasekolah 16 bit. Prescaler digunakan untuk membagi frekuensi sinyal dasar (biasanya 80 MHz), yang kemudian digunakan untuk menambah atau mengurangi penghitung waktu.

ESP32 adalah mikrokontroler system-on-chip berdaya rendah populer yang memiliki periferal Timer bawaan. Periferal pengatur waktu dapat digunakan untuk berbagai tujuan seperti menghasilkan interupsi setelah interval waktu tertentu, menghitung kejadian, dan mengukur durasi kejadian.

ESP32 memiliki beberapa modul pengatur waktu, masing-masing dengan jumlah saluran dan kemampuan berbeda. Beberapa modul pengatur waktu umum di ESP32 meliputi:

1. Grup Timer 0 (TIMER0): Grup timer ini memiliki dua modul timer, masing-masing dengan dua saluran.

1. Grup Timer 1 (TIMER1): Grup timer ini memiliki satu modul timer dengan empat saluran.

Timer dapat digunakan dalam mode yang berbeda seperti one-shot, periodik, dan lainnya. Untuk menggunakan periferal pengatur waktu, Anda perlu mengonfigurasi modul pengatur waktu dan salurannya sesuai kebutuhan spesifik Anda. Ini dapat dilakukan dengan menggunakan pustaka ESP-IDF (Espressif IoT Development Framework) dan memprogram ESP32 menggunakan bahasa pemrograman C.

Contoh Program untuk menjalankan timer pada ESP32

```
from machine import Timer

tim0 = Timer(0)
tim0.init(period=5000, mode=Timer.ONE_SHOT, callback=lambda t:print(0))

tim1 = Timer(1)
tim1.init(period=2000, mode=Timer.PERIODIC, callback=lambda t:print(1))
```

UCP1: Timer

ATTENDANCE
Presensi Pertemuan 05

ASSIGNMENT
UCP 1



UCP 1 assessment:

1. Bring your own Laptop (50%)
2. Bring your own ESP32 share 1 esp max 2 students (10%)
3. Successfully install micropython firmware (15%)
4. Successfully execute the program using software loop (10%) OR using hardware Timer (25%)

Total 100%

UART and I2C

ATTENDANCE

Presensi Pertemuan 06

ASSIGNMENT

Remidial Exam (UCP1)

Serial (UART dan I2C)

Istilah "Serial" mengacu pada metode komunikasi yang memungkinkan data dikirimkan satu bit pada satu waktu melalui satu jalur komunikasi (atau jalur serial). Pada ESP32, antarmuka Serial digunakan untuk komunikasi dengan perangkat lain, seperti komputer, melalui antarmuka UART (Universal Asynchronous Receiver/Transmitter).

UART adalah protokol umum untuk komunikasi serial, yang memungkinkan komunikasi asinkron antar perangkat, artinya perangkat dapat mengirimkan data pada waktu yang berbeda dan pada kecepatan baud yang berbeda. ESP32 memiliki dua antarmuka UART, bernama UART0 dan UART1, yang dapat digunakan untuk komunikasi dengan perangkat lain.

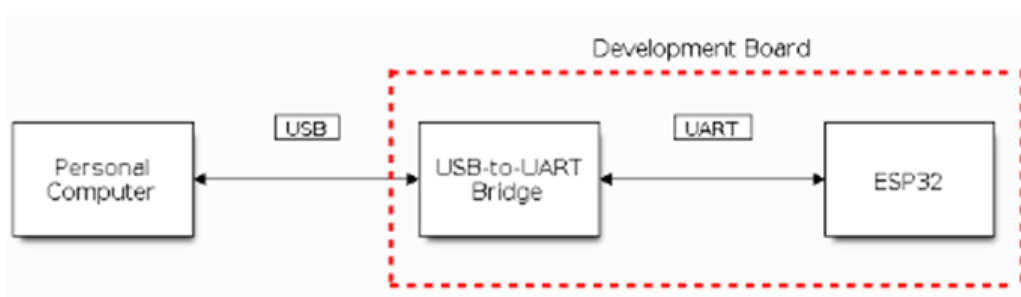
Untuk menggunakan antarmuka serial pada ESP32, Anda harus terlebih dahulu menginisialisasi objek serial dan menentukan kecepatan baud, yang merupakan jumlah bit yang dikirimkan per detik. Setelah objek serial diinisialisasi, Anda dapat menggunakan metode `Serial.println()` atau `Serial.print()` untuk mengirim data ke perangkat yang terhubung ke antarmuka serial, dan metode `Serial.read()` untuk menerima data dari perangkat.

