

Zero to Hero



Basic Arduino and ESP32 Tutorial pls.



Alright, follow me.

Send a message



ESL

To-do list

- Embedded System
- Boards
- Basic C language
- Learning Project
- Sensor & Module

What is Embedded System?

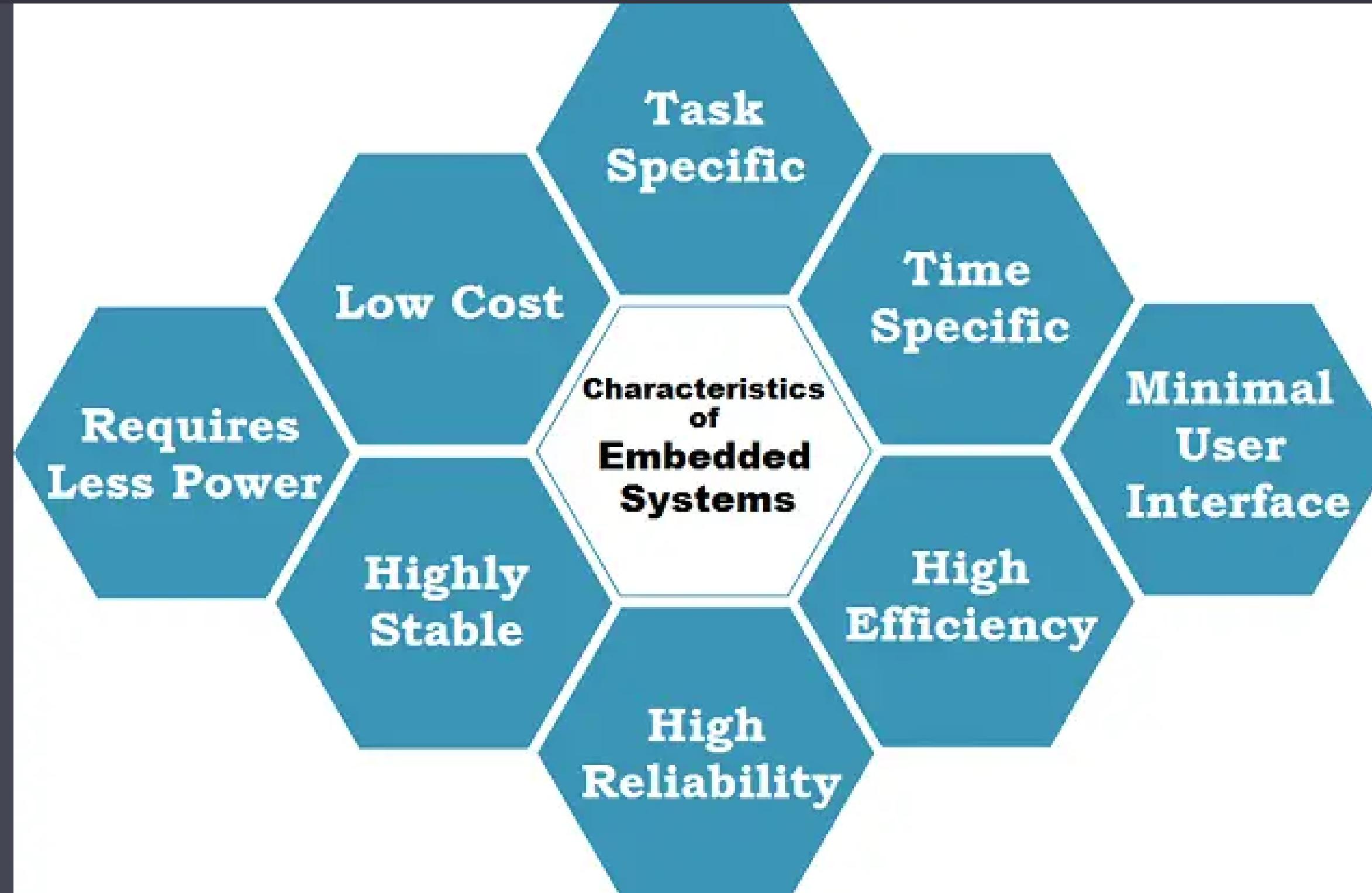


What is Embedded System?

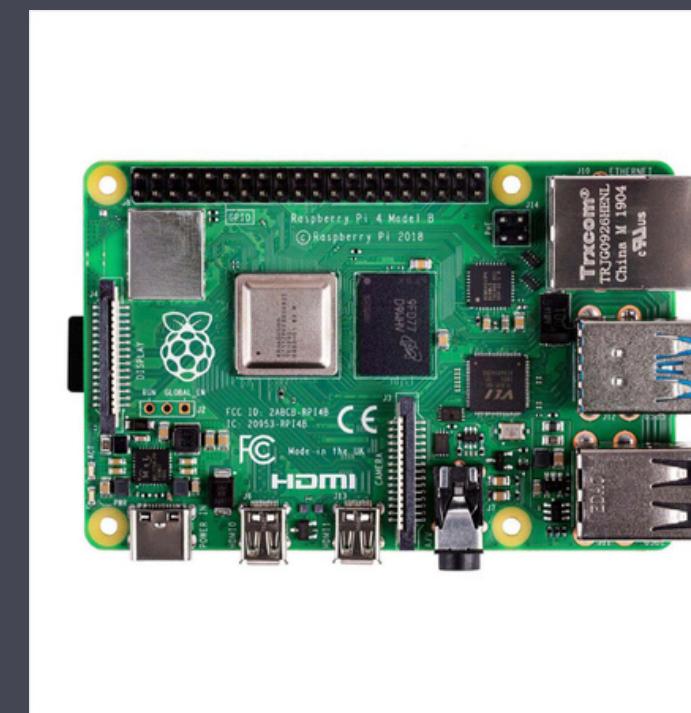
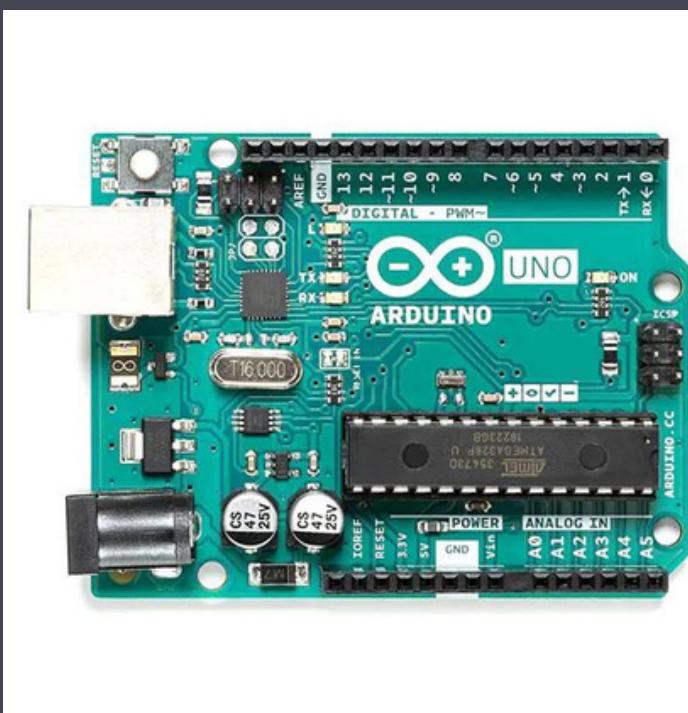


An embedded system is a combination of computer hardware and software designed for a **specific function**. Embedded systems may also function within a larger system. The systems can be programmable or have a fixed functionality.

Characteristics of Embedded System



What are Embedded System used for?



BOARDS

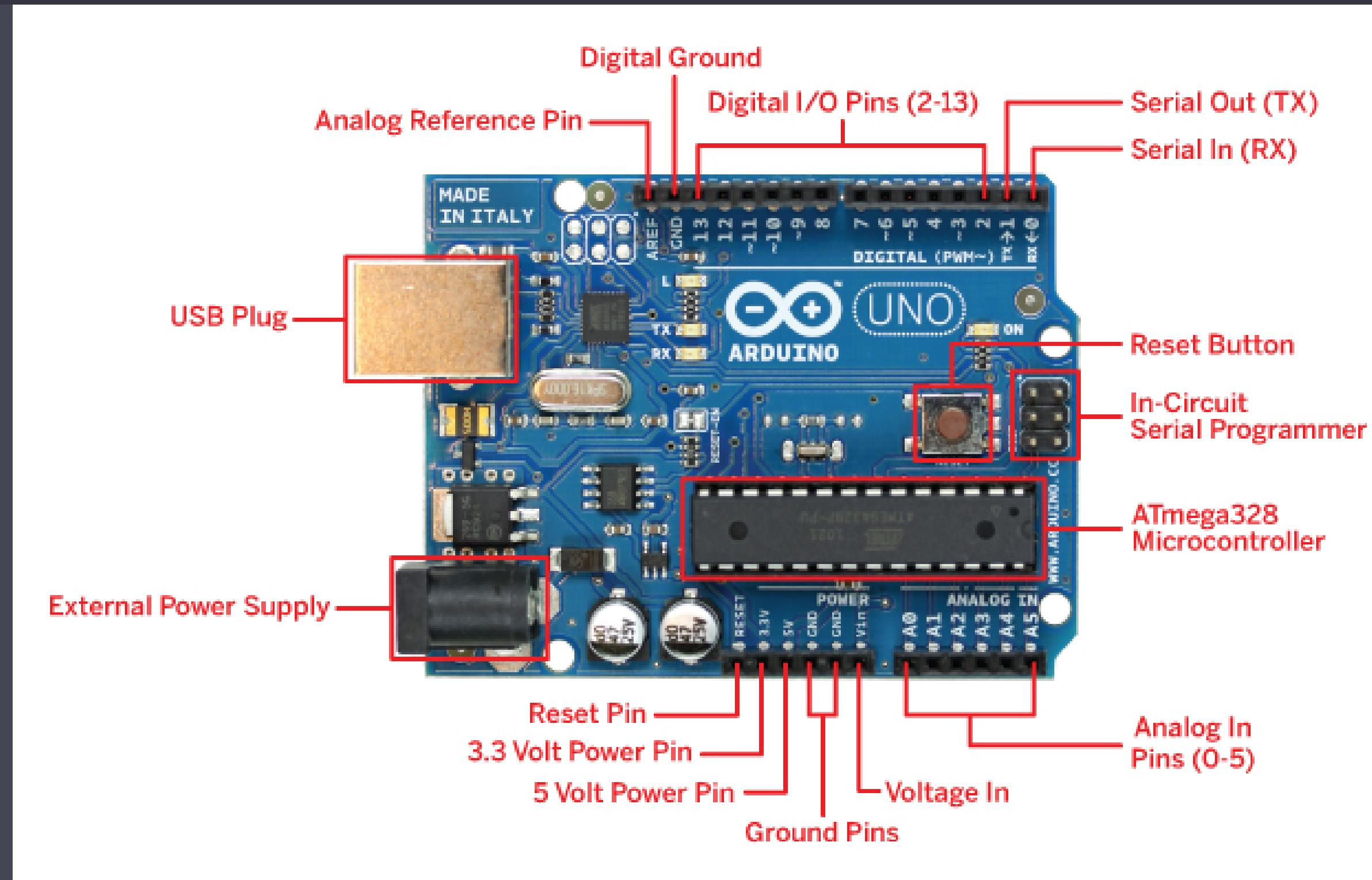
What is Arduino?

