

**LAPORAN PRAKTIKUM TUGAS AKHIR
ELEKTRONIKA MESIN LISTRIK DAN TEKNIK KENDALI
“TINYML-MOTION RECOGNITION USING RASPBERRY PI
PICO”**

Dosen Pengampu: Irfan Bahiuddin, ST, M.Phil., Ph.D.



Disusun Oleh:

Kelompok 1

Catur Wardana (19/441194/SV/16546)

Santi Rahayu (19/441209/SV/16561)

Yeyen Karunia (19/441215/SV/16567)

Aditya Bayu Maulana (19/447106/SV/16825)

Kelas: ARM 2

TEKNOLOGI REKAYASA MESIN

DEPARTEMEN TEKNIK MESIN SEKOLAH VOKASI

UNIVERSITAS GADJAH MADA

YOGYAKARTA

2022

KATA PENGANTAR

Puji Syukur kami panjatkan kehadirat Allah SWT karena atas berkat rahmat dan karunia-Nya Laporan Akhir Praktikum Elektronika Mesin Listrik dan Teknik Kendali ini dapat terselesaikan dengan baik tepat pada waktunya. Adapun tujuan penulisan laporan ini adalah untuk memenuhi tugas akhir kelompok Praktikum Elektronika Mesin Listrik dan Teknik Kendali Semester VI Tahun Pelajaran 2021/2022. Dalam penyelesaian laporan ini, kami banyak mengalami kesulitan, terutama disebabkan oleh kurangnya ilmu pengetahuan yang menunjang. Namun, berkat bimbingan dan bantuan dari berbagai pihak, akhirnya laporan ini dapat terselesaikan dengan cukup baik. Karena itu sudah sepantasnya jika kami mengucapkan terima kasih kepada:

1. Dosen Pembimbing dan Instruktur Praktikum Elektronika Mesin Listrik dan Teknik Kendali.
2. Semua pihak yang telah ikut membantu hingga tahap laporan praktikum ini selesai yang tidak dapat penulis sebutkan satu per satu.

Kami sadar sebagai seorang mahasiswa yang masih dalam proses pembelajaran, penulisan laporan ini masih banyak kekurangannya. Oleh karena itu, kami sangat mengharapkan adanya kritik dan saran yang bersifat positif, guna penulisan laporan yang lebih baik lagi di masa yang akan datang.

Yogyakarta, 28 Mei 2022

Penyusun

BAB I. PENDAHULUAN

1.1 Deskripsi Kasus

‘TinyML - Motion Recognition using Raspberry Pi Pico’ Motion Recognition pada kasus ini adalah sebuah klasifikasi gerakan dengan arah gerakan secara 3 dimensi gerakan sumbu X, sumbu Y, dan sumbu Z. Data tersebut akan diklasifikasikan berdasarkan gerakan yang akan kita tentukan menggunakan *Machine Learning*.

1.2 Latar Belakang

Ketika kita berpikir tentang deep learning dan machine learning, umumnya hal tersebut dimaksudkan untuk CPU/GPU yang lebih cepat dengan sejumlah RAM yang besar. Maka dari itu, ditinjau dari sudut pandang ini, implementasi algoritma ML dalam mikrokontroler dengan memori beberapa kb dan kecepatan clock dalam puluhan MHz akan terdengar aneh. Tapi, tetap hal tersebut memiliki kemungkinan untuk terjadi. Dalam program ini, kita dapat menggunakan sebuah website *Tensorflow* versi lite yang mungkinkan untuk mengonversi model jaringan saraf besar ke versi lite yang memerlukan beberapa kb ruang memori dan MCU.

Seri Cortex-M memiliki spesifikasi yang cukup untuk menjalankan model. (Catatan: Tensorflow-lite versi saat ini tidak dapat mengonversi semua model menjadi versi lite, ada beberapa batasan juga). Meskipun model ML di mikrokontroler memiliki banyak keterbatasan, model ini cocok untuk perangkat portabel, perangkat elektronik wearable yang dapat dioperasikan dengan mudah menggunakan baterai berukuran koin. Selain itu, karena model ML diimplementasikan secara lokal, di dalam MCU dan tidak perlu cloud interface. Hal ini sangat mengurangi latensi dan meningkatkan keamanan data, dalam kasus ini program akan dijalankan pada perangkat portable Raspberry PI PICO.

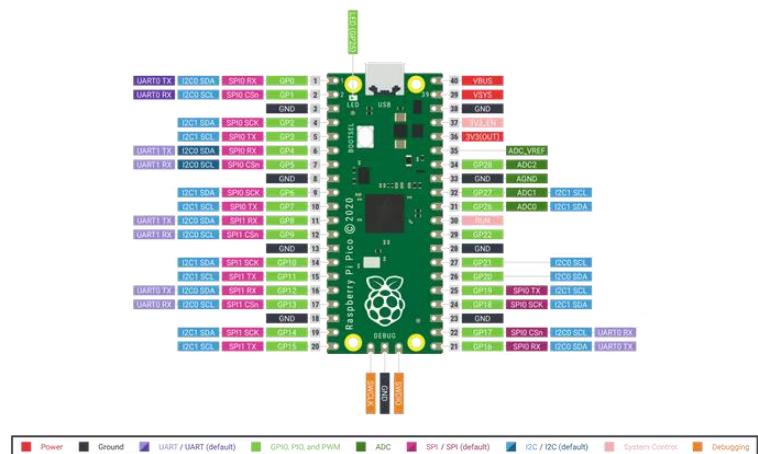
BAB II. PENDAHULUAN

2.1 RASPBERRY PI PICO

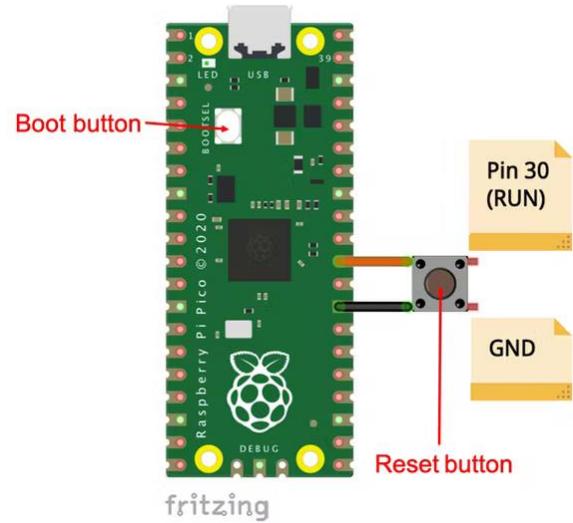
Raspberry Pi Pico adalah papan mikrokontroler berbiaya rendah dan berkinerja tinggi dengan flexible digital interfaces. Fitur utama meliputi:

- RP2040 microcontroller chip designed by Raspberry Pi Foundation

- Dual-core Arm Cortex M0+ processor, flexible clock running up to 133 MHz
- 264KB of SRAM, and 2MB of on-board Flash memory
- USB 1.1 with device and host support
- Low-power sleep and dormant modes
- 26 × multi-function GPIO pins
- 2 × SPI, 2 × I2C, 2 × UART, 3 × 12-bit ADC, 16 × controllable PWM channels
- Accurate clock and timer on-chip
- Temperature sensor
- Accelerated floating-point libraries on-chip
- 8 × Programmable I/O (PIO) state machines for custom peripheral support



menekan kedua tombol secara bersamaan dan program tersebut akan terunggah.

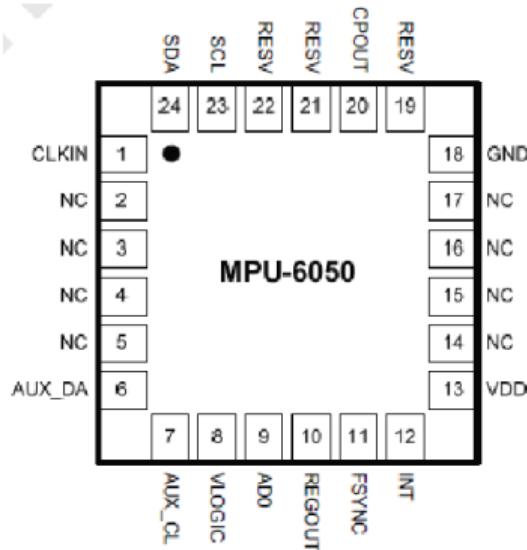


Gambar 2. Button pada Raspberry Pi Pico

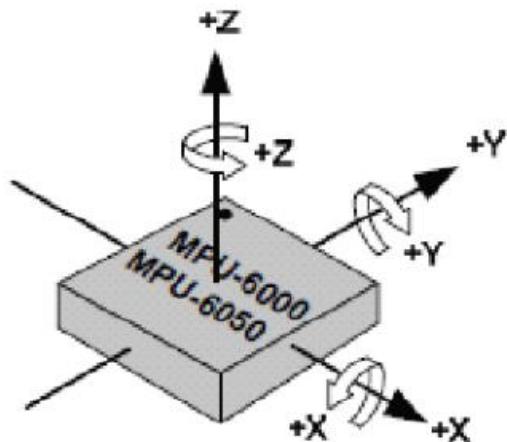
2.2 MCU 6050

Diagram pin MPU6050 ditunjukkan pada gambar 1. Gambar pin suplai VDD disuplai oleh 3.3V dimana CS merupakan sinyal pilih chip (chip select signal). Transfer data dilakukan dengan menggunakan SPI atau IIC interface.

Gambar 2 adalah tampilan tiga dimensi dari MPU6050 untuk deteksi sudut. MPU6050 memiliki 16 bit ADC yang berguna untuk mengumpulkan nilai percepatan 3 axis atau nilai gyro yang kemudian diubah menjadi output digital. Rentang pengukuran giroskop adalah plus atau minus 250 derajat, plus atau minus 500 derajat, plus atau minus 1000 derajat dan plus atau minus 2000 derajat serta rentang pengukuran akselerometer adalah + 2G, + 4G, + 8g, + 16g (Jian, H., 2016).



Gambar 3. MPU6050 Pins



Gambar 4. Tiga Axis Test Chart

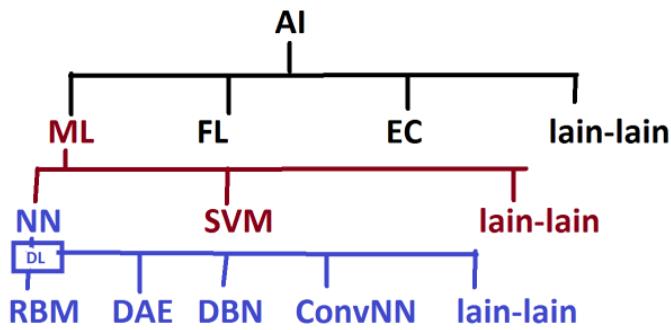
Sementara menurut Firman B (2016) MPU-6050 adalah chip dengan 3-axis Accelerometer(sensor percepatan) dan 3-axis Gyroscope(pengatur keseimbangan), atau dengan kata lain 6 degrees of freedom (DOF)IMU. Selain itu, MPU-6050 csendiri sudah memiliki Digital Motion Processors(DMP), yang akan mengolah data mentah dari masing-masing sensor. Sejumlah data tersebut akan diolah menjadi data dalam bentuk quaternions(4 Dimensi). DMP pada MPU6050 juga berfungsi meminimalisasi error yang dihasilkan. MPU 6050 adalah chip IC nverse yang didalamnya terdapat sensor Accelerometer dan Gyroscope yang sudah terintegrasi.

Accelerometer digunakan untuk mengukur percepatan, percepatan gerakan dan juga percepatan gravitasi. Accelerometer sering digunakan untuk menghitung sudut kemiringan, dan hanya dapat melakukan dengan nyata ketika statis dan tidak bergerak. Untuk mendapatkan sudut akurat kemiringan, sering dikombinasikan dengan satu atau lebih gyro dan kombinasi data yang digunakan untuk menghitung sudut. Gyroscope adalah perangkat untuk mengukur atau mempertahankan orientasi, yang berlandaskan pada prinsip-prinsip momentum sudut.

2.3 Machine Learning

Machine Learning (ML) atau pembelajaran mesin merupakan pendekatan dalam AI yang banyak digunakan untuk menggantikan atau menirukan perilaku manusia untuk menyelesaikan masalah atau melakukan otomatisasi. Sesuai namanya, ML mencoba menirukan bagaimana proses manusia atau makhluk cerdas belajar dan menggeneralisasi. Setidaknya ada dua aplikasi utama dalam ML yaitu, klasifikasi dan prediksi . Ciri khas dari ML adalah adanya proses pelatihan, pembelajaran, atau training. Oleh karena itu, ML membutuhkan data untuk dipelajari yang disebut sebagai data training. Klasifikasi adalah metode dalam ML yang digunakan oleh mesin untuk memilih atau mengklasifikasikan objek berdasarkan ciri tertentu sebagaimana manusia mencoba membedakan benda satu dengan yang lain.

Sedangkan prediksi atau regresi digunakan oleh mesin untuk menerka keluaran dari suatu data masukan berdasarkan data yang sudah dipelajari dalam training. Metode ML yang paling populer yaitu Sistem Pengambil Keputusan, Support Vector Machine (SVM) dan Neural Network.

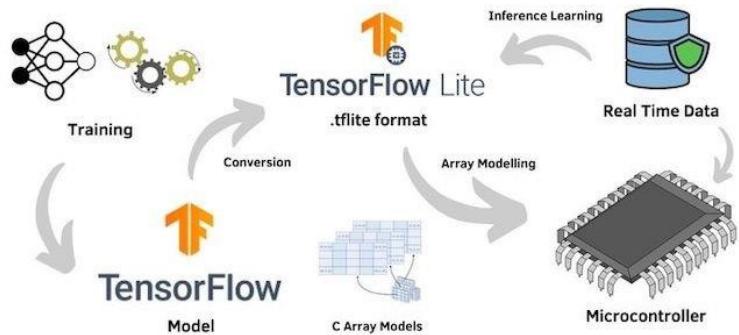


Gambar 5. Diagram AI

2.4 Tiny Machine Learning

Inferensi pembelajaran mesin di edge, khususnya dalam MCU berdaya sangat rendah, semakin diminati oleh komunitas ML. Minat ini berporos pada pembuatan platform yang sesuai di mana model ML dapat dieksekusi secara efisien dalam perangkat IoT. Ini telah membuka area penelitian yang berkembang dalam pembelajaran mesin tertanam yang disebut TinyML. TinyML adalah teknik pembelajaran mesin yang mengintegrasikan pembelajaran mesin yang dikompresi dan dioptimalkan agar sesuai dengan MCU berdaya sangat rendah. TinyML terutama berbeda dari pembelajaran mesin cloud (di mana model intensif komputasi diimplementasikan menggunakan komputer kelas atas di pusat data besar seperti Facebook, pembelajaran mesin seluler dalam hal konsumsi daya yang sangat rendah (rata-rata 0,1 W). TinyML menciptakan platform di mana model pembelajaran mesin didorong ke perangkat pengguna untuk menginformasikan pengalaman pengguna yang baik untuk beragam aplikasi dan memiliki keunggulan seperti efisiensi energi, pengurangan biaya, keamanan data, latensi rendah, dll., yang menjadi perhatian utama di cloud kontemporer teknologi komputasi. Colby dkk. mempresentasikan survei di mana arsitektur jaringan saraf (MicroNets) menargetkan unit mikrokontroler komoditas. Penulis secara efisien mem-porting MicroNets ke MCU menggunakan platform TensorFlow Lite Micro. Ada berbagai platform yang dikembangkan untuk porting algoritme ML dengan mudah ke lingkungan dengan sumber daya terbatas. Tabel 16 menyajikan daftar kerangka kerja TinyML yang tersedia yang umumnya diadopsi untuk

mendorong model ML ke berbagai perangkat terbatas sumber daya yang kompatibel.



Gambar 6. TinyML

Dalam banyak hal, perangkat lunak di balik alat dan konsep di balik TinyML adalah fitur terpentingnya. Secara umum, ekosistem yang paling populer dan terintegrasi untuk pengembangan TinyML adalah TensorFlow Lite for Microcontrollers (TF Lite Micro). TF Lite Micro dirancang khusus untuk tugas ML pada perangkat dengan sumber daya terbatas, dengan MCU sebagai fokusnya. Lingkungan berbasis Python, TF Lite Micro penuh dengan library dan toolkit bawaan untuk:

- Data Acquisition
- Pre processing
- Model Architecture
- Training
- Evaluation
- Optimization
- Quantization

2.5 Bahasa Pemrograman C

Bahasa Pemrograman C adalah sebuah bahasa pemrograman komputer yang bisa digunakan untuk membuat berbagai aplikasi (general-purpose programming language), mulai dari sistem operasi (seperti Windows atau Linux), antivirus, software pengolah gambar (image processing), hingga compiler untuk bahasa pemrograman, dimana C banyak

digunakan untuk membuat bahasa pemrograman lain yang salah satunya adalah PHP.