Selain komunikasi dasar, antarmuka serial ESP32 juga dapat digunakan untuk keperluan debugging, memungkinkan Anda untuk melihat output dari ESP32 di terminal serial di komputer Anda. Ini dapat membantu untuk memecahkan masalah dan men-debug kode Anda.

Membuat koneksi serial dengan perangkat target ESP32 dapat dilakukan menggunakan jembatan USB-ke-UART. Beberapa board memasang jembatan USB-ke-UART. Jika board tidak memiliki jembatan maka jembatan eksternal dapat digunakan.

1. USB-to-UART Bridge pada Development Board

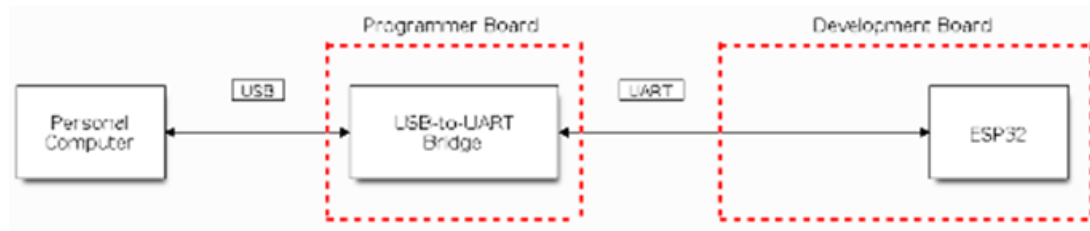
Untuk board dengan jembatan USB-ke-UART yang terpasang, koneksi antara komputer pribadi dan jembatan adalah USB dan antara jembatan dan ESP32 adalah UART.



2. External USB-to-UART Bridge



Terkadang jembatan USB-to-UART bersifat eksternal. Ini sering digunakan di papan pengembangan kecil atau produk jadi ketika ruang dan biaya sangat penting.



Contoh program UART (serial bus)

```
from machine import UART

uart1 = UART(1, baudrate=9600, tx=33, rx=32)
uart1.write('hello') # write 5 bytes
uart1.read(5)        # read up to 5 bytes
```

I2C (Inter-Integrated Circuit) adalah protokol komunikasi yang memungkinkan banyak perangkat berkomunikasi dengan satu perangkat master hanya menggunakan dua kabel, satu untuk jam (SCL) dan satu untuk data (SDA). Ini biasanya digunakan dalam aplikasi mikrokontroler untuk berkomunikasi dengan berbagai perangkat periferil seperti sensor, layar, dan chip memori. I2C adalah protokol komunikasi serial, sinkron, setengah dupleks yang memungkinkan koeksistensi beberapa master dan slave di bus yang sama. Bus I2C terdiri dari dua jalur: jalur data serial (SDA) dan jam serial (SCL). Kedua saluran membutuhkan resistor pull-up.

I2C beroperasi dalam mode master-slave, di mana perangkat master mengontrol komunikasi dan memulai transaksi dengan perangkat slave. Master menghasilkan sinyal clock dan mengontrol aliran data pada jalur SDA. Perangkat budak menanggapi permintaan master dan mengirimkan data ke master. Setiap perangkat pada bus I2C memiliki alamat yang unik, sehingga master dapat berkomunikasi dengan perangkat tertentu, atau dengan semua perangkat secara bersamaan dengan menggunakan alamat broadcast.

I2C dikenal dengan kesederhanaan dan jumlah pin yang rendah, menjadikannya pilihan yang menarik untuk sistem tertanam di mana sumber daya perangkat keras terbatas. Selain itu, bus I2C dapat mendukung banyak perangkat master, sehingga berguna untuk sistem kompleks di mana banyak mikrokontroler perlu berkomunikasi satu sama lain dan dengan perangkat periferil.

ESP32 memiliki 2 pengontrol I2C (juga disebut sebagai port), yang bertanggung jawab untuk menangani komunikasi pada bus I2C. Pengontrol I2C tunggal dapat beroperasi sebagai master atau slave.

