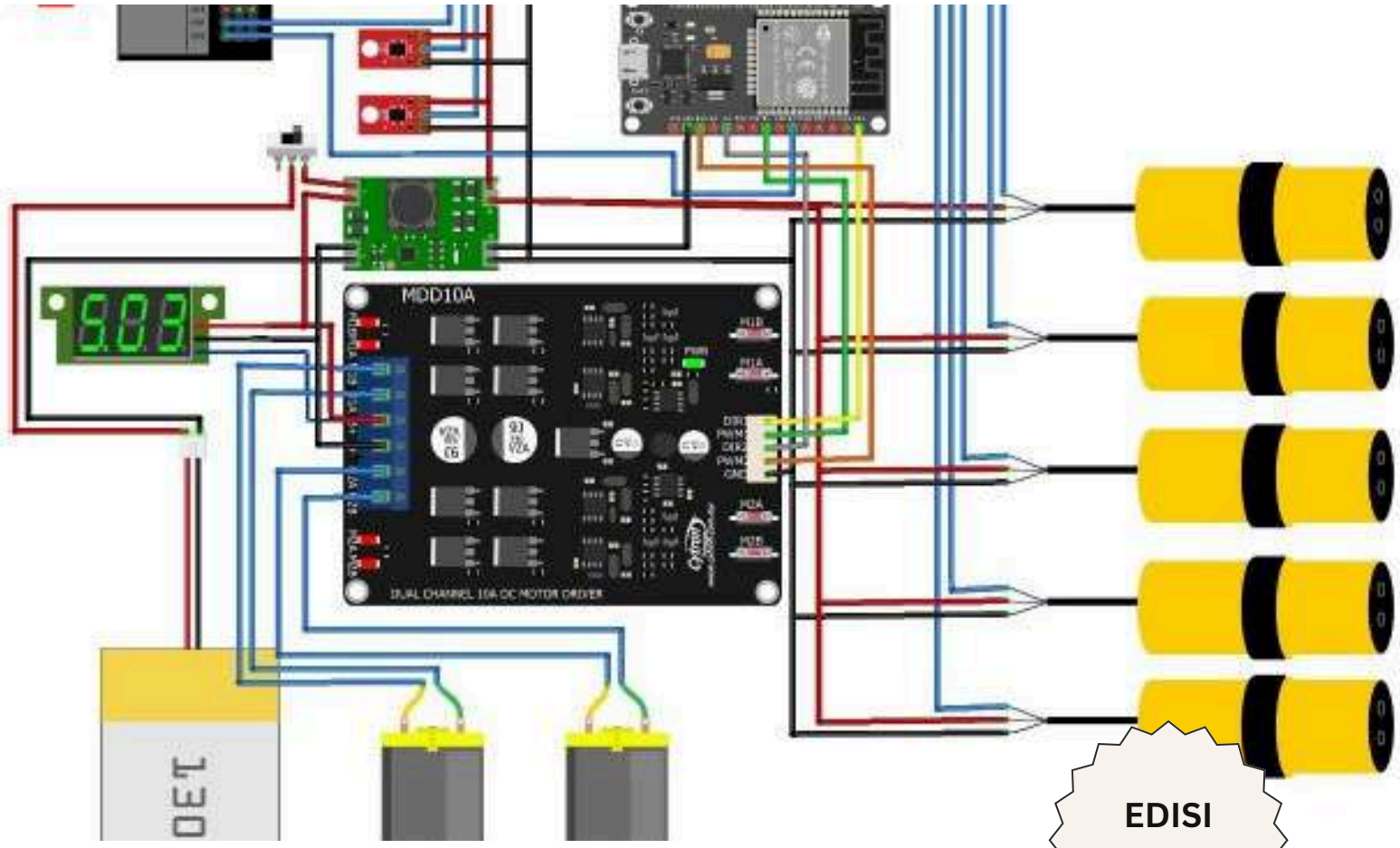


EDISI PERTAMA

Panduan Asas

ARDUINO DAN ESP32

PANDUAN MUDAH BINA PROJEK

BINA PROJEK YANG DIIMPIKAN

JULIYANNA ALIMAN
ARIFFUDDIN IBRAHIM

Panduan Asas

ARDUINO DAN ESP32

Edisi pertama

Edisi 1,2025

Diterbitkan oleh:

POLITEKNIK MERLIMAU MELAKA,
KM2.5, JALAN MERLIMAU-JASIN
77300, MERLIMAU,
MELAKA
Tel : 06-2636687
Fax : 06-2636678
https://pmm.mypolycc.edu.my/

e-ISBN 978-629-7737-16-4

PRAKATA

Buku ini disusun sebagai panduan lengkap untuk pelajar dan pemula yang ingin membangunkan projek berasaskan Arduino. Dengan perkembangan pesat teknologi pengawalmikro dan Internet of Things (IoT), Arduino telah menjadi salah satu platform yang paling digemari kerana kesederhanaan dan fleksibilitinya. Buku ini bertujuan untuk memberikan pemahaman mendalam tentang konsep asas, pemrograman, serta projek-projek praktikal yang boleh diaplikasikan dalam pelbagai bidang melalui kaedah simulasi.

Melalui buku ini, pembaca akan dibimbing langkah demi langkah, bermula dari pengenalan kepada Arduino, pemahaman konsep asas elektronik, penulisan kod pemrograman, sehingga kepada pelaksanaan projek-projek yang boleh diaplikasikan dalam dunia sebenar. Setiap bab dilengkapi dengan contoh-contoh praktikal dan ilustrasi yang jelas untuk memudahkan pemahaman.

Kami berharap buku ini dapat menjadi sumber inspirasi dan rujukan yang berguna bagi semua pembaca, sama ada yang baru mula atau yang sudah mempunyai pengalaman dalam bidang ini. Semoga buku ini dapat membantu anda mencapai kejayaan dalam projek berasaskan Arduino dan membuka pintu kepada lebih banyak inovasi di masa hadapan.

Selamat membaca dan selamat berinovasi!

Senarai

KANDUNGAN

BAB

SATU

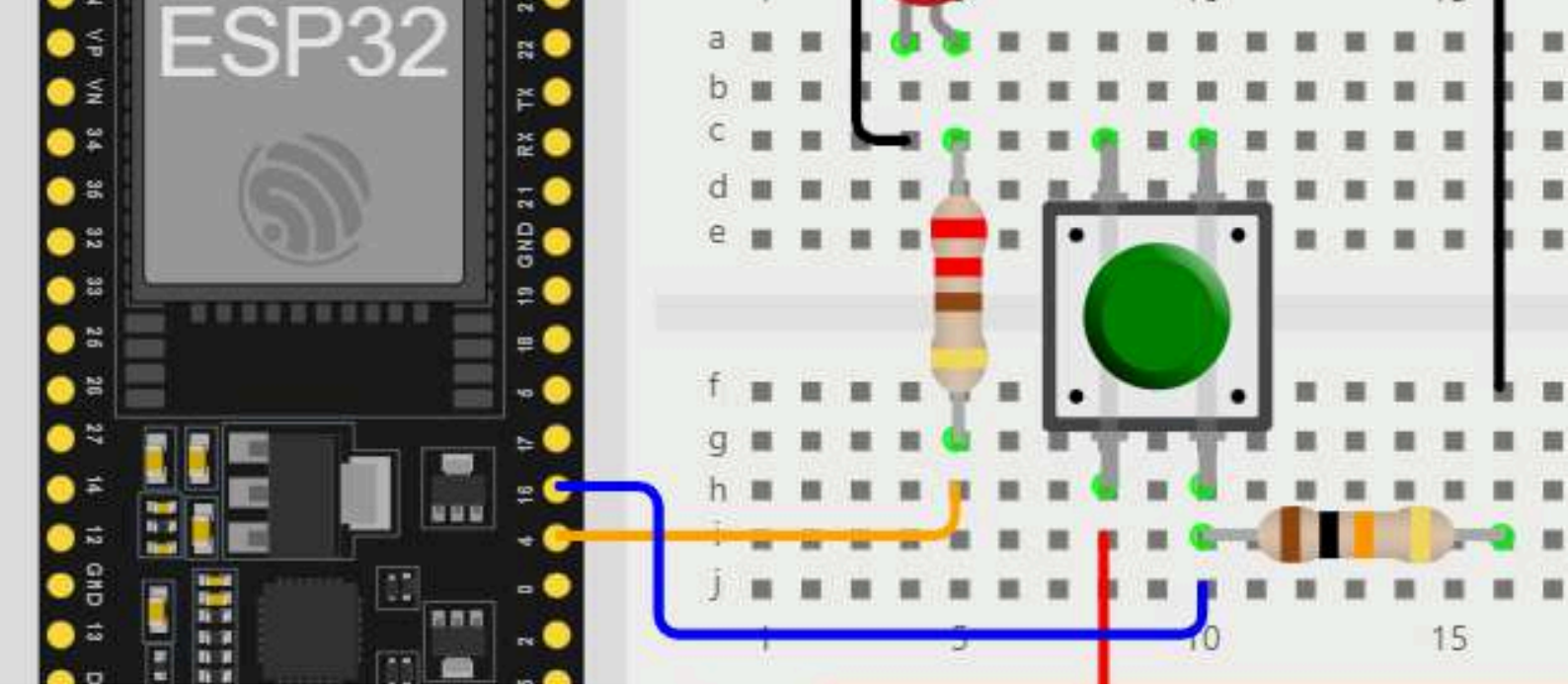
PENGENALAN KEPADA ARDUINO DAN ESP32

3 mukasurat

LET'S DO IT

kandungan bab

Kenali persamaan dan perbezaan arduino dan esp32



APA ITU ARDUINO?

Projek Tahun Akhir (Final Year Project - FYP) merupakan satu komponen penting dalam pengajian teknikal dan kejuruteraan, di mana pelajar perlu mengaplikasikan pengetahuan teori ke dalam projek praktikal. Dalam era Revolusi Perindustrian 4.0 (IR 4.0), teknologi Arduino dan ESP32 menjadi pilihan utama dalam membangunkan sistem automasi, Internet of Things (IoT), dan aplikasi pintar kerana sifatnya yang fleksibel, kos efektif, dan mudah diprogramkan.

Buku ini, "Panduan Asas: Arduino dan ESP32", bertujuan untuk memberikan panduan lengkap kepada pelajar dalam membangunkan projek mereka menggunakan mikropengawal Arduino dan ESP32. Dengan pendekatan yang sistematik, buku ini akan membantu pelajar memahami konsep asas, teknik pengaturcaraan, penyambungan komponen elektronik, dan pelaksanaan sistem secara praktikal.

Antara topik yang akan dibincangkan dalam buku ini termasuklah:

✅ Pengenalan kepada Arduino dan ESP32
✅ Struktur asas kod pengaturcaraan (C/C++ dalam Arduino IDE)
✅ Pemilihan sensor dan modul tambahan untuk projek FYP
✅ Teknik sambungan dan litar elektronik menggunakan breadboard

Buku ini sesuai untuk pelajar bidang kejuruteraan elektronik, mekatronik, robotik, dan sains komputer, serta mereka yang ingin membangunkan projek menggunakan teknologi terbuka (open-source hardware). Dengan adanya panduan ini, diharapkan pelajar dapat membina projek yang inovatif, berfungsi dengan baik, dan selaras dengan keperluan industri.

PERBANDINGAN ARDUINO UNO DAN ESP32

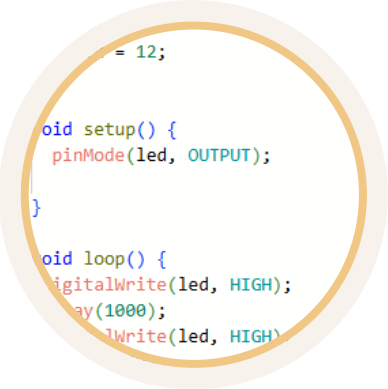
ARDUINO UNO	V S	ESP32
ATMEGA328P	MIKROPEMPROSES	DUAL-CORE TENSILICA XTENSA LX6
16 MHZ	KELAJUAN JAM	160 - 240 MHZ
5V	VOLTAN OPERASI	3.3V
14	BILANGAN I/O DIGITAL	34
6	BILANGAN I/O ANALOG	18
6	PWM PIN	SEMUA I/O DIGITAL
1 PORT	UART (SERIAL)	3 PORT
9600	BAUD RATE DEFAULT	115200
TIDAK	SAMBUNGAN WIFI/BLUETOOTH	YA (TERBINA DALAM)
32KB	MEMORIFLASH	4MB
2KB	SRAM (RAM)	520KB
ATMEGA16U2	MODUL USB KE SERIAL	CP2102/CH340

- JIKA PERLUKAN KOMUNIKASI TANPA WAYAR, GUNAKAN **ESP32**
- JIKA HANYA UNTUK PEMULA DAN ASAS, GUNAKAN **ARDUINO UNO**

Kelebihan
ARDUINO DALAM FYP

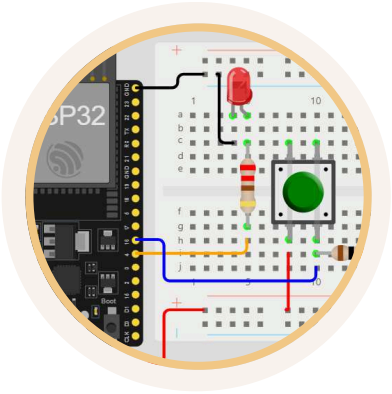
KOMUNITI BESAR DAN SOKONGAN MELUAS

Arduino mempunyai komuniti global yang aktif, membolehkan pelajar mendapatkan bantuan, berkongsi idea, dan menyelesaikan masalah melalui forum, blog, dan video tutorial.



MUDAH DIPROGRAMKAN MENGGUNAKAN ARDUINO IDE

Perisian Arduino IDE mesra pengguna dan menyokong pelbagai perpustakaan, memudahkan pemrograman tanpa perlu menulis kod dari awal.



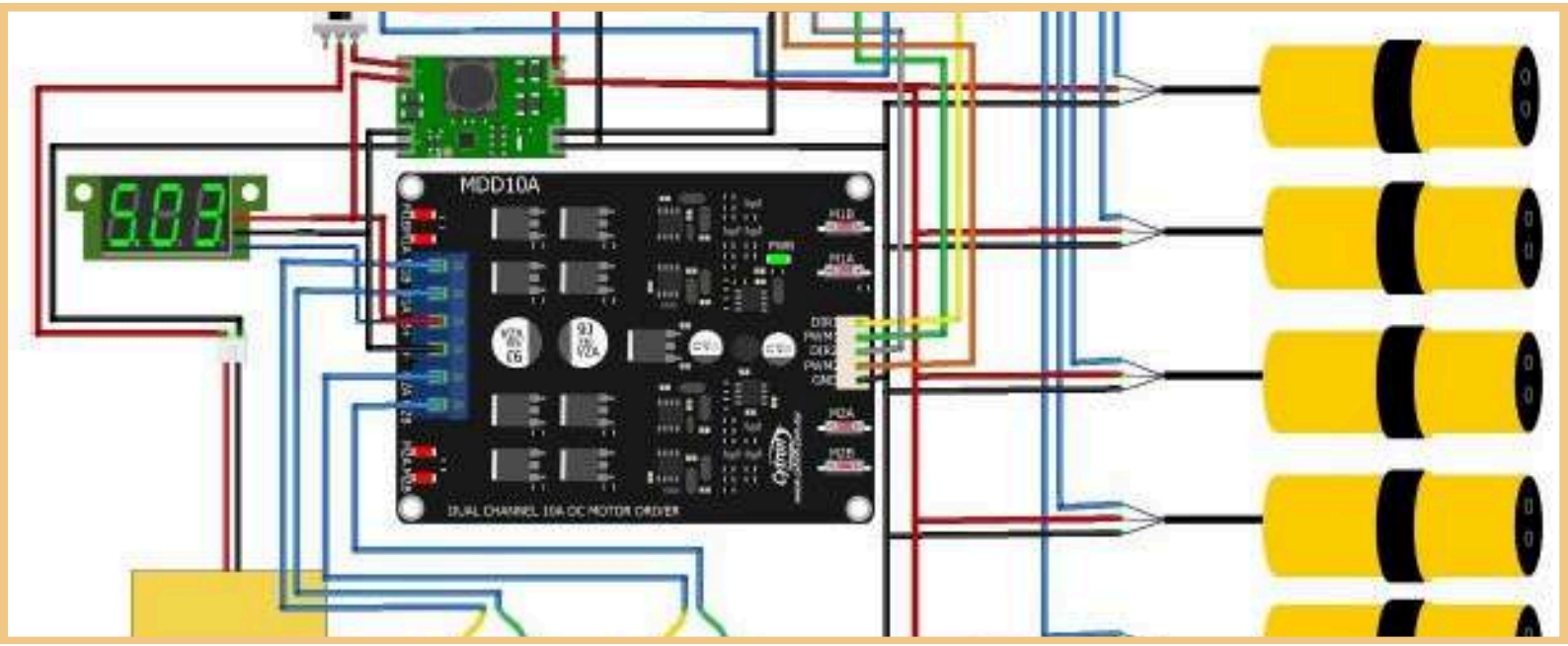
SESUAI UNTUK PROTOTAIP PANTAS

Arduino membolehkan pembangunan prototaip dengan cepat kerana ia menyokong penyambungan modul dan sensor dengan mudah menggunakan *breadboard* dan *jumper wires*, tanpa memerlukan litar bercetak yang kompleks.

