

ESP32 DAY2

Buzzer, Photoresistor, Joystick

Lab03. Varying brightness

1. 50% duty cycle brightness on the built LED

```
ledcAttach(LED_BUILTIN, 5000, 10);  
ledcWrite(LED_BUILTIN, 511);
```

2. Continuous fading of the builtin LED

```
void setup() {  
  ledcAttach(LED_BUILTIN, 5000, 10);  
  ledcWrite(LED_BUILTIN, 1022);  
}  
  
// the loop function runs over and over again forever  
void loop() {  
  ledcFade(LED_BUILTIN, 1022, 0, 1000);  
  delay(1000);  
  ledcFade(LED_BUILTIN, 0, 1022, 1000);  
  delay(1000);  
}
```

Review

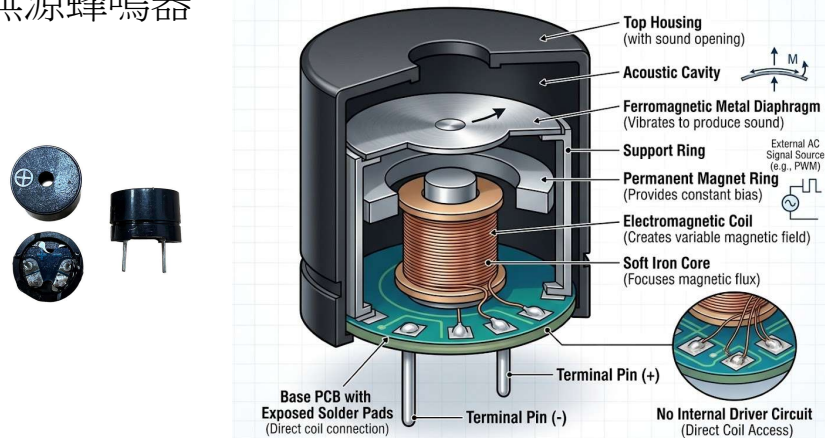
- Digital Signals
- Digital Output
- Digital Input
- Advanced Digital Output – PWM

Let's have some noise

無源蜂鳴器

INTERNAL STRUCTURE OF A MAGNETIC PASSIVE BUZZER

(For standard cylindrical through-hole type, like in IMAGE 0)



Lab04. Let's have make some noise!

- Super Mario Coin: Two quick notes—a rapid jump from B_5 (~988 Hz) to E_6 (~1319 Hz) with a slight delay (100ms, 400ms) between them.
- Laser Beam / Blaster: A fast frequency sweep downwards (e.g., looping from 2000 Hz down to 300 Hz in 2 ms steps).
- Emergency Siren: Alternating smoothly between two pitch extremes (e.g., 600 Hz and 1200 Hz using a sine wave function or rapid step changes).

- `pinMode(pin, OUTPUT);`
- `ledcAttach(13, 1000, 10);`
- `ledcWriteTone(pin, frequency)`
channel: it will generate the signal [0-15]
frequency: it will affect the type of sound produced
- `ledcWriteNote(pin, note_t, octave)`
note_t: scale

octave: it is a musical interval of eight notes [0-8]

NOTE_C	NOTE-Cs	NOTE_D	NOTE_Eb	NOTE_E	NOTE_F
NOTE_Fs	NOTE_G	NOTE_Gs	NOTE_A	NOTE_Bb	NOTE_B

<https://www.m5music.hk/dictionary/octave/><https://www.math.tu.edu.tw/front/News/news.php?ID=dGh1X21hdGmTmV3cw==&Sn=1107>

Lab05. Have some melody

Step	Note	Duration (ms)
1	NOTE_E	150
2	NOTE_E	300
3	NOTE_E	300
4	NOTE_C	150
5	NOTE_E	300
6	NOTE_G	500
7	NOTE_G	500