- Arduino (ອາດູຍໂນ) ເປັນບ່ອນດ microcontroller
- Software Open-source = Free!!
- easy-to-use hardware & software
- library ເຊັະ
- Board ຮາຄາງຸກ



[Arduino Uno Datasheet](#)

Arduino Uno



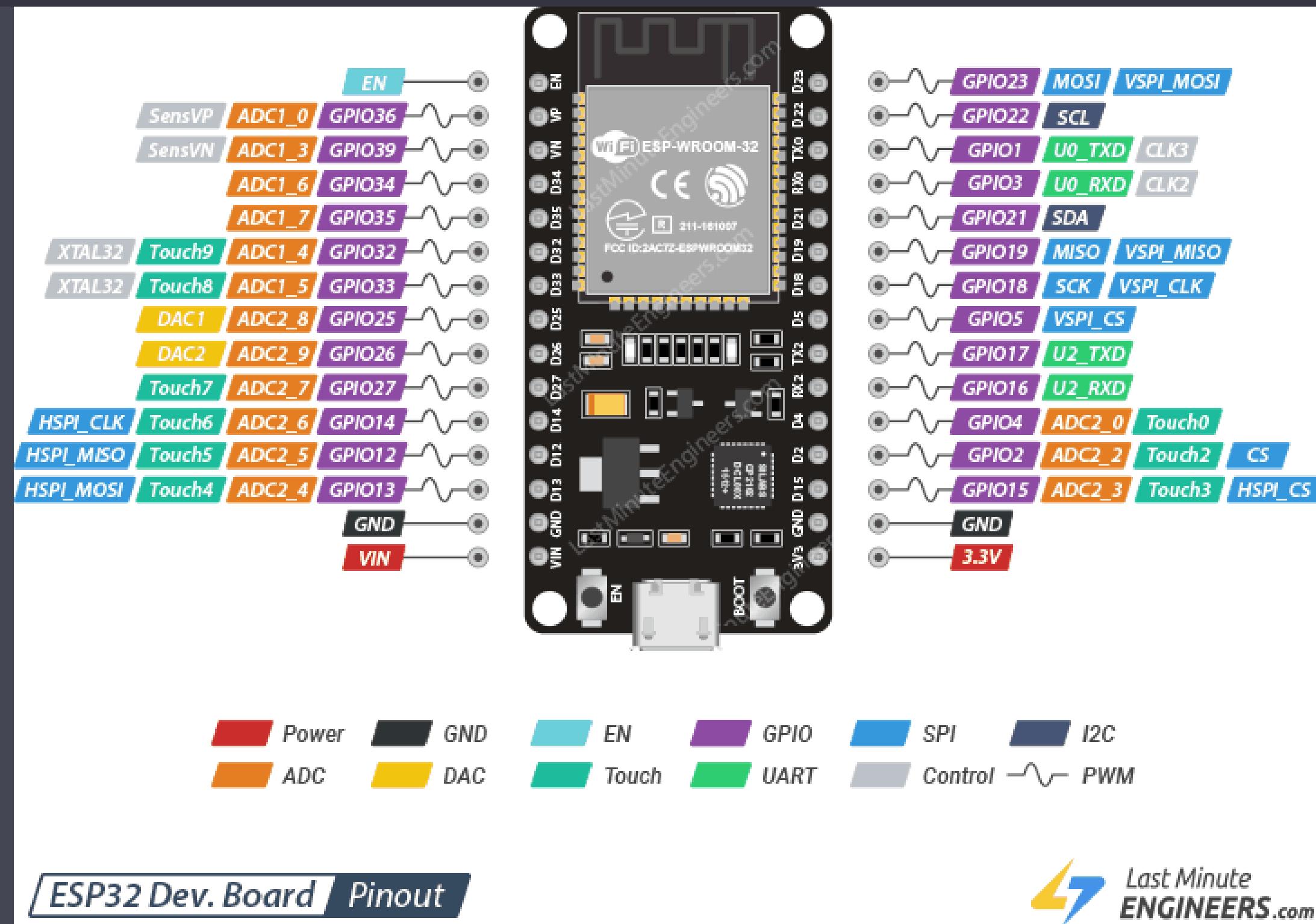
What is ESP32?

- ESP32 กำได้หมดเหมือน Arduino แคม spec ดีกว่า
- สามารถเชื่อมต่อกับ Wi-fi และใช้ Bluetooth ได้
- ประหยัดพลังงาน
- ราคายังคงกว่าบินเดิม



[ESP32 Datasheet](#)

ESP32



ESP32 vs Arduino Uno



Here's a comparison between the ESP32 and Arduino Uno in a table format:



Feature	ESP32	Arduino Uno
Microcontroller	ESP32-D0WDQ6 dual-core	ATmega328P
CPU	32-bit CPU, up to 240 MHz	8-bit CPU, 16 MHz
Digital I/O Pins	26 (some used for other functions)	14 (including 6 PWM outputs)
Analog Inputs	18 (12-bit resolution)	6 (10-bit resolution)
PWM Pins	16 (shared with digital I/O pins)	6 (dedicated PWM pins)
Flash Memory	4 MB	32 KB
SRAM	520 KB	2 KB
EEPROM	Not built-in	1 KB (ATmega328P)
Operating Voltage	3.3V	5V
Wi-Fi	Built-in Wi-Fi and Bluetooth	No built-in wireless communication
USB Interface	Micro USB	USB Type-B

Basic C language

Arduino Code Structure

Code run
top to
down

```
basic_tutorial.ino
1 void setup() {
2     // put your setup code here, to run once:
3 }
4
5 void loop() {
6     // put your main code here, to run repeatedly:
7 }
8
9 }
10
```

code ใน setup run ตอนเปิด
เพียงครั้งเดียว

code ใน loop run เสร็จ
จะกลับเข้า loop ใหม่ และทำงานไปเรื่อยๆ

code ใน setup
run ตอนเปิด
เพียงครั้งเดียว

Arduino Datatypes



Certainly! Here's the table with the Arduino data types:



Data Type	Size (bytes)	Range of Values
int	2	-32,768 to 32,767
unsigned int	2	0 to 65,535
long	4	-2,147,483,648 to 2,147,483,647
unsigned long	4	0 to 4,294,967,295
float	4	3.4028235E+38 to 3.4028235E+38 (approx.)
double	4	3.4028235E+38 to 3.4028235E+38 (approx.)
char	1	-128 to 127
boolean	1	true or false
byte	1	0 to 255

สังเกตได้ว่า datatype แต่ละชนิดจะใช้เก็บชนิดข้อมูลต่างกัน และมีระยะของค่าต่างกัน

Arduino Datatypes

วิธีประกาศตัวแปร

datatype variable = value;

datatype = ชนิดของตัวแปร

variable = ตั้งชื่อให้ตัวแปร

value = ค่าที่ต้องการกำหนดให้ตัวแปร

datatype variable;

เราสามารถตั้งชื่อตัวแปรไว้เลยๆ โดยไม่กำหนดค่าก็ได้

basic_tutorial.ino

```
1 int width = 75;  
2  
3 float hight = 172.4;  
4  
5 char name = 'p';  
6  
7 bool isReal = false;  
8
```

Variable Scope

Global variable

- เป็นการประกาศตัวแปรไว้ข้างนอกสุด ส่วนมากจะไว้ด้านบนของโปรแกรม
- Global var จะเก็บค่าตัวแปรจนกว่าจะทำงาน

Local variable

- เป็นการประกาศตัวแปรไว้ข้างใน function
- Local var สามารถใช้ได้แค่ภายใน function เมื่อ function ทำงานเสร็จก็จะหายไป

```
basic_tutorial.ino
1 // Global variable declaration
2 int height;
3 char name = 'g';
4
5 void setup() {
6     // Local variable declaration
7     int x, y;
8
9     // Actual initialization
10    x = 0;
11 }
12
13 void loop() {
14
15 }
16
```

Arduino Operators Arithmetic



Certainly! Here's a table listing common arithmetic operators used in Arduino programming along with their descriptions and examples:



Operator	Description	Example
+	Addition	<code>int sum = 5 + 3;</code>
-	Subtraction	<code>int difference = 10 - 4;</code>
*	Multiplication	<code>int product = 6 * 7;</code>
/	Division	<code>float quotient = 20 / 4;</code>
%	Modulus (remainder after division)	<code>int remainder = 17 % 3;</code>
++	Increment (by 1)	<code>int count = 0; count++;</code>
--	Decrement (by 1)	<code>int value = 10; value--;</code>

ใช้ +, -, *, / เมื่อ用คณิตศาสตร์กับไป (สามารถใช้งานได้ด้วย)

Arduino Operators Compound



Certainly! Here's a table listing common compound assignment operators used in Arduino programming along with their descriptions and examples:



Operator	Description	Example
<code>+=</code>	Add and assign	<code>int x = 5; x += 3;</code>
<code>-=</code>	Subtract and assign	<code>int y = 10; y -= 2;</code>
<code>*=</code>	Multiply and assign	<code>int z = 4; z *= 6;</code>
<code>/=</code>	Divide and assign	<code>int a = 20; a /= 5;</code>
<code>%=</code>	Modulus and assign	<code>int b = 17; b %= 4;</code>

`x += 1` \equiv `x = x + 1`

Arduino Operators Comparison



Of course! Here's a table listing common comparison operators used in Arduino programming along with their descriptions and examples:



Operator	Description	Example
<code>==</code>	Equal to	<code>if (x == y) { ... }</code>
<code>!=</code>	Not equal to	<code>if (x != y) { ... }</code>
<code>></code>	Greater than	<code>if (x > y) { ... }</code>
<code><</code>	Less than	<code>if (x < y) { ... }</code>
<code>>=</code>	Greater than or equal to	<code>if (x >= y) { ... }</code>
<code><=</code>	Less than or equal to	<code>if (x <= y) { ... }</code>

ใช้สำหรับเปรียบเทียบค่าได้

Arduino Operators Boolean



Certainly! Here's a table listing common boolean operators (logical operators) used in Arduino programming along with their descriptions and examples:



Operator	Description	Example
<code>&&</code>	Logical AND	<code>if (x > 0 && y < 10) { ... }</code>
<code> </code>	Logical OR	<code>if (x > 0 y > 0) { ... }</code>
<code>!</code>	Logical NOT	<code>if (!isTrue) { ... }</code>

ใช้สำหรับเชื่อมการเปรียบเทียบจัดการ logic ให้เงื่อนไข

Condition **if-else**: ใช้สำหรับสร้างเงื่อนไขให้โปรแกรมตามต้องการ

```
if (condition) {  
    1st command;  
}  
  
else {  
    2nd command;  
}
```

ถ้าเงื่อนไขใน **condition** เป็นจริง
จะทำ 1st command และจบ

แต่ถ้า **condition** เป็นเท็จ
จะไปกำกัตรง **else** (command 2)

สามารถต่อเงื่อนไขหลัง **else** ได้

```
1 int x = 5;  
2  
3 if (x > 20) {  
4     printf("wah wah");  
5 }  
6 else {  
7     printf("wib wab");  
8 }
```

```
if (condition1) {  
    1st command;  
}
```

```
else if (con2) {  
    2nd command;  
}
```

```
else {  
    3rd command;  
}
```

Condition if-else

```
if (condition1 boolean condition2) {  
    1st command;  
}
```

ใช้ Boolean operator มาเชื่อม 2 condition ได้

```
1 int x = 7;  
2  
3 if (x > 3 && x < 6) {  
4     printf("haha");  
5 }  
6 else {  
7     printf("huhi");  
8 }  
9 |
```

Switch-case

```
1 int num = 5;
2 switch (num){
3     case 3:
4         printf("3");
5         break; ←
6     case 4:
7         printf("4");
8         break;
9     case 5:
10        printf("5");
11        break;
12     default:
13         printf("i dont know")
14         break;
15 }
```