PEMBANGUNAN *OPEN SOURCE*

Perkakasan dan perisian Arduino bersifat sumber terbuka (*open-source*), membolehkan pengguna mengubah suai dan menyesuaikan kod serta reka bentuk papan mengikut keperluan projek tanpa sekatan.

BAB



DUA

PERKAKASAN DAN PERISIAN

14 mukasurat

LET'S DO IT

kandungan bab

Mengetahui perkakasan dan perisian yang berkaitan dan diperlukan sepanjang pembuatan projek

Rajah blok asas

APLIKASI PENGAWAL MIKRO

INPUT

Fungsi:

Menerima data atau isyarat dari persekitaran untuk diproses oleh pengawal mikro.

Boleh terdiri daripada sensor atau input kawalan seperti butang tekan dan komunikasi digital.

Contoh Input:

✅ Sensor Suhu (DHT11, DS18B20)
✅ Sensor Ultrasonik (HC-SR04)
✅ Butang Tekan (Push Button)

PROSES

Fungsi:

Memproses data yang diterima dari input dan membuat keputusan berdasarkan program yang telah dimuat naik.

Mengawal output berdasarkan logik yang telah diprogramkan.

Contoh Pengawal Mikro:

✅ Arduino Uno
✅ ESP32
✅ PIC16F877A
✅ STM32

OUTPUT

Fungsi:

Menghasilkan tindak balas berdasarkan pemprosesan data input.

Output boleh berupa kawalan isyarat elektrik, pemaparan maklumat, atau pergerakan mekanikal.

Contoh Output:

✅ LED
✅ Motor Servo
✅ Relay Module
✅Paparan LCD, OLED

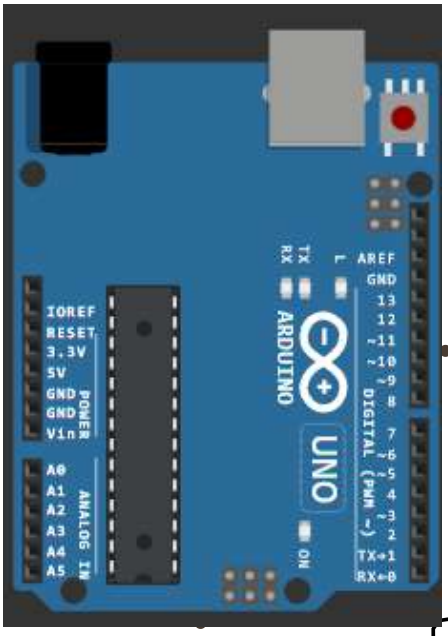
keperluan asas

PERKAKASAN & PERISIAN

1. **PENGAWAL MIKRO**
   Arduino UNO, ESP32
2. **ARDUINO IDE**
   Perisian yang digunakan untuk menulis, menyusun (compile), dan memuat naik kod ke papan pengawal mikro seperti Arduino Uno, dan ESP32
3. **BREADBOARD**
   bersaiz, mini atau penuh
4. **INPUT**
   suis, sensor, joystik
5. **RELAY/ MOTOR DRIVER**
   diperlukan untuk menggerakkan beban berinduktif seperti motor, solenoid
6. **OUTPUT**
   led, oled

ciri penting pin

ARDUINO UNO

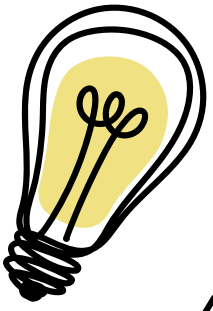


PIN DIGITAL

- Mempunyai 14 pin digital (D0 hingga D13).
- Pin ini boleh digunakan sebagai input atau output.
- 6 pin (D3, D5, D6, D9, D10, D11) menyokong PWM.
- D0 dan D1 digunakan untuk komunikasi serial, jadi elakkan penggunaannya jika Serial Monitor digunakan.

PIN ANALOG

- Mempunyai 6 pin analog (A0 hingga A5).
- Boleh membaca voltan antara 0V hingga 5V.
- Resolusi bacaan 10-bit (0 - 1023).



PWM

- 6 pin PWM (D3, D5, D6, D9, D10, D11).
- Resolusi PWM adalah 8-bit (0 - 255).

GPIO

- Mudah digunakan, tetapi bilangan GPIO terhad.

ciri penting pin

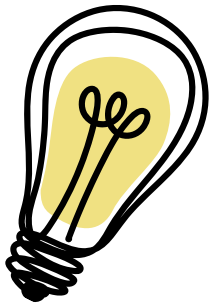
ESP32

PIN DIGITAL

- Mempunyai 34 pin digital yang boleh digunakan sebagai input/output.
- Hampir semua pin boleh digunakan untuk PWM, I2C, SPI, UART.
- Beberapa pin adalah pin khas:
- Pin 0: BOOT mode.
- Pin 6-11: Digunakan untuk memori flash dalaman, tidak boleh digunakan sebagai I/O biasa.
- Pin 34-39: Hanya boleh digunakan sebagai input (tidak boleh output).

PIN ANALOG

- Mempunyai 18 pin analog.
- Boleh membaca voltan antara 0V hingga 3.3V.
- Resolusi bacaan boleh ditetapkan antara 9-bit hingga 12-bit (0 - 4095 pada 12-bit).
- ADC lebih sensitif kepada noise, jadi perlu gunakan penapis jika membaca sensor analog.



PWM

- Semua pin digital boleh digunakan sebagai PWM.
- Resolusi boleh ditetapkan sehingga 16-bit (0 - 65535).
- Frekuensi PWM boleh ditetapkan dari 1Hz hingga 40MHz.

GPIO

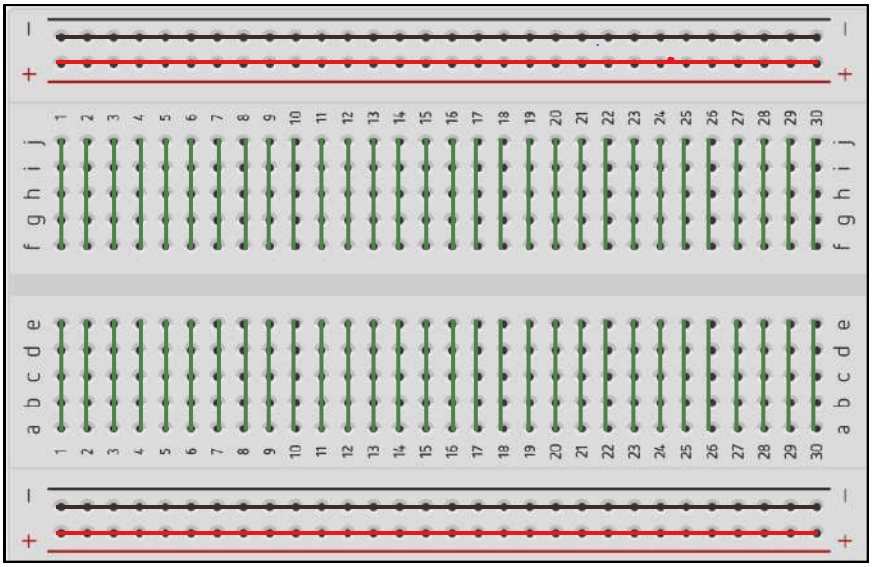
- Jumlah semua 34 GPIO, dengan beberapa pin khas.
- Ada fungsi tambahan seperti Touch Sensor, I2C, SPI, dan UART.
- Tidak semua pin boleh digunakan untuk output, contohnya pin 34-39 hanya boleh digunakan sebagai input.

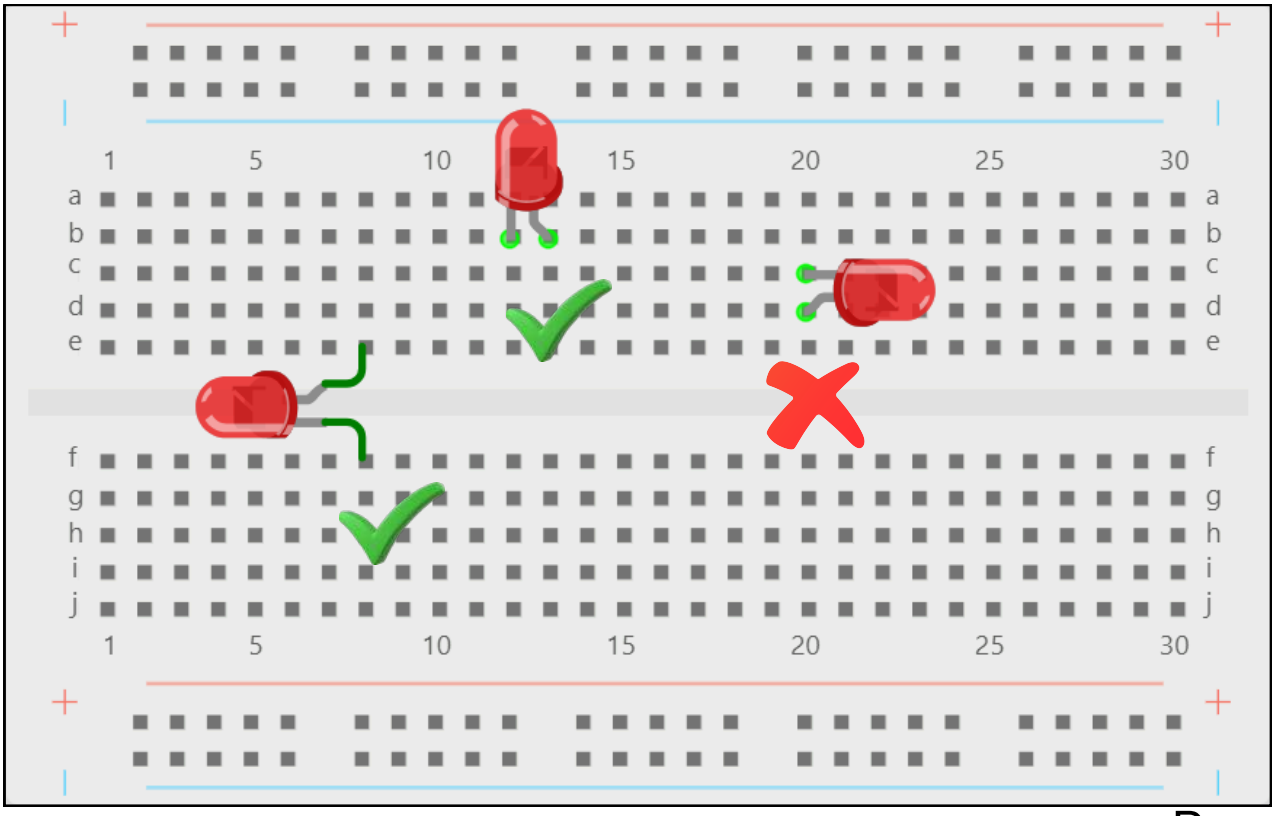
Perkakasan elektronik

BREADBOARD

Breadboard ialah papan litar tanpa solder yang digunakan untuk membina dan menguji litar elektronik tanpa perlu menyambungkan komponen secara kekal.

Ia sangat berguna untuk projek prototaip kerana membolehkan pengguna menukar sambungan dengan mudah sebelum litar dipateri secara tetap.





Perkakasan elektronik

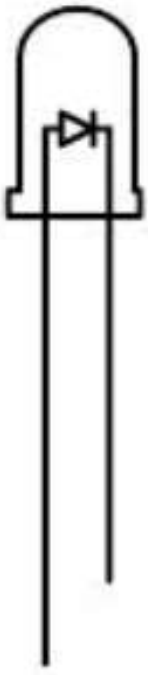
PERINTANG

Perintang (resistor) mengawal aliran arus dalam litar .

Hukum Ohm, menyatakan **V = IR**

Dalam aplikasi Arduino, perintang digunakan untuk mengawal arus ke LED, sensor, dan komponen lain bagi mengelakkan kerosakan. Pemilihan nilai perintang bergantung kepada keperluan voltan dan arus dalam litar.

LED



LED (Light Emitting Diode) adalah komponen elektronik yang memancarkan cahaya apabila dialiri arus elektrik.

Dalam Arduino, LED digunakan sebagai indikator, paparan status, atau elemen dalam projek pencahayaan.

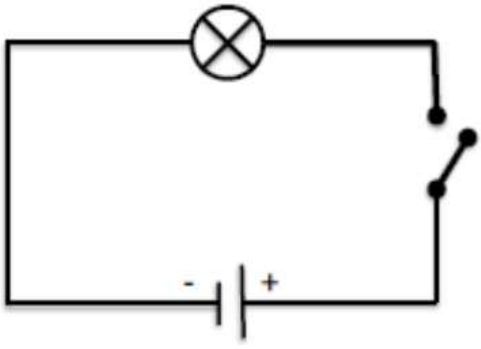
Perkakasan elektronik

SUIS

Suis adalah peranti yang digunakan untuk mengawal aliran arus elektrik dalam satu litar. Fungsi utama suis adalah untuk membuka atau menutup litar, membolehkan atau menghalang arus daripada mengalir.

Suis bekerja dengan cara membuka atau menutup litar:

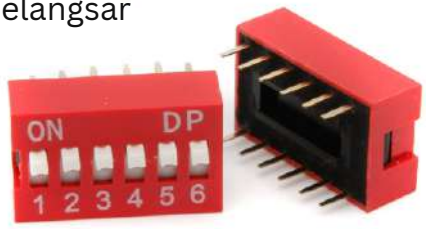
Buka Litar (Open Circuit): Apabila suis terbuka, arus tidak boleh mengalir, dan peranti tidak akan berfungsi.
Tutup Litar (Closed Circuit): Apabila suis ditutup, arus mengalir dalam litar dan membolehkan peranti berfungsi.



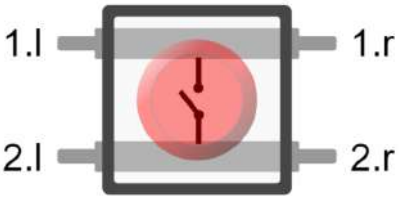
Suis buka

Suis Gelangsar

Suis butang tekan



Suis DIP



Sesentuh dalaman suis butang tekan empat kaki

Perkakasan elektronik

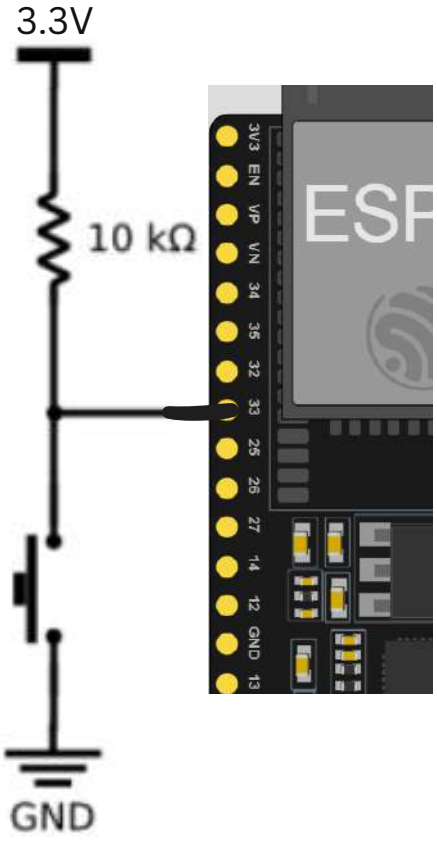
SUIS

Terdapat 2 jenis penyambungan suis yang memeberikan kesan berbeza apabila suis ditekan atau dilepaskan, iaitu:

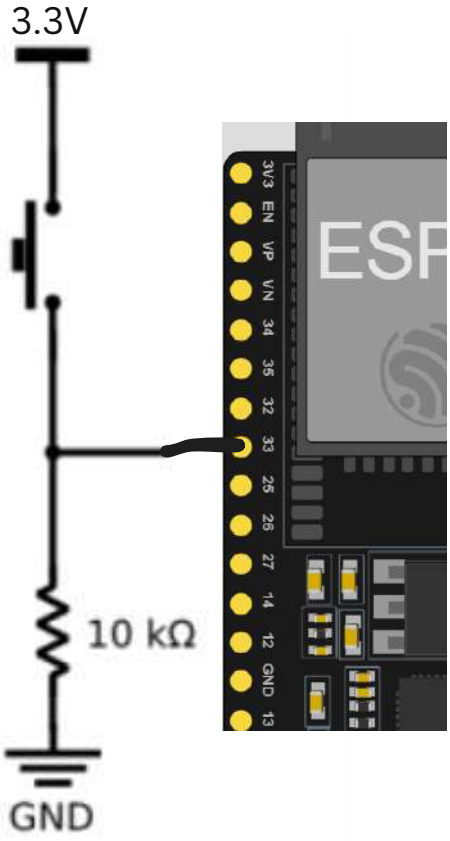
- Suis Aktif Tinggi
- Suis Aktif Rendah

Suis Aktif Tinggi:
Suis menghantar isyarat HIGH (5V/3.3V) apabila ditekan dan LOW (0V) apabila tidak ditekan.
Digunakan apabila kita mahu sistem bertindak balas apabila suis ditekan.