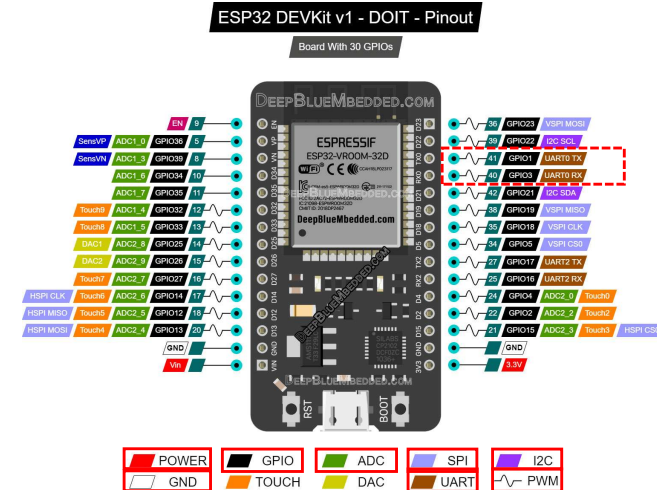
Step	Note	Duration (ms)
1	NOTE_G	500
2	NOTE_G	500
3	NOTE_G	500
4	NOTE_Eb	350
5	NOTE_Bb	150
6	NOTE_G	500
7	NOTE_Eb	350
8	NOTE_Bb	150
9	NOTE_G	600

Step	Note	Octave	Duration (ms)
1	NOTE_E	5	200
2	NOTE_Ds	5	200
3	NOTE_E	5	200
4	NOTE_Ds	5	200
5	NOTE_E	5	200
6	NOTE_B	4	200
7	NOTE_D	5	200
8	NOTE_C	5	200
9	NOTE_A	4	600

`ledcWriteNote(pin, note_t, octave)`

Sending messages from Arduino/ESP32 to our serial monitor

So it will be easier for you to debug



USB-Serial Bridge

- Many Arduino like **development boards** have a USB-serial bridge that converts between serial messages (later) and USB packets.
 - That is why we needed to install CP210x USB-UART driver package.
- They become virtual com ports on PCs.
- Popular ones
 - FTDI
 - Silicon Labs CP210x
 - Prolific
 - Nanjing Qinheng Microelectronics (WCH) CH340



https://en.wikipedia.org/wiki/Serial_port
<https://wiki.csie.ncku.edu.tw/embedded/USB>

Serial Functions (ESP32 → PC)

- `Serial.begin(115200);`
- `Serial.print(val)`
- `Serial.print(val, format)`
 - `print` → `println`
 - `val`: the value to print - any data type
 - For integer numbers, the format can be BIN, OCT, HEX, DEC (default)

<https://docs.arduino.cc/language-reference/en/functions/communication/serial/print/>

Lab07: Serial.print, Serial.println

- Output sin(x), cos(x) to the serial port.
- Use serial monitor to check your results
- Use serial plotter to show you results

Category	Method	Description
Setup & Config	begin(baud, config, rxPin, txPin, ...)	Initializes UART with custom baud, data bits, and remappable GPIO pins.
	end()	Disables the UART port and frees allocated GPIOs.
	setRxBufferSize(size) / setTxBufferSize(size)	Configures internal software buffer sizes (must be called before begin()).
	updateBaudRate(baud)	Modifies baud rate on the fly without restarting the peripheral.
Reading (Input)	available()	Returns the count of bytes waiting in the RX buffer.
	read()	Retrieves and removes the oldest byte in the buffer (returns -1 if empty).
	peek()	Previews the next incoming byte without clearing it from the queue.
	readBytes(buffer, length)	Reads binary data into an array until length is reached or timeout expires.
	readString() / readStringUntil(char)	Captures data into a String until timeout or reaching a delimiter (e.g., \n).
Writing (Output)	print() / println()	Outputs data as human-readable ASCII text (println appends a newline).
	printf(format, ...)	Sends formatted strings using C-style format specifiers (%d, %s, etc.).
	write(buf, len)	Emits raw binary bytes directly to the line.
	flush()	Waits until all buffered outgoing TX data has been physically transmitted.
Parsing & Control	setTimeout(ms)	Defines timeout threshold for stream operations (readBytes, find, etc.).
	parseInt() / parseFloat()	Extracts the next valid numerical value from the serial stream.
	find(target)	Scans the stream for a target string/character, returning true on match.

Serial Functions for Input (PC→ESP32)

- **Serial.available()**
 - Get the number of bytes (characters) available for reading from the serial port.