Meskipun termasuk general-purpose programming language, yakni bahasa pemrograman yang bisa membuat berbagai aplikasi, bahasa pemrograman C paling cocok merancang aplikasi yang berhubungan langsung dengan Sistem Operasi dan hardware. Ini tidak terlepas dari tujuan awal bahasa C dikembangkan.

Bahasa C tentunya adalah bahasa yang dapat dijadikan sebagai bahasa pemrograman pertama bagi pemula. Namun, perlu kalian ketahui bahwa bahasa C juga dikenal sebagai mother language, system programming language, procedure-oriented programming language, structured programming language, dan mid-level programming language. Berikut penjelasannya.

Bahasa C dikenal sebagai mother language karena sebagian besar compiler, kernel, dan lainnya dicatat dalam bahasa ini dan beberapa bahasa pemrograman lainnya mengikuti syntax bahasa ini seperti C++, Java, dan lainnya. Bahasa C sebagai system programming language dapat digunakan untuk melakukan low-level programming. Bahasa C sebagai procedural language menentukan beberapa langkah untuk program agar dapat menyelesaikan masalah. Bahasa C sebagai structured procedural language berarti bahasa ini dapat memecahkan sebuah program menjadi bagian-bagian sehingga dapat dimengerti dengan mudah. Bahasa C sebagai mid-level programming language mendukung kedua low-level dan high-level language.

Tentunya, bahasa C dapat digunakan untuk kehidupan sehari-hari kita karena bahasa ini menghasilkan kode yang berjalan hampir secepat kode yang ditulis dalam assembly language. Contoh penggunaan bahasa C antara lain Operating Systems, Language Compilers, Text Editors, Network Drivers, Databases, dan Utilities.

2.6 Bahasa Pemrograman Phyton

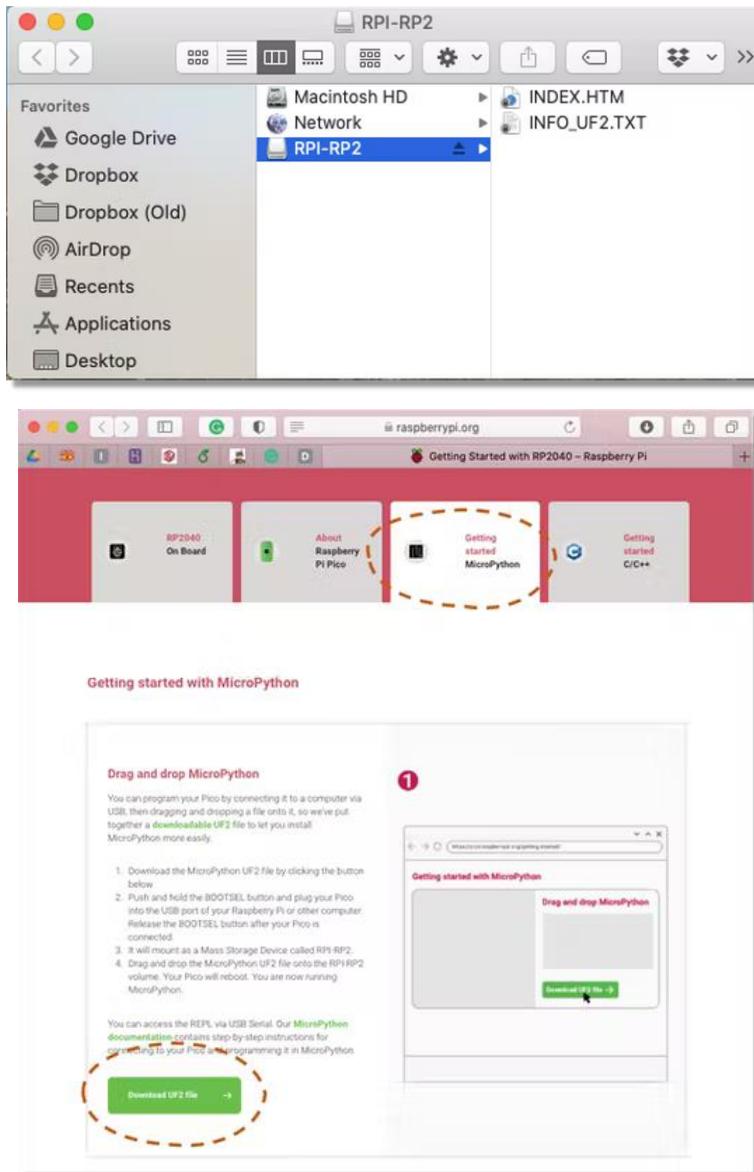
Python adalah bahasa pemrograman interpretatif multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. Python

diklaim sebagai bahasa yang menggabungkan kapabilitas, kemampuan, dengan sintaksis kode yang sangat jelas, dan dilengkapi dengan fungsionalitas pustaka standar yang besar serta komprehensif. Python juga didukung oleh komunitas yang besar.

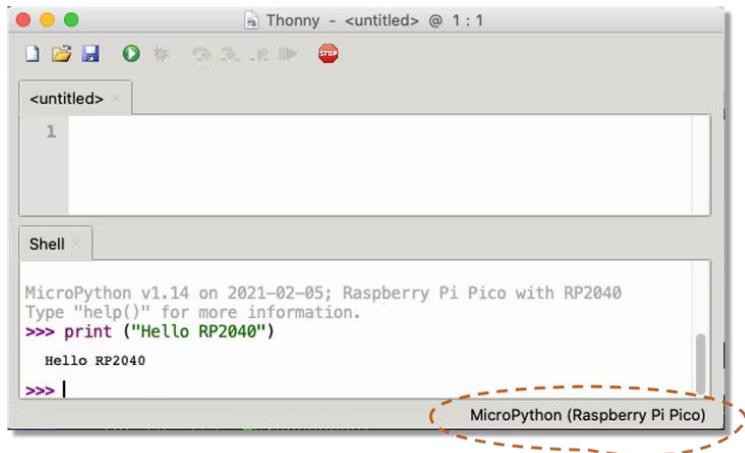
Python mendukung multi paradigma pemrograman, utamanya; namun tidak dibatasi; pada pemrograman berorientasi objek, pemrograman imperatif, dan pemrograman fungsional. Salah satu fitur yang tersedia pada python adalah sebagai Bahasa pemrograman dinamis yang dilengkapi dengan manajemen memori otomatis. Seperti halnya pada bahasa pemrograman dinamis lainnya, python umumnya digunakan sebagai bahasa script meski pada praktiknya penggunaan bahasa ini lebih luas mencakup konteks pemanfaatan yang umumnya tidak dilakukan dengan menggunakan bahasa script. Python dapat digunakan untuk berbagai keperluan pengembangan perangkat lunak dan dapat berjalan di berbagai platform sistem operasi. Saat ini kode python dapat dijalankan di berbagai platform sistem operasi. Python didistribusikan dengan beberapa lisensi yang berbeda dari beberapa versi. Namun pada prinsipnya Python dapat diperoleh dan dipergunakan secara bebas, bahkan untuk kepentingan komersial. Lisensi Python tidak bertentangan baik menurut definisi Open Source maupun General Public License (GPL).

Pada kasus ini untuk memprogram pico dapat dilakukan dengan menggunakan MicroPhyton, berikut merupakan cara-cara dalam menerapkan Pico pada MicroPhyton:

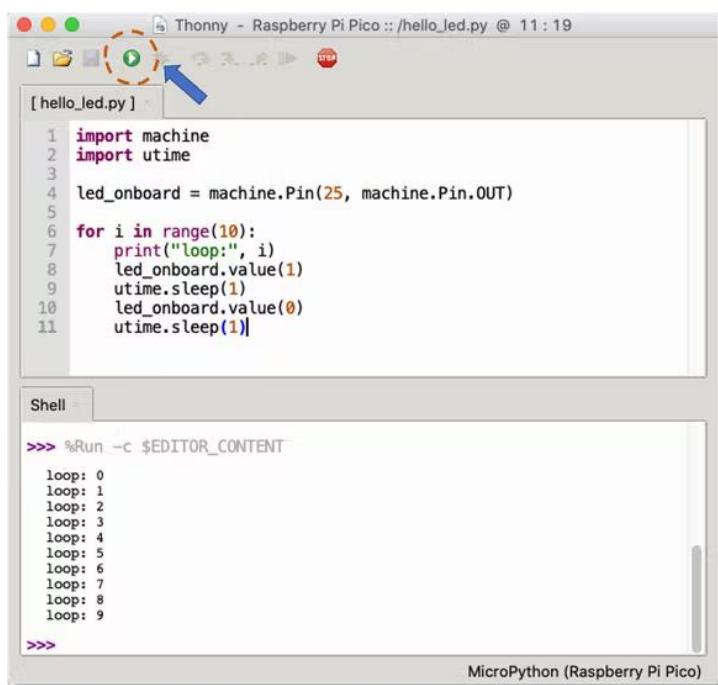
1. Menghubungkan Pico dengan menggunakan USB. Setelah Pico disambungkan pada komputer, (melalui USB) dan dengan menekan tombol BOOT (atau menekan Reset dan Boot, setelah koneksi), windows RPI-RP2 akan muncul, sebagai Mass Storage Device normal (sama sebagai Pen-Driver biasa). Lalu, klik INDEX.HTM untuk memulai aplikasi MicroPhyton.



2. Kemudian ikuti setiap instruksi yang berlaku. Unduh file UF2 yang akan berguna untuk penerjemah MicroPython di Pico, cukup drag file tersebut ke window RPI-RP2 dan Pico siap menerima skrip Python. Gunakan aplikasi Thonny sebagai IDE untuk menulis skrip python langsung di shell seperti di bawah ini, atau mengembangkan script di editor:



3. Menulis atau masuk dengan skrip Python seperti contoh blink di bawah ini:



Dengan tombol play (ditandai pada gambar di atas), skrip diunggah ke Pico. LED internal (pin 25) akan berkedip 10 kali, mencetak nomor loop pada Shell.

4. Untuk membaca sensor suhu internal, maka perlu membuat file log untuk menyatukan suhu Pico internal. Dalam contoh ini, file log temp.txt disimpan di dalam Pico, jadi perhatikan berapa banyak ruang memori yang mungkin diperlukan.

The screenshot shows the Thonny IDE interface. In the top tab bar, there are three tabs: [hello_led.py], [read_internal_temp.py], and [temps.txt]. The [read_internal_temp.py] tab is selected, displaying the following Python code:

```
1 import utime
2
3 sensor_temp = machine.ADC(machine.ADC.CORE_TEMP)
4
5 conversion_factor = 3.3/(65535)
6
7 file = open("temps.txt", "w")
8
9
10 while True:
11     reading = sensor_temp.read_u16() * conversion_factor
12     temperature = 27-(reading-0.706)/0.001721
13     print("Temp: ", temperature)
14     file.write(str(temperature) + "\n")
15     file.flush()
16     utime.sleep(5)
```

In the bottom panel, there is a 'Shell' tab which contains the command `>>> %Run -c $EDITOR_CONTENT`. Below this command, the output of the script is shown, displaying a series of temperature readings:

```
Temp: 22.36296
Temp: 24.70368
Temp: 24.70368
Temp: 24.70368
Temp: 24.70368
Temp: 24.70368
Temp: 24.23554
Temp: 24.70368
Temp: 24.70368
Temp: 24.70368
Temp: 24.70368
```

At the bottom right of the interface, it says "MicroPython (Raspberry Pi Pico)".

BAB III. METODE PELAKSANAAN DAN LANGKAH KERJA

3.1 Metode Pelaksanaan

Metode pelaksanaan dari pembuatan embedded Machine learning Motion Recognition MCU 6050 menggunakan Raspberry PI PICO ini dilakukan melalui beberapa proses berikut:

1. Persiapan perlengkapan hardware dan software pada komputer yang akan digunakan sebagai media pemrograman
2. Memastikan hardware yang digunakan dapat bekerja dengan baik
3. Pembuatan code untuk mengambil raw data
4. Pengambilan raw data sensor yang akan digunakan sebagai data training dan testing, melakukan klasifikasi data yang diambil
5. Melakukan training dan testing
6. Mengunduh data hasil training dan testing
7. Melakukan pembuatan firmware Raspberry PI PICO untuk melakukan klasifikasi pembacaan data berdasarkan klasifikasi yang telah dibuat pada pembuatan data training & testing
8. Melakukan pengujian perangkat embedded Motion Detection yang telah dibuat.

3.2 Langkah Kerja

1. Persiapan perlengkapan hardware dan software pada komputer yang akan digunakan sebagai media pemrograman

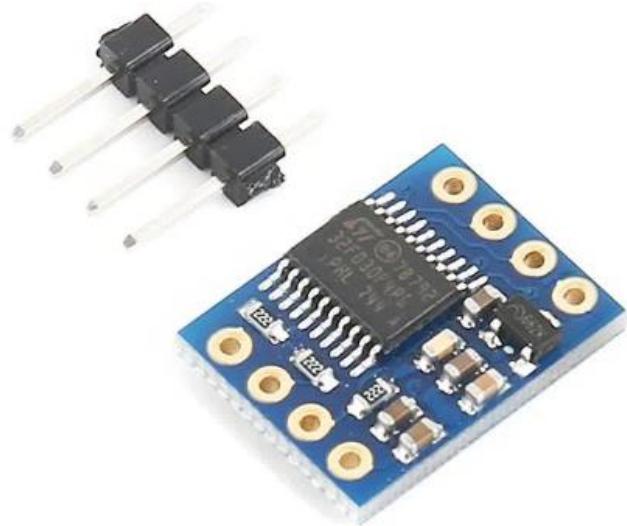
a. Persiapan Perlengkapan Hardware

- **Raspberry PI PICO**



Gambar 7. Raspberry PI PICO

- **MCU 6050 (Gyroscope Sensor)**



Gambar 8. MCU 6050 (Gyroscope Sensor)