Suis Aktif Rendah:
Suis menghantar isyarat LOW (0V) apabila ditekan dan HIGH (5V/3.3V) apabila tidak ditekan.
Digunakan dalam sistem keselamatan atau apabila kita mahu sistem bertindak balas apabila suis dilepaskan.



Suis Aktif Rendah



Suis Aktif Tinggi

bantuan perisian

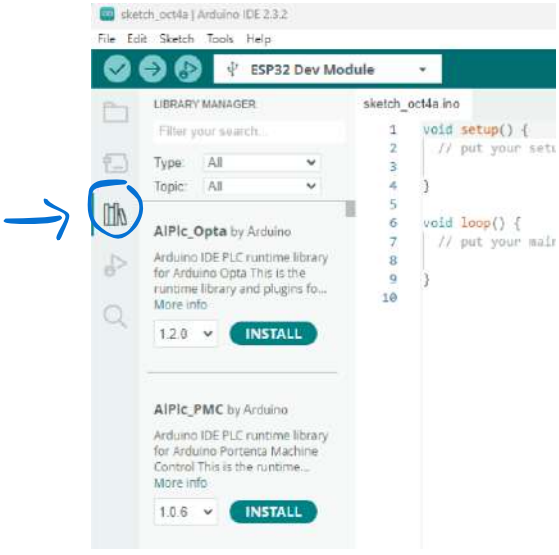
PENGGUNAAN *LIBRARY MANAGER*

Library Manager dalam Arduino IDE adalah alat yang membolehkan pengguna memuat turun, mengemas kini, dan menguruskan pustaka (libraries) dengan mudah. Banyak komponen elektronik memerlukan fail sokongan dari *Library Manager* sekiranya digunakan.

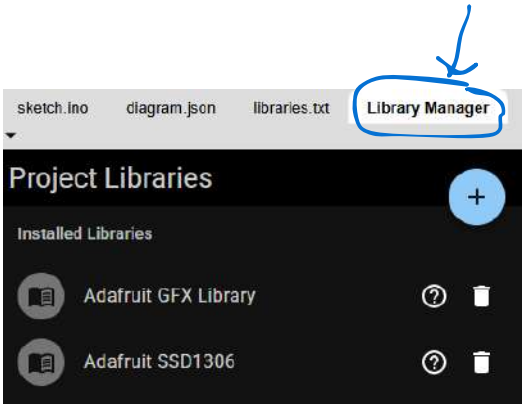
📌 Fungsi Utama *Library Manager*
✅ Memudahkan pemasangan pustaka – Tidak perlu memuat turun fail secara manual.
✅ Mengemas kini pustaka secara automatik – Pastikan versi terkini digunakan.
✅ Menyediakan pustaka rasmi dan pihak ketiga – Contoh: Adafruit, SparkFun, dan DFRobot.

📌 Cara Menggunakan *Library Manager*
1️⃣ Buka Arduino IDE
2️⃣ Pergi ke Sketch → Include Library → Manage Libraries...
3️⃣ Taip nama pustaka yang diperlukan dalam kotak carian.
4️⃣ Klik Install untuk memuat turun pustaka ke dalam Arduino IDE.
5️⃣ Gunakan #include <library_name.h> dalam kod untuk mengaktifkan pustaka.

Contoh pustaka popular:
🔹 Adafruit SSD1306 → Untuk OLED Display
🔹 DHT.h → Untuk sensor suhu & kelembapan
🔹 Servo.h → Untuk mengawal servo motor



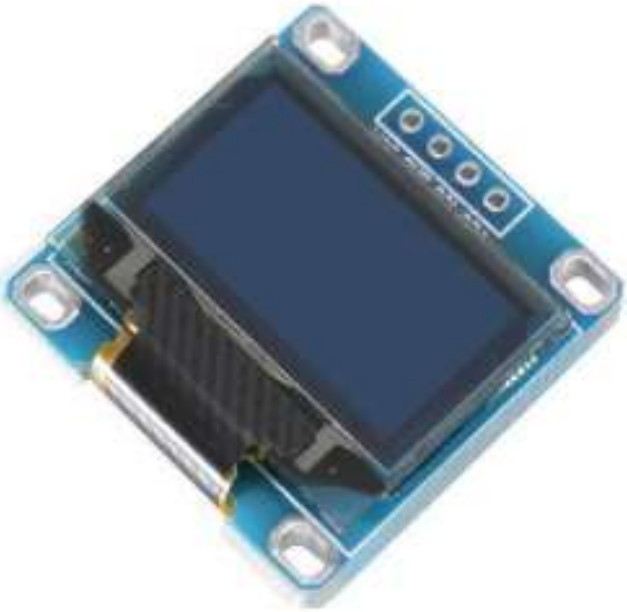
Library manager dalam Arduino IDE



Library manager dalam Wokwi

Perkakasan elektronik

OLED



OLED (Organic Light Emitting Diode) ialah teknologi paparan yang menggunakan bahan organik untuk menghasilkan cahaya apabila arus elektrik dialirkan. Ia digunakan secara meluas dalam paparan skrin kecil untuk projek elektronik seperti Arduino dan ESP32.

Ciri-Ciri OLED:
✅ Tidak memerlukan lampu latar – Berfungsi secara sendiri tanpa pencahayaan tambahan seperti LCD.
✅ Kontras tinggi dan jimat tenaga – Piksel boleh dimatikan sepenuhnya untuk menjimatkan tenaga.
✅ Skrin nipis dan fleksibel – Boleh dihasilkan dalam pelbagai saiz dan bentuk.

Jenis OLED yang Popular untuk Arduino & ESP32:
OLED 0.96 inci (128x64 atau 128x32 piksel) – Menggunakan komunikasi I2C atau SPI.
OLED Monochrome (Hitam-Putih) atau Warna – Sesuai untuk paparan teks dan grafik ringkas.
Kelebihan OLED Berbanding LCD:
✓ Lebih cerah & tajam – Piksel menyala secara individu.
✓ Sudut pandangan lebih luas – Paparan jelas dari pelbagai sudut.
✓ Kurang penggunaan tenaga – Hanya piksel yang diperlukan akan menyala.

- Penggunaan OLED memerlukan pustaka (library) supaya ESP32 atau Arduino dapat berkomunikasi dengan skrin OLED dengan lebih mudah.
- Untuk paparan OLED 0.96 inci (128x64) menggunakan I2C, dua library utama digunakan dalam Arduino IDE:

1️⃣ Adafruit SSD1306
2️⃣ Adafruit GFX

Perkakasan elektronik

LDR SENSOR

LDR (Light Dependent Resistor) ialah sejenis perintang (resistor) yang berubah nilai rintangannya berdasarkan keamatan cahaya yang jatuh ke atasnya.

📌 Ciri-Ciri Sensor LDR:
✅ Apabila cahaya meningkat, rintangan berkurang → Arus lebih mudah mengalir.
✅ Apabila cahaya berkurang (gelap), rintangan meningkat → Arus lebih sukar mengalir.
✅ Boleh digunakan untuk mengesan tahap pencahayaan dalam persekitaran.

📌 Contoh Aplikasi Sensor LDR:
🔹 Lampu Jalan Automatik – Hidup pada waktu malam dan padam pada siang hari.
🔹 Sistem Pengesanan Cahaya – Contohnya, kawalan automatik tirai atau panel suria.
🔹 Robot Pengikut Cahaya – Robot yang bergerak ke arah sumber cahaya.

📌 Cara Menggunakan LDR dengan ESP32 atau Arduino:
🔹 Output Analog (A0) → Memberikan bacaan voltan berkadar dengan intensiti cahaya.
🔹 Output Digital (D0, melalui modul LDR) → Memberikan HIGH (1) atau LOW (0) bergantung pada nilai ambang (threshold).

maklumat lanjut: https://docs.wokwi.com/parts/wokwi-photoresistor-sensor

Perkakasan elektronik

DHT 22

DHT22 ialah sensor digital yang digunakan untuk mengukur kelembapan (%) dan suhu (°C). Ia lebih tepat dan stabil berbanding model DHT11.

📌 Ciri-Ciri Sensor DHT22:
✅ Julat Suhu: -40°C hingga 80°C (ketepatan ±0.5°C)
✅ Julat Kelembapan: 0% hingga 100% (ketepatan ±2-5%)
✅ Jenis Output: Digital (bukan analog)
✅ Mempunyai sensor terbina dalaman dan tidak memerlukan litar tambahan
✅ Kelebihan: Lebih tepat dan mempunyai julat lebih luas berbanding DHT11

📌 Pin & Penyambungan Sensor DHT22
Pin Sensor DHT22 Fungsi Sambungan ke ESP32
VCC Bekalan Kuasa (3.3V - 5V) 3.3V / 5V ESP32
Data Isyarat Digital (Suhu & Kelembapan) Sambung ke GPIO ESP32 (contoh: GPIO 23)
NC (Not Connected) Tidak digunakan -
GND Sambungan ke Tanah GND ESP32

📌 Contoh Aplikasi Sensor DHT22
🔹 Sistem Kawalan Kelembapan & Suhu – Contohnya, rumah hijau pintar (smart greenhouse).
🔹 Pemantauan Cuaca – Digunakan dalam stesen cuaca DIY.
🔹 Penghawa Dingin Pintar – Mengawal kipas atau penghawa dingin secara automatik.

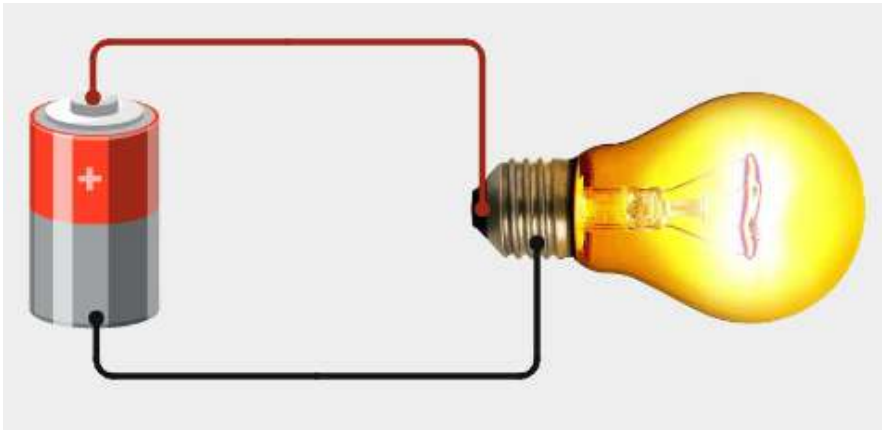
maklumat lanjut: https://docs.wokwi.com/parts/wokwi-dht22

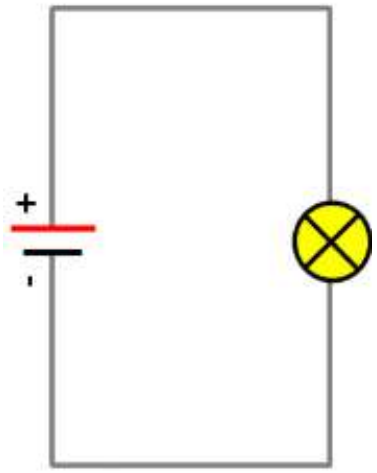
litar elektronik

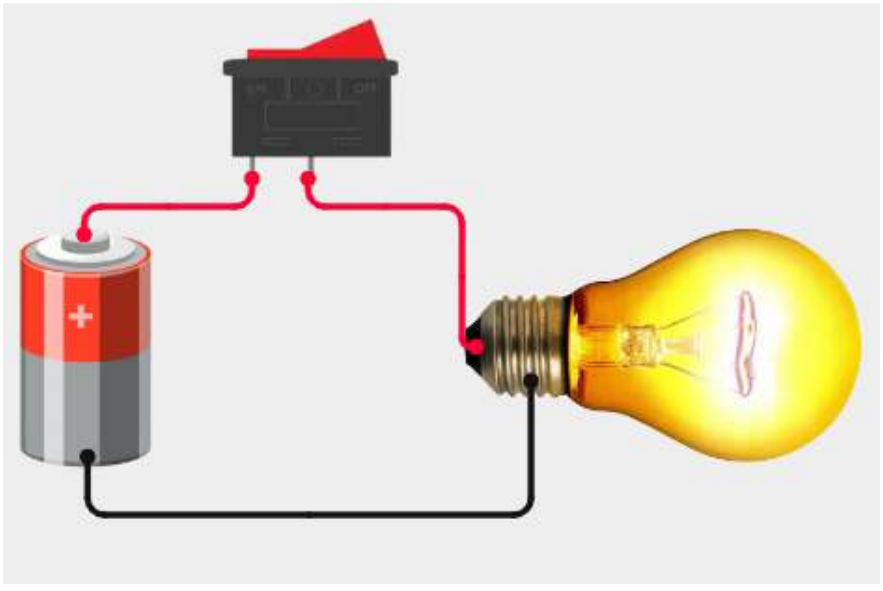
ASAS

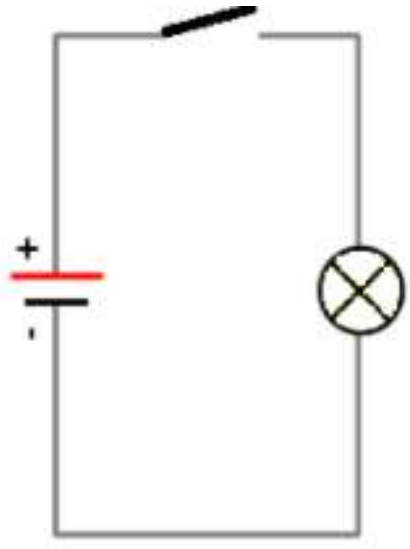
Litar asas elektrik terdiri daripada bateri sebagai sumber kuasa dan lampu sebagai beban. Apabila kedua-dua komponen ini disambungkan dengan wayar secara lengkap (tertutup), arus elektrik mengalir dari terminal positif bateri ke lampu dan kembali ke terminal negatif, menyebabkan lampu menyala.

Dalam litar kedua, suis ditambah untuk mengawal aliran arus. Apabila suis ditutup (ON), litar menjadi lengkap dan lampu menyala. Sebaliknya, apabila suis dibuka (OFF), litar terputus dan lampu padam.