กำเส็จอย่าลืมเบรค

เอาค่าที่มาจาก switch มาเช็คว่าเป็น case ไหน, ไม่ใช่อะไรเลยจะไปที่ Default

Function

```
1 void printHello() {  
2     printf("Hello World!");  
3 }  
4  
5 printHello();  
6  
7  
8
```

function แบบไม่ส่งค่า , รับค่าเลย

การส่งค่ากลับควรบอก

Datatype ให้ตรงกันด้วย

รับค่ามาให้กำหนด

Datatype ให้ตรงกันด้วย

```
1 int multiply(int a, int b) {  
2     return a * b;  
3 }  
4  
5  
6 printf("%d", multiply(2,4));  
7  
8
```

มีการส่งค่ากลับจุดที่เรียกใช้ function

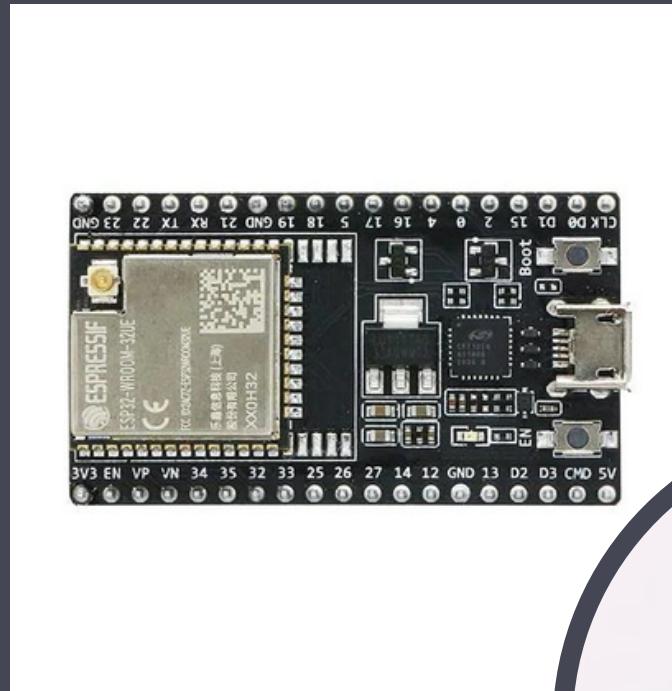
Parameter รับค่า
จากจุดที่เรียกใช้

Argument ส่งค่าให้
function

Learning Project

Material

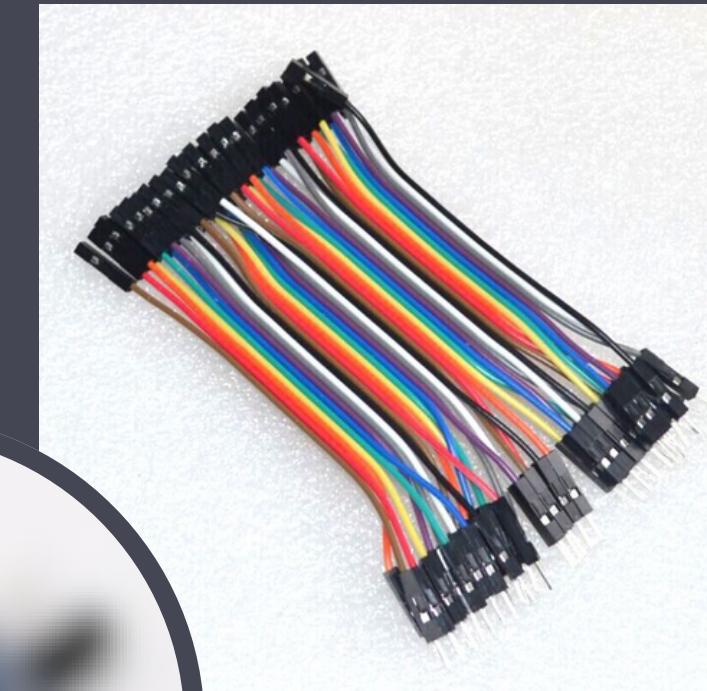
ESP32



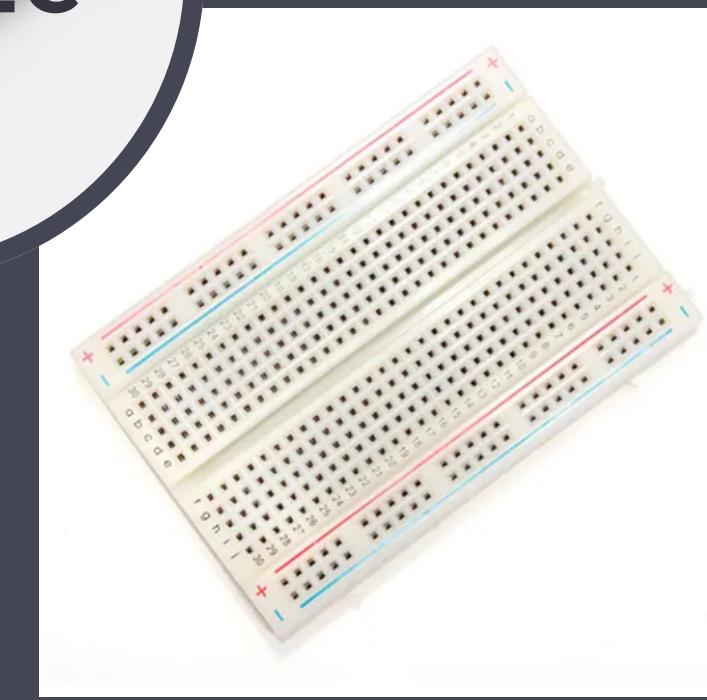
Micro USB Cable



Module

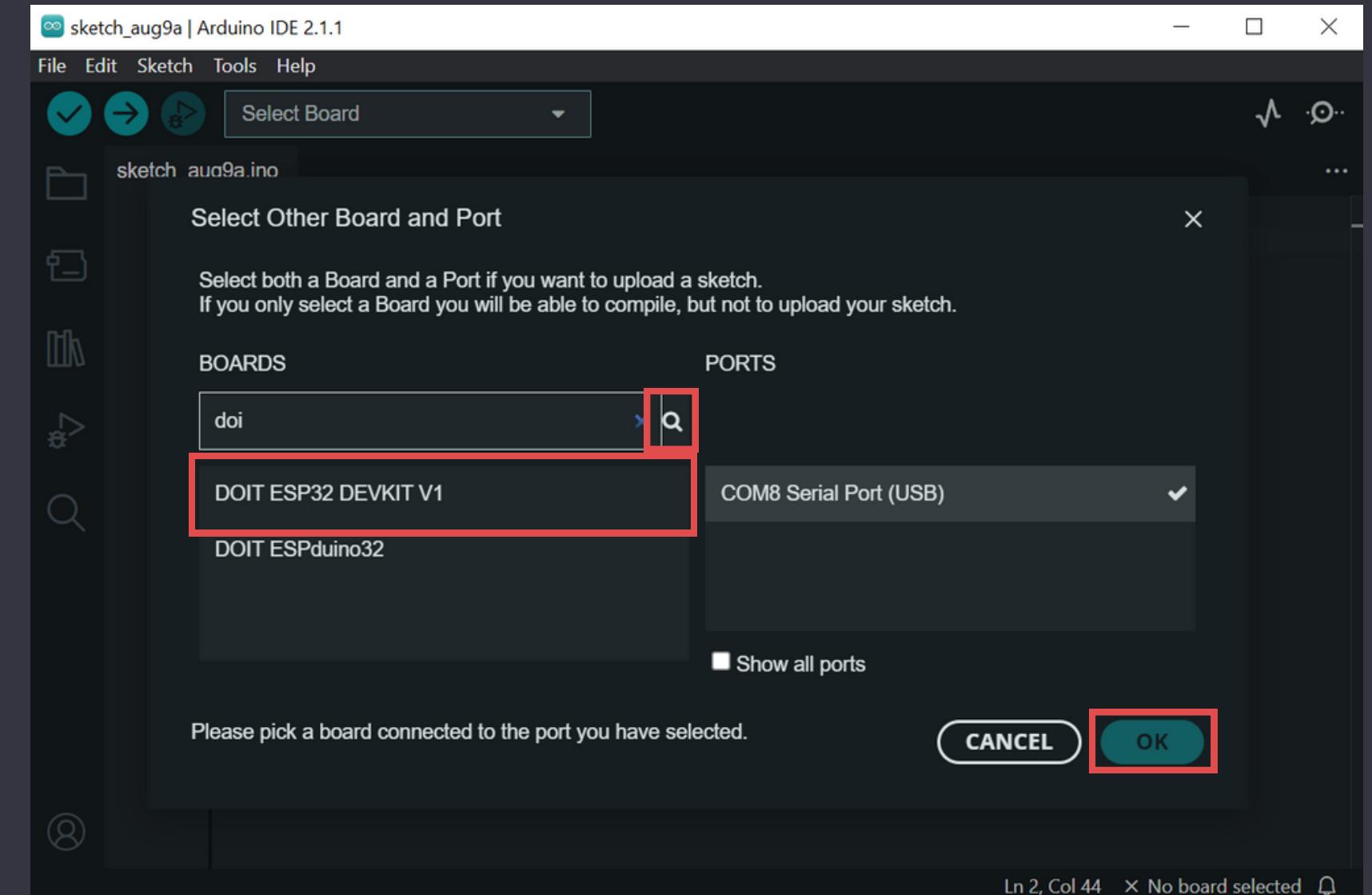
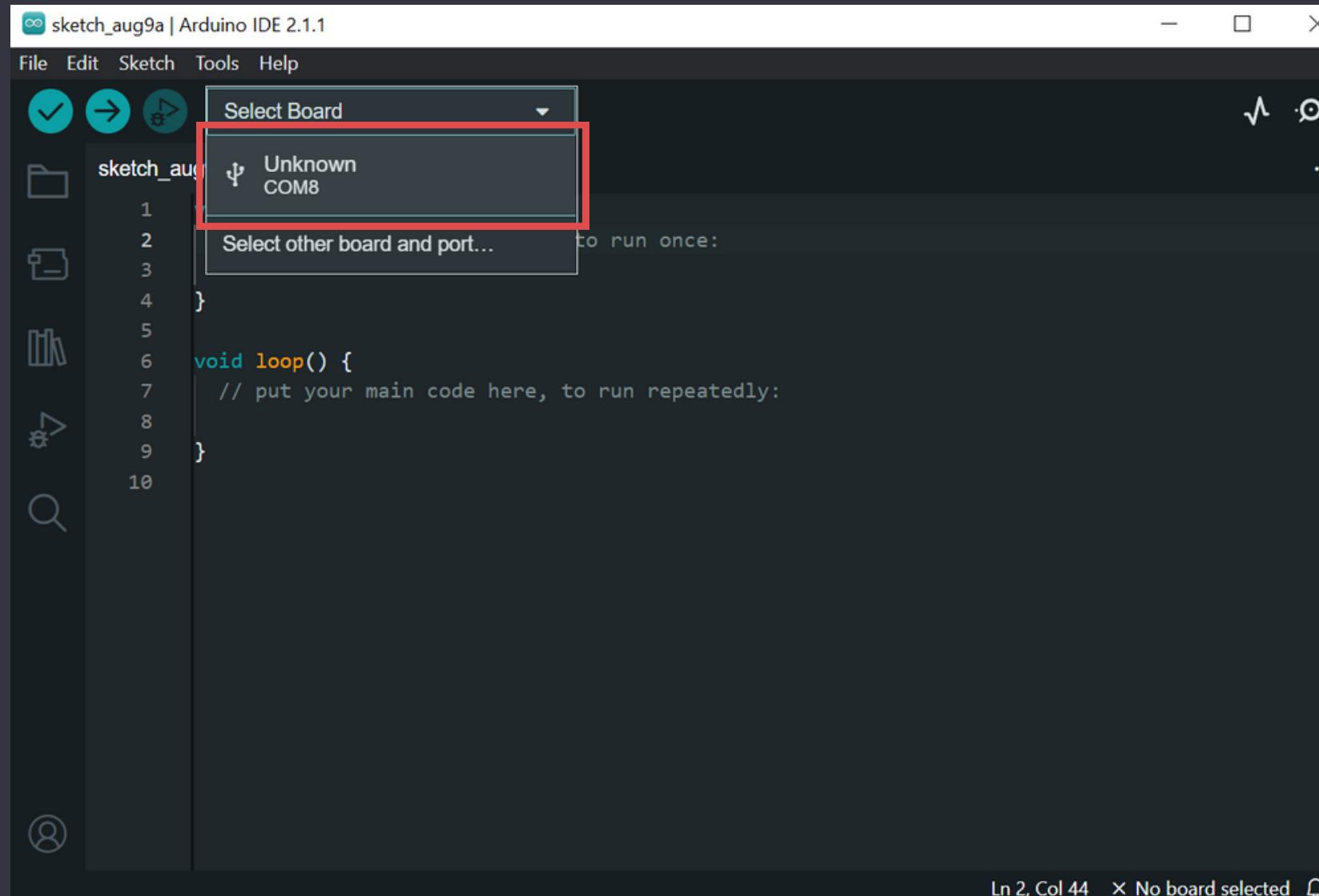