- **Kabel Jumper**



Gambar 9. Kabel Jumper

- **Kabel Data Micro Type**



Gambar 10. Kabel Data Micro Type

- **Perangkat Komputer/Laptop**

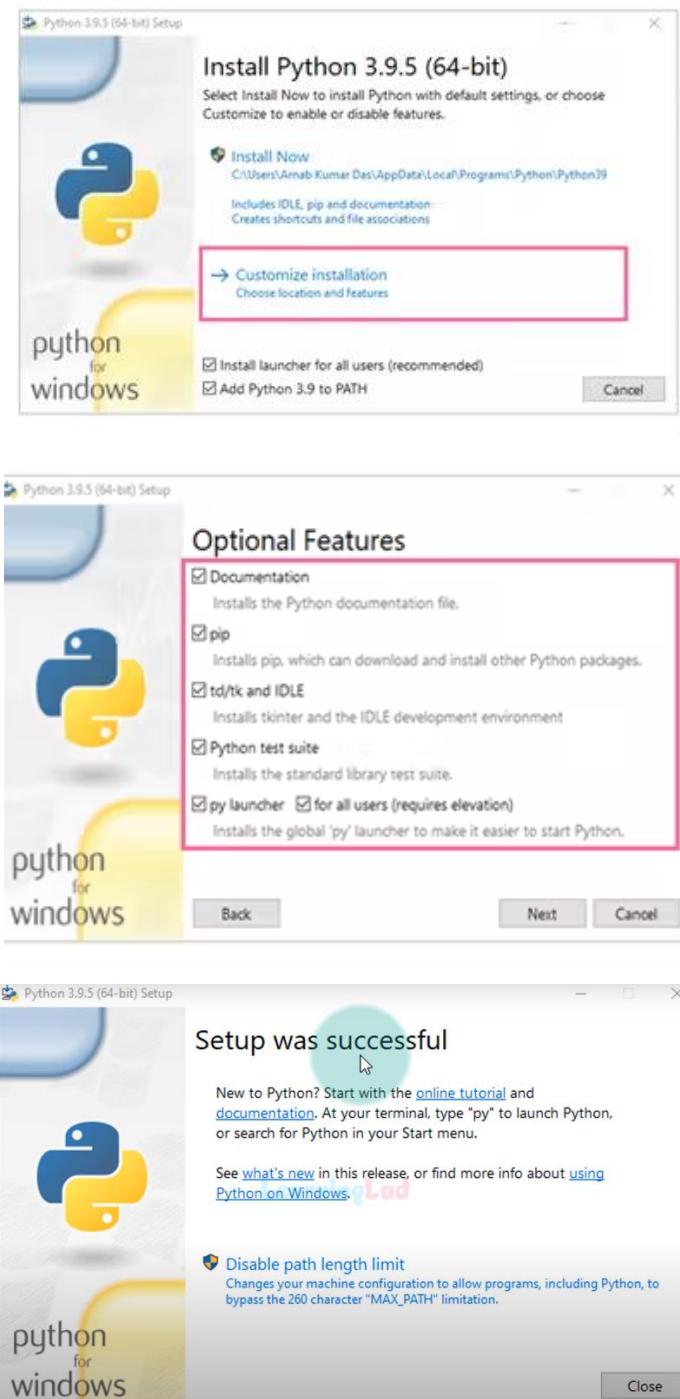


Gambar 11. Laptop

- b. Persiapan Perlengkapan Software**

- **Phyton 3.9**

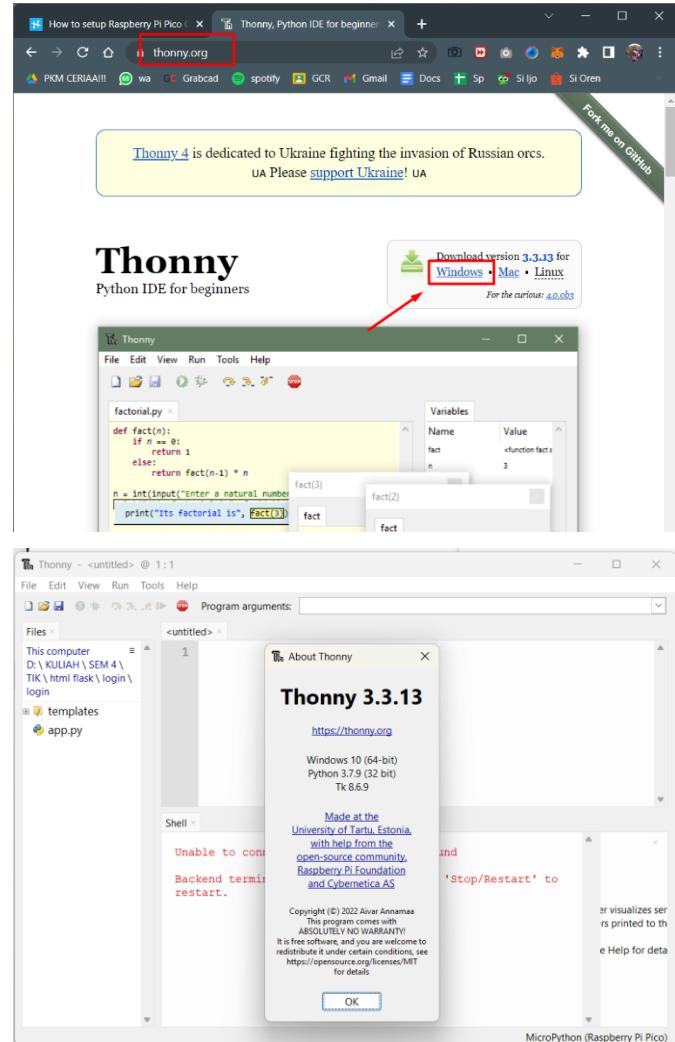
Pengunduhan software Phyton 3.9 dapat dilakukan melalui:
<https://www.python.org/downloads/windows/> dan pastikan bahwa PATH sudah di ceklis.



Gambar 12. Install Phyton 3.9

- **Thonny Phyton**

Pengunduhan software Thonny Phyton dapat dilakukan melalui: <https://thonny.org/>

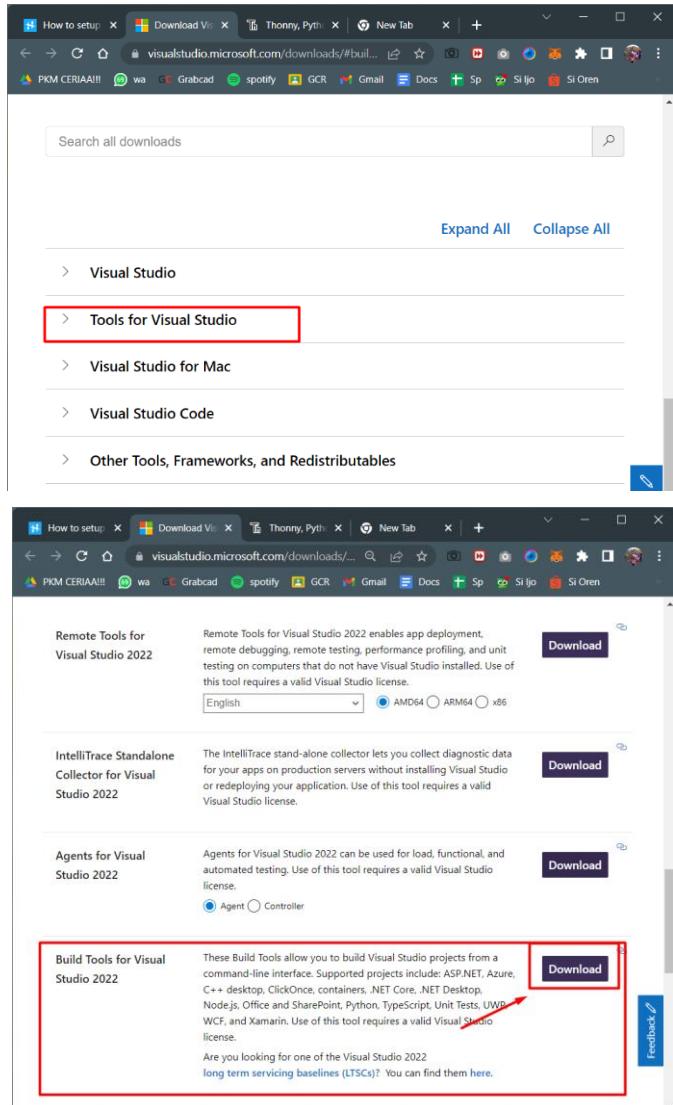


Gambar 13. Install Thonny

- Build Tool Visual Studio 2019 (Bahasa C++)

Pengunduhan software Build Tool Visual Studio 2019 dapat dilakukan melalui:

<https://visualstudio.microsoft.com/downloads/#build-tools-for-visual-studio-2019> , kemudian Scroll ke bawah dan temukan Tools for visual studio dropdown.

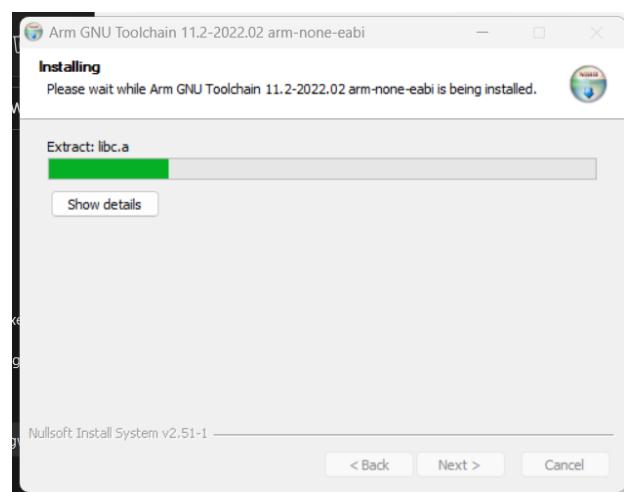
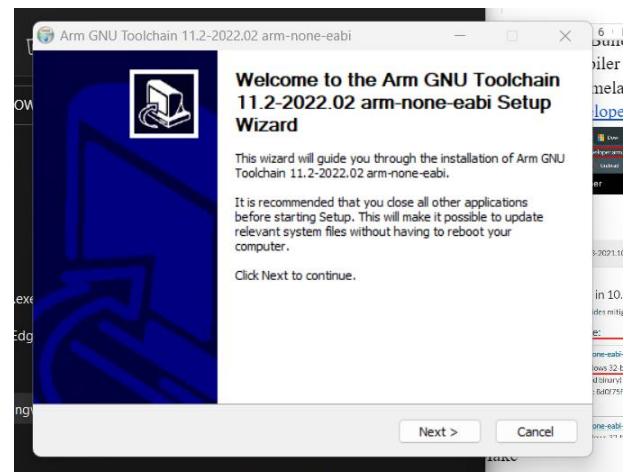
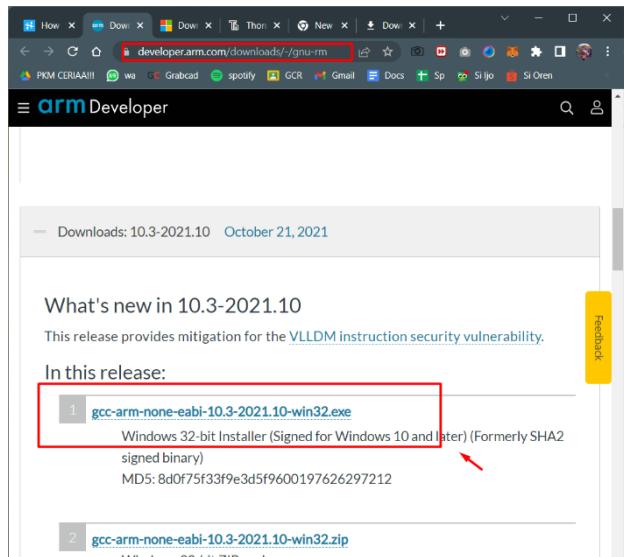


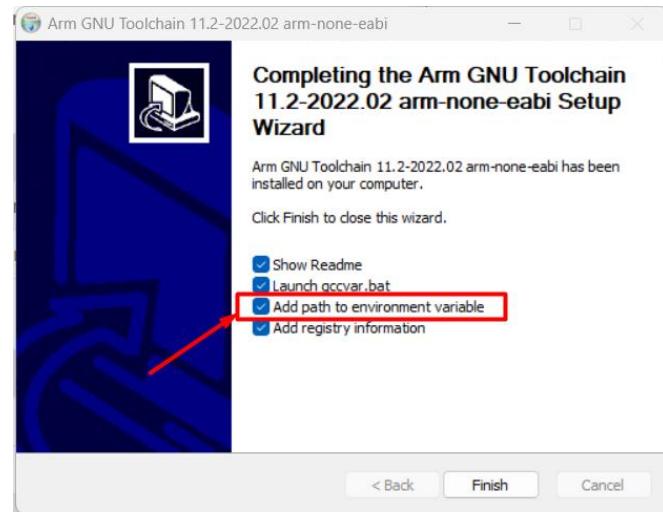
Gambar 14. Install Build Tool Visual Studio 2019

- **ARM GCC Compiler**

Pengunduhan software ARM GCC Compiler dapat dilakukan melalui: <https://developer.arm.com/downloads-/gnu-rm>

- a) Run Installer
- b) Set Default Destination folder path
- c) Check the options:
 - Add path to environment variable
 - Add registry information





Gambar 15. Install ARM GCC Compiler

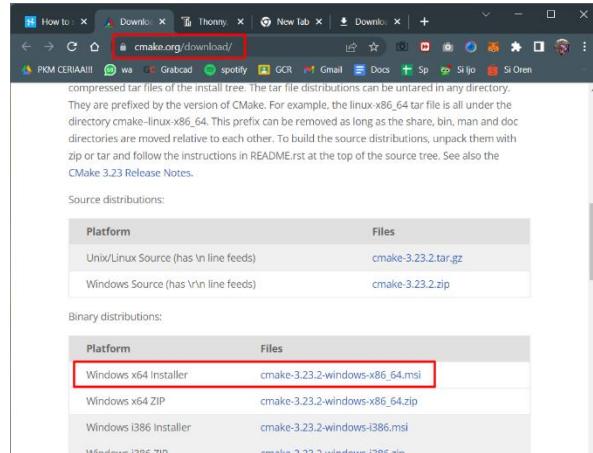
- **CMake**

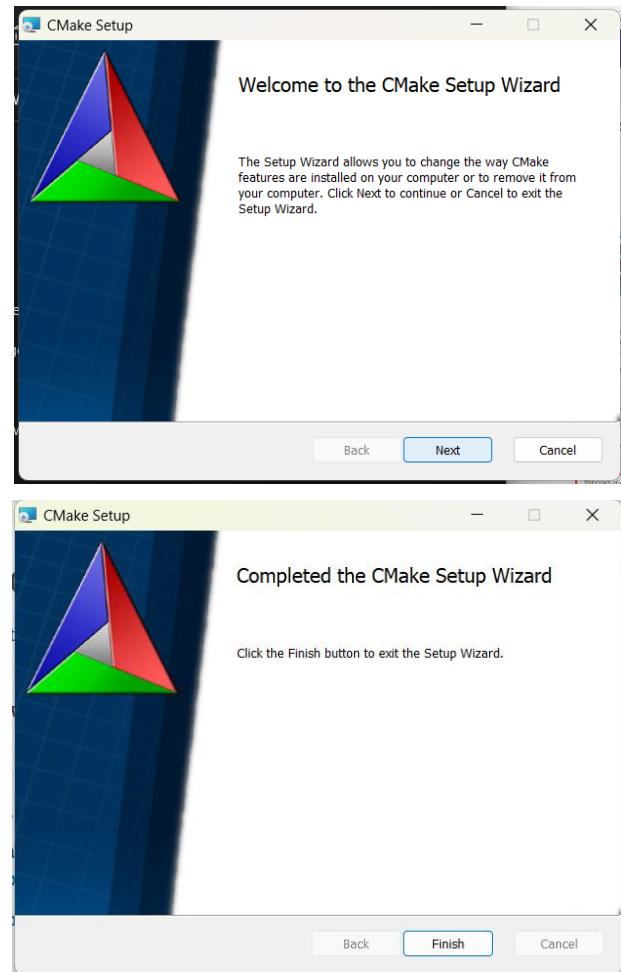
Pengunduhan software CMake dapat dilakukan melalui:

- a) Run Installer
- b) Check the option:

Add CMake to the system PATH for all the users

- c) Set Default Destination folder path

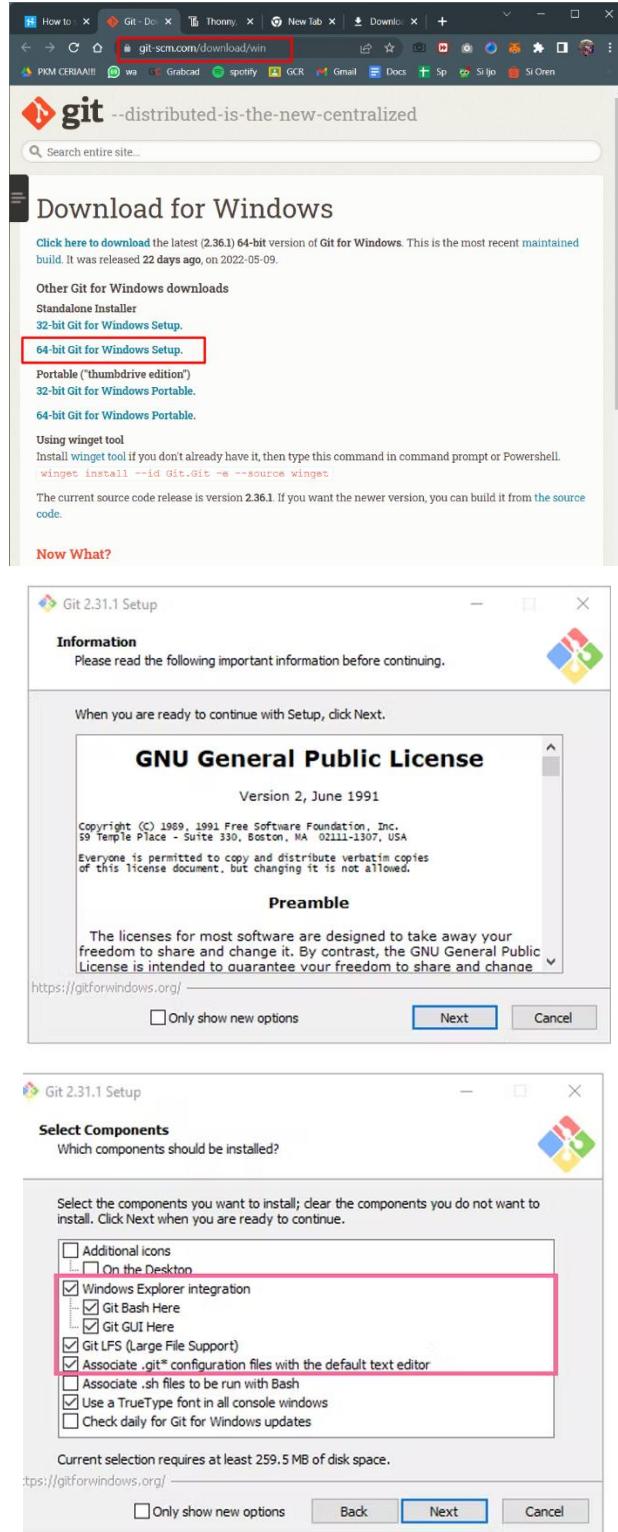




Gambar 16. Install CMake

- Git
 - a) Run Installer
 - b) Set Default Destination folder path
 - c) Select the options:
 - Git from the command line and also from 3rd-party software
 - Use OpenSSH
 - Use the OpenSSL library
 - Checkout as-is, commit as-is
 - Use Windows' default console window (optional)
 - 'git pull', Default (fast-forward or merge)