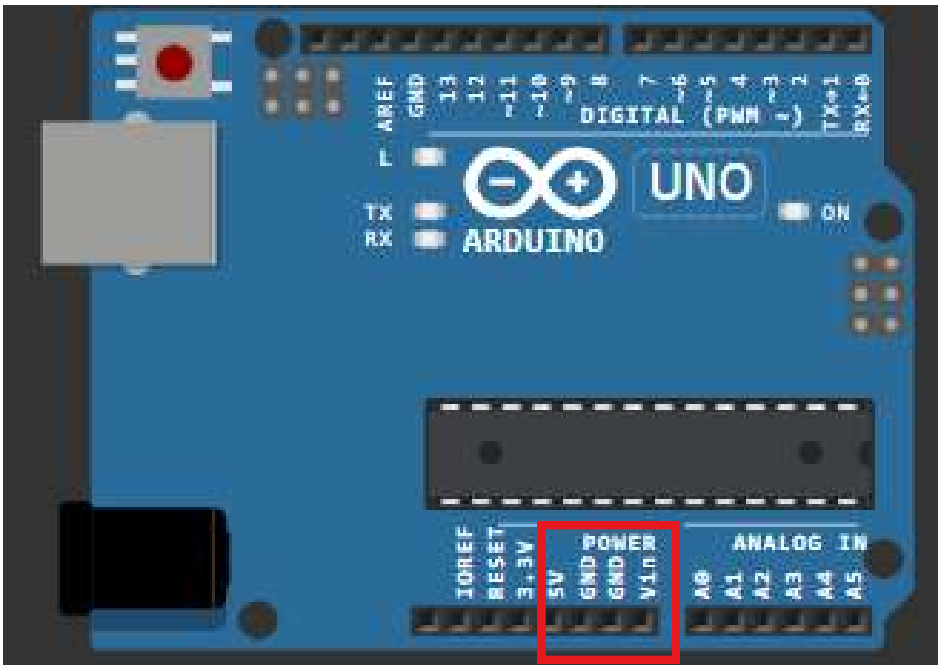




ASAS

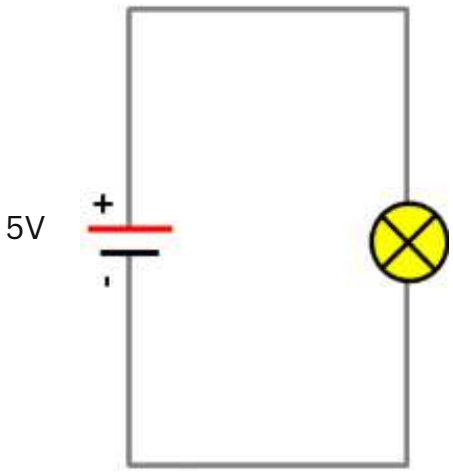
CONTOH 1: ON LED TANPA PROGRAM

Bina litar berikut menggunakan arduino dan breadboard di dalam perisian Wokwi.
Lihat keadaan LED



Gunakan Port POWER “5V” dan “GND” pada Arduino

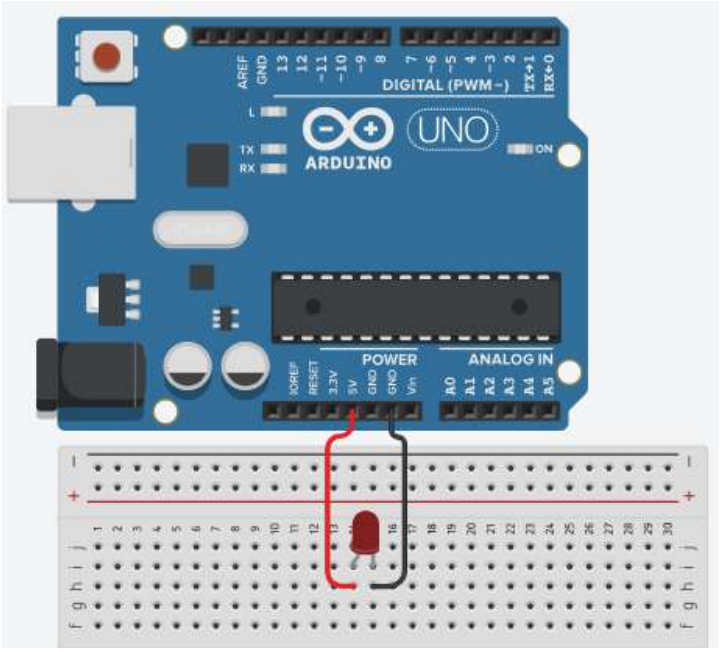
GND = Ground



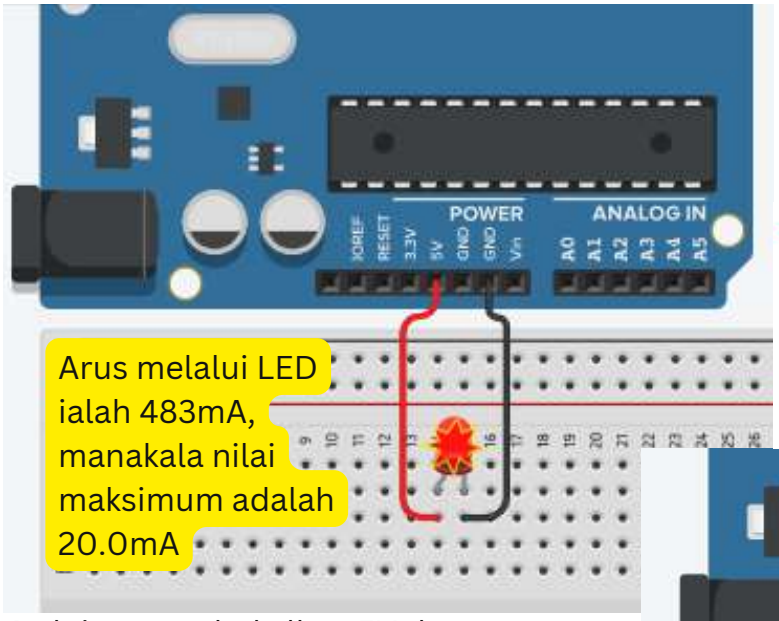
litar elektronik

ASAS

JAWAPAN CONTOH 1: ON LED TANPA PROGRAM

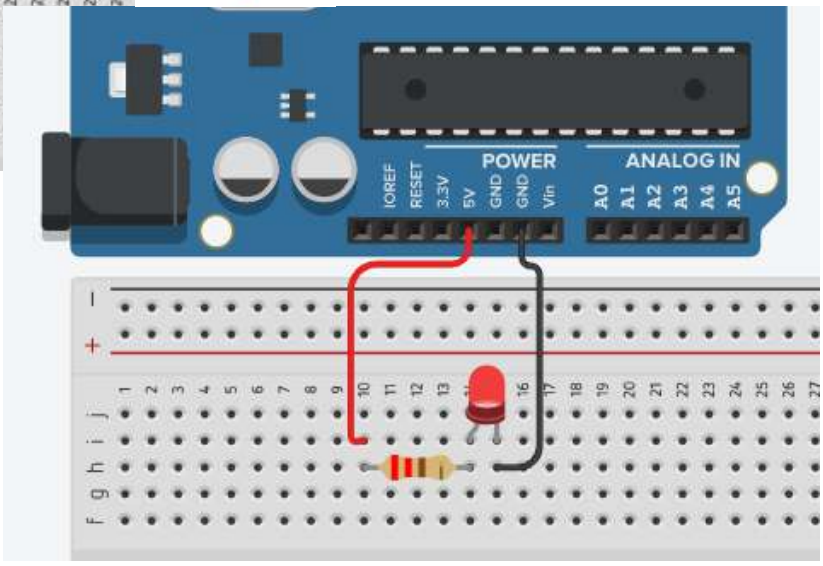


JIka nilai arus terlalu tinggi, LED akan musnah. Bagi mengelakkan masalah ini. resistor digunakan untuk menghadkan jumlah arus yang mengalir di dalam litar.



Arus melalui LED ialah 483mA, manakala nilai maksimum adalah 20.0mA

Untuk mengatasi isu ini, pasangkan perintang yang sesuai untuk menghadkan arus.



Arduino membekalkan 5V dan arus diperlukan LED ialah 20mA. maka, dengan

R= V/I

5V/0.02A=250 Ohm

Bolehlah memilih nilai rintangan 250 Ohm dan yang berhampiran untuk digunakan.

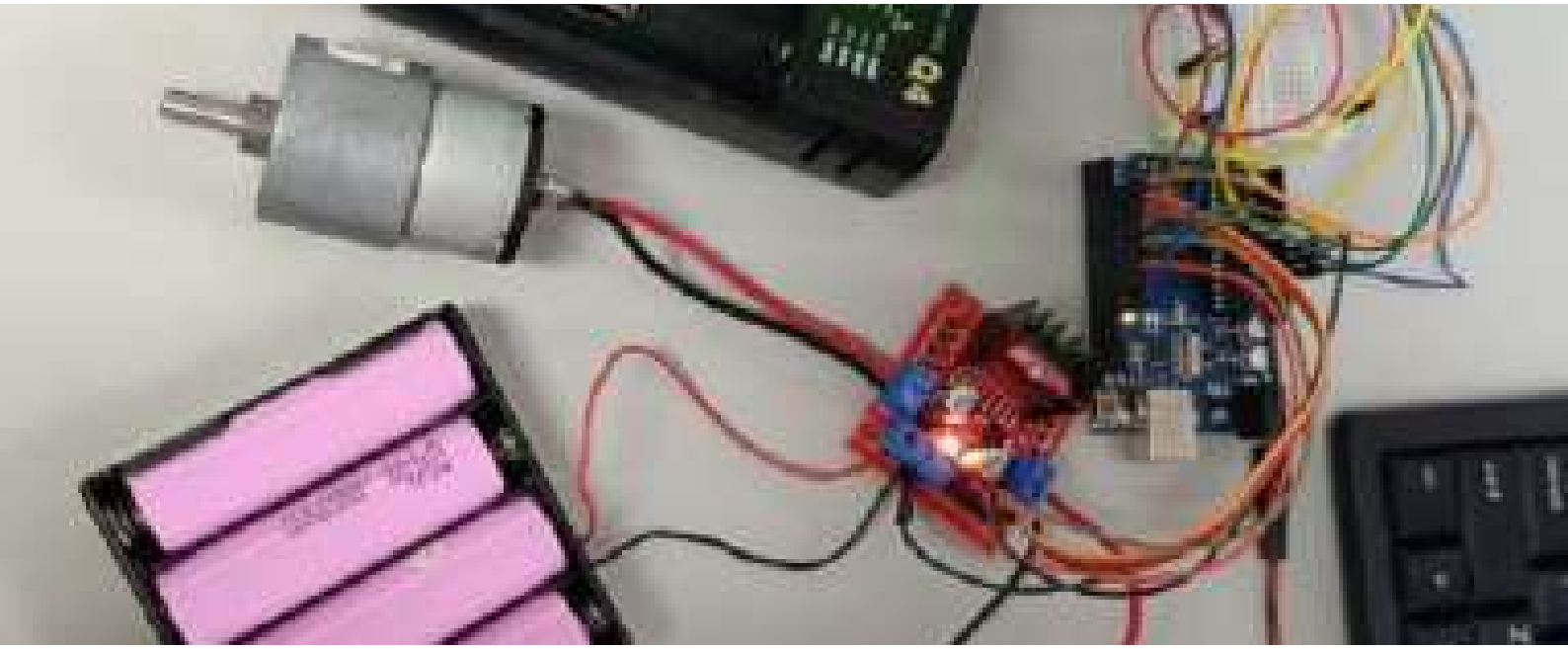
ESP32 membekalkan 3.3V dan arus diperlukan LED ialah 20mA. maka, dengan

R= V/I

3.3V/0.02A=165 Ohm

Bolehlah memilih nilai rintangan 165 Ohm dan yang berhampiran untuk digunakan.

BAB



TIGA

ASAS PEMPROGRAMAN & PENDAWAIAN ARDUINO

4 PAGES
CHECKLIST

LET'S DO IT

kandungan bab

Memahami struktur asas program yang penting dan mesti digunakan berulang kali

Struktur ASAS KOD

Dalam dunia pengaturcaraan Arduino, memahami struktur asas kod adalah langkah pertama untuk membangunkan projek yang berfungsi dengan baik. Sama seperti membina rumah yang memerlukan asas kukuh, kod Arduino juga perlu disusun dengan format yang betul agar ia dapat dikendalikan oleh mikropengawal tanpa ralat.

Kod Arduino terdiri daripada **tiga** bahagian utama:

Bahagian Deklarasi
Digunakan untuk mengisytiharkan pembolehubah, perpustakaan, dan pin yang akan digunakan dalam program.

Fungsi setup()

- Fungsi ini dijalankan sekali sahaja ketika papan Arduino dihidupkan atau di-reset.
- Digunakan untuk menetapkan mod pin (INPUT/OUTPUT), memulakan komunikasi serial, atau konfigurasi sensor.

Fungsi loop()

- Fungsi ini akan dijalankan berulang kali selepas setup() selesai.
- Digunakan untuk mengawal komponen elektronik seperti LED, sensor, atau motor dalam projek Arduino.

```
//struktur asas kod esp32


int led = 12;


void setup() {
  pinMode(led, OUTPUT);
}



void loop() {
 digitalWrite(led, HIGH);
 delay(1000);
 digitalWrite(led, LOW);
 delay(1000);
}
```

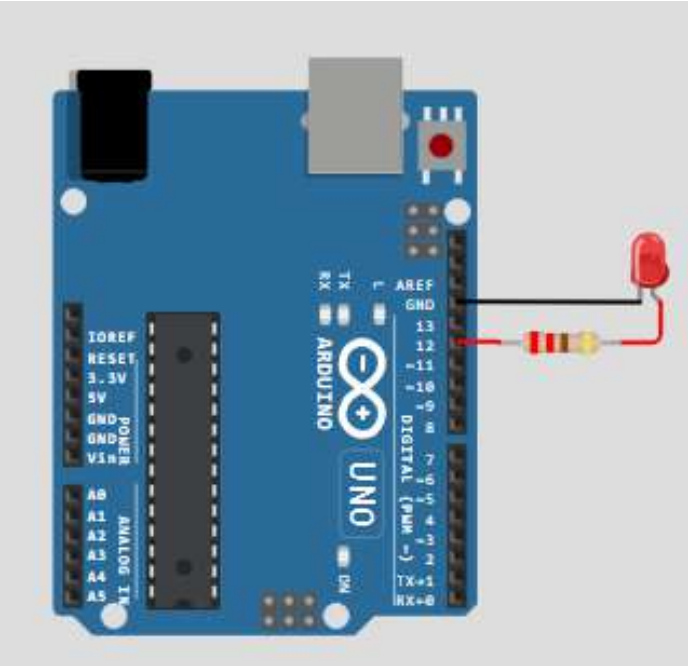
kaitan antara kod program dan

SAMBUNGAN LITAR

Arduino berfungsi sebagai pengawal utama dalam sistem elektronik, di mana kod yang ditulis dalam Arduino IDE akan mengawal komponen fizikal seperti LED. Untuk memastikan sistem berfungsi dengan baik, kod yang ditulis harus sesuai dengan pemasangan dan sambungan litar pada papan Arduino.

Komponen yang digunakan:
1 x Arduino UNO
1 x LED
1 x Resistor 220Ω
2 x Jumper wire

Pemasangan Litar:
Sambungkan kaki anod (positif) LED ke pin digital 12 pada Arduino melalui resistor 220Ω.
Sambungkan kaki katod (negatif) LED ke GND (ground) Arduino.




```
//struktur asas kod arduino uno

int led = 12;


void setup() {
  pinMode(led, OUTPUT);

}

void loop() {
 digitalWrite(led, HIGH);
 delay(1000);
 digitalWrite(led, HIGH);
 delay(1000);
}
```

simulasi lanjut: https://wokwi.com/projects/422899971350437889

kaitan antara kod program dan

SAMBUNGAN LITAR

ESP32 juga boleh disambungkan dengan LED dan resistor menggunakan kod yang sama dengan contoh di atas. Walaubagaimanapun, pengguna mestilah memahami persamaan dan perbezaan antara ESP32 dan Arduino Uno. Kedudukan, pernomboran dan fungsi pin bagi kedua-dua pengawalmikro adalah tidak sama.