Contoh program untuk SoftI2C

```
from machine import Pin, SoftI2C

i2c = SoftI2C(scl=Pin(5), sda=Pin(4), freq=100000)

i2c.scan() # scan for devices

i2c.readfrom(0x3a, 4) # read 4 bytes from device with address 0x3a
i2c.writeto(0x3a, '12') # write '12' to device with address 0x3a

buf = bytearray(10) # create a buffer with 10 bytes
i2c.writeto(0x3a, buf) # write the given buffer to the peripheral
```



Timer

Pada dasarnya, timer adalah sebuah jam, yang digunakan untuk mengukur dan mengontrol peristiwa waktu. memberikan waktu tunda yang tepat. Sebagian besar mikrokontroler memiliki pengatur waktu bawaan. Timer pada mikrokontroler tidak hanya digunakan untuk menghasilkan waktu tunda tetapi juga digunakan sebagai pencacah. Karakteristik pengatur waktu ini digunakan untuk banyak aplikasi. Timer dalam mikrokontroler dikendalikan oleh register fungsi khusus yang ditugaskan untuk operasi timer.

ESP32 memiliki dua grup pengatur waktu, masing-masing dengan dua pengatur waktu perangkat keras untuk keperluan umum. Semua pengatur waktu didasarkan pada penghitung 64 bit dan anak prasekolah 16 bit. Prescaler digunakan untuk membagi frekuensi sinyal dasar (biasanya 80 MHz), yang kemudian digunakan untuk menambah atau mengurangi penghitung waktu.

ESP32 adalah mikrokontroler system-on-chip berdaya rendah populer yang memiliki periferal Timer bawaan. Periferal pengatur waktu dapat digunakan untuk berbagai tujuan seperti menghasilkan interupsi setelah interval waktu tertentu, menghitung kejadian, dan mengukur durasi kejadian.

ESP32 memiliki beberapa modul pengatur waktu, masing-masing dengan jumlah saluran dan kemampuan berbeda. Beberapa modul pengatur waktu umum di ESP32 meliputi:

1. Grup Timer 0 (TIMER0): Grup timer ini memiliki dua modul timer, masing-masing dengan dua saluran.

1. Grup Timer 1 (TIMER1): Grup timer ini memiliki satu modul timer dengan empat saluran.

Timer dapat digunakan dalam mode yang berbeda seperti one-shot, periodik, dan lainnya. Untuk menggunakan periferal pengatur waktu, Anda perlu mengonfigurasi modul pengatur waktu dan salurannya sesuai kebutuhan spesifik Anda. Ini dapat dilakukan dengan menggunakan pustaka ESP-IDF (Espressif IoT Development Framework) dan memprogram ESP32 menggunakan bahasa pemrograman C.

Contoh Program untuk menjalankan timer pada ESP32

```
from machine import Timer

tim0 = Timer(0)
tim0.init(period=5000, mode=Timer.ONE_SHOT, callback=lambda t:print(0))

tim1 = Timer(1)
tim1.init(period=2000, mode=Timer.PERIODIC, callback=lambda t:print(1))
```

Midterm Exam



TIK untuk Pembangunan Indonesia Maju dan Berkelanjutan

GEMASTIK 2023

Pendaftaran PT dan Tim, Sinkron Data, Unggah Berkas Submisi Tahap Seleksi Nasional **Telah Dibuka** dan Akan Berakhir dalam

56 : **11** : **28** : **57**
Days Hours Minutes Second

Daftar Tim / PT

Panduan Alur Sistem Lomba



Membuat kelompok untuk proyek akhir (3 orang) terkait aplikasi Mikroprosesor. Setiap mahasiswa menulis 1 proposal meskipun proyek berkelompok. Tulis menggunakan MS Word dalam kertas A4 minimal 5 halaman maksimal 5 halaman. Format adalah sebagai berikut:

1. Nama kelompok dan anggota
2. Tema: (a) Kota Pintar atau (b) Piranti Cerdas, Sistem Benam & IoT
3. Latar belakang
4. Studi literature minimal 5 jurnal/conference
5. Rancangan sistem (arsitektur)
6. Hasil yang diharapkan

Proyek ini akan menjadi proyek akhir Anda dalam kursus ini. Pastikan untuk melengkapi dan menyerahkan makalah sebelum 21 Mei 2023.

===== translation =====

Create a group for the final task (3 people) related to microprocessor systems. Each student writes 1 report even though it is a group project. Write down your proposal on A4 paper, minimum 5 pages, maximum 5 pages using MS Word. The format is as follows:

1. Group name and members
2. Tema: (a) Smart City or Smart Devices, (b) Embedded System, and IoT
3. Background
4. Literature study of at least 5 papers (journals/conferences)
5. System design (architecture)
6. Expected results

This project will be your final project in this course. Make sure to complete and submit the paper before Mei 21, 2023.

Opened: Wednesday, 17 May 2023, 12:56 PM

Due: Sunday, 21 May 2023, 11:59 PM



Analog to Digital Conversion

ADC adalah singkatan dari Analog-to-Digital Converter. Ini adalah komponen elektronik yang mengubah sinyal analog, seperti tegangan atau arus, menjadi representasi digital yang dapat diproses oleh sistem digital, seperti mikrokontroler atau komputer.