Jump wire



Breadboard

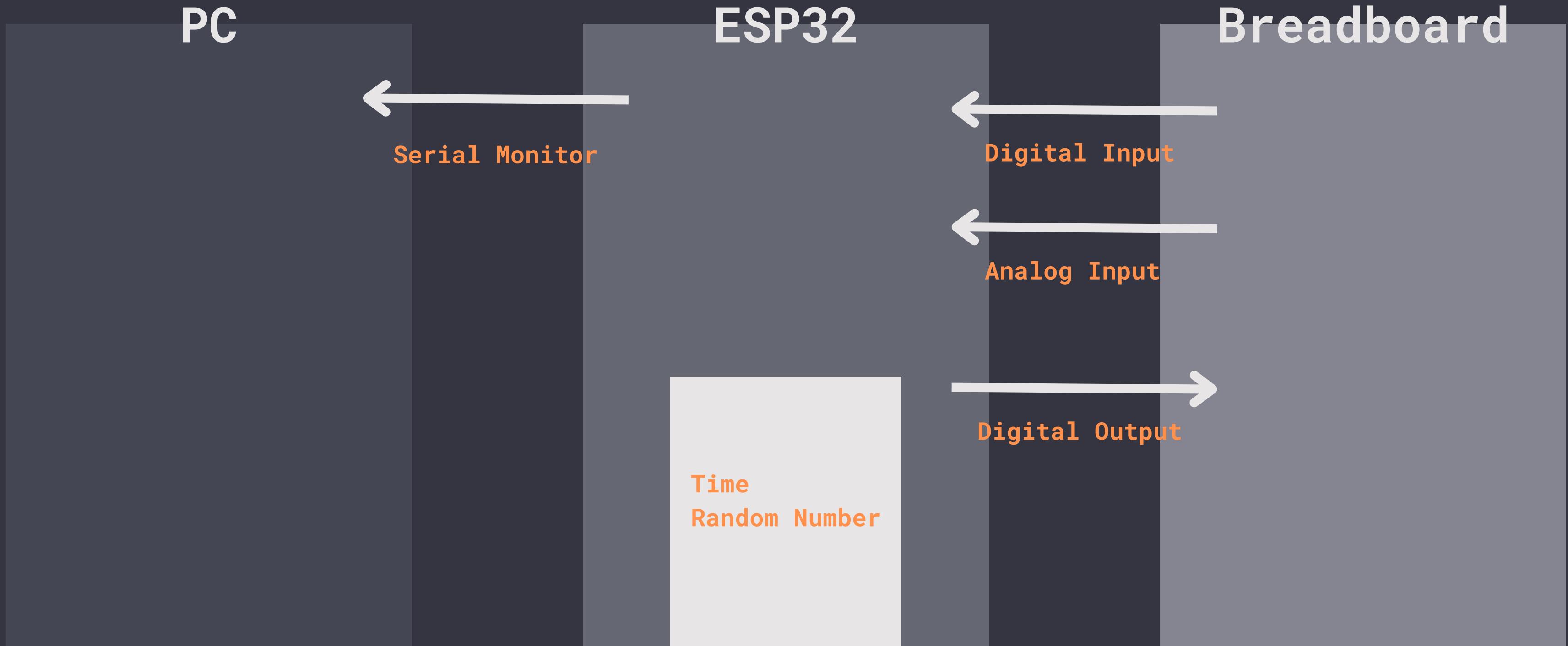
Hardware Preparation



Select Board > COM...

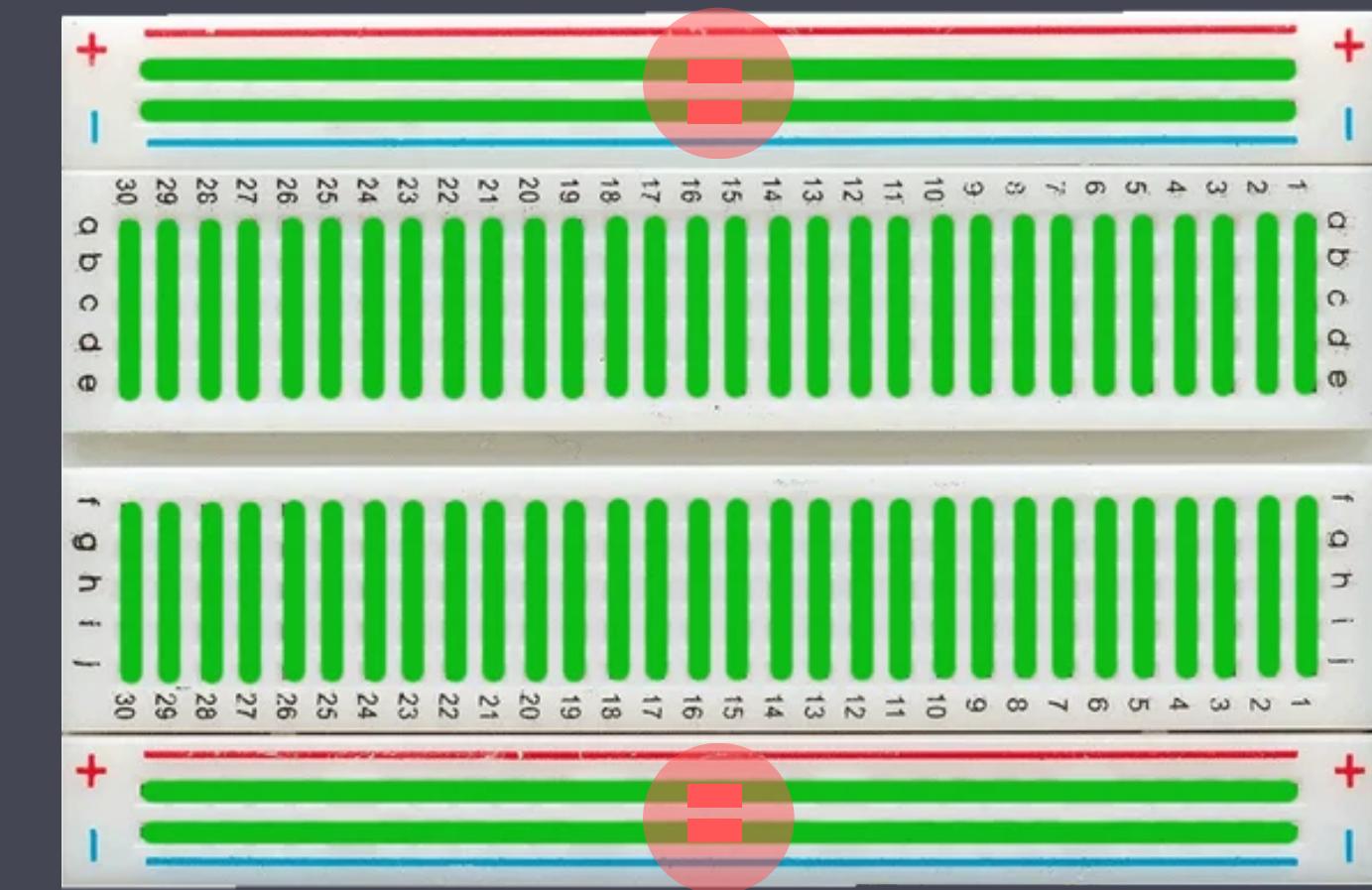
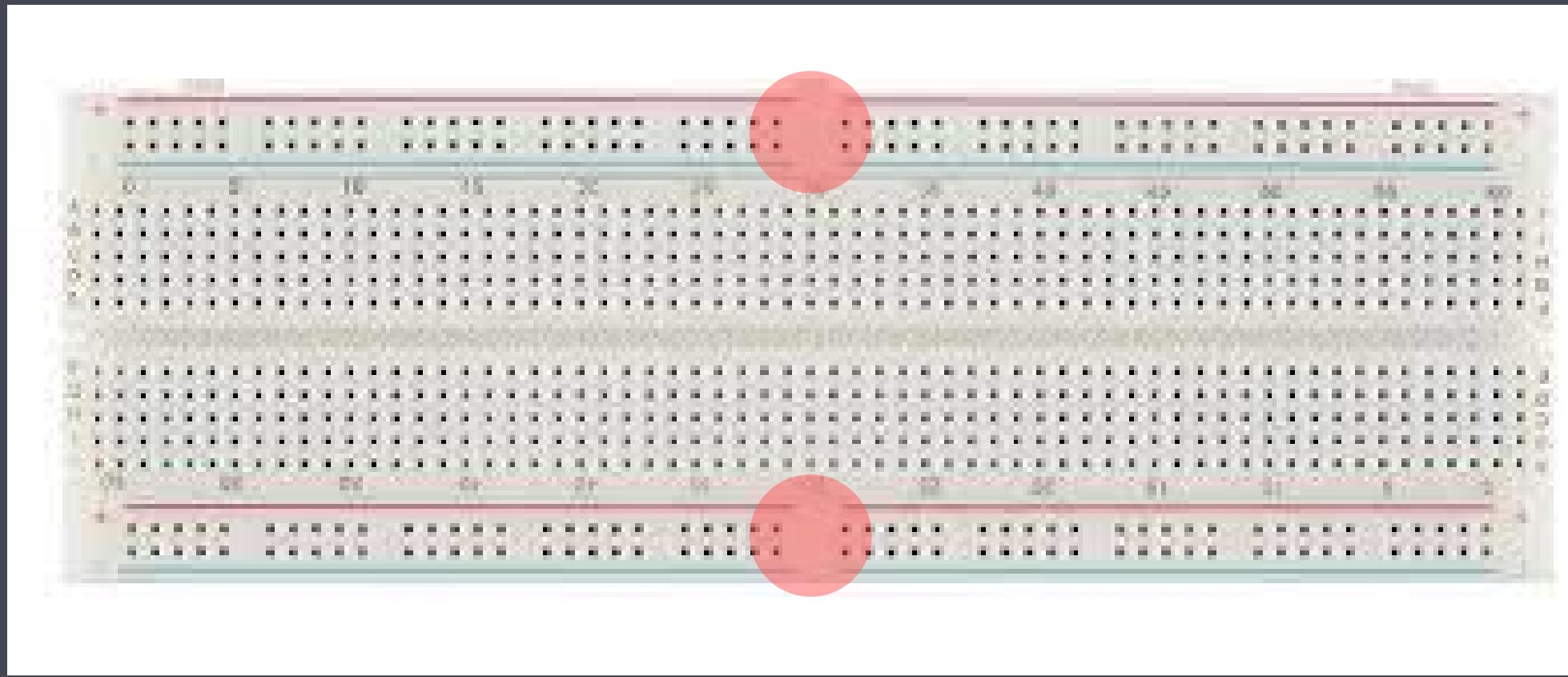
BOARDS > "DOIT ESP32 DEVKIT V1" > OK

What we will learn?