Komponen yang digunakan:
1 x ESP32
1 x LED
1 x Resistor 220Ω
2 x Jumper wire

Pemasangan Litar:
Sambungkan kaki anod (positif) LED ke pin digital 12 pada ESP32 melalui resistor 220Ω.
Sambungkan kaki katod (negatif) LED ke GND (ground) ESP32


```
//struktur asas kod esp32


int led = 12;


void setup() {
  pinMode(led, OUTPUT);
}



void loop() {
 digitalWrite(led, HIGH);
 delay(1000);
 digitalWrite(led, LOW);
 delay(1000);
}
```

Bagi memudahkan pembinaan projek akan datang, kebanyakan contoh yang dinyatakan adalah menggunakan ESP32

simulasi lanjut: https://wokwi.com/projects/422849316883693569

Input Output

ANALOG & DIGITAL

ISYARAT	OUTPUT	
	DIGITAL	ANALOG
	digitalWrite(LEDMERAH, HIGH); analogWrite(LEDMERAH, 185);	

	INPUT	
	DIGITAL	ANALOG
	digitalRead(SUIS); analogRead(LDR);	

Digital I/O= melibatkan komponen yang mempunyai 2 status seperti on@off, HIGH@LOW, menyala@padam dan seumpamanya. Cth: suis togol, LED on/off

Analog I/O= melibatkan komponen yang mempunyai status dari 0 hingga 255, tahap kecerahan cahaya yang pelbagai, tahap kelajuan motor yang pelbagai dan seumpamanya.Cth: sensor LDR, LED malap ke cerah

Susunan asas kod

DIGITAL INPUT & OUTPUT

DIGITAL OUTPUT

```
// declaration section
int outputname = pinnumber;



//setup section
void setup()
{
  pinMode(outputname, OUTPUT);
}

//main program or loop section
void loop()
{
  digitalWrite(outputname, MODE);
  }
```

//outputname refer to the output used in the circuit
//pinnumber refer to arduino number pin attached with the output
//MODE refer to HIGH=1, or LOW=0.
//MODE also refer to valinput if desired same status with digital input

DIGITAL INPUT

```
// declaration section
int inputname = pinnumber;
int valinput = 0;//variable to store the read value at initial which is 0

//setup section
void setup()
{
  pinMode(inputname, INPUT);
}

//main program or loop section
void loop()
{
  valinput=digitalRead(inputname);
//read the input pin
}
```

//inputname refer to the input used in the circuit
//pinnumber refer to arduino number pin attached with the input
//valinput must be include as to test the state of input

Rujukan: https://www.arduino.cc/reference/en/

Susunan asas kod

ANALOG INPUT & OUTPUT

ANALOG OUTPUT

```
// declaration section
int outputname = pinnumber;




//setup section
void setup()
{
  pinMode(outputname, OUTPUT);
}


//main program or loop section
void loop()
{
   analogWrite(outputname,value);
  //value from 0 to 255
}
```

//outputname refer to the output used in the circuit
//pinnumber refer to arduino number pin attached with input
//value must be between 0 until 255 fou UNO and ESP32

ANALOG INPUT

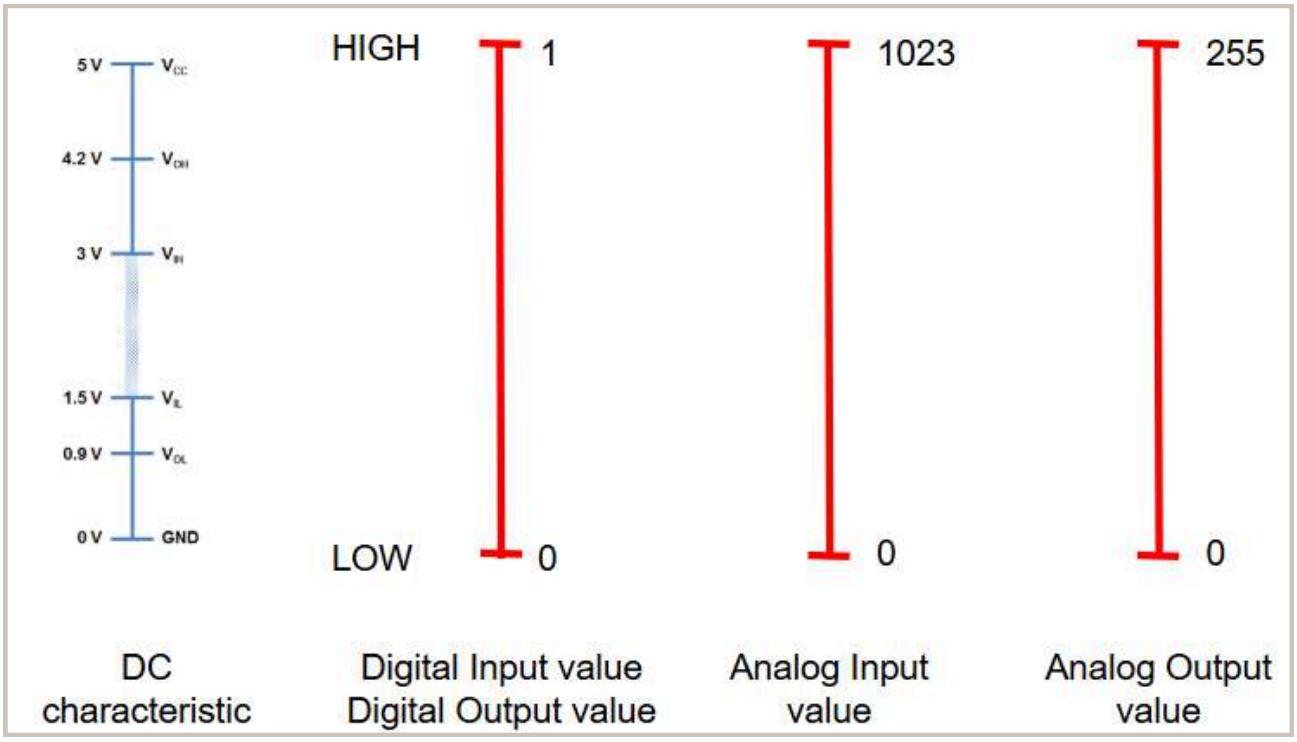
```
// declaration section
int inputname = pinnumber;
int valinput = 0;
//variable to store the read value at
//initial which is 0

//setup section
void setup()
{
  pinMode(inputname, INPUT);
}


//main program or loop section
void loop()
{
  valinput=analogRead(inputname);
   //read the input pin
}
```

//inputname refer to the input used in the circuit
//pinnumber refer to arduino number pin attached with input
//valinput must be include as to test the state of input

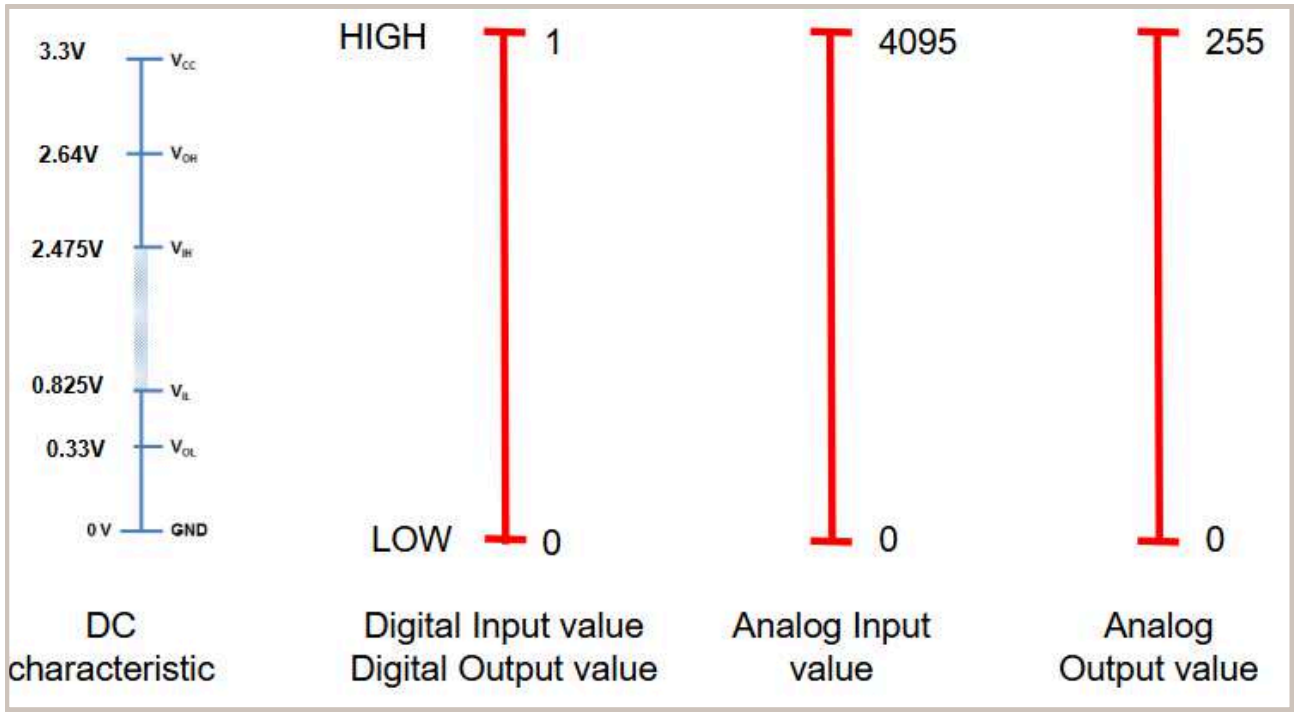
Rujukan: https://www.arduino.cc/reference/en/

Perbandingan ciri

DC ANALOG & DIGITAL

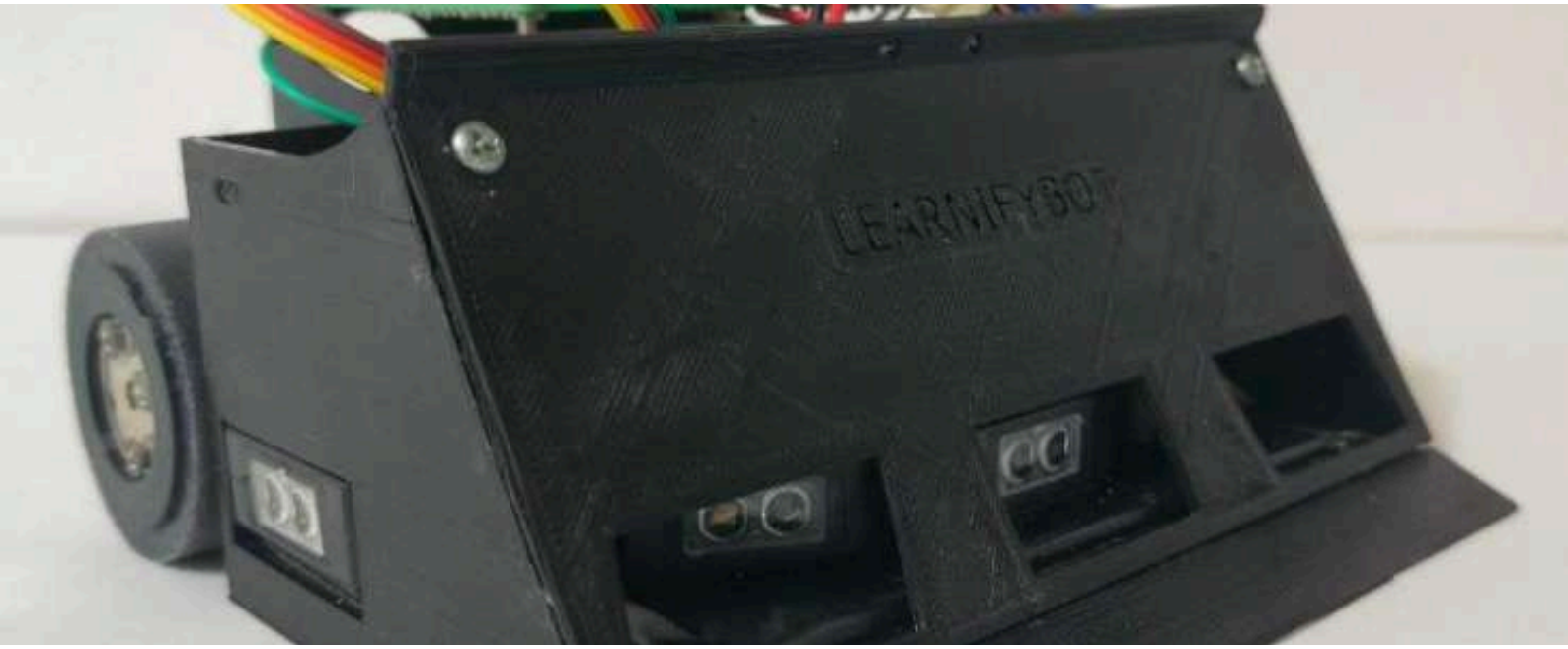


ARDUINO UNO



ESP 32

BAB



EMPAT

PROJEK ASAS ARDUINO

12
mukasurat

LET'S DO IT

kandungan bab

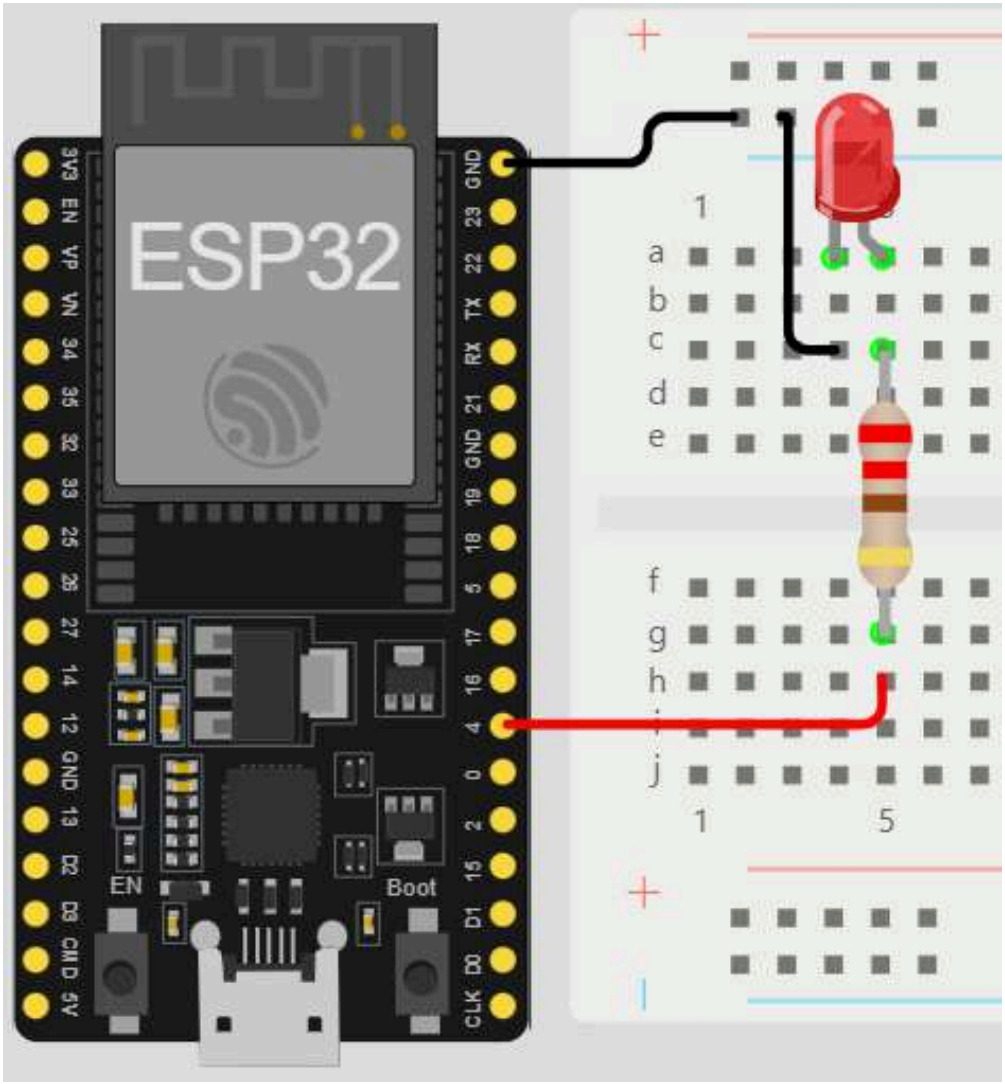
Membina projek asas bagi meningkatkan pemahaman dan kemahiran kendiri

CONTOH 2: ON LED MENGGUNAKAN PROGRAM

Komponen yang digunakan:
1 x ESP32
1 x LED
1 x Resistor 220Ω
1x Half Breadboard
Jumper wire secukupnya

Pemasangan Litar:
Sambungkan kaki anod (positif) LED ke pin digital 4 pada ESP32 melalui resistor 220Ω. Sambungkan kaki katod (negatif) LED ke GND (ground) Esp32

```
//Contoh 2: On LED

int LED=4;


void setup() {
  pinMode(LED, OUTPUT);
}

void loop() {
  digitalWrite(LED, HIGH);
  delay(1000);
  digitalWrite(LED, LOW);
  delay(1000);
}
```



Simulasi lanjut: https://wokwi.com/projects/423286971933259777

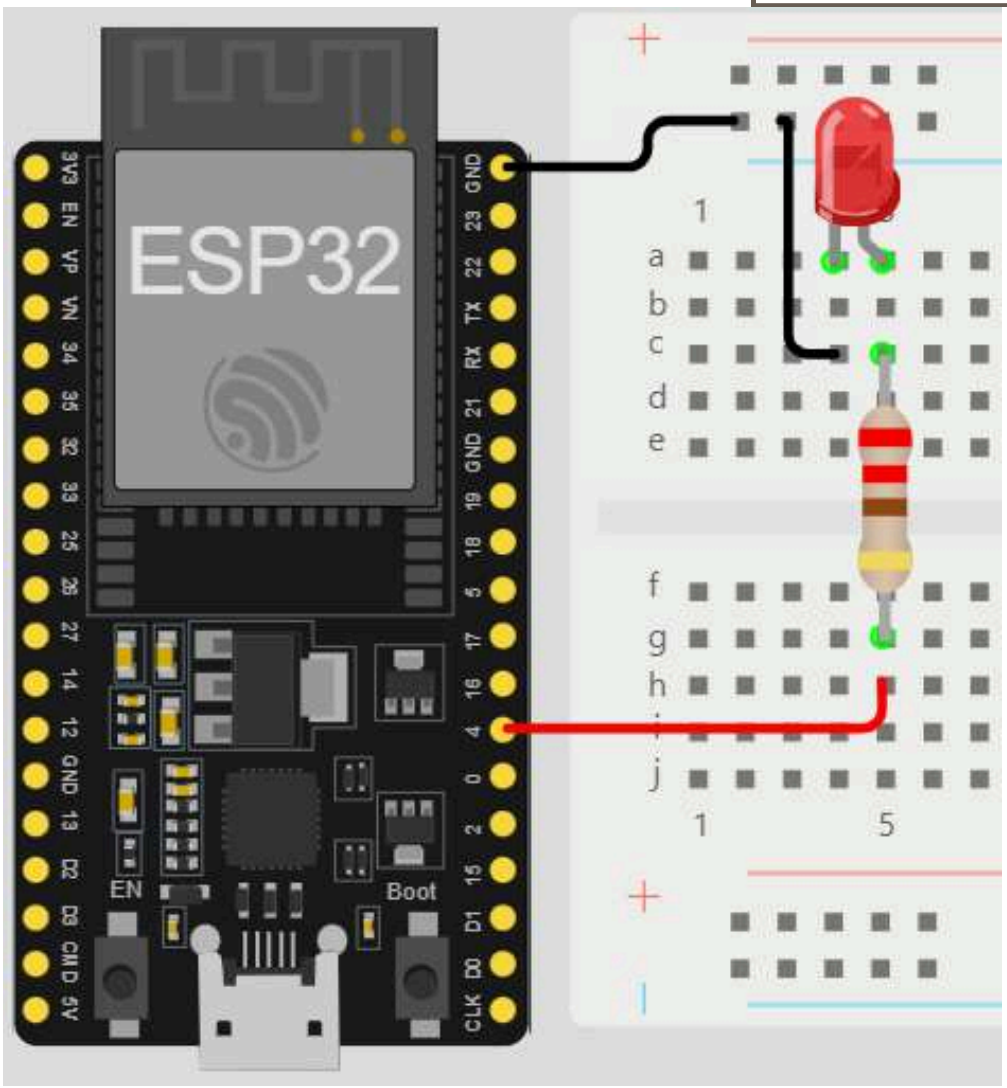
CONTOH 3: LED *BLINKING*

Komponen yang digunakan:
1 x ESP32
1 x LED
1 x Resistor 220Ω
1x Half Breadboard
Jumper wire secukupnya