- Enable file system caching
- Enable experimental support for pseudo consoles



Gambar 17. Install Git

- Visual Studio Code dan Library

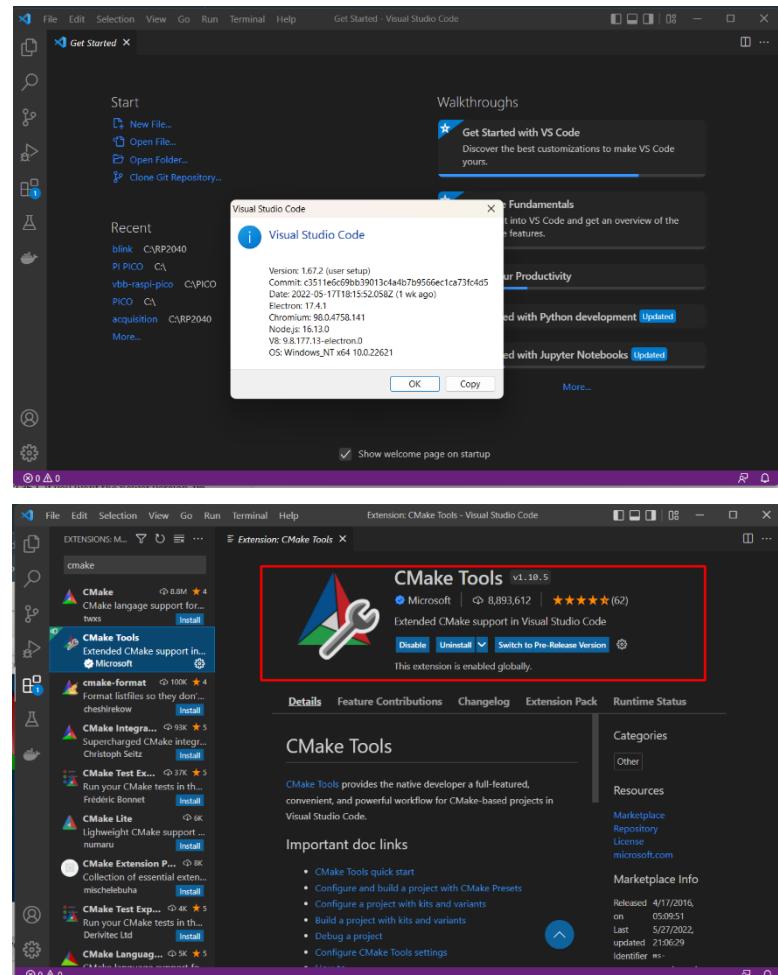
Pengunduhan software Visual Studio Code dapat dilakukan melalui: <https://code.visualstudio.com/download>

a) Run Installer

b) Check essential options:

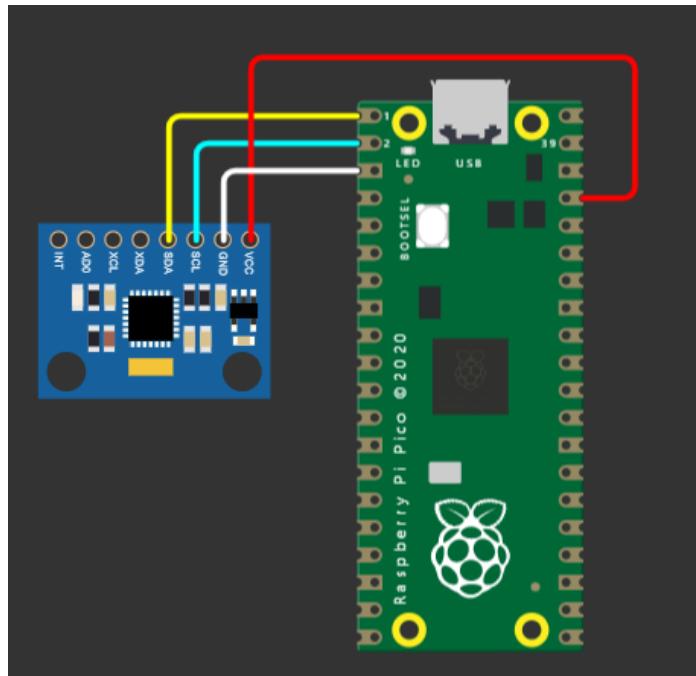
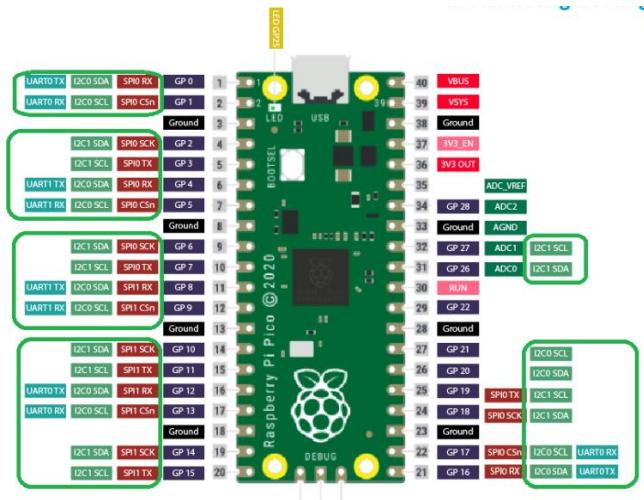
Desktop development with C++

Universal Windows Platform development



Gambar 18. Install Visual Studio Code

2. Persiapan perlengkapan hardware dan software pada komputer yang akan digunakan sebagai media pemrograman
 - a. Menyambungkan pin MPU 6050 ke raspberry pi pico



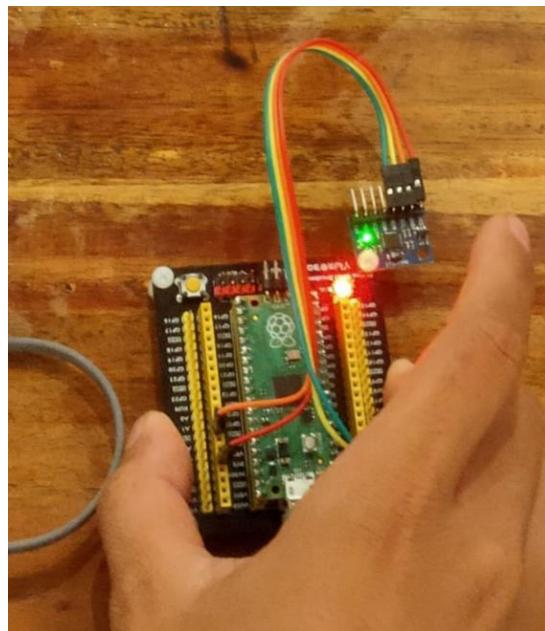
Gambar 19. Sambungan pin MPU 6050 ke raspberry pi pico

Pin GND MPU 6050 ke Pin GND Raspberry pi PICO

Pin VCC MPU 6050 ke Pin 3v3 Raspberry pi PICO

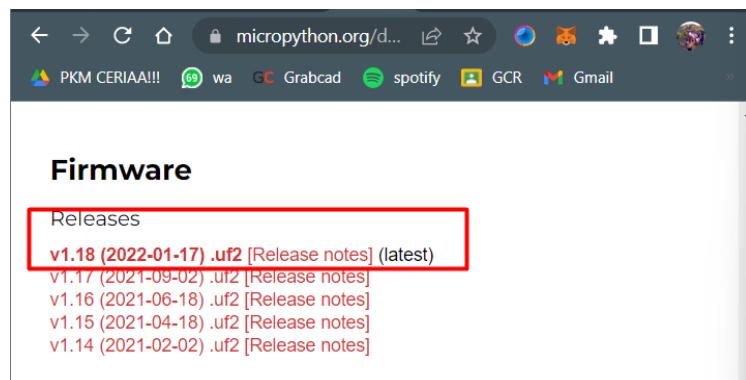
Pin SDA MPU 6050 ke Pin SDA Raspberry pi PICO (GPIO 0)

Pin SCL MPU 6050 ke Pin SCL Raspberry pi PICO (GPIO 1)



Gambar 20. Contoh Pengaplikasian

- b. Download Firmware UF2 raspberry pi pico dari website:
<https://micropython.org/download/rp2-pico/>



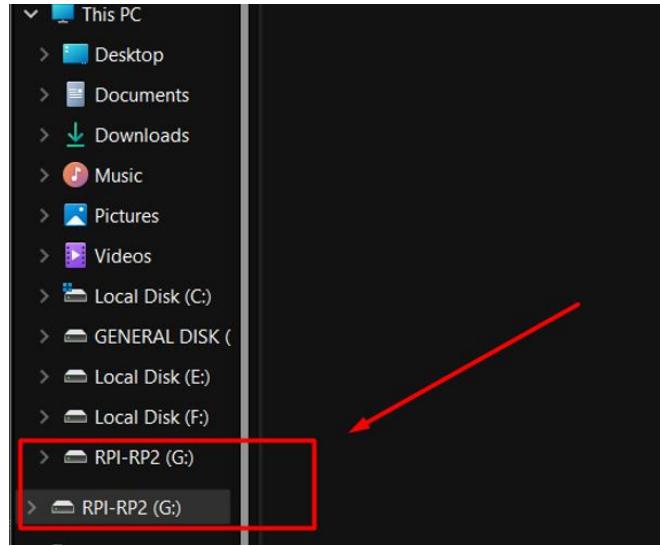
Gambar 21. Install Firmware

- c. Hubungkan Raspberry PI PICO ke komputer menggunakan USB micro sambil menekan botsel button.



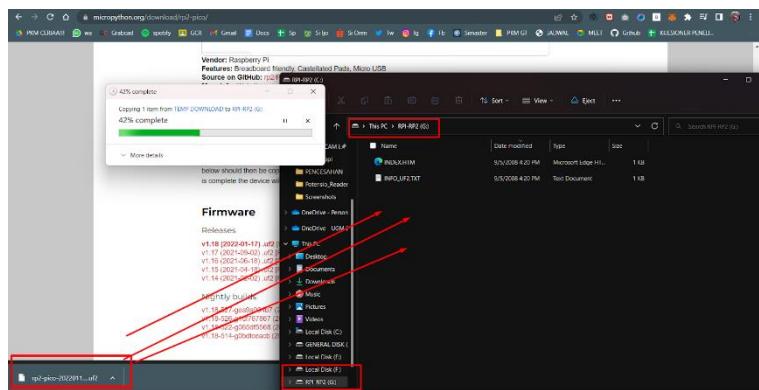
Gambar 22. Bootsel

- d. Pastikan Terdapat Partisi Drive baru pada list drive computer



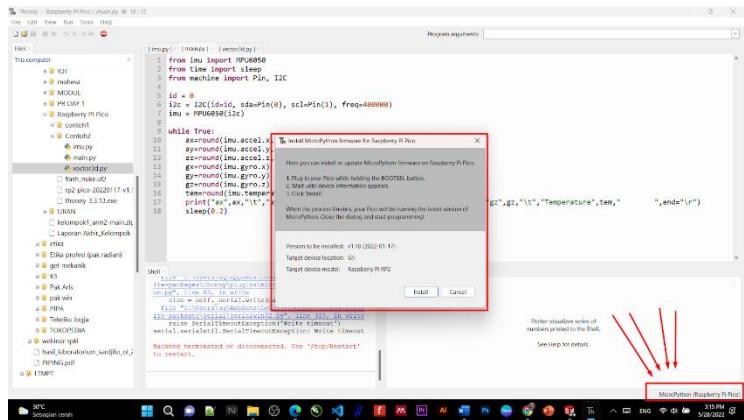
Gambar 23. Partisi Drive

- e. Drag & Drop file firmware UF2 ke dalam drive RPI-RP2



Gambar 24. Drag File

- f. Setelah selesai memasukkan file maka Raspberry pi akan melakukan restart dan masuk ke aplikasi Thonny, dan pastikan interpreter Thonny sudah terarah ke Raspberry PI PICO, pada kanan bawah aplikasi.

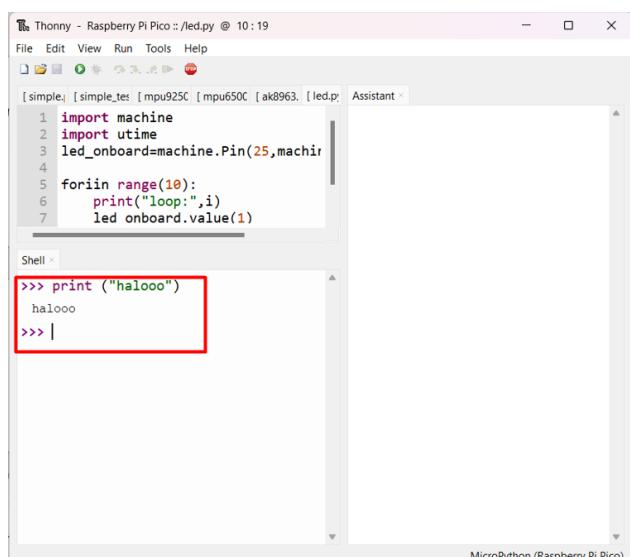


Gambar 25. Thonny terarah ke Raspberry PI PICO



Gambar 26. Install

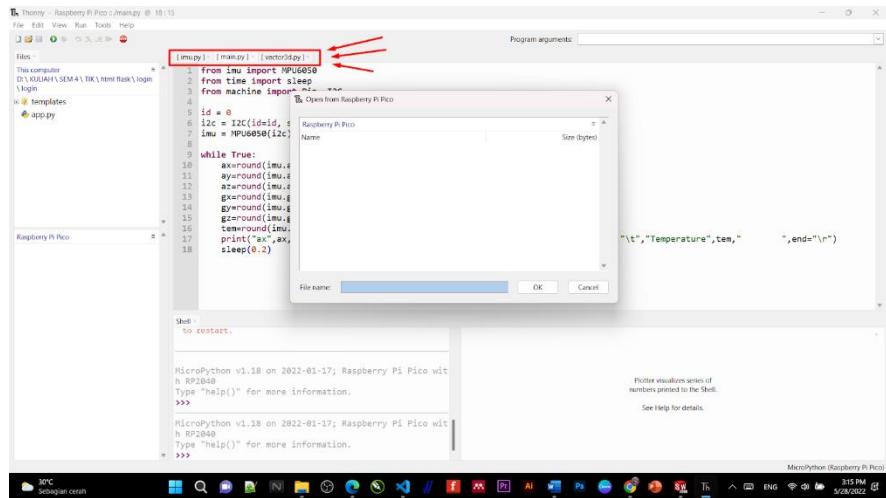
- g. Setelah Instalasi firmware raspberry selesai maka raspberry pi siap digunakan.
- h. Uji coba code.



Gambar 27. Uji Coba Kode

3. Pembuatan code untuk mengambil raw data

Pengambilan data pada tahap ini kami lakukan masih menggunakan coding bahasa Python.