Sinyal analog kontinu dan dapat mengambil nilai apa pun dalam rentang, sedangkan sinyal digital bersifat diskrit dan hanya dapat mengambil nilai tertentu. ADC mengambil sinyal analog sebagai input dan output sinyal digital yang mewakili amplitudo sinyal input pada titik waktu tertentu. Keakuratan dan resolusi ADC bergantung pada jumlah bit yang digunakan untuk mewakili sinyal digital. Semakin banyak bit yang digunakan ADC, semakin tinggi resolusi dan akurasinya. ADC merupakan komponen penting dalam banyak aplikasi yang memerlukan konversi sinyal analog menjadi data digital untuk diproses, disimpan, atau ditransmisikan.

Pada ESP32, fungsionalitas ADC tersedia pada pin 32-39 (blok ADC 1) dan pin 0, 2, 4, 12-15 dan 25-27 (blok ADC 2).

contoh program untuk membaca nilai ADC

```
from machine import ADC

adc = ADC(pin)          # create an ADC object acting on a pin
val = adc.read_u16()    # read a raw analog value in the range 0-65535
val = adc.read_uv()     # read an analog value in microvolts
```

Tegangan referensi ADC internal biasanya 1,1V, tetapi sedikit berbeda dari paket ke paket. ADC kurang linier dekat dengan tegangan referensi (terutama pada atenuasi yang lebih tinggi) dan memiliki tegangan pengukuran minimum sekitar 100mV, tegangan pada atau di bawah ini akan dibaca sebagai 0.

One Wire

One Wire

OneWire adalah protokol komunikasi yang digunakan untuk menghubungkan beberapa perangkat elektronik dengan satu kabel saja. Protokol ini dikembangkan oleh perusahaan Dallas Semiconductor (sekarang dikenal sebagai Maxim Integrated) dan digunakan dalam berbagai aplikasi seperti pengukuran suhu, pengukuran kelembaban, dan chip memori EEPROM.

Komunikasi OneWire dilakukan melalui satu kabel sinyal dan satu kabel ground (tanah), dimana kabel sinyal digunakan untuk transmisi data dan juga sebagai sumber daya listrik. Setiap perangkat yang menggunakan protokol OneWire memiliki alamat unik berupa 64 bit yang digunakan untuk mengidentifikasi perangkat tersebut di dalam jaringan.

Perangkat OneWire dapat beroperasi dalam mode daya parasit, di mana perangkat tersebut mengambil daya dari sinyal data, bukan dari sumber daya listrik terpisah. Hal ini dilakukan dengan menggunakan kapasitor untuk menyimpan muatan selama sinyal data dalam keadaan rendah, dan kemudian menggunakan muatan tersebut untuk memberikan daya pada perangkat ketika sinyal data dalam keadaan tinggi.



Protokol OneWire menggunakan arsitektur master-slave, di mana perangkat master memulai komunikasi dengan perangkat slave. Perangkat master mengirimkan perintah dan data ke perangkat slave, dan perangkat slave merespon dengan mengirimkan data. Komunikasi biasanya dilakukan dengan kecepatan yang rendah, yang membuatnya cocok untuk aplikasi yang hanya membutuhkan transmisi data kecil.

Secara keseluruhan, protokol OneWire adalah cara yang sederhana dan efektif untuk berkomunikasi dengan perangkat yang memiliki kebutuhan daya yang rendah dan beroperasi di lingkungan yang keras di mana beberapa kabel tidak praktis.

Driver OneWire diimplementasikan dalam perangkat lunak dan berfungsi di semua pin:

```
from machine import Pin
import onewire

ow = onewire.OneWire(Pin(12)) # create a OneWire bus on GPIO12
ow.scan()                    # return a list of devices on the bus
ow.reset()                   # reset the bus
ow.readbyte()                # read a byte
ow.writebyte(0x12)           # write a byte on the bus
ow.write('123')              # write bytes on the bus
ow.select_rom(b'12345678') # select a specific device by its ROM code
```

PWM

ATTENDANCE

Presensi Pertemuan 11

PWM

PWM (Pulse Width Modulation) yang merupakan teknik modulasi sinyal analog di mana lebar pulsa dari sinyal digital diubah untuk mengendalikan daya yang diberikan ke sebuah perangkat atau sistem.