Pemasangan Litar:
Sambungkan kaki anod (positif) LED ke pin digital 4 pada ESP32 melalui resistor 220Ω. Sambungkan kaki katod (negatif) LED ke GND (ground) Esp32

delay(1000) bersamaan 1 saat.
Nilai 1000 boleh diubah mengikut keperluan.

```
//Contoh 3: LED blinking

int LED=4;


void setup() {
  pinMode(LED, OUTPUT);
}

void loop() {
  digitalWrite(LED, HIGH);
  delay(1000);
  digitalWrite(LED, LOW);
  delay(1000);
}
```

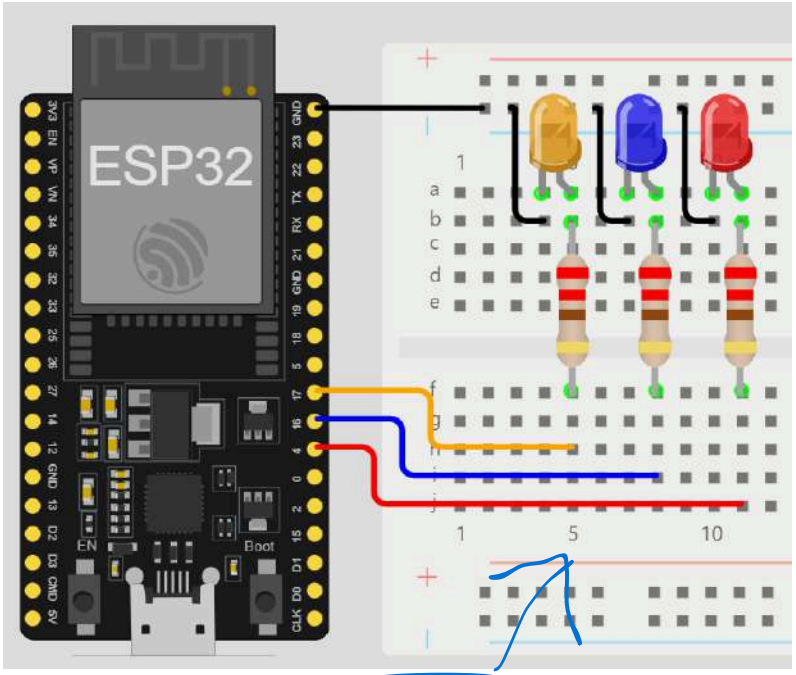


Simulasi lanjut: https://wokwi.com/projects/423287952236700673

CONTOH 4: LED *BLINKING* SERENTAK

Komponen yang digunakan:
1 x ESP32
3 x LED
3 x Resistor 220Ω
1x Half Breadboard
Jumper wire secukupnya

Pemasangan Litar:
Sambungkan kaki anod (positif) LED1 ke pin digital 4 pada ESP32 melalui resistor 220Ω.
Anod LED2 ke kaki 10 dan anod LED3 ke kaki 17 melalui perintang 220Ω.
Sambungkan kaki katod (negatif) LED1, LED2 dan LED3 ke GND (ground) Esp32



Langkah 1: tambahkan komponen dan pendawaian litar

```
//Contoh 4: 3 LED blinking serentak

int LED1=4;
int LED1=16;
int LED1=17;

void setup() {
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);

}

void loop() {
  digitalWrite(LED1, HIGH);
  digitalWrite(LED2, HIGH);
  digitalWrite(LED3, HIGH);
  delay(1000);
  digitalWrite(LED1, LOW);
  digitalWrite(LED2, LOW);
  digitalWrite(LED3, LOW);
  delay(1000);
}
```

Langkah 2: tambahkan senarai komponen dan nombor pin kaki ESP32 yang disambungkan pada perintang setiap LED.
Pastikan menamakan setiap LED dengan nama yang berlainan. COntohnya LED1, LED2, LED3.

Langkah 3: tambahkan senarai komponen dan menamakan output mengikut pendawaian litar. Nyatakan status OUTPUT bagi LED

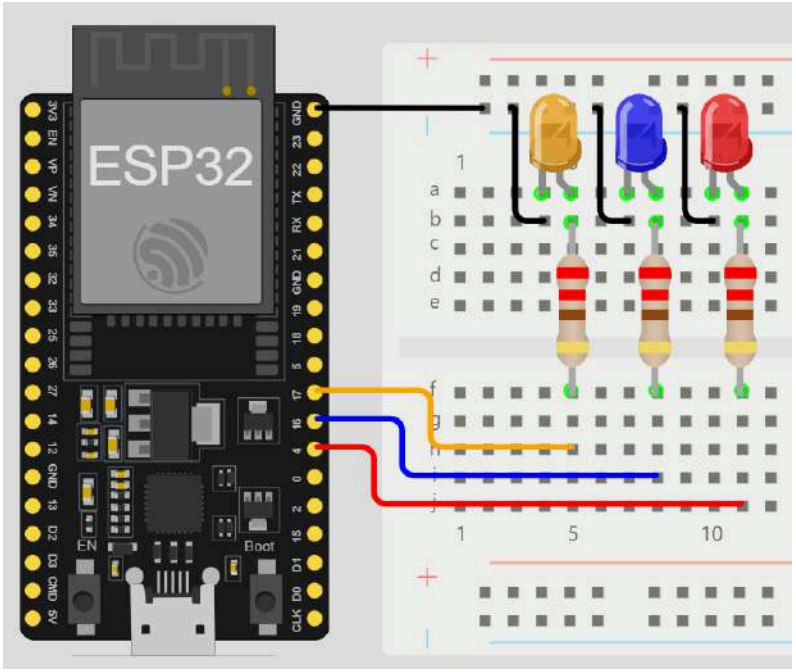
Langkah 4: tambahkan senarai komponen dan menamakan output mengikut pendawaian litar. Nyatakan status HIGH bagi LED menyala dan LOW bagi LED padam.

Simulasi lanjut:https://wokwi.com/projects/423288872552148993

CONTOH 5: LED *BLINKING* SECARA BERTURUTAN

Komponen yang digunakan:
1 x ESP32
3 x LED
3 x Resistor 220Ω
1x Half Breadboard
Jumper wire secukupnya

Pemasangan Litar:
Sambungkan kaki anod (positif) LED1 ke pin digital 4 pada ESP32 melalui resistor 220Ω.
Anod LED2 ke kaki 10 dan anod LED3 ke kaki 17 melalui perintang 220Ω.
Sambungkan kaki katod (negatif) LED1, LED2 dan LED3 ke GND (ground) Esp32




```
//Contoh 5: 3 LED blinking berturutan

int LED1=4;
int LED1=16;
int LED1=17;

void setup() {
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);

}

void loop() {
  digitalWrite(LED1, HIGH);
  delay(1000);
  digitalWrite(LED1, LOW);
  digitalWrite(LED2, HIGH);
  delay(1000);
  digitalWrite(LED2, LOW);
  digitalWrite(LED3, HIGH);
  delay(1000);
  digitalWrite(LED3, LOW);
}
```

Menggunakan kod terdahulu, tukarkan turutan LED dengan delay(1000) mengikut kreativiti sendiri. Lihat perubahan kerlipan LED.

Simulasi lanjut:https://wokwi.com/projects/423298790110119937

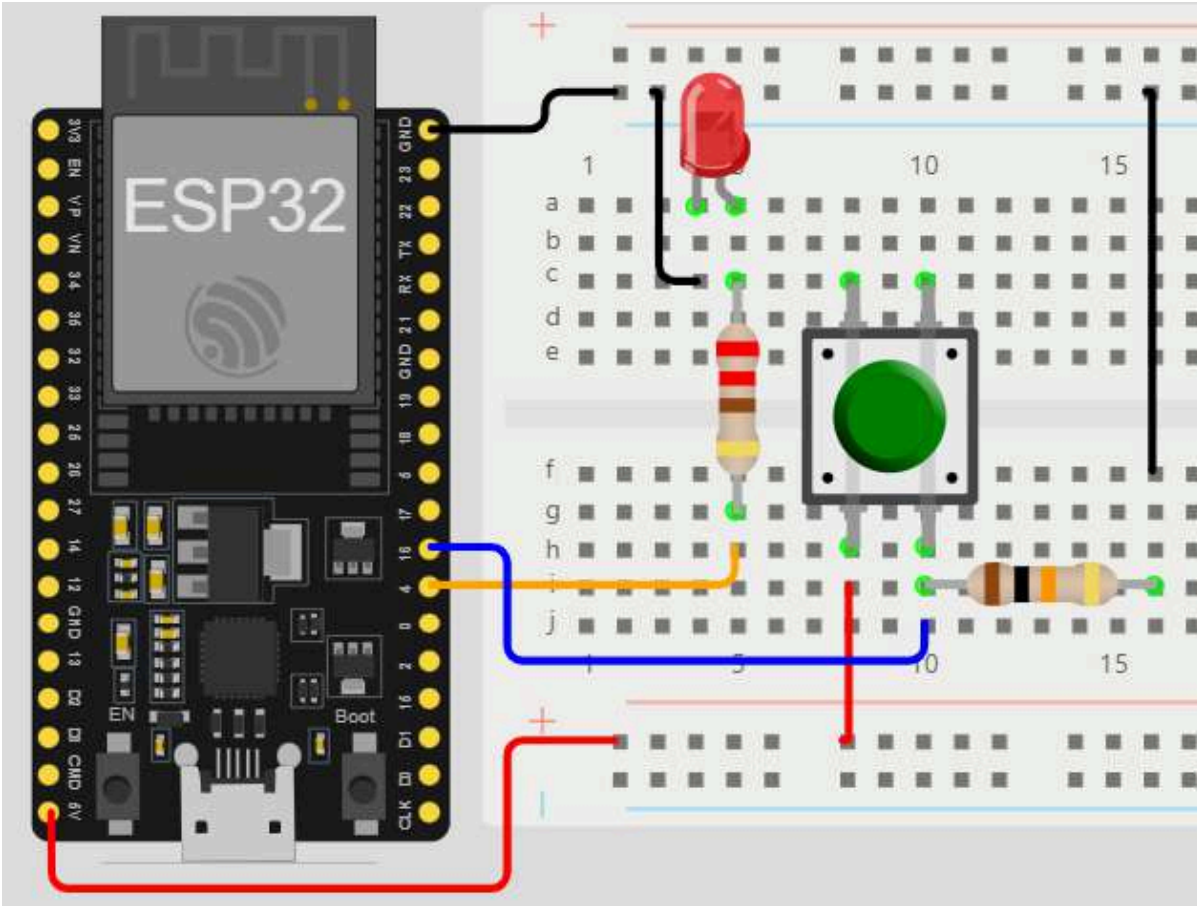
CONTOH 6: SUIS TEKAN DENGAN SATU LED

Komponen yang digunakan:
1 x Suis tekan
1 x ESP32
1 x LED
1 x Resistor 220Ω
1 x Resistor 10kΩ
1x Half Breadboard
Jumper wire secukupnya

Pemasangan Litar:
Sambungkan kaki anod (positif) LED ke pin digital 4 pada ESP32 melalui resistor 220Ω. Suis tekan disambungkan seperti rajah iaitu satu kaki ke 5V. Satu kaki lagi ke pin 16 ESP32 dan ke GND melalui resistor 10kΩ.

Fungsi Litar:
Suis tidak ditekan(**LOW**): LED akan padam
Suis ditekan(**HIGH**): LED akan menyala

```
//Contoh 6: Suis tekan dengan satu LED

int LED=4;
int suis=16;

int suisSTATE=0;

void setup() {
  pinMode(LED, OUTPUT);
  pinMode(suis, INPUT);
}

void loop() {
  suisSTATE = digitalRead(suis);

  if(suisSTATE ==HIGH)
    digitalWrite(LED, HIGH);
  else
    digitalWrite(LED, LOW);
}
```



Simulasi lanjut:https://wokwi.com/projects/423302938328213505

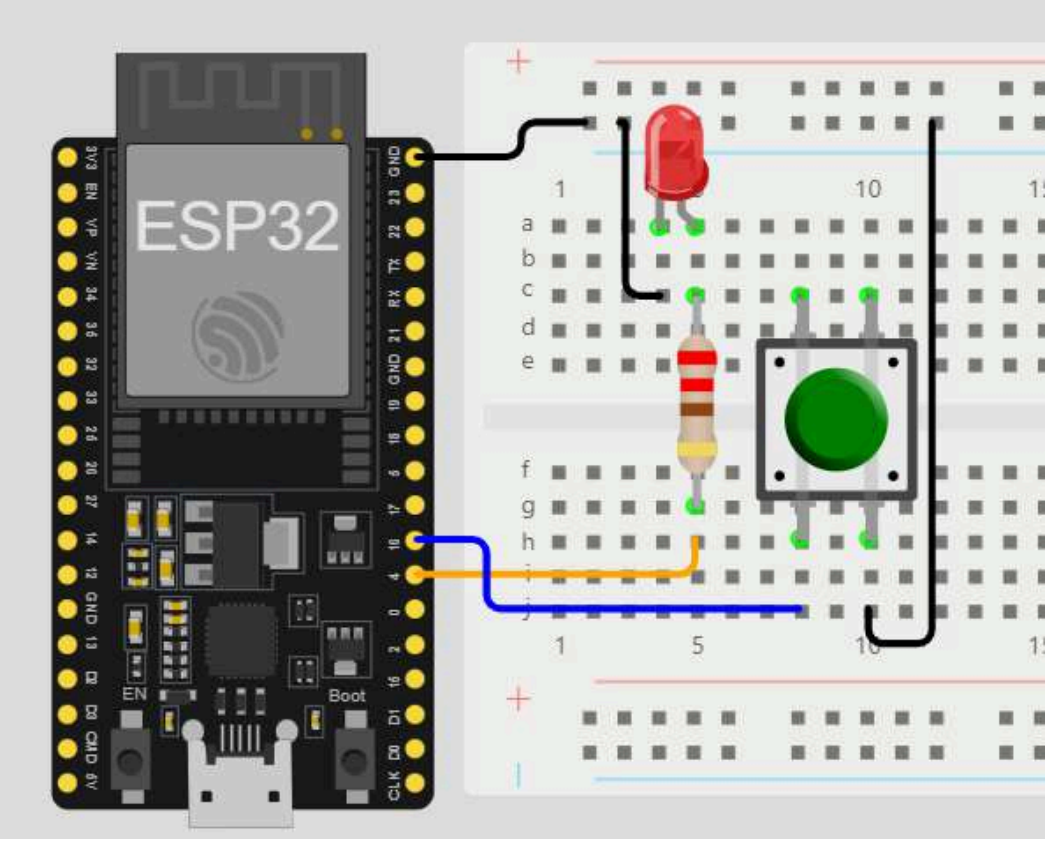
CONTOH 7: SUIS TEKAN DENGAN SATU LED (INPUT PULLUP)

Komponen yang digunakan:
1 x Suis tekan
1 x ESP32
1 x LED
1 x Resistor 220Ω
1x Half Breadboard
Jumper wire secukupnya

Pemasangan Litar:
Sambungkan kaki anod (positif) LED ke pin digital 4 pada ESP32 melalui resistor 220Ω. Suis tekan disambungkan seperti rajah iaitu satu kaki pin 16 ESP32 dan satu lagi ke GND.