Gambar 28. Raw Data

Buat 2 files library: imu.py, vector3d.py Dapat di download melalui: Library dapat didownload pada: <https://github.com/micropython-IMU/micropython-mpu9150.git>

Buat 1 files main: main.py

```
#main.py

from imu import MPU6050
from time import sleep
from machine import Pin, I2C

id = 0
i2c = I2C(id=id, sda=Pin(0), scl=Pin(1), freq=400000)
imu = MPU6050(i2c)

while True:
    ax=round(imu.accel.x,2)
    ay=round(imu.accel.y,2)
    az=round(imu.accel.z,2)

    gx=round(imu.gyro.x)
    gy=round(imu.gyro.y)
```

```

gz=round(imu.gyro.z)

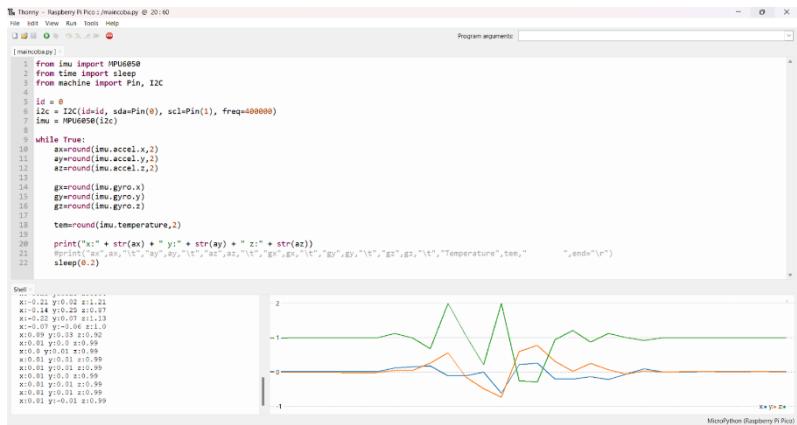
AVGgforce=((gx)+(gy)+(gz))/3

tem=round(imu.temperature,2)

print(str(ax) + "," + str(ay) + "," + str(az))

```

Kemudian save ketiga files ke dalam drive raspberry pi pico.



Gambar 29. Screenshot Raspberry Pi Pico

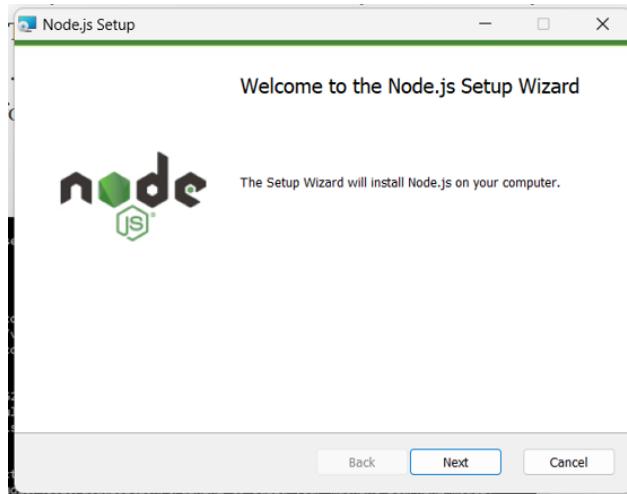
Hasil data screenshot masih menggunakan penamaan axis string x y z, script diatas hanya akan menampilkan nilai x y z yang dibutuhkan untuk digunakan sebagai raw data learning. Maka akan ditampilkan hasil print dari acceleration x y dan z.

- Pengambilan raw data sensor yang akan digunakan sebagai data training dan testing, melakukan klasifikasi data yang diambil.

- Install Node.js pada komputer

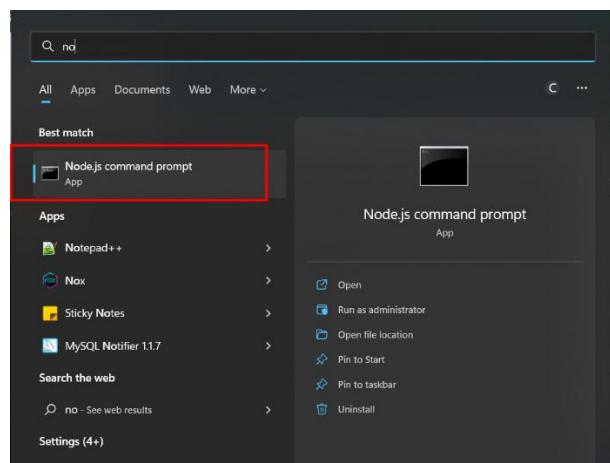
Link download: <https://nodejs.org/en/>





Gambar 30. Install Node.js

- **Buka Node.js command prompt**



Gambar 31. Command Prompt Node.js

- **Download & install edge impulse**

Ketik: npm install -g edge-impulse-cli --force

```
PS C:\Users\uq>npm install -g edge-impulse-cli --force
Your environment has been set up for using Node.js 16.15.0 (x64) and npm.

c:\Users\uq>npm install -g edge-impulse-cli --force
npm WARN using --force Recommended protections disabled.
npm WARN deprecated @zeit/dockerignore@0.5: "@zeit/dockerignore" is no longer maintained
npm WARN deprecated har-validator@5.1.5: this library is no longer supported
npm WARN deprecated request-promise@4.2.4: request-promise has been deprecated because it extends t
now deprecated request package. See https://github.com/request/request/issues/3142
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity Re
gression when used in a Node.js environment. It is recommended you upgrade to 3.2.7 or 4.3.1.
(https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated debug@4.1.1: Debug versions >>3.2.0 <3.2.7 || >>4 <4.3.1 have a low-severity Re
gression when used in a Node.js environment. It is recommended you upgrade to 3.2.7 or 4.3.1.
(https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated debug@4.1.1: Debug versions >>3.2.0 <3.2.7 || >>4 <4.3.1 have a low-severity Re
gression when used in a Node.js environment. It is recommended you upgrade to 3.2.7 or 4.3.1.
(https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated uuid@3.4.0: Please upgrade to version 7 or higher. Older versions may use Mat
h.random() in certain circumstances, which is known to be problematic. See https://v8.dev/blog/math
-random for details.
npm WARN deprecated request@2.88.0: request has been deprecated, see https://github.com/request/req
uest/issues/3142

changed 372 packages, and audited 373 packages in 1m

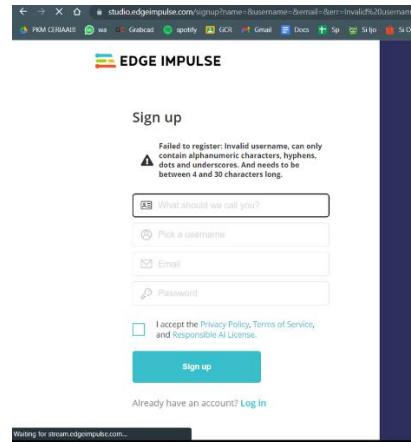
12 packages are looking for funding
  run 'npm fund' for details

5 vulnerabilities (2 moderate, 3 high)
```

Gambar 32. Install edge Impulse

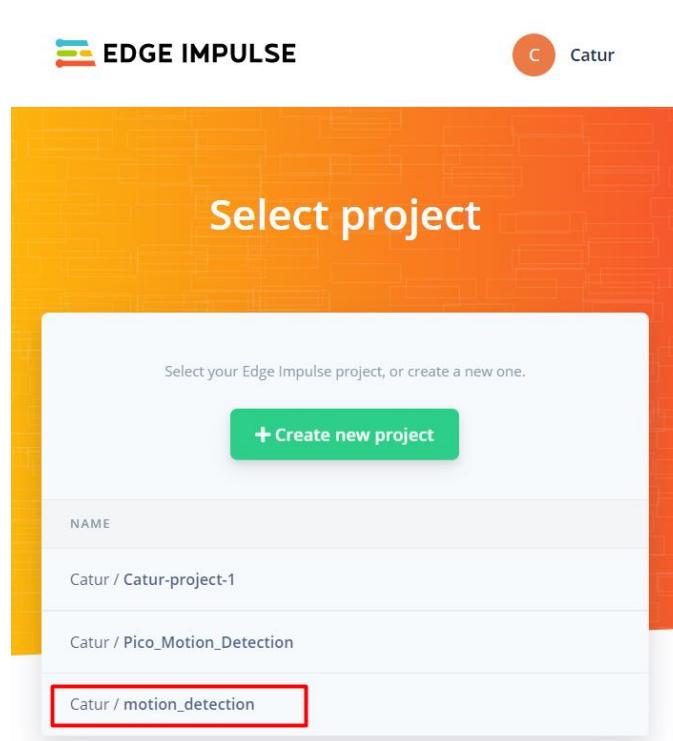
- **Buat akun pada website:**

<https://studio.edgeimpulse.com/signup>



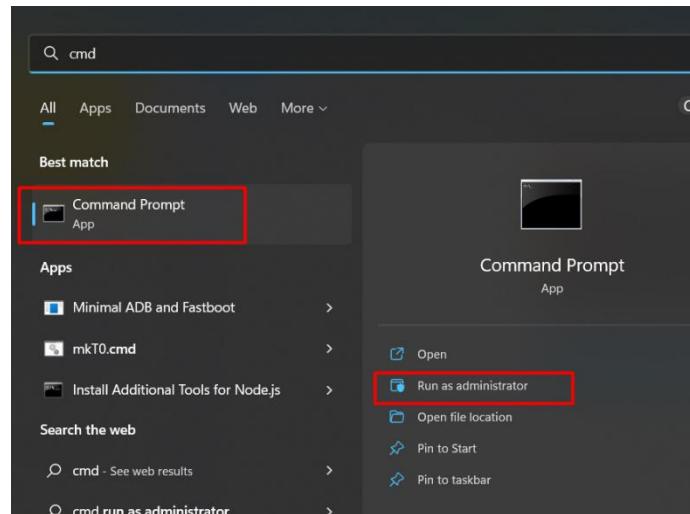
Gambar 33. Register akun Edge Impulse

- **Buat Akun Baru**



Gambar 34. New Project

- **Buka Cmd.exe**



Gambar 35. Cnd.Exe

Ketik: edge-impulse-data-forwarder

Masuk ke akun yang telah dibuat pada website.

```
C:\WINDOWS\system32\cmd.exe - "node" "C:\Users\uq\AppData\Roaming\npm\edge-impulse-data-forwarder\index.js"
Microsoft Windows [Version 10.0.22621.1]
(c) Microsoft Corporation. All rights reserved.

C:\Users\uq>edge-impulse-data-forwarder
Edge Impulse data forwarder v1.15.0
? What is your user name or e-mail address (edgeimpulse.com)?
? What is your password? [hidden]

? What is your user name or e-mail address (edgeimpulse.com)? Catur
? What is your password? [hidden]
Endpoints:
  Websocket: wss://remote-mgmt.edgeimpulse.com
  API: https://studio.edgeimpulse.com/v1
  Ingestion: https://ingestion.edgeimpulse.com
```

Gambar 36. Tampilan edge-impulse-data-forwarder

- **Tunggu edge impuls mendeteksi perangkat Raspberry PI PICO kita.**

Pastikan Raspberry berjalan.

```
To which project do you want to connect this device? Catur / motion_detection
[SER] Detecting data frequency...
[SER] Detected data frequency: 41Hz
: 3 sensor axes detected (example values: [0.01,-0.02,0.99]). What do you want to call them? Separate the names with ',':
: circle
: Invalid input. Got 1 axes (circle), but expected 3
: 3 sensor axes detected (example values: [0.01,-0.02,0.99]). What do you want to call them? Separate the names with ',':
: rectangle
: Invalid input. Got 1 axes (rectangle), but expected 3
: 3 sensor axes detected (example values: [0.01,-0.02,0.99]). What do you want to call them? Separate the names with ',':
: triangle
: Invalid input. Got 1 axes (triangle), but expected 3
: 3 sensor axes detected (example values: [0.01,-0.02,0.99]). What do you want to call them? Separate the names with ',':
: x,y,z
? What name do you want to give this device? pico_
```

Gambar 37. Raspberry Berjalan

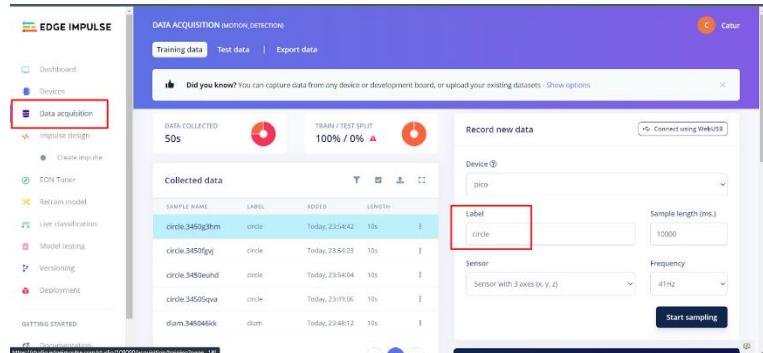
Beri nama per pembacaan sensor yaitu x y dan z

Beri nama device disini kami beri nama “pico”

- **Masuk dashboard web edge impuls:**

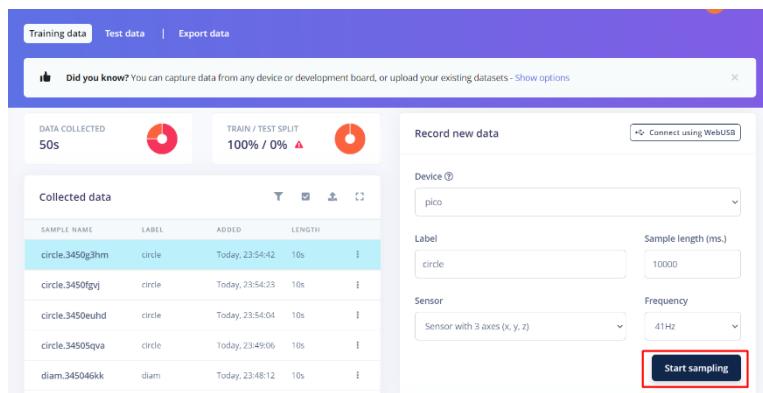
<https://studio.edgeimpulse.com/studio/>

Buat nama class yang akan kita buat, pada kasus ini kita akan mendeteksi 3 jenis gerakan yaitu: circle, rectangle dan diam.



Gambar 38. Membuat Nama Class

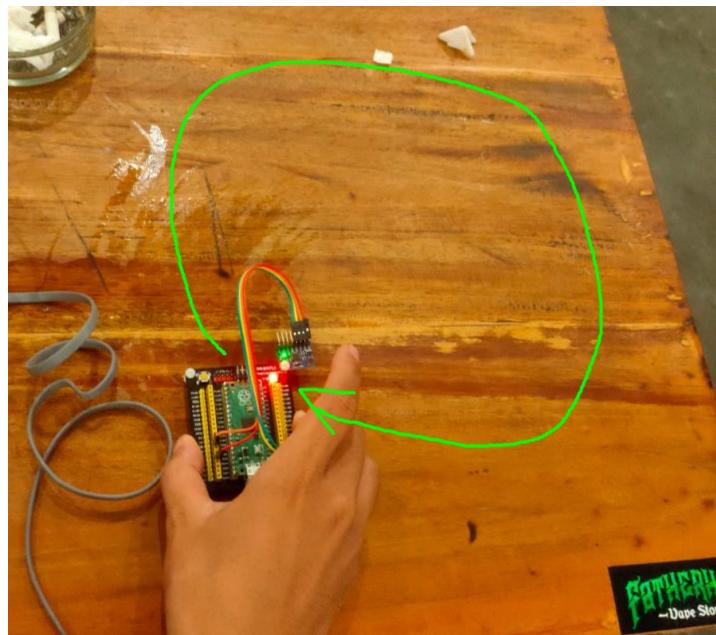
- **Merecord data Circle, Rectangle dan diam dengan klik tombol start sampling**



