Este documento contiene distintos tutoriales y conceptos teóricos sobre los diferentes sensores, actuadores y otros componentes electrónicos que se pueden usar en la plataforma ESP32

Apuntes de componentes embebidos usando el ESP32

Sistemas Operativos Avanzados
Ing. Esteban A. Carnuccio

Contenido

1. Pinout 3

2. Conexión de sensores y actuadores al ESP32 4

a. Pulsadores: 4

b. LEDS 4

c. Reles con Brekout 4

d. Sensor PIR 4

e. Potenciómetro 4

f. Sensor de distancia (HC-Sr04) 4

g. SERVOMOTORES (funciona con SG90 y MG996R) 4

h. PWM con LED SIMPLE 5

- **USO DE LEDCWRITE** 5
- **USO DE LA BIBLIOTECA ANALOGWRITE PARA ESP32** 6

i. PWM con LED RGB con breakout 6

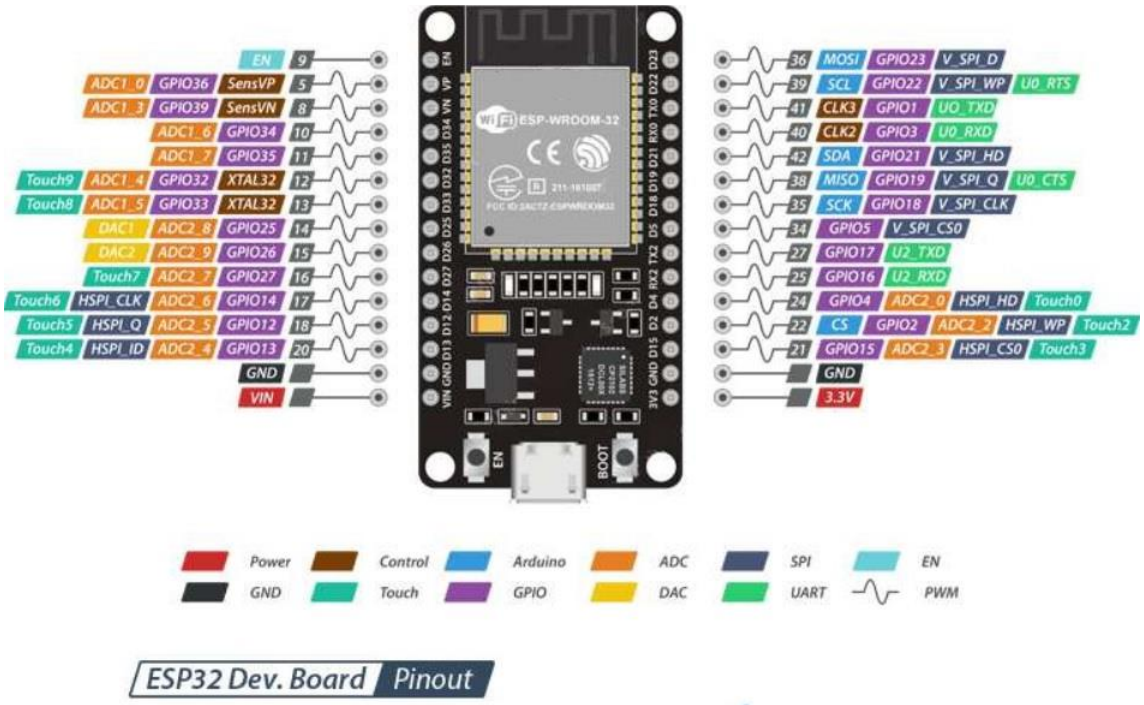
j. PWM con Buzzer (función Tone) 6

k. LDR (No se pudo probar) 7

l. Pagina con varios tutoriales y conexión de distintas placas con el mismo sensor 7

Conexiones ESP32

1. Pinout



CARACTERISTICAS

- https://programarfacil.com/esp8266/esp32/ (Tipos de modelos del ESP32 y sus características)
- https://www.prometec.net/instalando-esp32/
- Se alimenta a través de cable USB con 5V, pero internamente existe un regulador que lo trabaja a 2.55 a 3.6V.
- Se puede alimentar a través del Pin VIN. Pero el valor del voltaje necesario de alimentación de ese pin depende mucho del regulador, generalmente posee el AMS1117
- **Los Pines trabajan a 3.3v. Ninguno trabaja a 5V**

Link donde se explican cada uno de los PINOUT

- https://descubrearduino.com/esp32-modulo-esp32-wroom-gpio-pinout/
- https://circuits4you.com/2018/12/31/esp32-devkit-esp32-wroom-gpio-pinout/
- https://components101.com/microcontrollers/esp32-devkitc

FORMA DE PROGRAMAR CON EL IDE ARDUINO

- https://programarfacil.com/esp8266/programar-esp32-ide-arduino/

ESP32 ALIMENTADO POR BATERIAS.