Fungsi Litar:
Dalam kaedah ini, **ESP32** akan menggunakan **resistor pull-up dalaman,** menjadikan **pin input b**erada dalam keadaan **HIGH (1)** secara lalai**.**
Suis tidak ditekan(**HIGH**): LED akan padam
Suis ditekan(**LOW**): LED akan menyala

```
//Contoh 7: Suis tekan dengan satu LED
// menggunakan input pullup

int LED=4;
int suis=16;

int suisSTATE=0;

void setup() {
  pinMode(LED, OUTPUT);
  pinMode(suis, INPUT_PULLUP);
}

void loop() {
  suisSTATE = digitalRead(suis);

  if(suisSTATE ==LOW)
    digitalWrite(LED, HIGH);
  else
    digitalWrite(LED, LOW);
}
```



Simulasi lanjut:https://wokwi.com/projects/423304446433610753

LATIHAN : RUJUK CONTOH 7

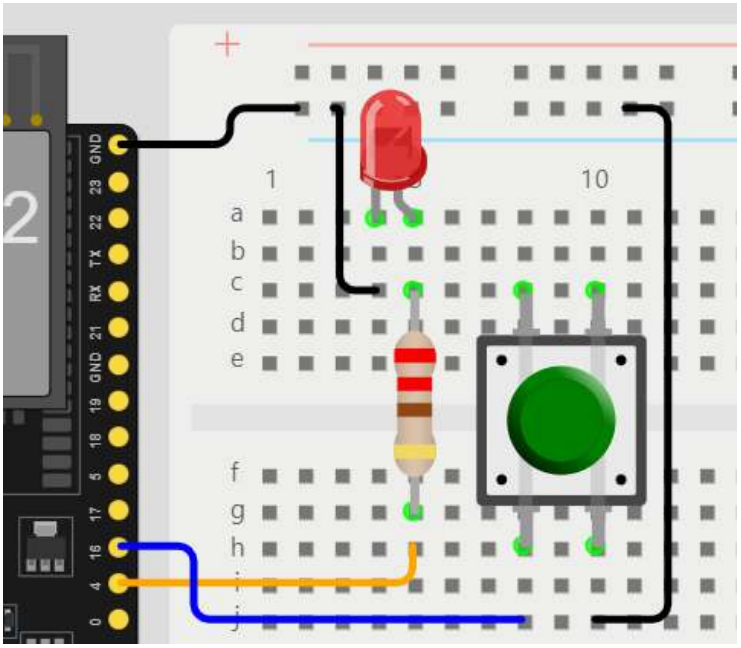
Soalan 1:
Tukarkan operasi litar kepada “apabila suis ditekan, LED akan berkelip selama 0.5 saat”.

Soalan 2:
Ubah keadaan apabila suis tidak ditekan, LED akan menyala dan apabila suis ditekan, LED akan padam.

Soalan 1:
Tukarkan operasi litar kepada “apabila suis ditekan, LED akan berkelip selama 0.5 saat”.

Jawapan:

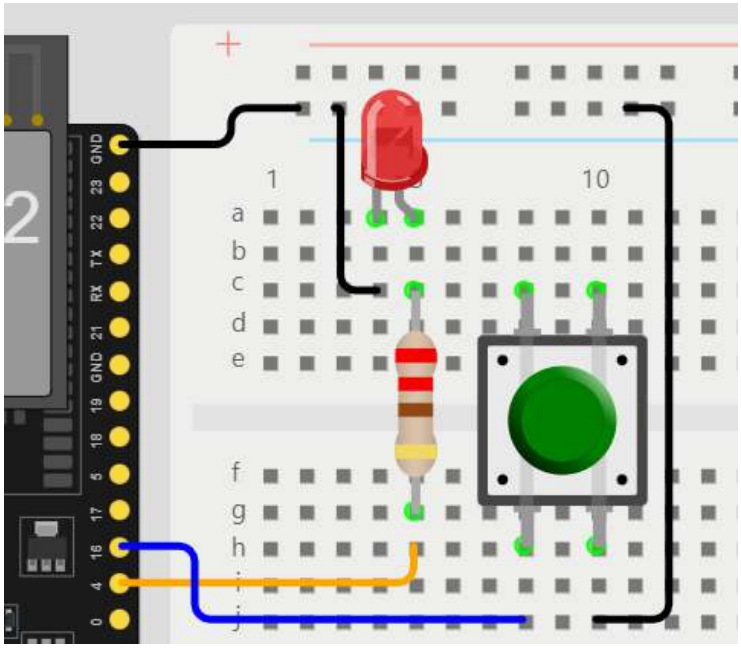
```
int LED=4;
int suis=16;

int suisSTATE=0;

void setup() {
  pinMode(LED, OUTPUT);
  pinMode(suis, INPUT_PULLUP);
}

void loop() {
  suisSTATE = digitalRead(suis);

  if(suisSTATE ==LOW)
    digitalWrite(LED, HIGH);
    delay(500);
    digitalWrite(LED, LOW);
    delay(500);
  else
    digitalWrite(LED, LOW);
}
```



Soalan 2:
Ubah keadaan apabila suis tidak ditekan, LED akan berkelip dan apabila suis ditekan, LED akan padam.

Jawapan:

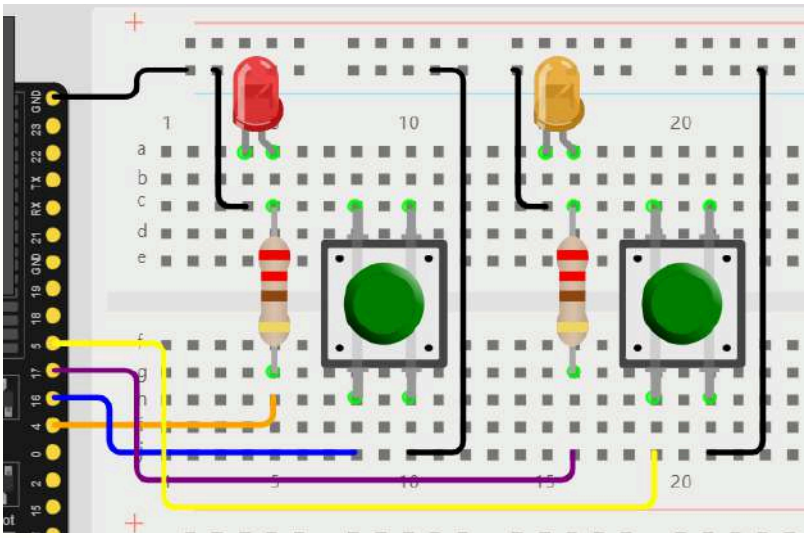
```
int LED=4;
int suis=16;

int suisSTATE=0;

void setup() {
  pinMode(LED, OUTPUT);
  pinMode(suis, INPUT_PULLUP);
}

void loop() {
  suisSTATE = digitalRead(suis);

  if(suisSTATE ==HIGH)
    digitalWrite(LED, HIGH);
    delay(500);
    digitalWrite(LED, LOW);
    delay(500);
  else
    digitalWrite(LED, LOW);
}
```



Apakah perbandingan yang boleh dilihat daripada contoh 7 dan kedua-dua latihan di atas?

CONTOH 8: 2 UNIT SUIS TEKAN DENGAN DUA LED (INPUT PULL-UP)

```
//Contoh 8: 2 Suis mengawal 2 LED
//(input pull-up)

int LED1=4;
int suis1=16;
int LED2=17;
int suis2=5;

int suisSTATE1=0;
int suisSTATE2=0;

void setup() {
  pinMode(LED1, OUTPUT);
  pinMode(suis1, INPUT_PULLUP);
  pinMode(LED2, OUTPUT);
  pinMode(suis2, INPUT_PULLUP);

}

void loop() {
  suisSTATE1 = digitalRead(suis1);
  suisSTATE2 = digitalRead(suis2);


  if(suisSTATE1 ==LOW){
    digitalWrite(LED1, HIGH);
    delay(500);
    digitalWrite(LED1, LOW);
    delay(500);}
  else
    {digitalWrite(LED1, LOW);}

if(suisSTATE2 ==LOW){
    digitalWrite(LED2, HIGH);
    delay(500);
    digitalWrite(LED2, LOW);
    delay(500);}
  else
    {digitalWrite(LED2, LOW);}

}
```



Berikut adalah operasi bagi litar di atas yang menggunakan 2 suis untuk mengawal 2 LED pada ESP32 menggunakan kaedah INPUT_PULLUP:

Keadaan Normal (Tiada Suis Ditekan)

- Kedua-dua suis berada dalam keadaan HIGH kerana menggunakan pull-up dalaman ESP32.
- Kedua-dua LED akan padam kerana arahan dalam kod menetapkan bahawa LED hanya menyala jika suis ditekan.

Apabila Suis 1 Ditekan

- Pin suis 1 (pin 16) akan berubah dari HIGH → LOW kerana ia disambungkan ke GND.
- Arduino membaca perubahan ini dan menghidupkan LED 1 (pin 4).

Apabila Suis 2 Ditekan

- Pin suis 2 (pin 17) akan berubah dari HIGH → LOW.
- Arduino membaca perubahan ini dan menghidupkan LED 2 (pin 5).

Apabila Kedua-dua Suis Dilepaskan

- Suis kembali ke keadaan asal HIGH kerana pull-up dalaman.
- Kedua-dua LED akan padam.

CONTOH 9: SUIS TEKAN DENGAN LED BERUBAH KECERAHAN

```
//Contoh 9: Suis tekan dengan LED berubah kecerahan
//(input pull-up)

int LED=4;
int brightness=0;
int fadeAmount=10;

void setup()
  {
  pinMode(LED, OUTPUT);
  }

void loop() {
  analogWrite(LED, brightness);

  brightness = brightness+fadeAmount;

  if(brightness ==0 || brightness==255)
    {fadeAmount = -fadeAmount;}
  delay(30);
}
```

brightness mengawal tahap kecerahan LED (0-255).
fadeAmount menetapkan nilai perubahan setiap kitaran (10 tahap)

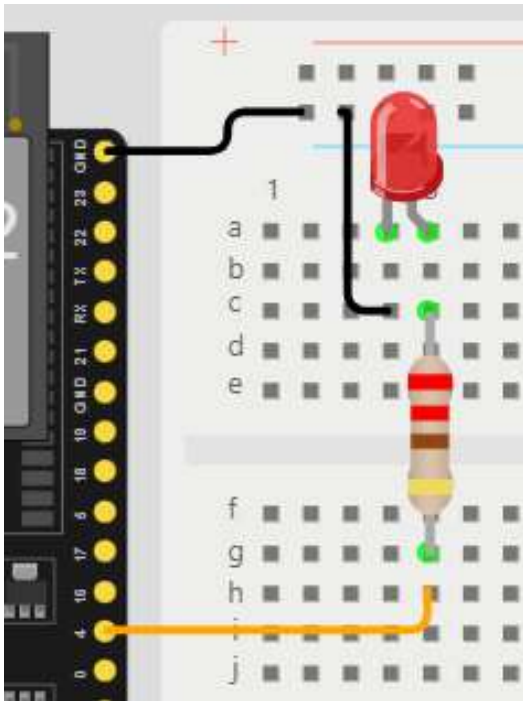
Pin LED ditetapkan sebagai output supaya ESP32 boleh mengawalnya.

Fungsi analogWrite(LED, brightness); digunakan untuk mengawal kecerahan LED antara nilai 0 (gelap) hingga 255 (terang).

Nilai brightness akan bertambah sebanyak fadeAmount

Apabila brightness mencapai 255 (maksimum) atau 0 (minimum), nilai fadeAmount akan dibalikkan (-fadeAmount).
Ini menyebabkan LED akan mencerah dan malap secara berulang.

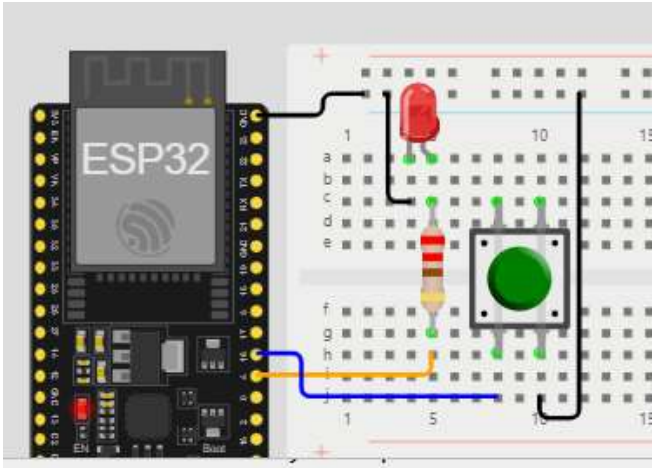
Menangguhkan kod selama 30ms supaya perubahan kecerahan LED lebih perlahan dan nampak halus.



Bagaimana LED Berfungsi?
✅ LED mula padam kerana brightness = 0.
✅ Kecerahan LED bertambah sebanyak +10 setiap kali loop().
✅ Apabila brightness mencapai 255, perubahan dibalikkan (fadeAmount = -10).
✅ Kecerahan LED mula berkurang sehingga brightness = 0, dan kitaran berulang.

Simulasi lanjut:https://wokwi.com/projects/423461625357370369

CONTOH 10: SERIAL MONITOR SEBAGAI PAPARAN OUTPUT




```
Suis TIDAK ditekan, LED padam
Suis TIDAK ditekan, LED padam
Suis TIDAK ditekan, LED padam
Suis TIDAK ditekan, LED padam
Suis TIDAK ditekan, LED padam
```

Serial Monitor ialah alat dalam Arduino IDE yang membolehkan kita berkomunikasi dengan mikrokontroler seperti Arduino dan ESP32 menggunakan komunikasi serial (UART). Ia digunakan untuk:

✅ Memaparkan data dari sensor atau pembolehubah dalam kod.
✅ Mengesan ralat (debugging) dengan mencetak mesej atau nilai pembolehubah.
✅ Menghantar arahan dari komputer ke mikrokontroler.

Cara Menggunakannya:
1️⃣ Tambah Serial.begin(115200); dalam setup() untuk memulakan komunikasi.
2️⃣ Gunakan Serial.print() atau Serial.println() untuk memaparkan data.
3️⃣ Buka Serial Monitor di Arduino IDE dengan menekan Ctrl + Shift + M atau klik ikon Serial Monitor.
4️⃣ Pastikan baud rate sama dengan yang ditetapkan dalam kod (115200 atau 9600).

```
1   //Contoh 10: Suis tekan dengan satu LED
2   // dan serial monitor
3
4   int LED=4;
5   int suis=16;
6
7   int suisSTATE=0;
8
9   void setup() {
10    pinMode(LED, OUTPUT);
11    pinMode(suis, INPUT_PULLUP);
12    Serial.begin(115200);
13  }
14
15  void loop() {
16    suisSTATE = digitalRead(suis);
17
18    if(suisSTATE ==LOW)
19      {digitalWrite(LED, HIGH);
20      Serial.println("Suis DITEKAN, LED menyala");}
21    else
22      {digitalWrite(LED, LOW);
23      Serial.println("Suis TIDAK ditekan, LED padam");}
24
25  delay(100);
26  }
```

Memulakan komunikasi Serial Monitor dengan baud rate 115200 (pastikan nilai ini dipilih dalam Arduino Serial Monitor).

Akan dipaparkan dalam Serial Monitor jika suis ditekan dan LED menyala.

Akan dipaparkan dalam Serial Monitor jika suis tidak ditekan dan LED padam.

Mengelakkan teks berulang terlalu cepat dalam Serial Monitor.

Simulasi lanjut:https://wokwi.com/projects/423464419071610881

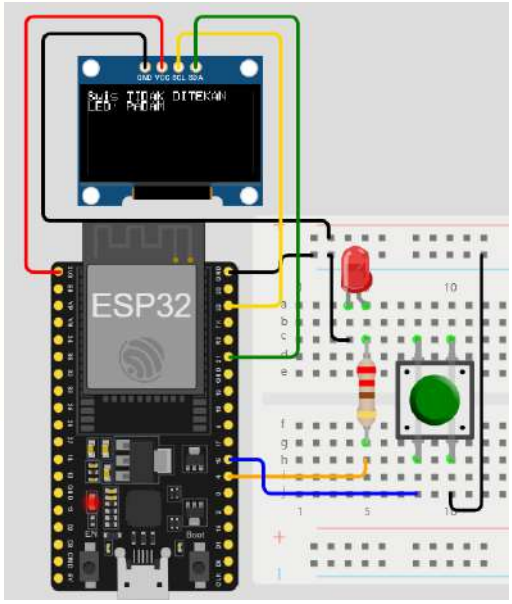
CONTOH 11: SERIAL MONITOR SEBAGAI PAPARAN OUTPUT

```ino
//Contoh 11: Suis tekan dengan satu LED
// serial monitor dan OLED

#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128  // Lebar OLED (128 piksel)
#define SCREEN_HEIGHT 64  // Tinggi OLED (64 piksel)
#define OLED_RESET -1     // Reset tidak digunakan
#define SCREEN_ADDRESS 0x3C  // Alamat I2C bagi OLED

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

int LED = 4;      // LED disambungkan ke pin 4
int suis = 16;    // Suis disambungkan ke pin 16
int suisSTATE = 0; // Pembolehubah untuk menyimpan status suis

void setup() {
  pinMode(LED, OUTPUT);
  pinMode(suis, INPUT_PULLUP);
  Serial.begin(115200);

  // Memulakan OLED
  if (!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
    Serial.println(F("Gagal memulakan OLED"));
    for (;;);
  }

  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0, 0);
  display.println("OLED SIAP!");
  display.display();
  delay(2000);
}

void loop() {
  suisSTATE = digitalRead(suis);  // Membaca status suis

  display.clearDisplay();
  display.setCursor(0, 0);

  if (suisSTATE == LOW) {  // Jika suis ditekan
    digitalWrite(LED, HIGH);
    Serial.println("Suis DITEKAN, LED menyala");

    display.println("Suis DITEKAN");
    display.println("LED: MENYALA");
  } else {  // Jika suis tidak ditekan
    digitalWrite(LED, LOW);
    Serial.println("Suis TIDAK ditekan, LED padam");

    display.println("Suis TIDAK DITEKAN");
    display.println("LED: PADAM");
  }

  display.display();
  delay(100);
}
```

Dua library utama yang perlu dipasang dalam Arduino IDE atau Wokwi jika menggunakan OLED.