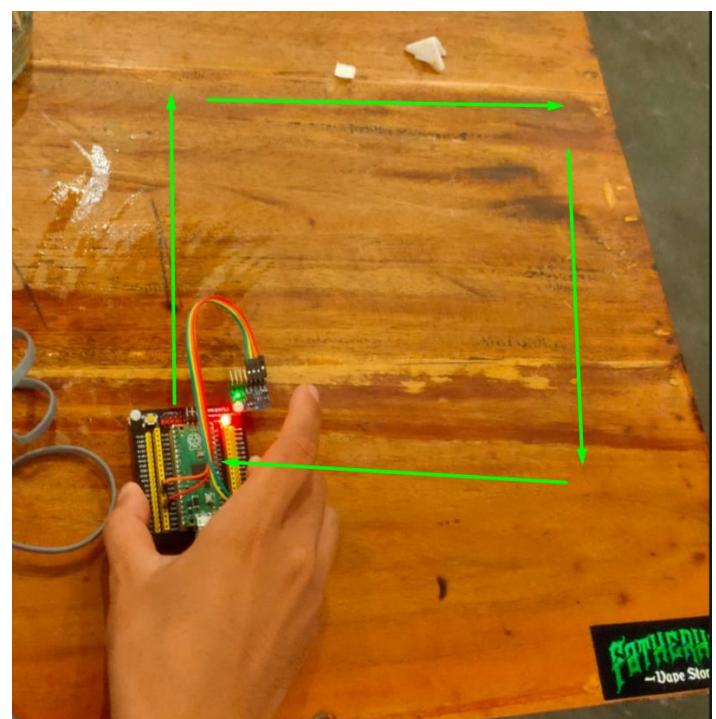
```
C:\WINDOWS\system32\cmd.exe - node "C:\Users\up\AppData\Roaming\npm\node_modules\edge-impulse-cli\build\cli\data-forwarder.js"
[ 0.22, 0.37, 1.34 ],
[ 0.05, -0.04, 0.83 ],
[ 0.14, -0.11, 0.599999 ],
[ 0.02, -0.14, 0.75 ],
[ 0.2, 0.07, 0.46 ],
[ 0.06, -0.31, 0.49 ],
[ 0.14, -0.45, 0.5 ],
[ 0.15, -0.21, 0.86 ],
[ 0.04, -0.36, 1.57 ],
[ 0.02, -0.43, 0.6 ],
[ 0.01, -0.19, -0.01 ],
[ -0.14, -0.42, 1.12 ],
[ -0.06, -0.28, 0.82 ],
[ 0.06, -0.39, 1.08 ],
[ 0.09, -0.2, 0.73 ],
[ 0.15, -0.13, 0.94 ],
[ 0.25, 0.02, 0.79 ],
[ 0.31, 0.02, 1.03 ],
[ 0.19, -0.08, 1.35 ],
[ 0.35, -0.08, 1.26 ],
[ 0.4, 0.08, 1.2 ],
[ 0.17, -0.07, 1.4 ],
[ 0.36, -0.04, 0.64 ],
[ 0.16, -0.42, 1.08 ],
[ 0.26, 0.2, 2 ],
[ 0.38, -0.54, -0.66 ],
[ 0.5, 0.35, 0.56 ],
[ 0.2, 0.05, 2 ],
[ 0.01, -0.19, 1.76 ],
[ 0.21, -0.09, 0.81 ],
[ 0.02, -0.14, 1.27 ],
[ -0.28, -0.01, 0.79 ],
[ -0.07, -0.13, 1.87 ],
[ -0.24, 0.13, 0.87 ],
[ -0.4, 0.12, 0.78 ],
[ -0.27, -0.23, -0.07 ],
[ 0.41, 0.08, 0.85 ],
[ -0.31, 0.12, 1.05 ],
[ -0.27, -0.19, 1.15 ],
[ -0.17, -0.05, 1.05 ],
[ -0.1, 0.08, 0.99 ],
... 305 more items
```

Gambar 39. Record Program

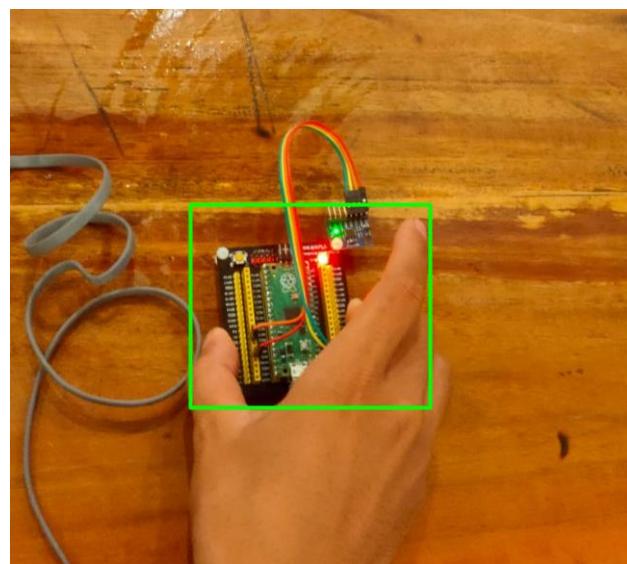
(CIRCLE) Memutar perangkat ke arah jarum jam selama sampling



Gambar 40. Perangkat Diputar SJJ
(RECTANGLE) menggerakan perangkat seperti gerakan persegi selama sampling

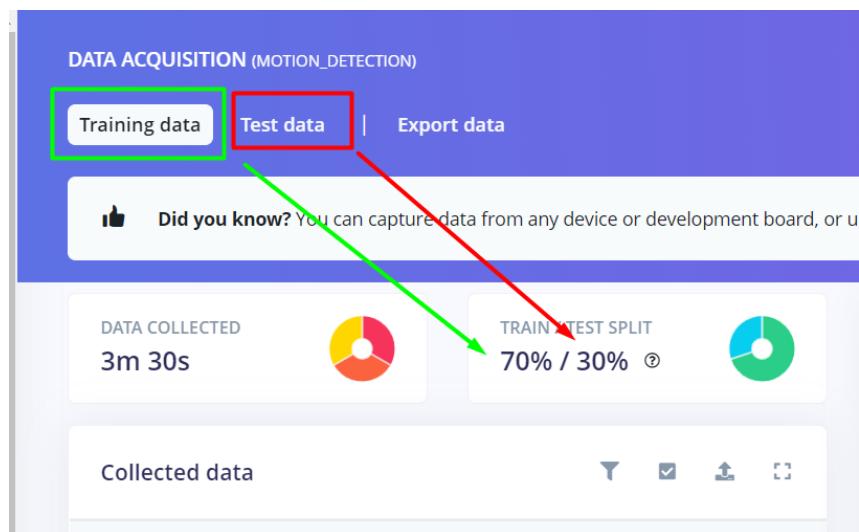


Gambar 41. Gerakan Perangkat Membentuk Persegi (DIAM) Mendiamkan perangkat selama sampling



Gambar 42. Diamkan Perangkat

Data merupakan data training dan testing

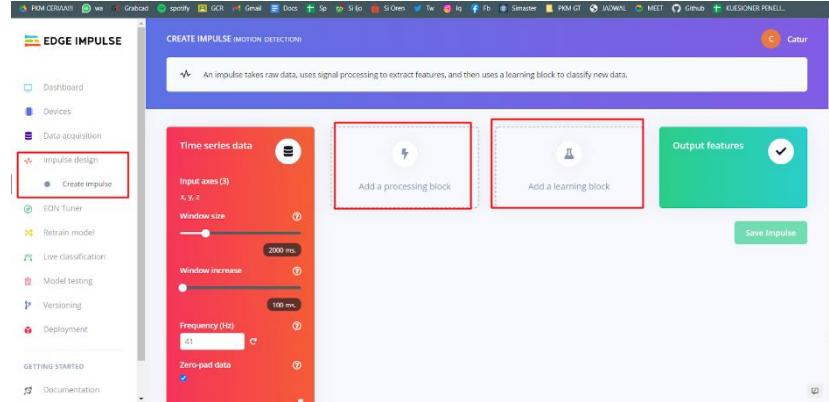


Gambar 43. Data Training dan Data Testing

Keseluruhan data memiliki waktu sampling 3 menit 30 detik
Dengan 70% data training dan 30% data testing

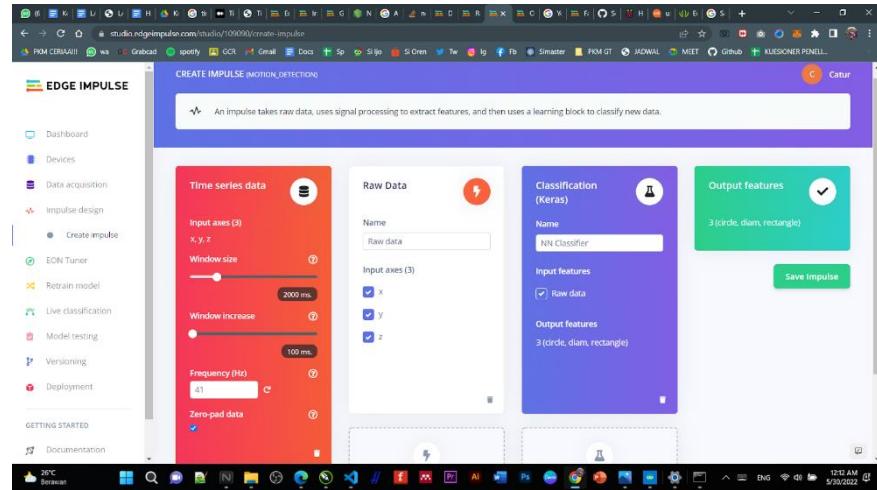
5. Melakukan training dan testing

a. Masuk tab Create Impuls



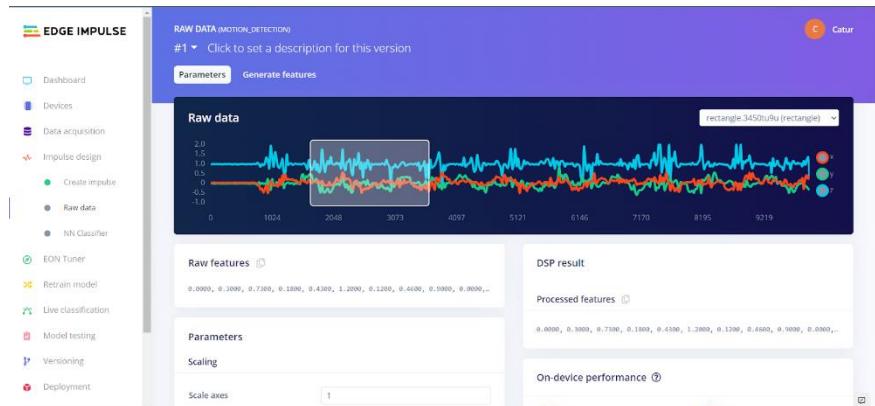
Gambar 44. Tab Create Impuls

b. Setting dengan data yang sudah kita punya, Kemudian klik save impulse



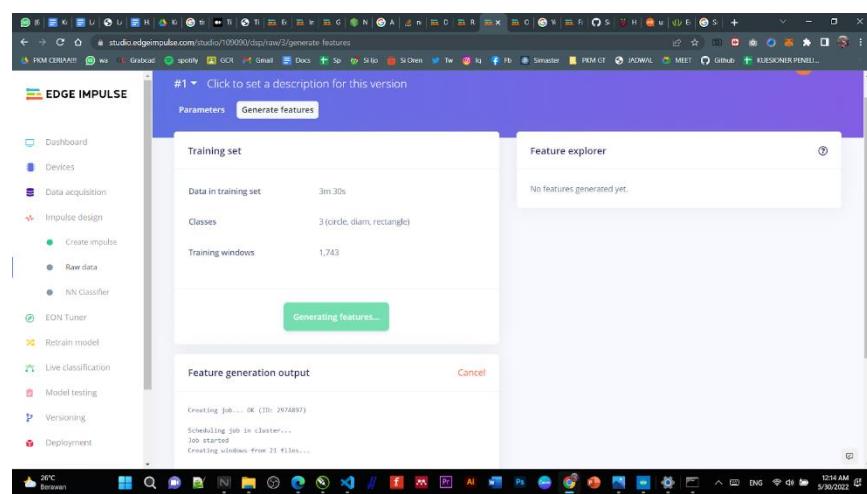
Gambar 45. Setting dan Save Impuls

c. Maka pada tab raw data akan tampil keseluruhan data impulse kita



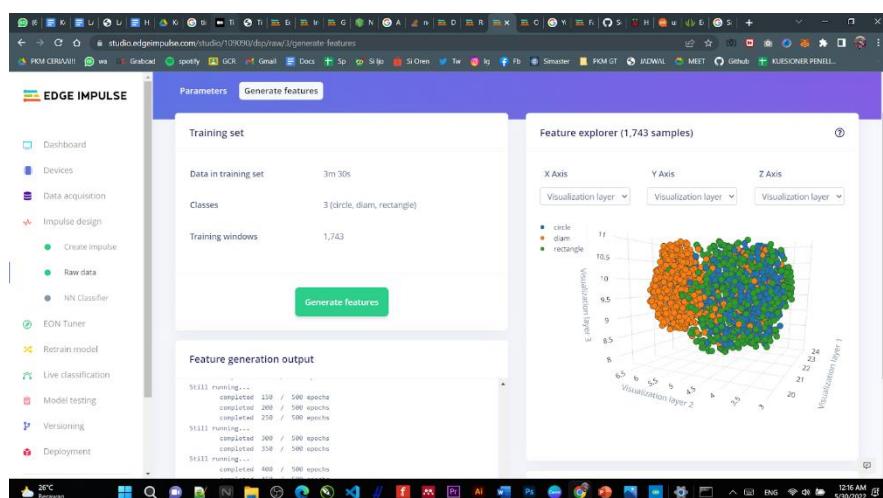
Gambar 46. Raw Data

d. Kemudian klik Generate Feature



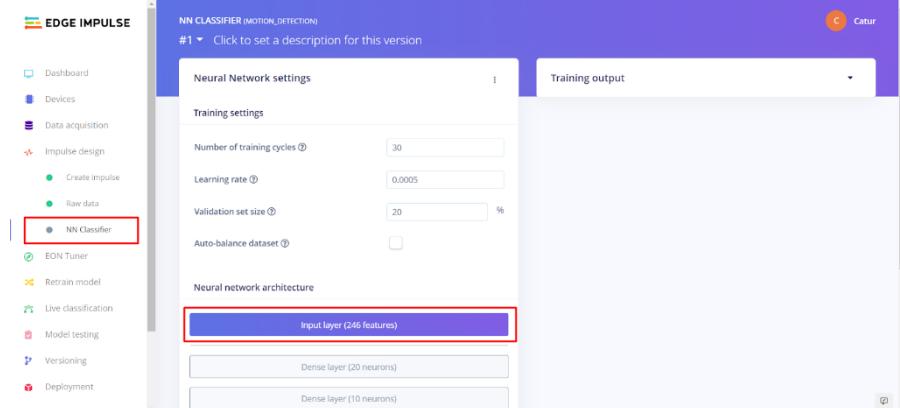
Gambar 47. Generate Feature

e. Kemudian, akan tampil letak neuron-neuron pada data sampling



Gambar 48. Neuron pada Sampling

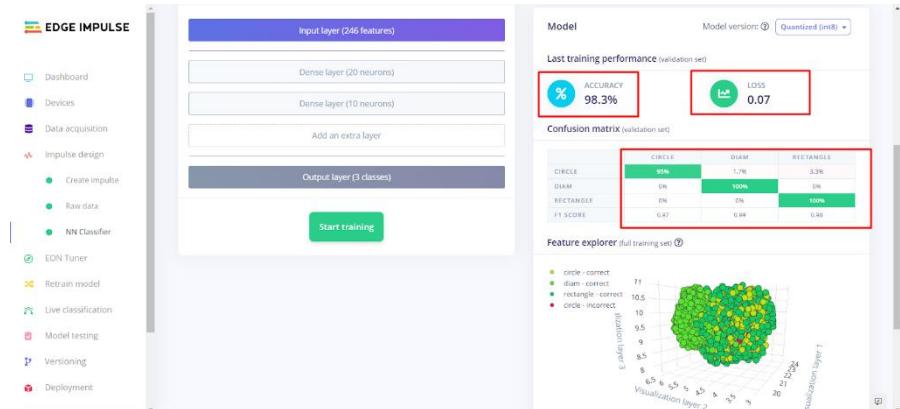
- f. Masuk pada tab NN classifier dan klik input layer untuk melakukan training dan testing pada data yang kita punya.