[Returns The number of bytes available to read.]
- **Serial.read()**
 - Reads incoming serial data.

[Returns The first byte of incoming serial data available (or -1 if no data is available).
Data type: int.]

```
while (Serial.available() > 0) {
  data= Serial.read();
  delay(2);
}
```

Use AI to summarize important functions

- help me summarize esp32 serial functions into a table.
 - <https://share.gemini.google/wQl4gJrgQseV>
 - <https://copilot.microsoft.com/shares/BTniagyu6ck1dLHi6UXtj>
 - <https://claude.ai/share/db9e4e91-c7e7-47b1-baae-a0cefa6a27c4>
 - <https://chatgpt.com/share/6a952045-d31c-83ee-a896-e29f5a6448b1>
 - https://grok.com/share/bGVnYWN5_166d7fcb-48ac-460e-9385-e38e15fcb057
 - <https://www.perplexity.ai/computer/tasks/12de561f-9aa9-46d5-8bd6-a7ad850e8e6c?view=thread>

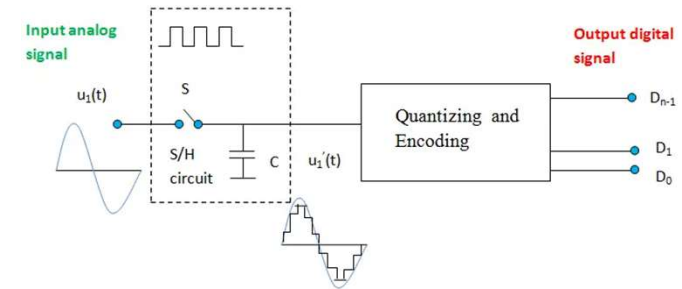
Lab08. Reading from Serial on ESP32

Enter digits (0-9) in the serial monitor on your computer, and display the digit in your 4-digit 7-segment display:

Input Display

1	__1
2	__12
3	__123
4	1234
5	2345
6	3456
7	4567

ADC process



<https://www.electrical4u.com/analog-to-digital-converter/>

ADC: Analog-Digital Convertor

Quantizing/Quantization and Encoding

- Resolution:
$$\Delta V = \frac{V_{ref}}{2^N - 1}$$
- V_{ref} : reference voltage, N: ADC resolution or width
- Quantizing or quantization is the process of dividing the reference signal into several discrete levels, or quanta, and then matching the input signal to the correct level.
- Encoding assigns a unique digital code to each discrete level (quantum) of the input signal. The process of quantizing and encoding is demonstrated in the table on the RHS.
- The figure on the RHS has ADC width of 3 bits and V_{ref} of 7V. The resolution of this ADC is 1V, and the quantization error is $\pm 0.5V$.

Analog signal			Digital o/p
7.5	7	$7\Delta = 7V$	111
6.5	6	$6\Delta = 6V$	110
5.5	5	$5\Delta = 5V$	101
4.5	4	$4\Delta = 4V$	100
3.5	3	$3\Delta = 3V$	011
2.5	2	$2\Delta = 2V$	010
1.5	1	$1\Delta = 1V$	001
0.5	0	$0\Delta = 0V$	000

<https://www.electrical4u.com/analog-to-digital-converter/>

ADC in ESP32, single-shot mode

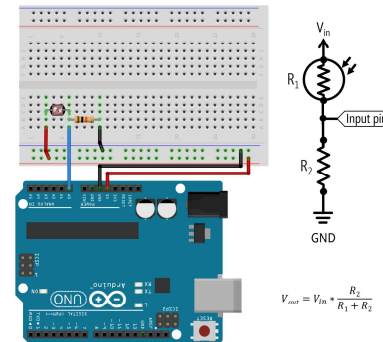
```
uint16_t analogRead(uint8_t pin);
uint32_t analogReadMilliVolts(uint8_t pin);
void analogReadResolution(uint8_t bits);
void analogSetAttenuation(adc_attenuation_t attenuation);
```

```
typedef enum {
    ADC_0db, // 100mV - 950mV
    ADC_2_5db, // 100mV - 1250mV
    ADC_6db, // 150mV - 1750mV
    ADC_11db, // 150mV - 3100mV
    ADC_ATTENDB_MAX
} adc_attenuation_t;
```