Breadboard or protoboard

เป็นอุปกรณ์ใช้ทดลองต่อวงจรก่อน เพื่อไม่ให้ผิดพลาดตอนนำไปใช้งานจริง



บางครั้ง breadboard ที่มีความยาวจะแบ่งเส้นแบนตอนเป็น 2 เส้น (ไม่เชื่อมกัน)

Syntax #1: Serial Monitor

Serial Monitor ใช้ในการสื่อสารระหว่าง PC กับ Board(ESP32) โดยคุยกันผ่าน Serial โดยต้องมีความเร็วเท่ากัน

Syntax: **Serial.begin(baudrate);**

Benefit: เป็นการเริ่มใช้ Serial Monitor พร้อมกำหนดความเร็ว

Parameter:

baudrate = ความเร็วในการส่งสัญญาณต่อวินาที

300, 600, 1200, 2400, 4800, 9600, 14400,
19200, 28800, 38400, 57600, 115200

Syntax #1: Serial Monitor

Syntax: `Serial.print(val);`

Benefit: แสดงค่าผ่าน Serial Monitor

Syntax: `Serial.println(text);`

Benefit: แสดงค่าผ่าน Serial Monitor พร้อมขึ้นบรรทัดใหม่

Parameter:

`val` = ค่าที่ต้องการส่งไป

Example #1: Serial Monitor

basic_tutorial.ino

```
1 void setup() {  
2     Serial.begin(9600);  
3  
4     Serial.print("hello"); // print -> ໃນຂໍ້ມູນກັດໃໝ່ print ຜ່ານ Serial Monitor ໄດ້  
5     Serial.print("1");  
6  
7     Serial.println(" ");  
8  
9     Serial.println("hello"); // println -> ຂໍ້ມູນກັດໃໝ່  
10    Serial.println("2");  
11 }  
12  
13 void loop() {  
14     Serial.println("hello1");  
15     Serial.println("hello");  
16     Serial.println("2");  
17 }
```

Output Serial Monitor X

Message (Enter to send message to 'DOIT ESP32 DEVKIT V1' on 'COM8')

New Line 9600 baud

Ln 7, Col 17 DOIT ESP32 DEVKIT V1 on COM8 ⚡ 2

The screenshot shows the Arduino IDE interface. On the left, the code for 'basic_tutorial.ino' is displayed. It includes a setup function that initializes the serial port at 9600 baud and prints "hello" and "1" to the serial monitor. It also includes a loop function that prints "hello1", "hello", and "2" to the serial monitor. On the right, the 'Serial Monitor' window is open, showing the received messages. The window has tabs for 'Output' and 'Serial Monitor'. The status bar at the bottom indicates the current line (Ln 7, Col 17), the board (DOIT ESP32 DEVKIT V1), the port (COM8), and the baud rate (9600). There are also icons for a refresh, a power button, and a help menu.

Upload Code

The screenshot shows the Arduino IDE interface. On the left is the code editor with the file named 'basic_tutorial.ino'. The code contains a setup function that initializes serial communication at 9600 baud and prints "hello" and "1" to the serial monitor. It then prints a blank line and repeats the process. The status bar at the bottom shows the output of the compilation and upload process.

```
basic_tutorial.ino
1 void setup() {
2     Serial.begin(9600);
3
4     Serial.print("hello");
5     Serial.print("1");
6
7     Serial.println(" ");
8
9     Serial.println("hello");
10    Serial.println("2");
11}
12
13
```

Output: Sketch uses 260221 bytes (19%) of program storage space. Maximum is 1310720 bytes.
Global variables use 21344 bytes (6%) of dynamic memory, leaving 306336 bytes for local variables.
esptool.py v4.5.1
Serial port COM8
Connecting.....

Uploading... Done compiling.

เมื่อขึ้น connecting..... > ให้กดปุ่ม Boot ค้าง

The screenshot shows the Arduino IDE interface with the same sketch as the first window. A red arrow points from the 'Connecting.....' text in the first window to the 'Uploading...' progress bar in this window. The progress bar is nearly full, indicating the upload is almost complete. The status bar at the bottom shows the upload progress and device information.

```
basic_tutorial.ino
1 void setup() {
2     Serial.begin(9600);
3
4     Serial.print("hello");
5     Serial.print("1");
6
7     Serial.println(" ");
8
9     Serial.println("hello");
10    Serial.println("2");
11}
12
13
```

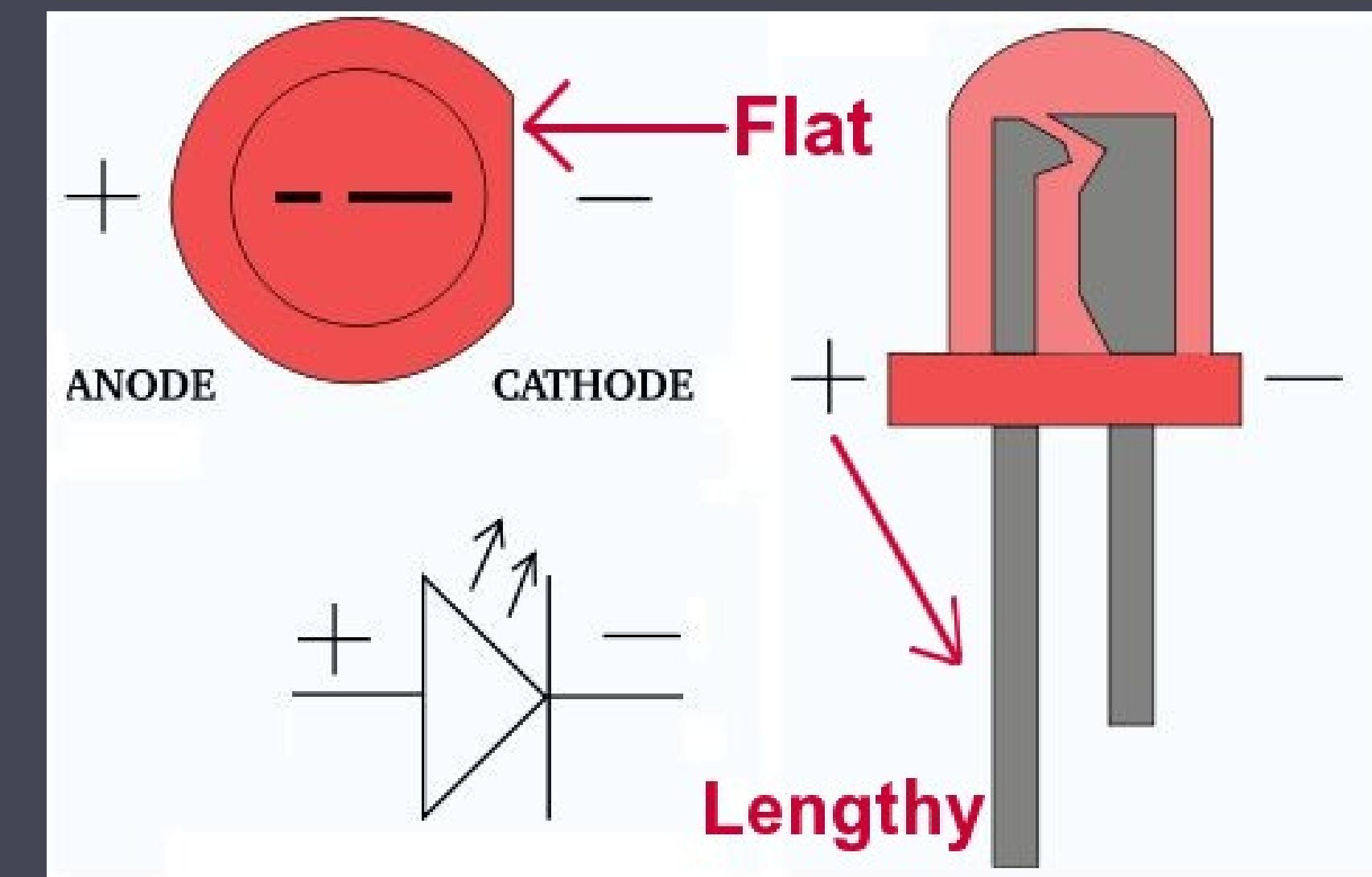
Output: Compressed 3072 bytes to 146
Writing at 0x00008000... (100 %)
wrote 3072 bytes (146 compressed) at 0x00008000 in 0.1 seconds (effective 186.1 kbit/s)...
Hash of data verified.
Compressed 8192 bytes to 146
Writing at 0x0000e000... Uploading...

Ln 7, Col 17 DOIT ESP32 DEVKIT V1 on COM8 ⌚ 3

ปล่อยเมื่อขึ้น (100 %)

LED: Light Emitting Diode

เป็นไดโอดที่ทำมาแบบพิเศษ เมื่อไดรับแรงดันไฟอัลตรอน จะสามารถเปล่งแสงออกมานะ (ไม่สามารถเกิน 20mA)



เวลาต่อ LED อย่าลืมต่อตัวต้านทาน (170 - 220 Ohms)

Syntax #2: Digital Output

กำหนดขาเป็น OUTPUT และจึงส่งสัญญาณดิจิทัล (0, 1) ไปที่ขาบีน เพื่อใช้กับอุปกรณ์ต่าง ๆ

Syntax: `pinMode(pin, mode);`

Benefit: ใช้ในการกำหนดขาของบอร์ดว่าต้องการตั้ง mode อะไร

Syntax: `digitalWrite(pin, logic);`

Benefit: กำหนด output logic ในขาที่กำหนด (HIGH = VCC, LOW = GND)

Parameter:

`pin` = ขาที่จะกำหนด

`logic` = HIGH or LOW

`mode` = INPUT, OUTPUT or INPUT_PULLUP

Syntax #3: Time

จัดการเกี่ยวกับเวลาใน Arduino

Syntax: `delay(time);`

Benefit: หน่วงเวลาให้เท่ากับ time มิลลิวินาที โดยระหว่างนี้ทั้งโปรแกรมจะหยุดรอ

Syntax: `millis();`

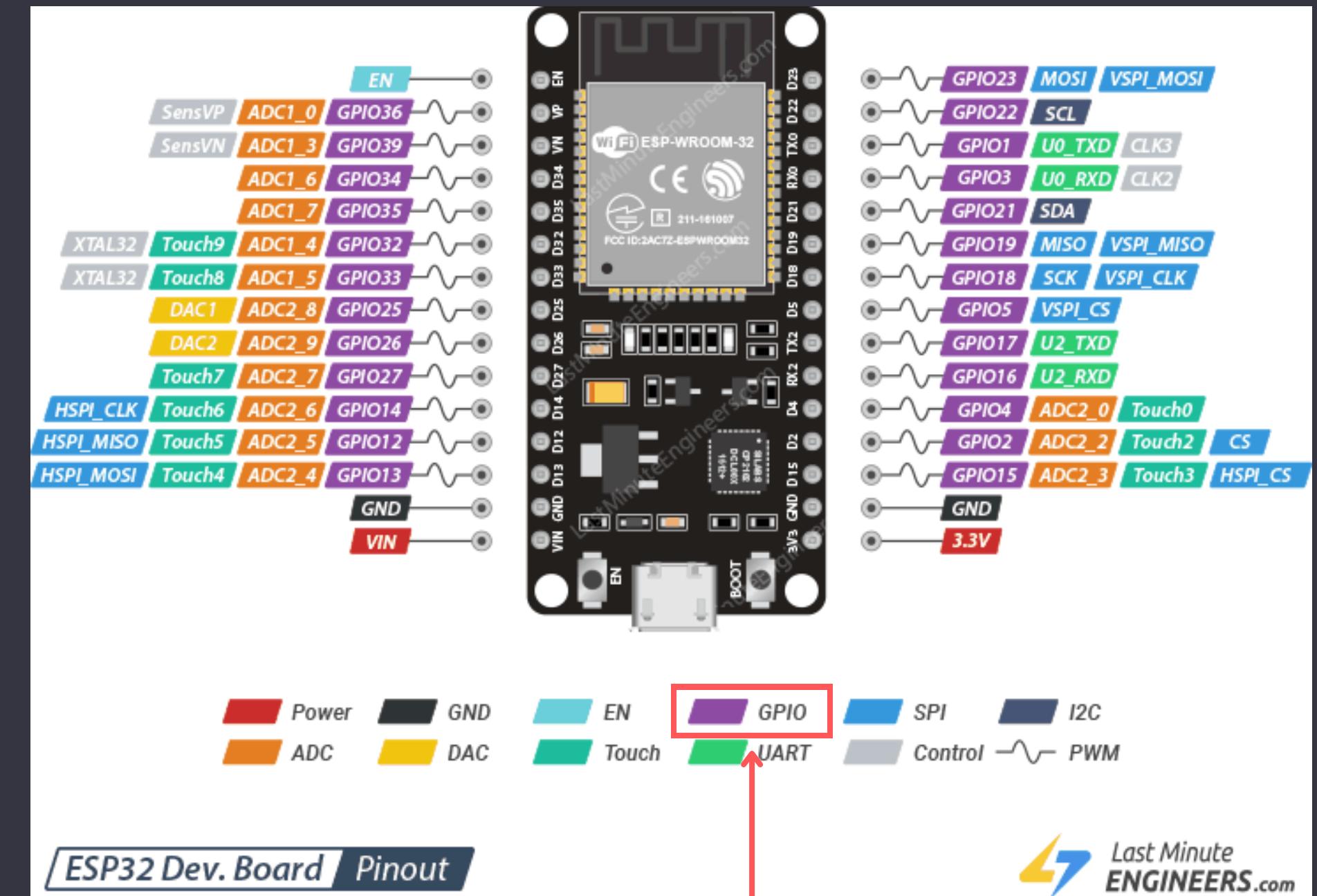
Benefit: function ที่ส่งค่าจำนวนมิลลิวินาที โดยนับตั้งแต่โปรแกรมเริ่มทำงาน

Parameter:

`time` = เวลาที่ต้องการหน่วง (millisec)

Example #2: Digital Output

```
basic_tutorial.ino
1 #define LED_pin 4
2
3 void setup() {
4     pinMode(LED_pin, OUTPUT);
5 }
6
7 void loop() {
8     digitalWrite(LED_pin, HIGH);
9     delay(250);
10    digitalWrite(LED_pin, LOW);
11    delay(250);
12 }
13
```

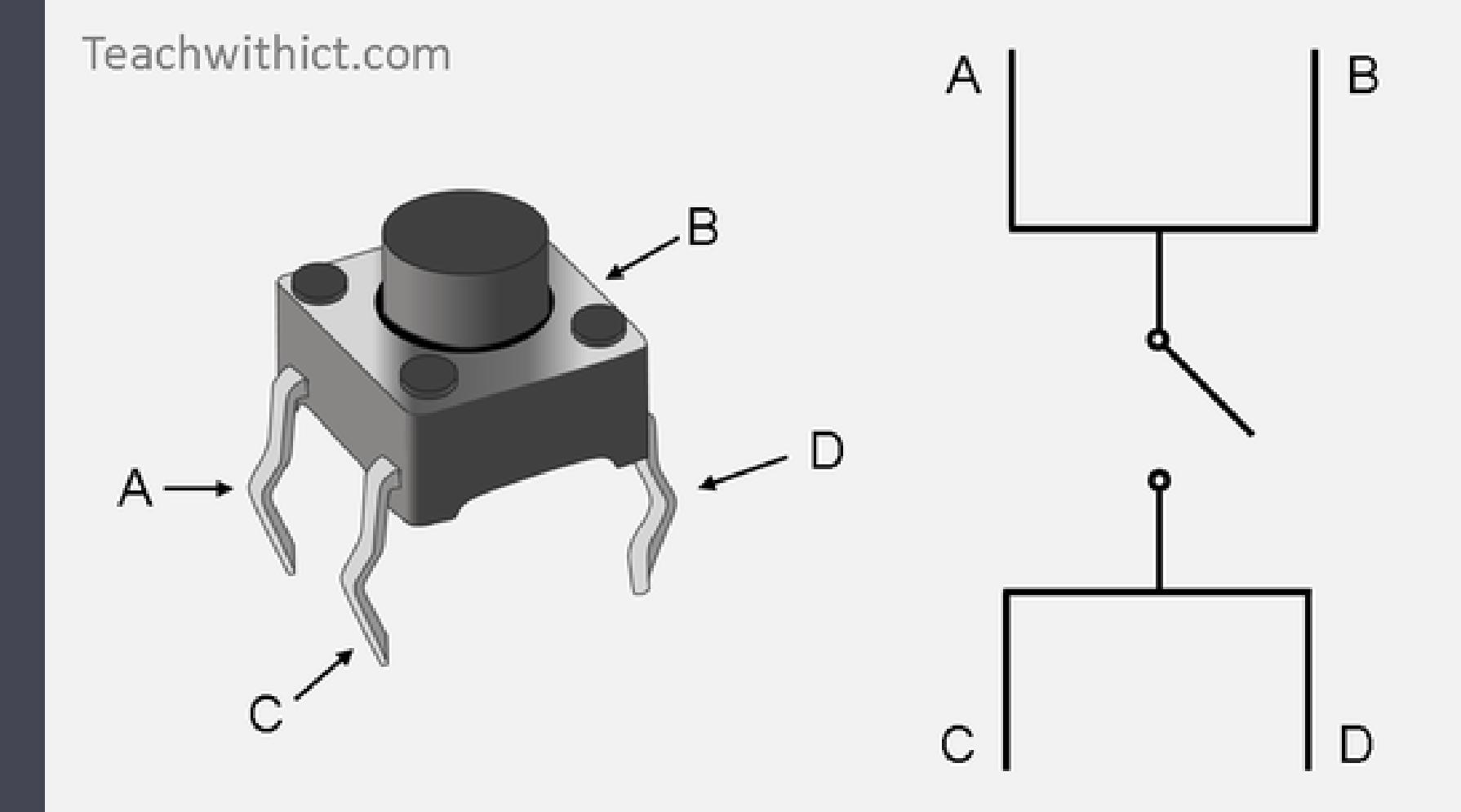


ຈະໄດ້ LED ກຣະວິສຸປົມ ທ່ານ

digital pin = GPIO

Button

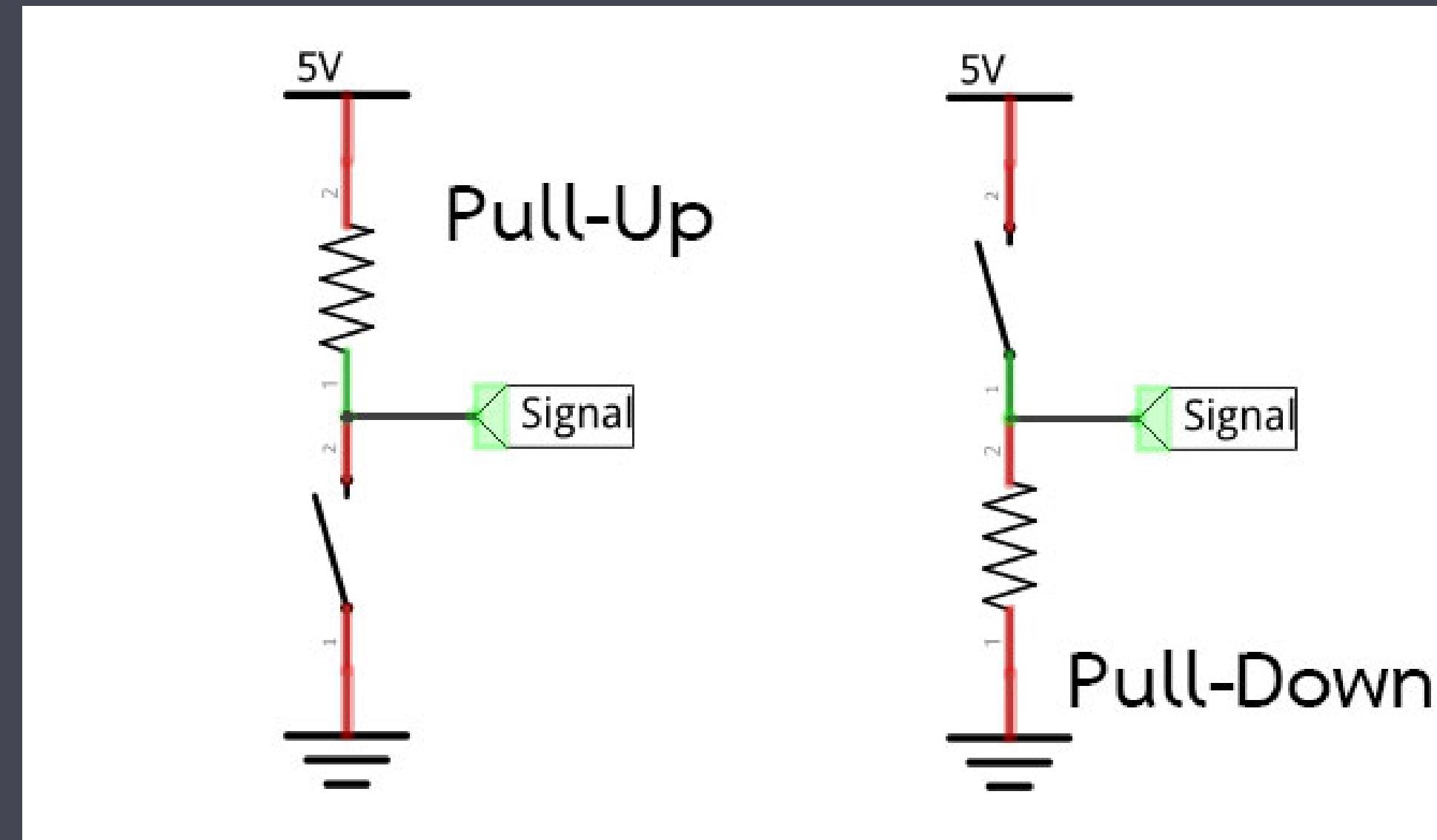
ปุ่มกดธรรมด้า ๆ แต่สามารถนำไปใช้ประยุกต์ได้หลากหลาย ใช้คันระหว่างสายไฟเมื่อกดจะทำให้เป็นวงจรปิด



A - B, C - D มองเป็นเส้นเดียวกัน เวลาใช้งานให้จับคู่คนละเส้น (eg. A - C, A - D)

Button

การต่อปุ่มกดมี 2 แบบ Pull-Up และ Pull-Down



Pull-Up = ปกติ ดึงเป็น HIGH
เมื่อกดกล้ายเป็น LOW

Pull-Down = ปกติ ดึงเป็น LOW
เมื่อกดกล้ายเป็น HIGH

Syntax #4: Digital Input

กำหนดขาเป็น INPUT และจึงอ่านสัญญาณดิจิทัล (0, 1) จากขาที่นี่ มาใช้งานต่อในโปรแกรม