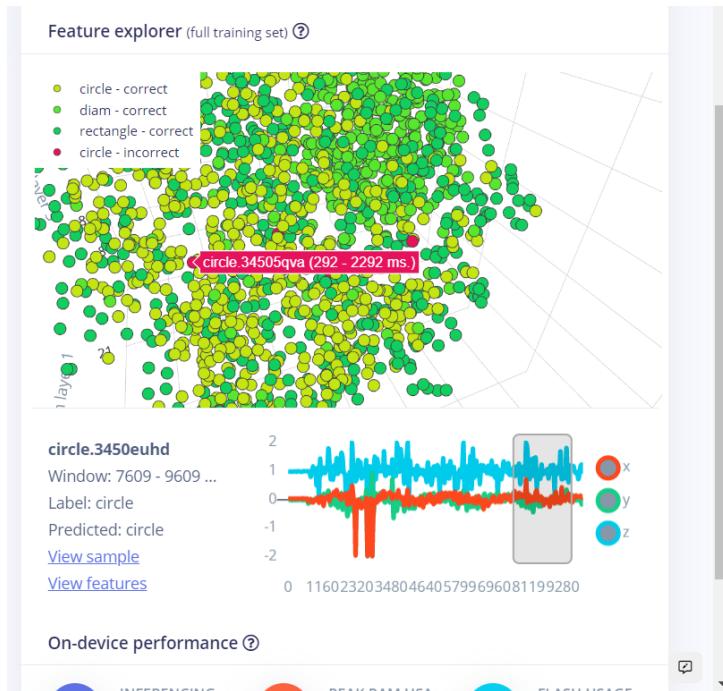
Gambar 49. Tata Cara Training dan Testing Sss

Hasil:

Akurasi data 98% dengan loss 0.07



Gambar 50. Hasil Training dan Testing

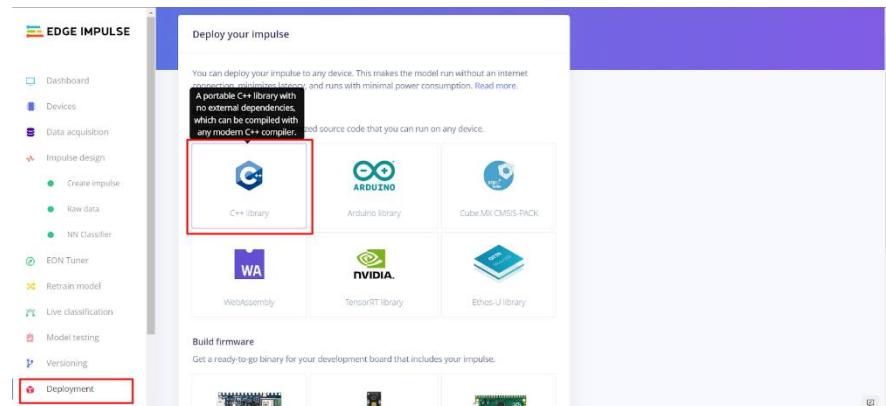


Gambar 51. Data yang mengalami Loss

Beberapa data yang mengalami loss berwarna merahwah (circle)

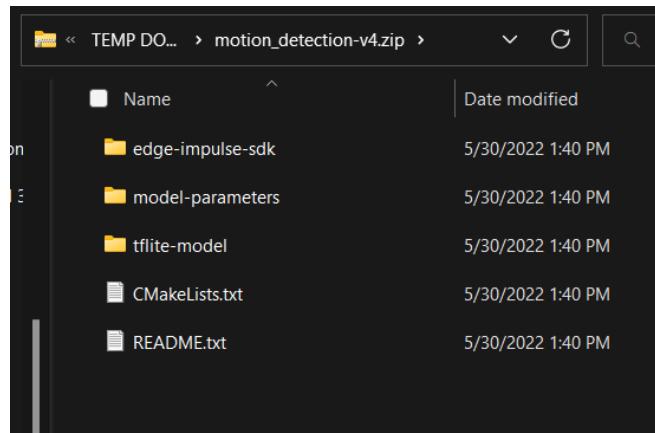
6. Mengunduh data hasil training dan testing

- Masuk pada tab Development dan unduh menggunakan bahasa C++ binary. **Harus C++ karena pada pembuatan firmware harus di compile menggunakan bahasa C++.**



Gambar 52. Unduh hasil Training dan Testing

- Maka hasil download file berupa .zip berisi:



Gambar 53. ZIP hasil download

7. Melakukan pembuatan firmware Raspberry PI PICO untuk melakukan klasifikasi pembacaan data berdasarkan klasifikasi yang telah dibuat pada pembuatan data training & testing.

Melakukan Cloning PICO-SDK

```
C:\RP2040> git clone -b master
https://github.com/raspberrypi/pico-sdk.git

C:\RP2040> cd pico-sdk

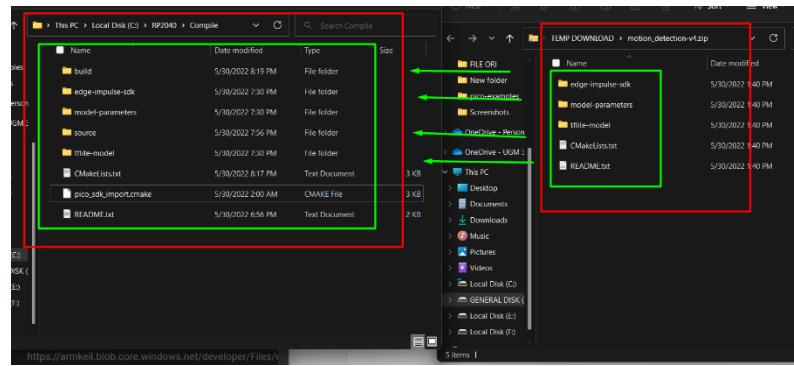
C:\RP2040\pico-sdk> git submodule update --init

C:\RP2040\pico-sdk> cd ..

C:\RP2040> git clone -b master
https://github.com/raspberrypi/pico-examples.git
```

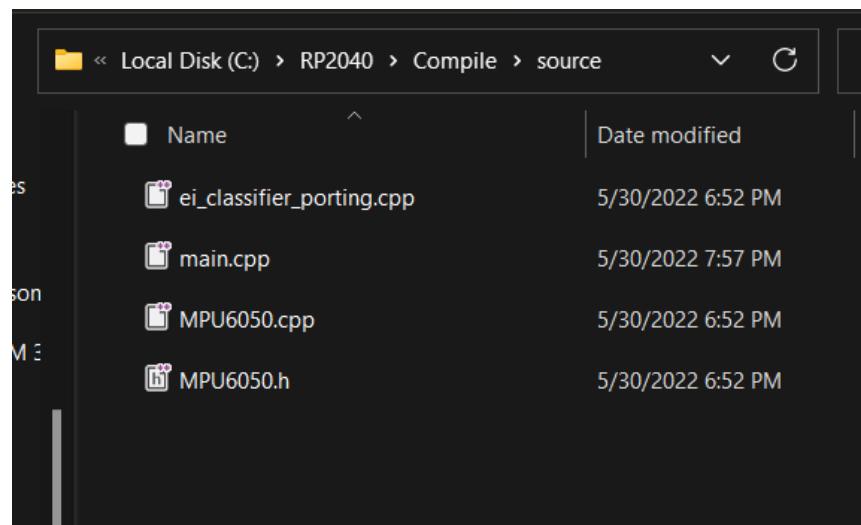
Melakukan Compiling Datasheet hasil training dengan file dari tutorial:<https://www.hackster.io/mjrobot/tinyml-motion-recognition-using-raspberry-pi-pico-6b6071>.

Pada file yang ada di tutorial library, sensor yang digunakan adalah ADX335 yang dimana sensor tersebut membaca nilai adc dari masing masing axis X Y Z. Sedangkan pada sensor yang kita gunakan yaitu MPU 6050 yang gerakan acceleration dibaca melalui SDA dan SCL. untuk itu beberapa perubahan pada main.cpp kami lampirkan pada laporan ini. Masukkan semua isi datasheet data training ke folder “C:\RP2040\Compile”



Gambar 54. Masukkan semua isi data sheet data training ke
C:\RP2040\Compile

Masukkan Library MPU6050



Gambar 55. Library MPU6050

Edit main.cpp (pada include ada peringatan karena library terletak pada folder sdk yang mana library tersebut akan terakses oleh compiler)

```

1 //include <stdio.h>
2 //include <string.h>
3 //include "pico/stdlib.h"
4 //include "pico/stdio.h"
5 //include "pico/stdio.info.h"
6 //include "title-model.h"
7 //include "ei_rpi_classifier.h"
8 //include "MPU6050.h"
9
10 static bool debug_on = false; // set this to true to see e.g. features generated from the raw signal
11
12 const uint LED_PIN = PICO_DEFAULT_LED_PIN;
13
14 #define I2C_I2C_SDA 1
15
16 int main()
17 {
18     stdio_init_all();
19
20     gpio_init(LED_PIN);
21     gpio_set_dir(LED_PIN, GPIO_OUT);
22     gpio_put(LED_PIN, 0);
23
24     printf("Hello, it starts...\n");
25
26     // This example will use I2C on the default SDA and SCL pins (4, 5 on a Pi)
27     i2c_init(I2C_DEFAULT, 400 * 1000);
28     gpio_set_function(PICO_DEFAULT_I2C_SDA_PIN, GPIO_FUNC_I2C);
29     gpio_set_function(PICO_DEFAULT_I2C_SCL_PIN, GPIO_FUNC_I2C);
30     gpio_pull_up(PICO_DEFAULT_I2C_SDA_PIN);
31     gpio_pull_up(PICO_DEFAULT_I2C_SCL_PIN);
32
33     // Set the clock frequency to 100kHz
34     i2c_set_clock(I2C_DEFAULT, 100000);
35 }

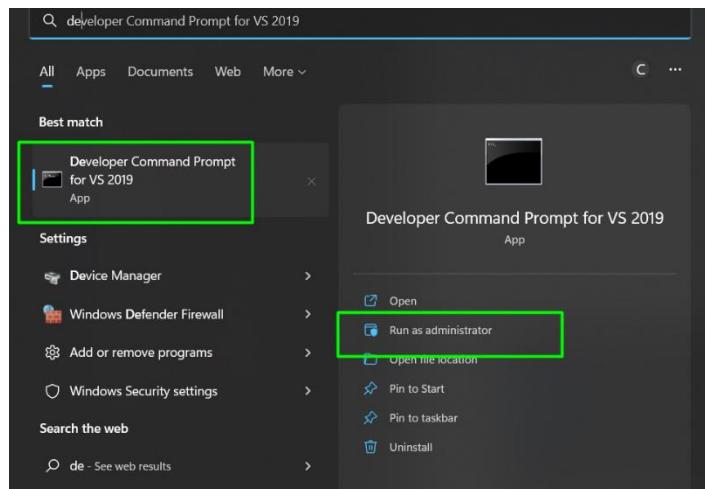
```

Gambar 56. Edit Main.cpp

Edit CMakeList.txt sesuai dengan isi folder yang akan di compile

Gambar 57. Edit CMakeList.txt

Buka Developer command prompt dari visual studio



Gambar 58. Developer command prompt dari visual studio

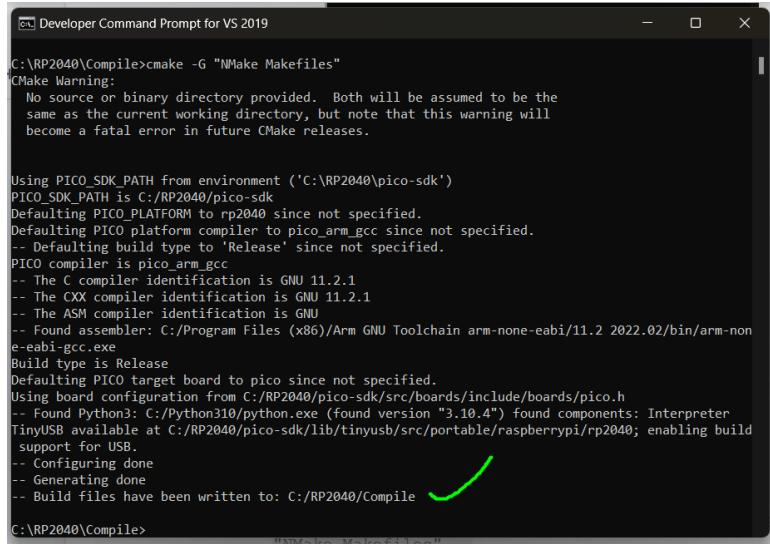
Arahkan Folder ke folder Compile dan set PICO SDK ke path

```
Developer Command Prompt for VS 2019
*****
** Visual Studio 2019 Developer Command Prompt v16.8.3
** Copyright (c) 2020 Microsoft Corporation
*****

C:\Program Files (x86)\Microsoft Visual Studio\2019\Community>cd C:\RP2040\Compile ->
C:\RP2040\Compile>setx PICO_SDK_PATH "C:\RP2040\pico-sdk" ->
SUCCESS: Specified value was saved. ->
C:\RP2040\Compile>
```

Gambar 59. set PICO SDK ke path

Kemudian jalankan perintah compile dengan cmake -G "NMake Makefiles" ..

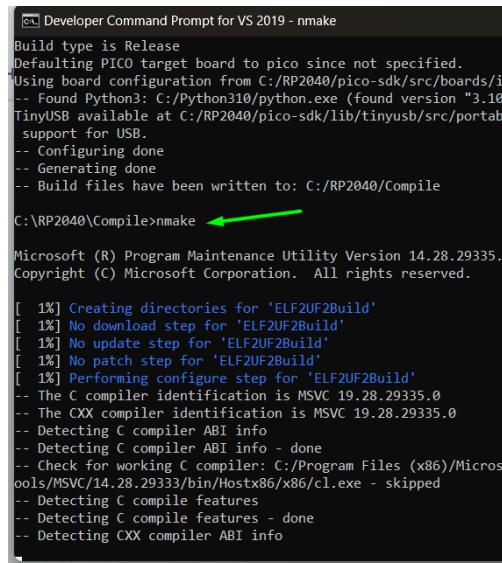


```
C:\RP2040\Compile>cmake -G "NMake Makefiles"
CMake Warning:
  No source or binary directory provided. Both will be assumed to be the
  same as the current working directory, but note that this warning will
  become a fatal error in future CMake releases.

Using PICO_SDK_PATH from environment ('C:\RP2040\pico-sdk')
PICO_SDK_PATH is C:/RP2040/pico-sdk
Defaulting PICO_PLATFORM to rp2040 since not specified.
Defaulting PICO platform compiler to pico_arm_gcc since not specified.
-- Defaulting build type to 'Release' since not specified.
PICO compiler is pico_arm_gcc
-- The C compiler identification is GNU 11.2.1
-- The CXX compiler identification is GNU 11.2.1
-- The ASM compiler identification is GNU
-- Found assembler: C:/Program Files (x86)/Arm GNU Toolchain arm-none-eabi/11.2 2022.02/bin/arm-none-eabi-gcc.exe
Build type is Release
Defaulting PICO target board to pico since not specified.
Using board configuration from C:/RP2040/pico-sdk/src/boards/include/boards/pico.h
-- Found Python3: C:/Python310/python.exe (found version "3.10.4") found components: Interpreter
TinyUSB available at C:/RP2040/pico-sdk/lib/tinyusb/src/portable/raspberrypi/rp2040; enabling build
support for USB.
-- Configuring done
-- Generating done
-- Build files have been written to: C:/RP2040/Compile ✓
```

Gambar 60. Jalankan Perintah Compile

Kemudian ketikkan nmake untuk memverifikasi dan melakukan building firmware Raspberry pi PICO berextensi .UF2 dengan perintah nmake



```
C:\RP2040\Compile>nmake ←
```

```
Microsoft (R) Program Maintenance Utility Version 14.28.29335.0
Copyright (C) Microsoft Corporation. All rights reserved.

[ 1%] Creating directories for 'ELF2UF2Build'
[ 1%] No download step for 'ELF2UF2Build'
[ 1%] No update step for 'ELF2UF2Build'
[ 1%] No patch step for 'ELF2UF2Build'
[ 1%] Performing configure step for 'ELF2UF2Build'
-- The C compiler identification is MSVC 19.28.29335.0
-- The CXX compiler identification is MSVC 19.28.29335.0
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: C:/Program Files (x86)/Microsoft Visual Studio/14.28.29333/bin/Hostx86/x86/cl.exe - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Detecting CXX compiler ABI info
```

Gambar 61. verify dan building firmware Raspberry pi PICO
berextensi.UF2

Tunggu hingga Proses selesai 100%

```
[ 99%] Building C object CMakeFiles/EICarStateRecog.dir/C_/RP204
[100%] Building C object CMakeFiles/EICarStateRecog.dir/C_/RP204
[100%] Building C object CMakeFiles/EICarStateRecog.dir/C_/RP204
[100%] Linking CXX executable EICarStateRecog.elf
collect2.exe: fatal error: CreateProcess: No such file or direct
compilation terminated.
NMAKE : fatal error U1077: 'C:\PROGRA~2\ARMGNU~1\112202~1.02\bin
Stop.
NMAKE : fatal error U1077: '"C:\Program Files (x86)\Microsoft Vi
Stop.
NMAKE : fatal error U1077: '"C:\Program Files (x86)\Microsoft Vi
Stop.

C:\RP2040\Compile>.vscode_
```

Note:

Saat melakukan compile terdapat beberapa kegagalan compiling sehingga saat proses 100% file firmware tidak terbentuk, beberapa langkah telah kami coba dengan melakukan compile menggunakan Cmake dari Visual studio code namun masih menemukan error. Beberapa screenshot error compiling terdapat pada lampiran.

8. Melakukan pengujian perangkat *embedded Motion Detection* yang telah dibuat.

Pada proses ini kami belum dapat melakukan demo alat atau pengujian alat dikarenakan Firmware yang tertanam TinyML tidak dapat di compile dan kami belum sampai melakukan installing firmware ke perangkat Raspberry PI PICO.