<https://docs.espressif.com/projects/arduino-esp32/en/latest/api/adc.html>

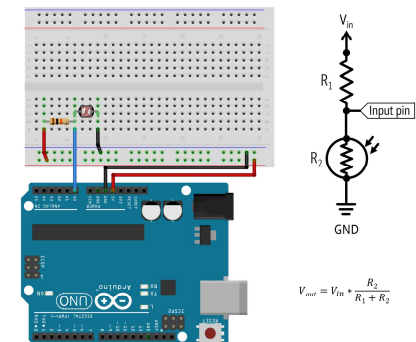
“LIGHTNESS” SENSOR CONFIGURATION

V_{out} increases as light levels on the photoresistor increase



“DARKNESS” SENSOR CONFIGURATION

V_{out} increases as light levels on the photoresistor decrease



<https://makeabilitylab.github.io/physcomp/sensors/photoresistors.html>

光敏電阻

- Photoresistor or Light Dependent Resistor(LDR)
- 為光線偵測元件
 - 光線越強，內部電阻越小。
 - 光線越弱，內部電阻越大。



<https://makeabilitylab.github.io/physcomp/sensors/photoresistors.html>

<https://physcourse.thu.edu.tw/wp-content/uploads/sites/8/2018/09/%E5%85%89%E6%95%8F%E9%9B%BB%E9%98%BB1025.pdf>

Lab 09 – ADC with Photoresistor

- Add Photoresistor in serial with resistor to make it into a “lightness” sensor. See results in the serial plotter.
- Change your configuration so that it becomes a “darkness” sensor. See results in the serial plotter.
- Create a night-light using the LED_BUILTIN, with its brightness varies with the lighting condition. The LED is off when the environment is bright; when LED becomes brighter and brighter (PWM controlled) when the environment becomes darker and darker.

可變電阻 Potentiometer

可變電阻可經由滑動(轉動)而改變滑動端與兩個固定端間電阻值的電子零件。可變電阻沒有分正負極，中間接上類比訊號Pin即可。

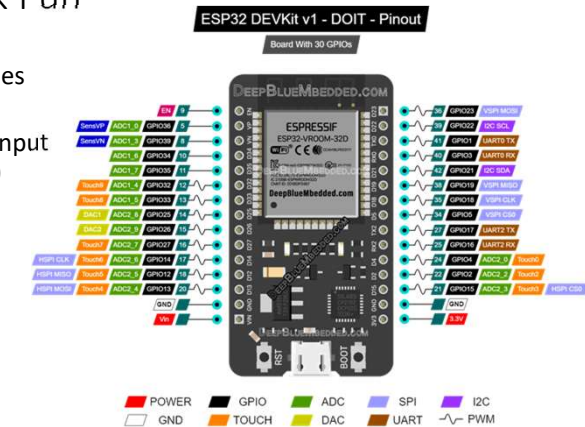


Potentiometer symbol (IEC standard)

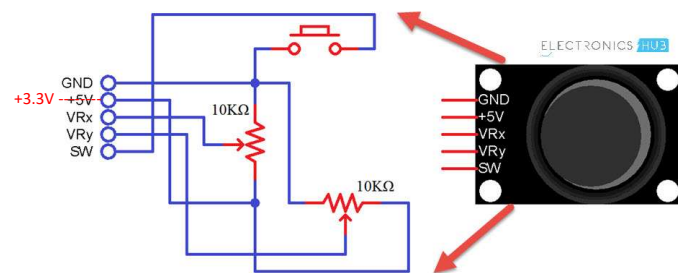
Potentiometer symbol (ANSI standard)

Lab 10. Joystick Fun

- Printout the ADC values from VRx and VRy
- Also print the digital input from SW (LOW, HIGH)



Joystick



<https://arduinogetstarted.com/tutorials/arduino-joystick>
<https://www.electronicshub.org/arduino-joystick-interface-control-servo/>

Homework

- Use joystick to:
 - Left-Right: switch melody to the previous / next
 - Up-Down: control speed of play
 - Switch: mute/resume