Pada dasarnya, PWM adalah sebuah sinyal digital yang berosilasi antara keadaan ON (aktif) dan OFF (tidak aktif), dengan frekuensi tetap, namun dengan lebar pulsa yang dapat diubah-ubah sesuai dengan kebutuhan. Lebar pulsa ini ditentukan sebagai persentase siklus kerja (duty cycle), yang merupakan rasio antara durasi sinyal ON dan durasi siklus sinyal. Jika siklus kerja adalah 50%, artinya lebar pulsa adalah setengah dari durasi siklus sinyal.

Keuntungan utama dari PWM adalah efisiensi energi yang lebih baik dibandingkan dengan teknik modulasi sinyal lainnya, karena daya yang diberikan ke perangkat atau sistem dapat diatur dengan presisi tinggi. Selain itu, PWM juga dapat menghasilkan sinyal analog yang halus dengan menggunakan filter yang tepat.

PWM dapat diaktifkan pada semua pin yang mendukung keluaran. Frekuensi dasar dapat berkisar dari 1Hz hingga 40MHz tetapi ada tradeoff (sebagai frekuensi dasar meningkatkan resolusi tugas menurun).




```

from machine import Pin, PWM

pwm0 = PWM(Pin(0), freq=5000, duty_u16=32768) # create PWM object from a pin
freq = pwm0.freq() # get current frequency
pwm0.freq(1000) # set PWM frequency from 1Hz to 40MHz

duty = pwm0.duty() # get current duty cycle, range 0-1023 (default 512, 50%)
pwm0.duty(256) # set duty cycle from 0 to 1023 as a ratio duty/1023, (now 25%)

duty_u16 = pwm0.duty_u16() # get current duty cycle, range 0-65535
pwm0.duty_u16(2**16*3//4) # set duty cycle from 0 to 65535 as a ratio duty_u16/65535, (now 75%)

duty_ns = pwm0.duty_ns() # get current pulse width in ns
pwm0.duty_ns(250_000) # set pulse width in nanoseconds from 0 to 1_000_000_000/freq, (now 25%)

pwm0.deinit() # turn off PWM on the pin

pwm2 = PWM(Pin(2), freq=20000, duty=512) # create and configure in one go
print(pwm2) # view PWM settings

```

asd

LCD

ATTENDANCE

Presensi Pertemuan 12

LCD (Liquid Crystal Display)

Merupakan tampilan visual yang menggunakan teknologi cairan kristal untuk menampilkan gambar atau teks. LCD sering digunakan dalam berbagai perangkat elektronik seperti telepon seluler, laptop, monitor komputer, televisi, dan kalkulator.

Pada dasarnya, sebuah LCD terdiri dari dua panel kaca yang ditempatkan di atas satu sama lain, dengan lapisan cairan kristal di antaranya. Ketika listrik diberikan ke lapisan kristal, kristal akan berubah bentuk dan memungkinkan cahaya untuk melewati panel.

Untuk mengontrol tampilan pada LCD, umumnya digunakan mikrokontroler atau mikroprosesor yang terhubung ke LCD melalui interface seperti I2C atau SPI. Ada banyak jenis dan ukuran LCD yang tersedia, dari yang hanya dapat menampilkan beberapa karakter hingga yang mampu menampilkan gambar berwarna dengan resolusi tinggi.

Beberapa jenis LCD juga dilengkapi dengan backlight, yang memberikan cahaya latar belakang pada tampilan LCD sehingga dapat terlihat lebih jelas dalam kondisi cahaya rendah. Backlight biasanya diatur dengan menggunakan sumber daya listrik terpisah dan dapat dikontrol dengan transistor atau MOSFET.