- https://www.radioshuttle.de/es/medias-es/informaciones-tecnicas/esp32-alimentado-por-bateria/

2. Conexión de sensores y actuadores al ESP32

ANTES QUE NADA, CONVIENE VER EL DATASHEET DE LOS SENSORES PARA CONSTATAR CUAL ES EL VOLTAJE MINIMO DE FUNCIONAMIENTO Y EN BASE A ESO ALIMENTARLOS. LO SIGUIENTE ES UNICAMENTE A MODO DE GUIA.CONVIENE CONSTATARLO CON EL DATASHEET DE CADA SENSOR Y ACTUADOR

a. Pulsadores:

CARACTERISTICAS

- Se alimentan a 3.3V
- Para no usar una resistencia se pueden usar los pines señalados como INPUT_PULLUP. Pero se debe indicar que es INPUT _PULLUP E

b. LEDS

CARACTERISTICAS

- El código fuente es igual al Arduino UNO.
- Se alimentan a 3.3V
- Usar la calculadora de resistencias para obtener su valor.(Tener en cuenta los 3,3,v de entrada)

c. Reles con Brekout

CARACTERISTICAS

- El código fuente es igual al Arduino UNO.
- VCC debe conectarse a 5V

d. Sensor PIR

CARACTERISTICAS

- El código fuente es igual al Arduino UNO.
- VCC debe conectarse a 5V

e. Potenciómetro

CARACTERISTICAS

- El código fuente es igual al Arduino UNO.
- Tener en cuenta que en ESP32 el analógico va desde 0 hasta 4096. En cambio en Arduino UNO va desde 0 a 1024
- VCC debe conectarse a 3.3V

f. Sensor de distancia (HC-Sr04)

CARACTERISTICAS

- El código fuente es igual al Arduino UNO.
- VCC debe conectarse a 5V

g. SERVOMOTORES (funciona con SG90 y MG996R)

CARACTERISTICAS

- Usando las siguientes bibliotecas, funciona igual que con Arduino UNO.

- Estas bibliotecas tienen las mismas funciones que la bibliotecas Servo.h
- VCC debe conectarse a 5V. Depende mucho de los valores de voltaje del Servo.

BIBLIOTECA SERVO.H PARA ESP32

- https://randomnerdtutorials.com/esp32-servo-motor-web-server-arduino-ide/ (Tutorial donde usar una biblioteca para ESP32 llamada Servo.h)
- https://github.com/RoboticsBrno/ServoESP32 (Link a la biblioteca Servo.h para ESP32)

BIBLIOTECA SERVO.H PARA ESP32Servo.h

- Se debe instalar desde el administrador de librerías del ESP32. Se busca con el nombre ESP32Servo
- Una vez descargada fijarse en el Arduino Ide los ejemplos de esta biblioteca.
- https://www.arduino.cc/reference/en/libraries/esp32servo/

h. PWM con LED SIMPLE

CARACTERISTICAS

- En ESP32 no se usa Analogwrite por defecto, sino que se usa la función ledcWrite() en su lugar
- Existe una biblioteca llamada AnalogWrite.h, que utiliza las mismas instrucciones que en Arduino UNO. Ya que es wrapping de ledcWrite()