Syntax: `pinMode(pin, mode);`

Benefit: ใช้ในการกำหนดขาของบอร์ดว่า ขาที่ไหนต้องการรับข้อมูลเข้า (input)
หรือส่งข้อมูลออก (output)

Syntax: `digitalRead(pin);`

Benefit: อ่านค่าที่ได้จากขาที่กำหนด (HIGH = 1, LOW = 0)

Parameter:

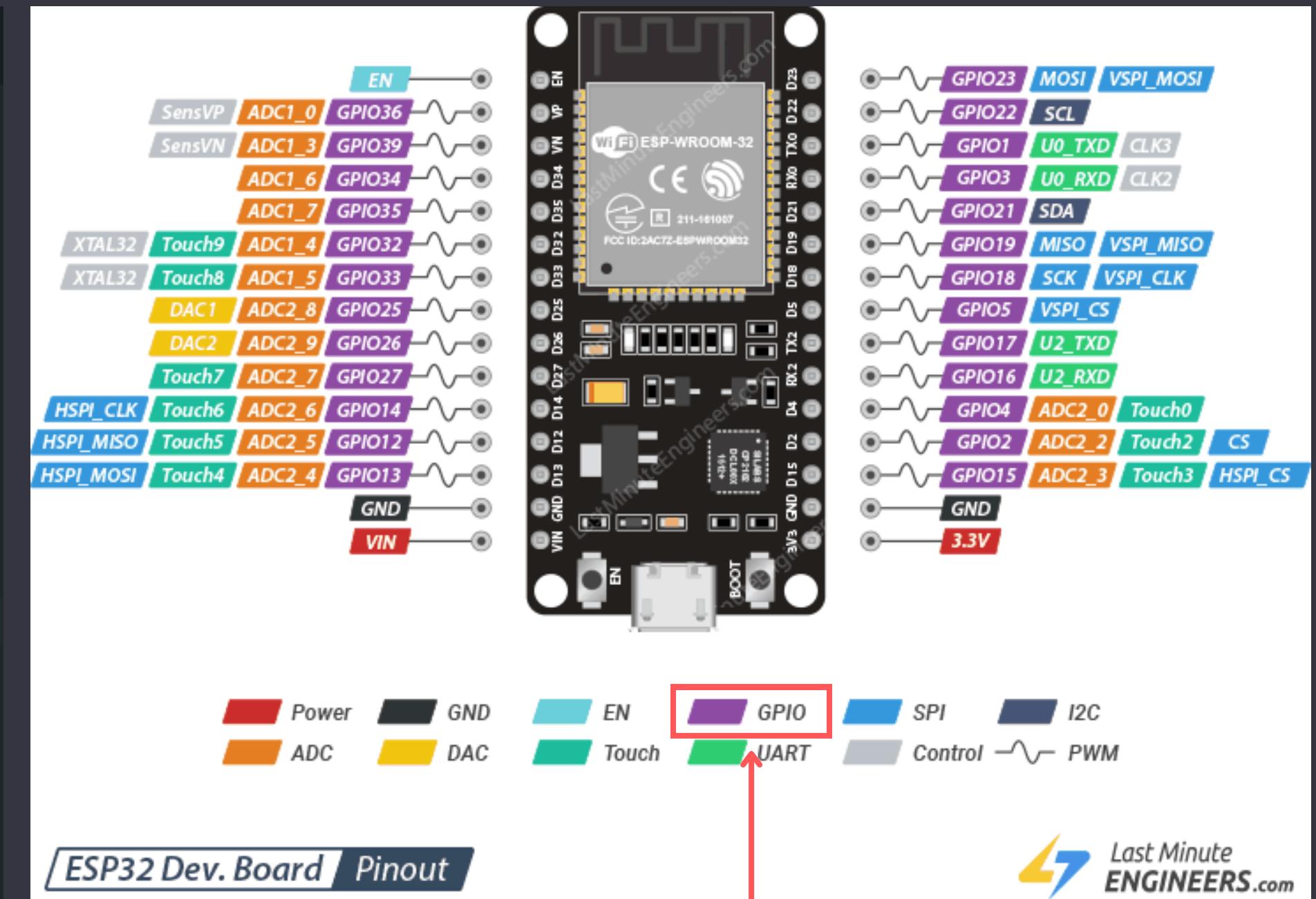
`pin` = ขาที่จะกำหนด

`mode` = INPUT(Pull-Down), OUTPUT or INPUT_PULLUP

Example #3: Digital Input

```
basic_tutorial.ino

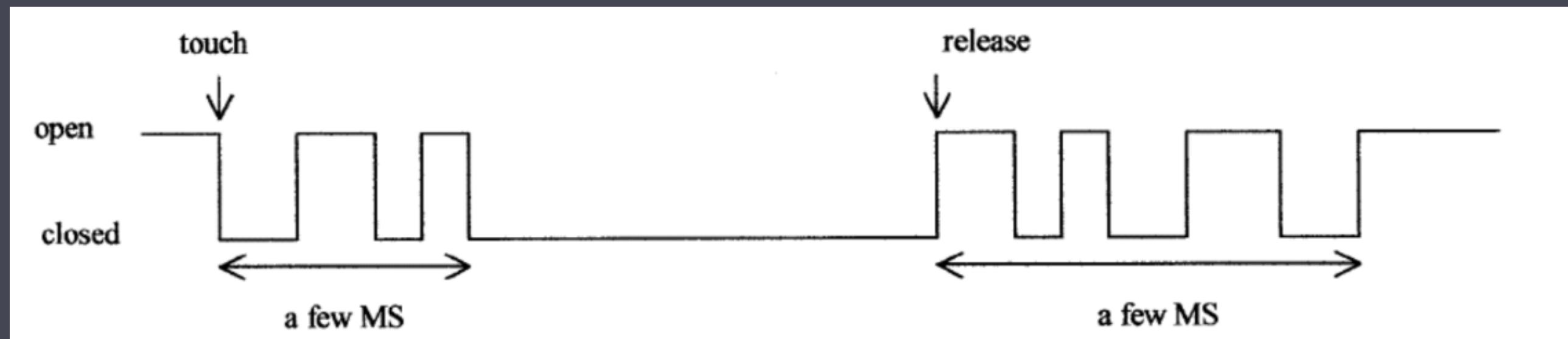
1 #define button 5 // switch input Active Low
2 #define pressed LOW
3
4 void setup(){
5     Serial.begin(9600);
6     pinMode(button, INPUT_PULLUP);
7 }
8 void loop(){
9     int ReadSwitch = digitalRead(button);
10
11     if(ReadSwitch == pressed)
12     {
13         Serial.println(ReadSwitch);
14         delay(500);
15     }
16 }
17 }
```



ເສື້ອງກວ່າປຸນຄດຍັງ

Button: Bounce Problem

ในการกดสวิตซ์ 1 ครั้ง จะมีช่วงเวลาสั้นๆ ที่มีสัญญาณเหมือนการกดสวิตซ์หลายครั้ง



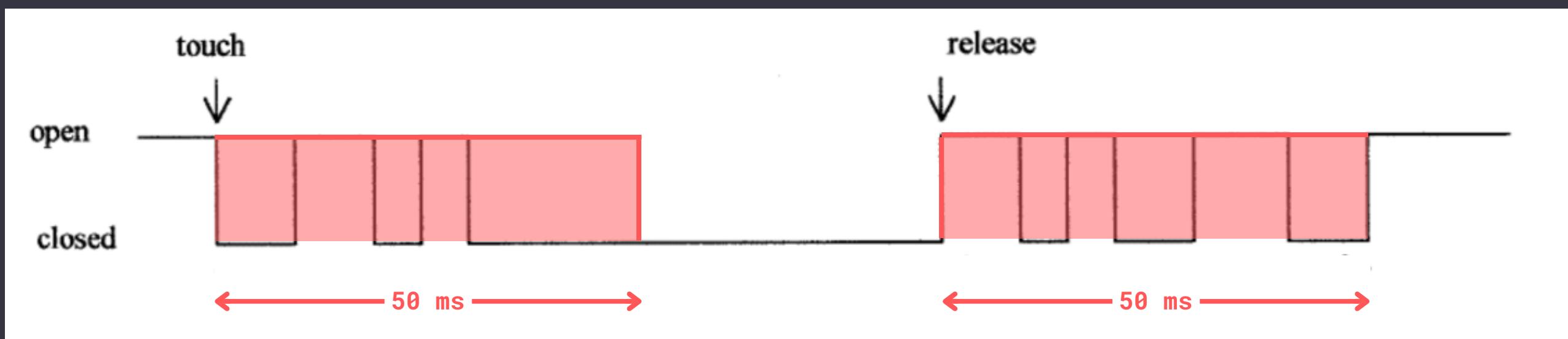
เป็น hardware error ที่เกิดจากตัวปุ่มกดเอง

เราเมวีรีแก้ปัญหานี้

Button: Software De-bouncing

ໄອເດີຍຄ່ອງເຮົາຈະໜ່ວງເວລາ 50ms (ກັ້ນຕອນກົດແລະຕອນປໍລ່ອຍ) ເພື່ອຂ້າມໜ່ວງທີ່ເກີດ bounce ແລ້ວຈຶ່ງອ່ານຄ່າຕ່ອງ

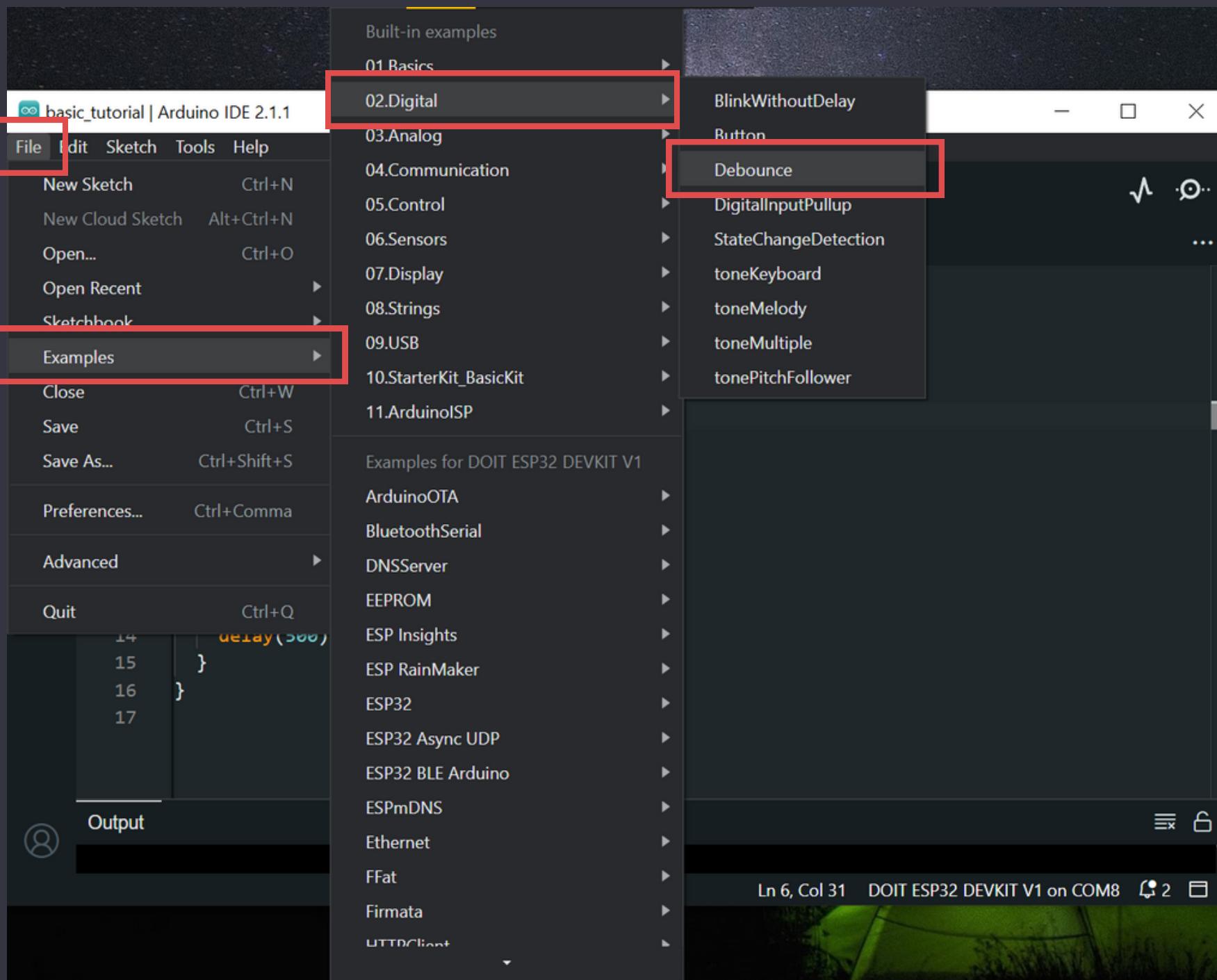
```
basic_tutorial.ino
...
1 const int buttonPin = 2;
2 const int ledPin = 13;
3
4 int ledState = HIGH;
5 int buttonState;
6 int lastButtonState = LOW;
7
8 unsigned long lastDebounceTime = 0;
9 unsigned long debounceDelay = 50;
10
11 void setup() {
12     pinMode(buttonPin, INPUT);
13     pinMode(ledPin, OUTPUT);
14
15     digitalWrite(ledPin, ledState);
16 }
17
```