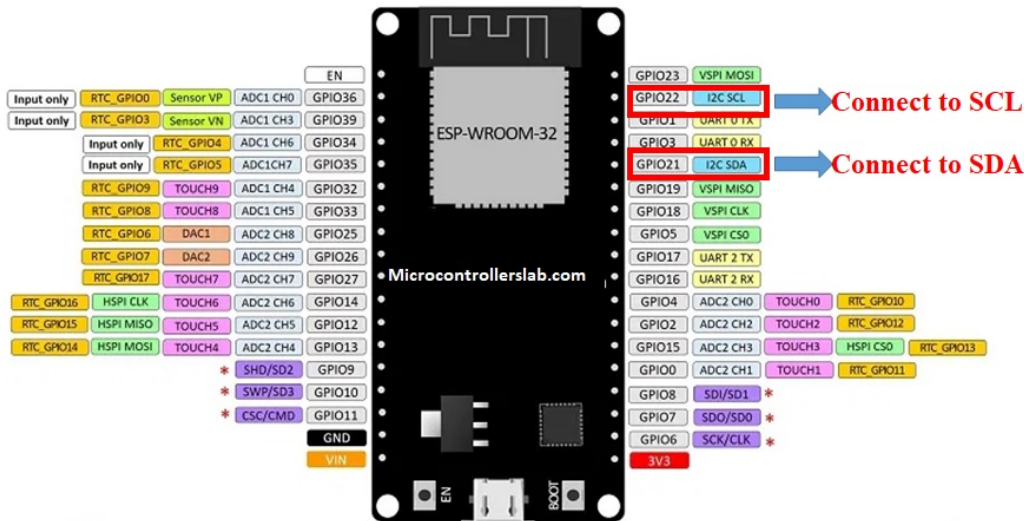
LCD sering digunakan dalam aplikasi yang membutuhkan tampilan teks atau angka yang sederhana dan mudah dibaca, seperti pengukur suhu atau voltase, jam digital, atau alat pengukur yang lain. Namun, dengan teknologi terbaru, LCD juga dapat menampilkan gambar dan video dengan kualitas yang tinggi, dan digunakan dalam aplikasi yang lebih kompleks seperti televisi atau monitor komputer.

Untuk mengkoneksikan LCD dengan ESP32 dapat dilihat pada gambar dibawah



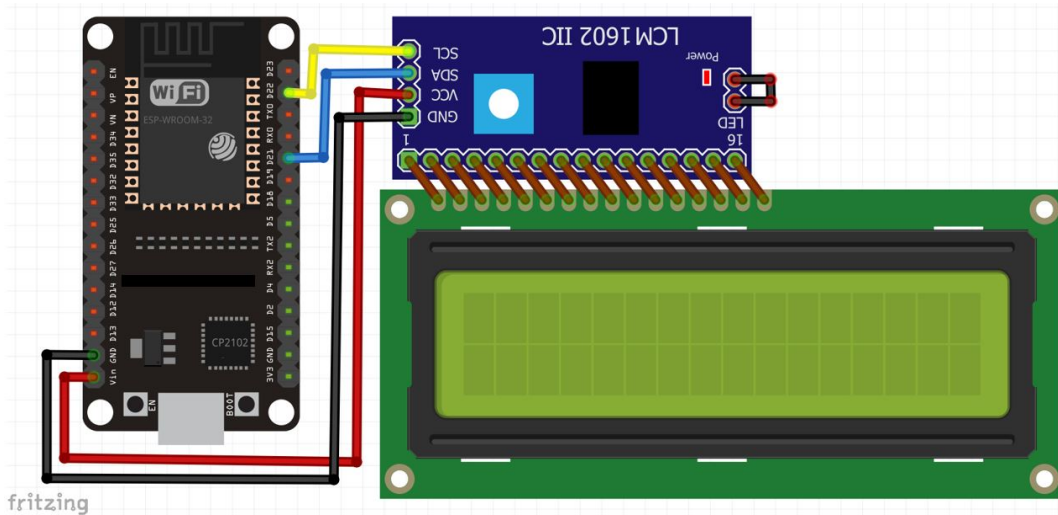
ESP32 DEVKIT V1 – DOIT

version with 36 GPIOs



* Pins SCK/CLK, SDO/SD0, SDI/SD1, SHD/SD2, SWP/SD3 and CSC/CMD, namely, GPIO6 to GPIO11 are connected to the integrated SPI flash integrated on ESP-WROOM-32 and are not recommended for other uses.

sambungkan perangkat ESP32 dengan LCD I2C seperti yang ditunjukkan pada diagram skematik di bawah ini:



Program ini akan memindai perangkat I2C yang terhubung dengan ESP32 dan akan menentukan jumlah perangkat dengan alamat di konsol shell.

```
import machine
sdaPIN=machine.Pin(21) #for ESP32
sclPIN=machine.Pin(22)
i2c=machine.I2C(sda=sdaPIN, scl=sclPIN, freq=10000)
devices = i2c.scan()
if len(devices) == 0:
    print("No i2c device !")
else:
    print('i2c devices found:',len(devices))
for device in devices:
    print("At address: ",hex(device))
```



```
Shell x
>>> %Run -c $EDITOR_CONTENT
i2c devices found: 1
At address: 0x27
>>>
```