- **USO DE LEDCWRITE**

PASOS PARA SU CONFIGURACION

Tutorial2

```
#define LEDC_CHANNEL_0     0    // use first channel of 16 channels (started from zero)
#define LEDC_TIMER_8_BIT   8    // use 8 bit precission for LEDC timer
#define LEDC_BASE_FREQ     5000 // use 5000 Hz as a LEDC base frequency
#define LED_PIN            5    // // Led Verde

int brillo = 0;
int sensorT0 = 0;
int sensorT3 = 0;

void setup() {
  Serial.begin(115200);

  ledcSetup(LEDC_CHANNEL_0, LEDC_BASE_FREQ, LEDC_TIMER_8_BIT); // Setup timer and attach timer to a led pin
  ledcAttachPin(LED_PIN, LEDC_CHANNEL_0);

  pinMode(23, INPUT_PULLUP); // Pulsador
  pinMode(18, OUTPUT);       // Led Rojo
}

void loop() {

  sensorT0 = touchRead(T0);
  sensorT3 = touchRead(T3);

  if (sensorT0 < 35 and sensorT0 != 0) {
    brillo = brillo + 5;
    if (brillo >= 255) {
      brillo = 255;
    }
```

```
Serial.println("T3=" + String(sensorT3) + " T0=" + String(sensorT0) + " Brillo=" + String(brillo));

ledcWrite(LEDC_CHANNEL_0, brillo);
```

LEDChannel :va de 0 a 16 canales PWM que tiene el ESP32
LEDCTIMER_8BIT : es la prescision del timer hasta 16 bits. Conviene ponerlo en 8 para que cuente valores hasta 255 (como en Arduino UNO)
LEDCBASE_Freq frecuencia de base
LED_PIN es el GPIO.

TUTORIALES

- https://techexplorations.com/guides/esp32/begin/pwm/ (tutorial donde hace una anología entre AnalogWrite de Arduino y ledcWrite de ESP32.
- https://www.youtube.com/watch?v=bYTqqBU6M7Q&t=567s
- https://programmerclick.com/article/593373513/

- **USO DE LA BIBLIOTECA ANALOGWRITE PARA ESP32**

CARACTERISTICAS

- Se debe instalar desde el administrador de librerías del ESP32. Se busca con el nombre ESP32AnalogWrite
- Tener cuidado que solo sirve para 1 solo usar un solo actuador de PWM. Ya que las funciones de configuración del PWM de su código fuente, confiuguran todos los canales juntos dentro de un for. Por lo que si se quiere usar más de un PWM, se debe modificar el código de la biblioteca para que funcione.

TUTORIALES

- https://github.com/ERROPiX/ESP32_AnalogWrite (código fuente de la biblioteca con ejemplos)

i. PWM con LED RGB con breakout

CARACTERISTICAS

- Funciona con una resistencia de 200 ohm en pin rojo
- Se debe usar el GND de 3,3V
- https://datasheetspdf.com/pdf-file/1402027/Joy-IT/KY-016/1 (Datasheet RGB Breakout)

j. PWM con Buzzer (función Tone)

CARACTERISTICAS

- El buzzer se debe conectar a 5V
- ESP32 no tiene implementado por defecto la función tone. Entonces hay que descargar la biblioteca correspondiente para poder utilizarla.
- Esta biblioteca es la que se llama ESP32Servo.h . Además de poder manejar el servo con esta biblioteca, permite manejar la función como en Arduino.

- Se debe instalar desde el administrador de librerías del ESP32. Se busca con el nombre ESP32Servo
- Una vez descargada fijarse en el Arduino Ide los ejemplos de esta biblioteca.

  https://www.arduino.cc/reference/en/libraries/esp32servo/

TUTORIALES

- https://github.com/madhephaestus/ESP32Servo/blob/master/examples/ToneExample/ToneExample.ino (Ejemplo de uso)

k. LDR (No se pudo probar)

CARACTERISTICAS

- Debe ser utilizado junto a una resistencia de 10K(esto según datasheet)
- El breakout ya la incorpora la resistencia de 10k con el smd 103
- Puede ser conectado a 3,3v.

l. Pagina con varios tutoriales y conexión de distintas placas con el mismo sensor

- https://elosciloscopio.com/tutorial-sensores-ultrasonicos-arduino-esp8266-esp32/
- https://elosciloscopio.com/tutorial-pantalla-led-segmento-arduino-esp8266-esp32/
- https://elosciloscopio.com/tutorial-sensor-lluvia-arduino-esp8266-esp32/
- https://elosciloscopio.com/tutorial-led-arduino-esp8266-esp32/
- https://deepbluembedded.com/esp32-lcd-display-16x2-without-i2c-arduino/
- https://esp32io.com/tutorials/esp32-light-sensor