Simbol #include < xxxxx> menunjukkan keperluan untuk plug-in library tambahan untk membolehkan sesuatu komponen elektronik digunakan.

Contoh kod menunjukkan beberapa bahagian kod tambahan yang wajib ada sekiranya ingin menggunakan OLED sebagai output pemapar.



Operasi Litar:

Keadaan Suis	Bacaan digitalRead (suis)	Keadaan LED	Paparan OLED	Serial Monitor
Tidak Ditekan	HIGH (1)	Padam	"Suis TIDAK DITEKAN" "LED: PADAM"	Suis TIDAK ditekan, LED padam
Ditekan	LOW (0)	Menyala	"Suis DITEKAN" "LED: MENYALA"	Suis DITEKAN, LED menyala

Simulasi lanjut:https://wokwi.com/projects/423466903715919873_

BAB

LIMA

PROJEK LANJUTAN - SENSOR

4 PAGES
CHECKLIST

LET'S DO IT

kandungan bab

Meningkatkan kemahiran untuk terus mencipta projek terkini

CONTOH 12: SENSOR LDR DENGAN SERIAL MONITOR

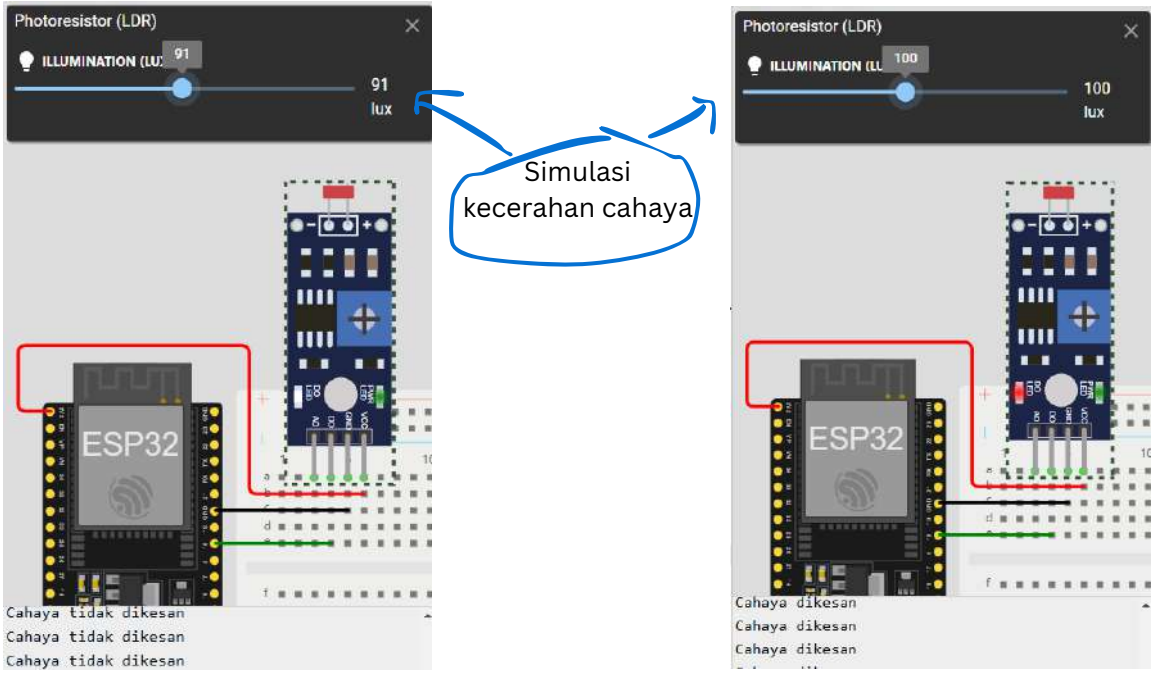
```ino
//contoh 12: Sensor LDR dengan serial monitor

// Deklarasi pin yang digunakan untuk LDR
const int ldrPin = 18;  // GPIO 18 pada ESP32

void setup() {
  // Inisialisasi komunikasi Serial
  Serial.begin(115200);

  // Inisialisasi pin LDR sebagai input
  pinMode(ldrPin, INPUT);
}

void loop() {
  // Membaca nilai dari pin digital sensor LDR
  int ldrStatus = digitalRead(ldrPin);

  // Jika cahaya terdeteksi
  if (ldrStatus == LOW) {
    Serial.println("Cahaya dikesan");
  }
  // Jika tidak ada cahaya
  else {
    Serial.println("Cahaya tidak dikesan");
  }

  // Beri jeda sejenak
  delay(500);  // Jeda selama 500 ms
}
```

Komponen yang digunakan:
1 x ESP32
1 x LDR Sensor
1x Half Breadboard
Jumper wire secukupnya

Pemasangan Litar:
Sambungkan kaki VCC LDR ke pin 3v3. Kaki GND LDR disambung ke GND esp32. Kaki D0 disambungkan kepada pin 18 ESP32.

Fungsi Litar:
D0 digunakan sebagai output digital, yang bermaksud ia akan memberikan isyarat HIGH (1) atau LOW (0) bergantung kepada tahap cahaya.
Sensor LDR ini mempunyai potensiometer yang boleh dilaraskan untuk menetapkan nilai ambang.

ESP32 membaca output digital dari sensor LDR melalui pin GPIO 18.
Jika nilai **LDR > 100**, sensor mengeluarkan **LOW (0)**, **bermaksud cahaya dikesan**.
Jika nilai **LDR < 100**, sensor mengeluarkan **HIGH (1)**, **bermaksud cahaya tidak dikesan**.
Data ini akan dihantar ke Serial Monitor.

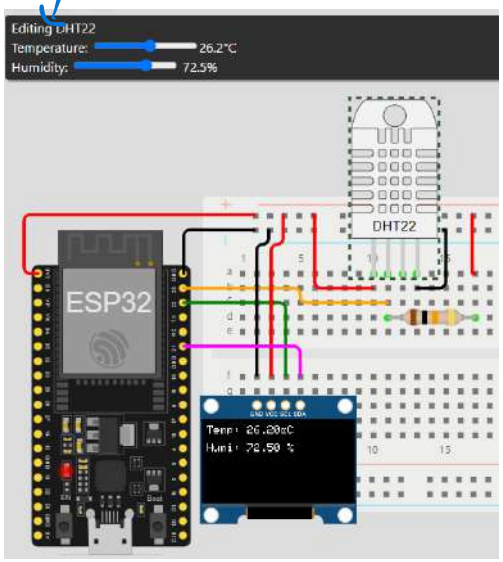


Simulasi lanjut:https://wokwi.com/projects/423734722570481665

CONTOH 13: SENSOR DHT 22 DENGAN OLED

```ino
//contoh 13: sensor DHT22 dengan OLED

#include <Adafruit_GFX.h>      // Perpustakaan grafik OLED
#include <Adafruit_SSD1306.h>  // Perpustakaan untuk OLED SSD1306
#include <Wire.h>              // Perpustakaan I2C
#include <DHT.h>               // Perpustakaan DHT sensor

#define SCREEN_WIDTH 128       // Lebar OLED 128 piksel
#define SCREEN_HEIGHT 64       // Tinggi OLED 64 piksel

// Definisikan OLED dengan I2C address 0x3C
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

#define DHTPIN 23              // Pin untuk DHT22 sensor
#define DHTTYPE DHT22          // Definisikan jenis DHT sensor (DHT22)
DHT dht(DHTPIN, DHTTYPE);      // Inisialisasi DHT sensor

void setup() {
  // Inisialisasi OLED display
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("Gagal untuk memulakan OLED"));
    for(;;); // Hentikan program jika OLED gagal diinisialisasi
  }

  display.clearDisplay();      // Bersihkan paparan OLED
  display.setTextSize(1);      // Saiz teks
  display.setTextColor(SSD1306_WHITE);  // Warna teks putih
  display.display();           // Paparkan perubahan pada OLED

  // Inisialisasi DHT sensor dan Serial monitor
  dht.begin();
  Serial.begin(115200);
  Serial.println("Hello, ESP32!");
}

void loop() {
  float humidity = dht.readHumidity();        // Bacaan kelembapan
  float temperature = dht.readTemperature();  // Bacaan suhu

  // Bersihkan paparan OLED sebelum mengemaskini data
  display.clearDisplay();

  // Paparkan suhu di OLED
  display.setCursor(0, 0);
  display.print("Temp: ");
  display.print(temperature);
  display.print((char)223);
  display.print("C");

  // Paparkan kelembapan di OLED
  display.setCursor(0, 16);
  display.print("Humi: ");
  display.print(humidity);
  display.print(" %");

  // Kemaskini paparan OLED
  display.display();

  // Selang 1 saat sebelum bacaan seterusnya
  delay(1000);
}
```

Simulasi suhu dan kelembapan persekitaran



Penyambungan Litar:

Sambungan DHT22 ke Pengawal Mikro ESP32:

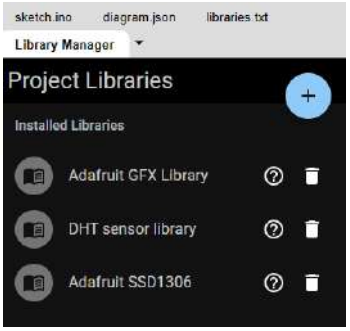
Pin VCC: Sambungkan ke pin 3.3V atau 5V (kedua-duanya boleh digunakan, tetapi disarankan 3.3V untuk keserasian dengan ESP32).

Pin GND: Sambungkan ke GND.

Pin Data: Sambungkan ke mana-mana GPIO (contoh: GPIO 23).

Resistor Pull-up: Letakkan resistor 10 kΩ antara pin Data dan VCC untuk komunikasi yang stabil.

Nota: DHT22 memerlukan masa tindak balas yang lebih lama, jadi perlu menunggu sekurang-kurangnya 2 saat antara setiap bacaan untuk mendapatkan data yang tepat.



Simulasi lanjut:https://wokwi.com/projects/423739645591378945

Tips mudah faham

PROGRAMMING DALAM ARDUINO IDE

01

FAHAMI STRUKTUR ASAS DALAM ARDUINO IDE

1. Declaration (Pernyataan Awal)
Digunakan untuk mengisytiharkan pin, pembolehubah, dan pustaka yang diperlukan.
2. Setup() – Kod yang Dijalankan Sekali
Digunakan untuk memulakan pin input/output dan tetapan awal.
3. Loop() – Kod yang Berulang-Ulang
Bahagian utama di mana program akan terus berjalan selagi Arduino beroperasi.

02

KENALI CIRI-CIRI BOARD ARDUINO YANG DIGUNAKAN

🔹 Arduino UNO: 5V Logic, 14 pin digital, 6 pin analog
🔹 ESP32: 3.3V Logic, lebih banyak pin digital & analog, ada WiFi & Bluetooth

03

FAHAMI KONSEP INPUT & OUTPUT

Input: Data masuk → Sensor, butang tekan, potensiometer
Output: Data keluar → LED, motor, buzzer

04

GUNAKAN LIBRARY MANAGER UNTUK MEMUDAHKAN PROJEK

Library Manager dalam Arduino IDE membolehkan pemasangan pustaka secara automatik.

05

MENINGKATKAN TAHAP KESUKARAN PENGEKODAN SECARA BERPERINGKAT

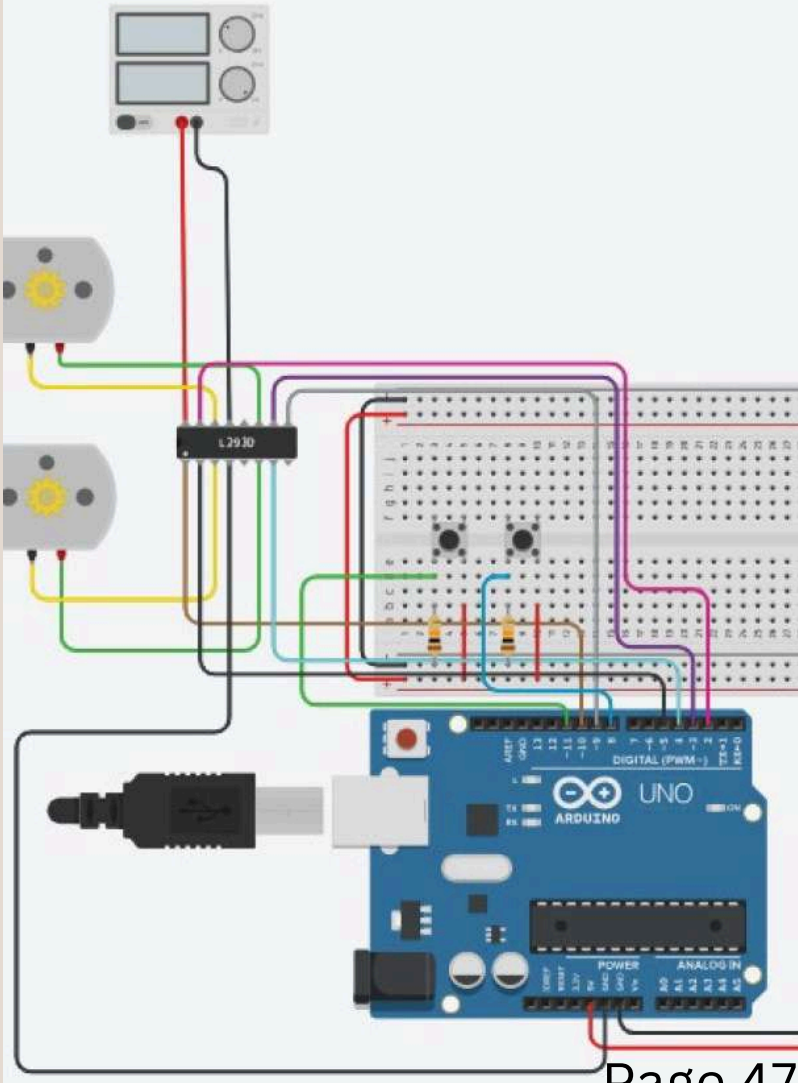
Mulakan dengan program mudah, kemudian tambahkan cabaran.

MENIGKATKAN PEMAHAMAN DAN PERSEDIAAN

Segala contoh program dan sambungan litar adalah sebagai latihan bagi meningkatkan kefahaman sebelum menggunakan peralatan sebenar. Langkah ini dapat menjimatkan kos dan mempercepat pemahaman. Setelah benar-benar bersedia dan faham asas penggunaan Arduino dan ESP32, bolehlah menggunakan peralatan dan persekitaran Arduino IDE yang sebenar.

LATIHAN BERTERUSAN

Latihan berterusan amat diperlukan begi meningkatkan kemahiran ini serta perlu kreatif dalam menyelesaikan masalah atau isu yang berlaku.



Senarai RUJUKAN

01 BLUM, J. (2020). EXPLORING ARDUINO: TOOLS AND TECHNIQUES FOR ENGINEERING WIZARDRY, (2ND ED.). JOHN WILEY & SONS, INC.

02 GEDDES, M. (2016). ARDUINO PROJECT HANDBOOK. WILLIAM POLLOCK PRODUCTION.

03 HALVORSEN, H.-P. (2018). PROGRAMMING WITH ARDUINO. HALVORSEN PUBLISHER.

04 O'REILLY, M. B. (2007). PROGRAMMING EMBEDDED SYSTEMS (2ND ED ED.). O'REILLY.

Panduan Asas ARDUINO DAN ESP32 Edisi pertama

e ISBN 978-629-7737-16-4

POLITEKNIK MERLIMAU

(online)