Kemudian untuk menampilkan teks pad LCD dapat menggunakan contoh program dibawah ini:

```
import machine
```

```
from machine import Pin, SoftI2C
```

```
from lcd_api import LcdApi
```

```
from i2c_lcd import I2cLcd
```

```
from time import sleep
```

```
I2C_ADDR = 0x27
```

```
totalRows = 2
```

```
totalColumns = 16
```

```
i2c = SoftI2C(scl=Pin(22), sda=Pin(21), freq=10000) #initializing the I2C method for ESP32
```

```
#i2c = I2C(scl=Pin(5), sda=Pin(4), freq=10000) #initializing the I2C method for ESP8266
```

```
lcd = I2cLcd(i2c, I2C_ADDR, totalRows, totalColumns)
```

```
while True:
```

```
    lcd.putstr("Halo")
```

```
    sleep(2)
```

```
    lcd.clear()
```

```
    lcd.putstr("Ayo Hitung 0-10!")
```

```
    sleep(2)
```

```
    lcd.clear()
```

```
    for i in range(11):
```

```
        lcd.putstr(str(i))
```

```
        sleep(1)
```

```
        lcd.clear()
```



SPI (Serial Peripheral Interface)

SPI (Serial Peripheral Interface) adalah antarmuka komunikasi serial sinkron yang biasa digunakan untuk komunikasi jarak pendek antara mikrokontroler dan perangkat periferal, seperti sensor, layar, dan perangkat memori. Mikrokontroler ESP32 memiliki dukungan untuk komunikasi SPI, yang membuatnya mudah untuk berinteraksi dengan berbagai perangkat SPI.

Untuk menggunakan antarmuka SPI pada ESP32, Anda harus terlebih dahulu mengkonfigurasi pin SPI dan mengatur parameter ESP32 mendukung hingga 3 antarmuka SPI, berlabel SPI, HSPI, dan VSPI. Setiap antarmuka dapat dikonfigurasi dengan pin dan pengaturan yang berbeda.

Software SPI (menggunakan bit-banging) bekerja pada semua pin, dan diakses melalui class `machine.SoftSPI`:

```
from machine import Pin, SoftSPI

# construct a SoftSPI bus on the given pins
# polarity is the idle state of SCK
# phase=0 means sample on the first edge of SCK, phase=1 means the second
spi = SoftSPI(baudrate=100000, polarity=1, phase=0, sck=Pin(0), mosi=Pin(2), miso=Pin(4))

spi.init(baudrate=200000) # set the baudrate

spi.read(10)           # read 10 bytes on MISO
spi.read(10, 0xff)     # read 10 bytes while outputting 0xff on MOSI

buf = bytearray(50)    # create a buffer
spi.readinto(buf)      # read into the given buffer (reads 50 bytes in this case)
spi.readinto(buf, 0xff) # read into the given buffer and output 0xff on MOSI

spi.write(b'12345')    # write 5 bytes on MOSI

buf = bytearray(4)     # create a buffer
spi.write_readinto(b'1234', buf) # write to MOSI and read from MISO into the buffer
spi.write_readinto(buf, buf) # write buf to MOSI and read MISO back into buf
```

Ada dua saluran SPI perangkat keras yang memungkinkan laju transmisi lebih cepat (hingga 80MHz). Hal ini dapat digunakan untuk apa pun yang mendukung arah yang diperlukan dan sebaliknya tidak digunakan (lihat Pin dan GPIO) tetapi jika pin tersebut dikonfigurasi ke pin defaultnya, maka pin tersebut harus melewati lapisan tambahan multiplexing GPIO, yang dapat memengaruhi kecepatan tersebut. Keandalan pada kecepatan tinggi, saluran SPI perangkat keras dibatasi hingga 40MHz bila digunakan pada pin selanjutnya yang tercantum di bawah ini

	HSPI (id=1)	VSPI (id=2)
sck	14	18
mosi	13	23
miso	12	19

Hardware SPI dapat diakses melalui class `machine.SPI` dan memiliki metode yang sama dengan perangkat lunak SPI di atas:



```
from machine import Pin, SPI
```

```
hspi = SPI(1, 10000000)
```

```
hspi = SPI(1, 10000000, sck=Pin(14), mosi=Pin(13), miso=Pin(12))
```

```
vspi = SPI(2, baudrate=80000000, polarity=0, phase=0, bits=8, firstbit=0, sck=Pin(18), mosi=Pin(23), miso=Pin(19))
```

ASSIGNMENT

UCP 3 - Demo OLED Display

Bluetooth

ATTENDANCE

Presensi Pertemuan 14

Mikrokontroler ESP32 memiliki dukungan bawaan untuk konektivitas Bluetooth, membuatnya mudah untuk berinteraksi dengan perangkat Bluetooth. ESP32 mendukung protokol Bluetooth klasik dan Bluetooth Low Energy (BLE).