Button: Software De-bouncing

```
18 void loop() {
19     int reading = digitalRead(buttonPin);
20
21     if (reading != lastButtonState) {
22         lastDebounceTime = millis();
23     }
24
25     if ((millis() - lastDebounceTime) > debounceDelay) {
26         if (reading != buttonState) {
27             buttonState = reading;
28
29             if (buttonState == HIGH) {
30                 ledState = !ledState;
31             }
32         }
33     }
34
35     digitalWrite(ledPin, ledState);
36
37     lastButtonState = reading;
38 }
39
```

Example #4: Software De-bouncing

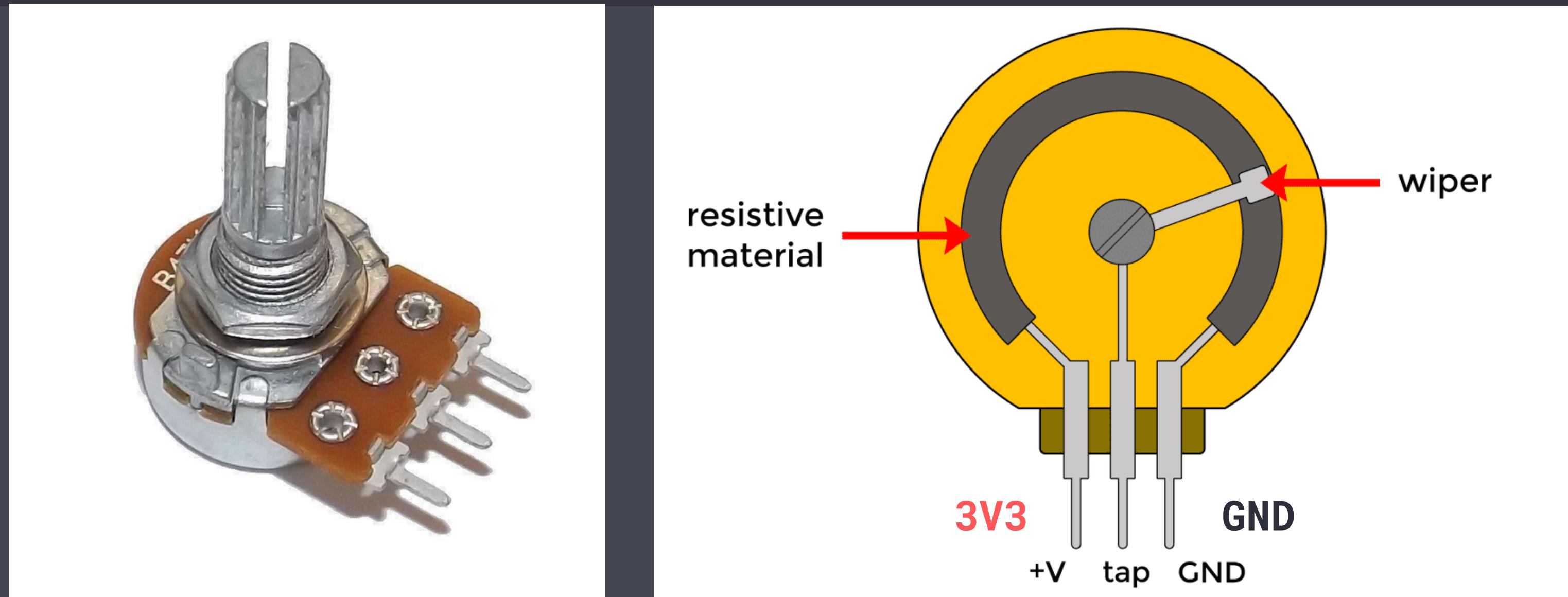


Copy example code
มาแก้ไขเป็น Pin ที่เราจะใช้
buttonPin > (input)
ledPin > (output)

File > Example > 02.Digital > Debounce

Variable Resistor: ตัวต้านทานปรับค่าได้

ต่อขา analog ของบอร์ด กับขากลางของ VR ถ้าเราหมุนไปฝั่ง VCC จะได้ค่ามาก ตรงข้ามจะได้ค่าน้อย



ตัวต้านทานปรับค่าได้ (VR) จะมีค่าความต้านทานสูงสุดเมื่อยกที่ข้างบน

Syntax #5: Analog Input

กำหนดขาเป็น INPUT และจึงอ่านสัญญาณอนาล็อก (0-4095) จากขาที่ 0 มาใช้งานต่อในโปรแกรม

Syntax: **analogRead(pin);**

Benefit: value 0-4095(0-3.3V)

Parameter:

pin = ขาที่จะกำหนด (analog)

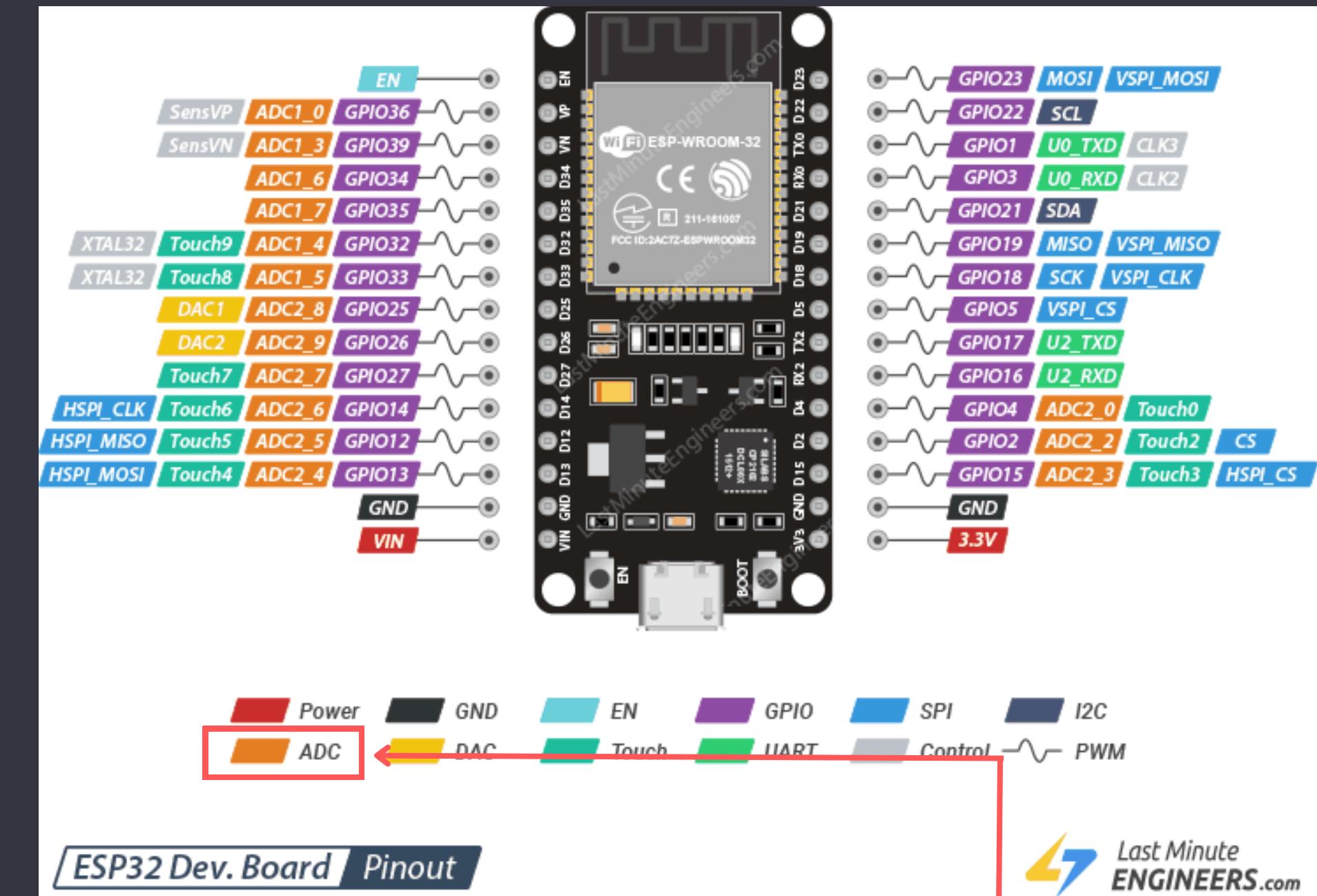
mode = INPUT, OUTPUT, INPUT_PULLUP

ใช้ขา analog สำหรับอ่านค่า analog อย่างเดียวจึงไม่ต้อง set pinMode

Example #5: Analog Input

basic_tutorial.ino

```
1 int potPin = A5;
2 int potVal = 0;
3
4 void setup() {
5     Serial.begin(9600);
6     delay(1000);
7 }
8
9 void loop() {
10    potVal = analogRead(potPin);
11    mapVal = map(potVal,0,4096,0,100)
12    Serial.println(mapVal);
13    delay(500);
14 }
15
```



Bonus Syntax: `map(val, fromLow, fromHigh, toLow, toHigh)`

eg. แปลงค่าที่ปรับได้เป็น 0 - 100

analog pin = ADC

Syntax #6: Random Number

ใช้ในการสุ่มเลข โดยต้องมีการ randomSeed ก่อน เพื่อไม่ให้ random ออกมาซ้ำ pattern

Syntax: `randomSeed(analogRead(A0));`

Benefit: initial random number generator

Syntax: `random(max)` or `random(min, max);`

Benefit: ทำการสุ่มระหว่างเลข min ไปจนถึง (max - 1)

Parameter:

`min` = เลขที่น้อยที่สุด

`max` = เลขที่มากที่สุด

Sensor & Module

Get your **Experience**

- ໂນດູລນັ້ນໃຊ້ກໍາວະໄສ Input ກັບ Output
- ມີກີ່ pin ຕ່ອຂາວະໄຮບ້າງ
- ຕ້ອງໃຊ້ Library ວະໄຮບ້າງ
- ຄິດໄມ້ອອກຄາມ Google

Assistant

Hardware subscriptions

THANK YOU

Gridsada Raksakun | kidsadoi