BAB IV. KESIMPULAN

Ide umum proyek ini adalah untuk mempelajari cara memprogram Raspberry Pi Pico dan melakukan pembuktian konsep yang memungkinkan untuk melakukan Machine Learning dengan MCU ini, yang belum didukung secara resmi oleh Edge Impulse dan Arduino. Kami harap hal tersebut segera terjadi karena ini akan sangat mempercepat semua proses pengkodean untuk pengembang non-ahli.

Perangkat embedded machine learning mencapai tahap 90% dengan tahap terakhir yaitu compiling firmware yang tertanam Machine learning. Namun saat melakukan compiling kami menemukan beberapa error, kami sudah mencoba beberapa jenis aplikasi kompiler namun hasilnya masih sama.

Kendala:

- 1) Kesulitan melakukan compile firmware yang sudah ada machine learning dari data training.
- 2) Keterbatasan referensi motion recognition yang mengaplikasikan sensor MPU 6050. Sehingga saat membuat main.cpp untuk firmware kami mengalami kesulitan.
- 3) Keterbatasan pemahaman bahasa C++ yang diaplikasikan pada Firmware Raspberry PI pico.
- 4) Belum begitu mengenal dan memahami perangkat Raspberry pi PICO serta bahasa pemrograman yang digunakan oleh perangkat ini.

Saran:

Bagi mahasiswa praktikum berikutnya yang mengerjakan kasus serupa, akan lebih baik mengusahakan se bisa mungkin menggunakan perangkat untuk melakukan pemrograman yang memiliki OS bersih. Se bisa mungkin menggunakan OS yang lebih kompatibel untuk melakukan programing seperti LINUX OS, UBUNTU OS, dan semacamnya, dikarenakan bisa saja error saat melakukan programing berasal dari beberapa add-on pada perangkat yang bertabrakan fungsi.

Bagi pelaksanaan praktikum berikutnya diharapkan untuk mendapatkan materi tentang bahasa pemrograman yang akan diaplikasikan pada perangkat yang akan digunakan seperti bahasa C++ untuk membuat firmware raspberry pi PICO.

Lampiran Video Pengujian perangkat dan Sensor MPU 6050 pada saat persiapan:

<https://youtube.com/shorts/1DyYVycop3s?feature=share>

File yang belum tercompile terlampir pada drive.

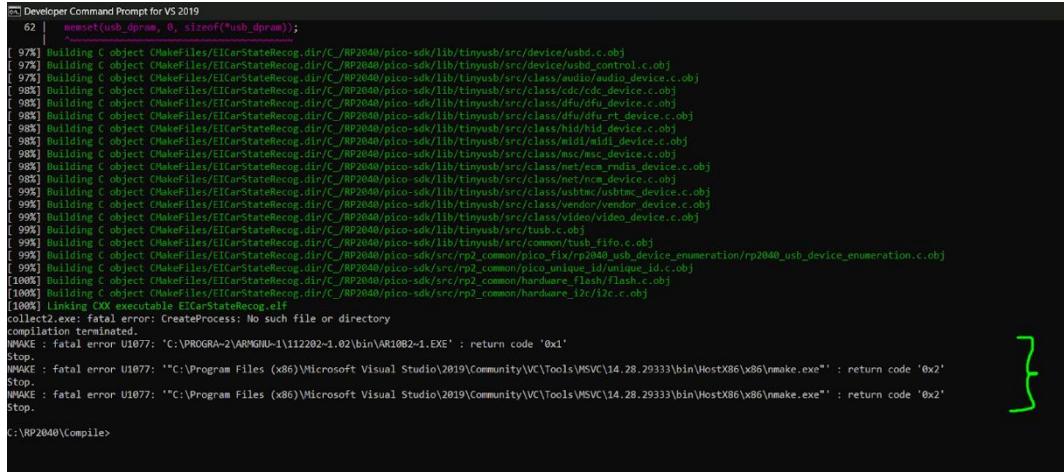
DAFTAR PUSTAKA

- Firman, B., 2016. Implementasi Sensor IMU MPU6050 Berbasis Serial I2C pada Self-Balancing Robot. *Jurnal Teknologi Technoscientia*, pp.18-24.
- Jian, H., 2016, June. Design of angle detection system based on MPU6050. In 7th International Conference on Education, Management, Information and Computer Science (ICEMC 2017) (pp. 6-8). *Atlantis Press*.
- Hakim, M.A.I. dan Putra, Y.H., 2013. Pemanfaatan Mini PC Raspberry Pi Sebagai Pengontrol Jarak Jauh Berbasis WEB Pada Rumah. *Jurusan Teknik Komputer, UNIKOM*.
- Nuruzzamanirridha, M., Dyah, I. dan Hariyani, Y.S., 2016. Implementasi Jaringan Komputer Berbasis Software Defined Network Menggunakan Ryu Controller Dan Openvswitch. *eProceedings of Applied Science*, 2(2).

Lampiran 1. Imu.py dan Vector3d.py code

Download link : <https://github.com/micropython-IMU/micropython-mpu9150.git>

Lampiran 2. Screenshot error compiling



The terminal output shows several errors during the compilation of ElCarStateRecog.exe:

```
62 |     memset(usb_dpram, 0, sizeof(*usb_dpram));
```

```
[97%] Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/device/usbd.c.obj
97%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/device/usbd_control.c.obj
97%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/device/usbd_audio_device.c.obj
98%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/class/dfu/dfu_device.c.obj
98%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/class/dfu/dfu_rt_device.c.obj
98%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/class/hid/hid_device.c.obj
98%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/class/midi/midi_device.c.obj
98%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/class/msc/msc_device.c.obj
98%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/class/net/enum_ndis_device.c.obj
98%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/class/net/nan_device.c.obj
99%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/class/usbm/usbmc_device.c.obj
99%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/class/video/vendor_device.c.obj
99%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/class/video/video_device.c.obj
99%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/common/tusb.c.obj
99%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/common/tusb_fifo.c.obj
99%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/src/rp2_common/pico_fix/rp2040_usb_device_enumeration.c.obj
[99%] Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/src/rp2_common/pico_unique_id/unique_id.c.obj
[100%] Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/src/rp2_common/hardware_flash/flash.c.obj
[100%] Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/src/rp2_common/hardware_i2c/i2c.c.obj
[100%] Linking CXX executable ElCarStateRecog.exe
collect2.exe: fatal error: createProcess: No such file or directory
compilation terminated.
NMAKE : fatal error U1077: 'C:\PROGRA~2\ARMGU~1\112202\1.02\bin\AR1082-1.EXE' : return code '0x1'
Stop.
NMAKE : fatal error U1077: '"C:\Program Files (x86)\Microsoft Visual Studio\2019\Community\VC\Tools\MSVC\14.28.29333\bin\HostX86\x86\nmake.exe"' : return code '0x2'
Stop.
NMAKE : fatal error U1077: '"C:\Program Files (x86)\Microsoft Visual Studio\2019\Community\VC\Tools\MSVC\14.28.29333\bin\HostX86\x86\nmake.exe"' : return code '0x2'
Stop.
```

Following this, the terminal shows a warning about a string overflow:

```
C:/RP2040/compile>memset(usb_dpram, 0, sizeof(*usb_dpram));
```

Then it continues with the compilation of ElCarStateRecog.exe, showing more warnings about string overflows:

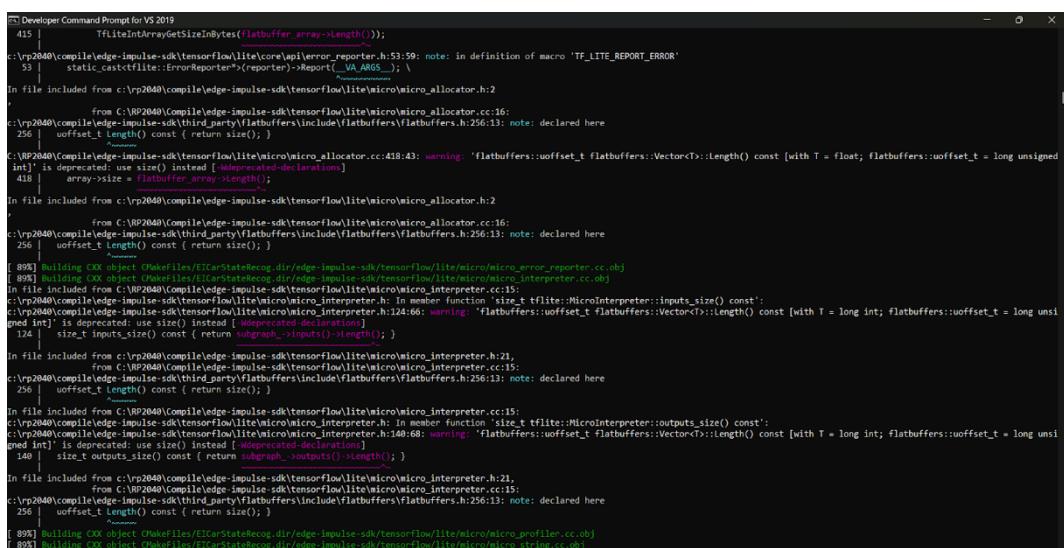
```
[96%] Building CXX object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/portable/raspberry_pi/rp2040/rp2040_usb.c.obj
[96%] Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/portable/raspberry_pi/rp2040/rp2040_usb_init.c.obj
C:/RP2040/pico-sdk/lib/tinyus[...]src/portable/raspberry_pi/rp2040/rp2040_usb.c: In function 'rp2040_rp2040_usb_init':
C:/RP2040/pico-sdk/lib/tinyus[...]src/portable/raspberry_pi/rp2040/rp2040_usb.c:61:3: warning: 'memset' writing 156 bytes into a region of size 0 overflows the destination [-Wstringop-overflow=]
61 |     memset(usb_dpram, 0, sizeof(*usb_dpram));
|     ^~~~~~
```

More warnings follow, including one about a deprecated declaration:

```
C:/RP2040/pico-sdk/lib/tinyus[...]src/portable/raspberry_pi/rp2040/rp2040_usb.c:62:3: warning: 'memset' writing 4996 bytes into a region of size 0 overflows the destination [-Wstringop-overflow=]
```

Finally, the terminal shows a note about a macro definition:

```
[97%] Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/device/usbd.c.obj
97%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/device/usbd_control.c.obj
97%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/device/usbd_audio_device.c.obj
98%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/class/dfu/dfu_device.c.obj
98%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/class/dfu/dfu_rt_device.c.obj
98%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/class/hid/hid_device.c.obj
98%| Building C object CMakelinks/ElCarStateRecog.dir/C:/RP2040/pico-sdk/lib/tinyusb/src/class/midi/midi_device.c.obj
```



The terminal output shows several errors during the compilation of edge-impulse-micromouse-micro_interpreter.cc:

```
415 |     TfliteInTArrayGetSizeInBytes(flatbuffer_array->Length());
```

```
c:/rp2040/compile>edge-impulse-sdk/tensorflow/lite/core/api/error_reporter.h:53:59: note: in definition of macro 'TF_LITE_REPORT_ERROR'
53 |     static __cast_type__(reporter)->ReportError(VA_ARGS...) \
```

In file included from c:/rp2040/compile/edge-impulse-sdk/tensorflow/lite/micro/micro_allocator.h:2:

```
    from C:/RP2040/Compile/edge-impulse-sdk/tensorflow/lite/micro/micro_allocator.cc:16;
```

c:/rp2040/compile/edge-impulse-sdk/third_party/flatbuffers/include/flatbuffers/flatbuffers.h:256:13: note: declared here

```
256 |     uoffset_t Length() const { return size(); }
```

c:/rp2040/compile/edge-impulse-sdk/tensorflow/lite/micro/micro_allocator.cc:418:43: warning: 'flatbuffers::uoffset_t flatbuffers::Vector<T>::Length() const [with T = float; flatbuffers::uoffset_t = long unsigned int]' is deprecated: use size() instead [-Wdeprecated-declarations]
418 | array->size = flatbuffer_array->Length();

In file included from c:/rp2040/compile/edge-impulse-sdk/tensorflow/lite/micro/micro_allocator.h:2:

```
    from C:/RP2040/Compile/edge-impulse-sdk/tensorflow/lite/micro/micro_allocator.cc:16;
```

c:/rp2040/compile/edge-impulse-sdk/third_party/flatbuffers/include/flatbuffers/flatbuffers.h:256:13: note: declared here

```
256 |     uoffset_t Length() const { return size(); }
```

[89%] Building CXX object CMakelinks/ElCarStateRecog.dir/edge-impulse-sdk/tensorflow/lite/micro/micro_error_reporter.cc.obj
[89%] Building CXX object CMakelinks/ElCarStateRecog.dir/edge-impulse-sdk/tensorflow/lite/micro/micro_interpreter.cc.obj

In file included from C:/RP2040/compile/edge-impulse-sdk/tensorflow/lite/micro/micro_interpreter.cc:15:

```
c:/rp2040/compile/edge-impulse-sdk/tensorflow/lite/micro/micro_interpreter.h: In member function 'size_t tflite::MicroInterpreter::inputs_size() const':
```

c:/rp2040/compile/edge-impulse-sdk/tensorflow/lite/micro/micro_interpreter.h:124:66: warning: 'flatbuffers::uoffset_t flatbuffers::Vector<T>::Length() const [with T = long int; flatbuffers::uoffset_t = long unsigned int]' is deprecated: use size() instead [-Wdeprecated-declarations]
124 | size_t inputs_size() const { return subgraph_inputs->Length(); }

In file included from c:/rp2040/compile/edge-impulse-sdk/tensorflow/lite/micro/micro_interpreter.h:21,

```
    from C:/RP2040/Compile/edge-impulse-sdk/tensorflow/lite/micro/micro_interpreter.cc:15;
```

c:/rp2040/compile/edge-impulse-sdk/third_party/flatbuffers/include/flatbuffers/flatbuffers.h:256:13: note: declared here

```
256 |     uoffset_t Length() const { return size(); }
```

[89%] Building CXX object CMakelinks/ElCarStateRecog.dir/edge-impulse-sdk/tensorflow/lite/micro/micro_profiler.cc.obj
[90%] Building CXX object CMakelinks/ElCarStateRecog.dir/edge-impulse-sdk/tensorflow/lite/micro/micro_string.cc.obj

```

[100%] Building C object CMakeFiles/ElCarStateRecog.dir/_/RP2040/pico-sdk/lib/tinyusb/src/class/video/device/device.c.o
[100%] Building C object CMakeFiles/ElCarStateRecog.dir/_/RP2040/pico-sdk/lib/tinyusb/src/class/video/device.cobj
[100%] Building C object CMakeFiles/ElCarStateRecog.dir/_/RP2040/pico-sdk/lib/tinyusb/src/class/video/device_id/video_id.c.o
[100%] Building C object CMakeFiles/ElCarStateRecog.dir/_/RP2040/pico-sdk/src/rp2_common/pico_fix_uefi_id/unique_id.c.o
[100%] Building C object CMakeFiles/ElCarStateRecog.dir/_/RP2040/pico-sdk/src/rp2_common/pico_uniq_uefi_id/unique_id.cobj
[100%] Building C object CMakeFiles/ElCarStateRecog.dir/_/RP2040/pico-sdk/src/rp2_common/hardware_flash/flash.c.o
[100%] Building C object CMakeFiles/ElCarStateRecog.dir/_/RP2040/pico-sdk/src/rp2_common/hardware_flash/flash.cobj
[100%] Linking CXX executable ElCarStateRecog
collect2.exe fatal error: CreateProcess: No such file or directory
compilation terminated.
make : fatal error U087: 'C:\Program Files (x86)\Microsoft Visual Studio\2019\Community\VC\Tools\VSVC\14.28.29333\bin\HostX86\x86\make.exe' : return code '0x1'
Stop.
make : fatal error U087: "C:\Program Files (x86)\Microsoft Visual Studio\2019\Community\VC\Tools\VSVC\14.28.29333\bin\HostX86\x86\make.exe" : return code '0x2'
Stop.
:~/RP2040/02_el_carstate_recog/build>

```

```

cmake] Omake Error at CMakeLists.txt:20 (project):
cmake]   Running
cmake]   'cmake' '--'
cmake]   failed with:
cmake]   The system cannot find the file specified
cmake] -- Configuring incomplete, errors occurred!
cmake] See also "C:/RP2040/02_el_carstate_recog/CMakeFiles/ElCarStateRecog.dir/build/CMakeOutput.log".
cmake] See also "C:/RP2040/02_el_carstate_recog/CMakeFiles/ElCarStateRecog.dir/build/CMakeError.log".

```