Untuk menggunakan Bluetooth di ESP32, maka yang harus dilakukan yaitu mengonfigurasi tumpukan Bluetooth terlebih dahulu, lalu menggunakan API yang sesuai untuk berinteraksi dengan perangkat Bluetooth.

Modul ini menyediakan antarmuka ke pengontrol Bluetooth pada board. Saat ini ESP32 mendukung Bluetooth Low Energy (BLE) dalam peran Central, Peripheral, Broadcaster, dan Observer, serta GATT Server and Client dan L2CAP connection-oriented-channels. Perangkat dapat beroperasi dalam beberapa peran secara bersamaan. Pairing (dan bonding) didukung pada beberapa port.

API ini dimaksudkan untuk mencocokkan protokol Bluetooth tingkat rendah dan menyediakan blok penyusun untuk abstraksi tingkat lebih tinggi seperti jenis perangkat tertentu.

Microcontroller Applications in Medical Field

ATTENDANCE

Presensi Pertemuan 15

Mikrokontroler adalah perangkat mandiri yang mencakup CPU, ROM, RAM, sirkuit jam, dan sirkuit I/O dalam satu paket sirkuit terpadu (IC). Mikrokontroler tidak memerlukan chip pendukung yang menyertainya untuk operasinya seperti yang dilakukan mikroprosesor konvensional.

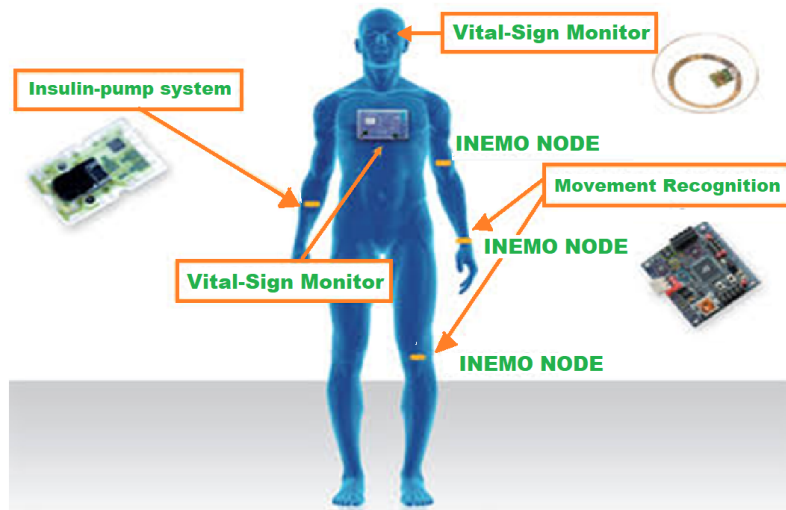
Penggunaan mikrokontroler umumnya menghasilkan lebih sedikit paket IC sehingga mengurangi kompleksitas. Jadi ini menawarkan keunggulan biaya dan ruang. Selain itu, kemampuan mikrokontroler untuk mengkalibrasi sendiri dan mendeteksi kesalahan meningkatkan keandalan instrumen biomedis.



Mikrokontroler dapat menyediakan kalibrasi sendiri untuk sistem pengukuran biomedis, urutan kejadian otomatis, dan cara mudah untuk memasukkan data pasien seperti berat badan, tinggi badan, dll. untuk menghitung kinerja yang diharapkan atau normal. Semua fungsi ini dimungkinkan oleh struktur prinsip sistem mikrokontroler.

Mikrokontroler yang merupakan sistem yang berdiri sendiri untuk aplikasi dalam sistem akuisisi data dan kontrol memiliki konverter analog-ke-digital pada chip, yang memungkinkannya digunakan langsung dalam instrumentasi biomedis.

Applications of Embedded Systems in The Medical Field



www.TheEngineeringProjects.com

Final Exam (Project)

ATTENDANCE

Presensi Pertemuan 16

QUIZ

UCP4

Opened: Friday, 7 July 2023, 8:00 AM


Closed: Friday, 7 July 2023, 11:55 PM





Navigation

Dashboard

 Site home

Site pages

My courses

EE4402P

EE6604

EE4406P

EE4405

Participants

 Competencies

 Grades

General

Introduction and Kontrak Belajar

Installing Thonny and Micropython

GPIO

Timer

UCP1: Timer

UART and I2C

Timer

Midterm Exam

ADC

One Wire

PWM

LCD

UCP 3 - SPI

Bluetooth

Microcontroller Applications in Medical Field

Final Exam (Project)

EE4404

EE4403

EE4402

EE4401

BS2205



