

**MODULO PRINCIPAL ESP32** 



# Guía de usuario



O ODAVINCI.TECH

### **DESCRIPCIÓN GENERAL**

El módulo principal es el que contiene el micro controlador, en este caso el ESP32, se encargara del procesamiento, comunicación con los módulos accesorios y cuenta con conectividad WiFi y bluetooth y 2 núcleos de 32bits. Sus pines llamados GPIOs en esta placa se distribuyen en los 6 lados del hexágono, de este modo cada módulo accesorio se conectará del lado o los lados cuyos GPIOs sean compatibles con su función. También tiene un sensor de temperatura, sensor de nivel de luz y led RGB, los cuales te servirán para ensayar tus primeros proyectos de loT; estos son fáciles de usar ya que en la misma PCB se indica los GPIOs en los que estos están conectados, sin embargo, en este documento encontraras todos los detalles.





PINOUT

DAC ADC ALIMENTACIÓN 12C JUART PROPOSITO GENERAL SPI RESET





#### **INSTALACION**

- Instalar el IDE Arduino 1.8.16 o superior https://downloads.arduino.cc/arduino-1.8.19-windows.exe
- En el IDE de Arduino ir a Archivo -> preferencias -> Gestor de URLs adicionales de Tarjetas
- Pegar el siguiente enlace:

•

https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package\_esp32\_index.json

💿 sketch_sep07a Arduino 1.8.16	- 🗆 X		
Arcie o Editar Programa Herramientas Ayuda			
Nutvo Ctrl+N	Q		
Abrir Ctrl+O			
Abrir Reciente >			
Proyecto >	Preferencias	×	
Ejemplos >	Ajustes Red		
Cerrar Ctrl+W	Lastinación de prevente		
Salvar Ctrl+S			
Guardar Como Ctrl+Mayús+S	C: \Users \Swings Computers \Documents \Arduino	Explorar	
Configurar Página Ctrl+Mayús+P	Editor de idioma: System Default v (requiere reiniciar Arduino)		
Imprimir 🔶 Ctrl+P	Editor de Tamaño de Fuente: 21		
Preferencias Ctrl+Coma	Escala Interfaz: 🗸 Automático 100 ≑ % (requiere reiniciar Arduino)		
Salir Ctrl+Q	Tema: Tema por defecto 🗸 (requiere reiniciar Arduino)		
91	Mostrar salida detallada mientras; 🗌 Compilación 🔲 Subir		
- 1	Advertencias del compilador: Naguno 🗸		
	Mostrar números de línea 🗌 Habilitar Plegado Código		
	✓ Verificar código después de subir □ Usar editor externo		
	Comprobar actualizaciones al iniciar		
	Use accessibility features		
	Gestor de URLs Adicionales de Tarjetas https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json	C	
Ildd, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS),	2 Más preferencias pueden ser editadas directamente en el fichero		
	C:\Users\Swings Computers\AppData\Local\Arduino15\preferences.txt		
(editar sólo cuando Arduino no está corriendo)			
	Ok	Cancelar	

#### **INSTALAMOS LA PLACA ESP32**

- Ir a Herramientas -> Placa: -> Gestor de tarjetas...
- En la barra de busqueda escribir: ESP32
- Por ultimo click en instalar.



## CONFIGURAMOS EL IDE PARA PROGRAMAR EL ESP32

## • Ir a Herramientas -> Placa: -> ESP32 Arduino -> ESP32 Dev Module

Archivo Editar Programa	Herramientas Ayuda			
	Auto Formato Archivo de programa.	Ctrl+T		2
sketch_oct12a 1 void set 2 // put	Reparar codificación & Recargar. Administrar Bibliotecas Monitor Serie Serial Plotter Blynk: Check for updates	Ctrl+Mayús+I Ctrl+Mayús+M Ctrl+Mayús+L		
3 4 } 5	Blynk: Example Builder Blynk: Run USB script WiFi101 / WiFiteINA Firmware Updater Placa: "ESP32 Dev Module"	k	Gestor de tarietas	
6 void loo	Upload Speed: "921600" CPLI Frequency: "240MHz (WiEi/RT)"		Arduino AVR Boards >	Δ
8 / // put	Flash Frequency: "80MHz" Flash Mode: "010"	L	ESF32 AIUUIIO	ESP32C3 Dev Module ESP32S2 Dev Module
9}	Flash Size: "4MB (32Mb)" Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)' Core Debug Level: "Ninguno" PSRAM: "Disabled" Arduino Runs On: "Core 1" Events Run On: "Core 1" Puerto: "COM3"	> > > > >		ESP32 Dev Module     ESP32 Wrover Module     ESP32 PICO-D4     ESP3252 Native USB     ESP32 Wrover Kit (all versions)     UM TinyPICO     UM FeatherS2     UM FeatherS2
	Programador Quemar Bootloader	>		UM TinyS2 S.ODI Ultra v1 microS2 MagicBit

## YA CASI LISTO...

## Asegúrate de que las casillas en el recuadro las tengas igual que en la siguiente imagen...

	Herramientas Ayuda				
	Auto Formato	Ctrl+T			
	Archivo de programa.				
	Reparar codificación & Recargar.				
	Administrar Bibliotecas	Ctrl+Mayús+I			
	Monitor Serie	Ctrl+Mayús+M			
	Serial Plotter	Ctrl+Mayús+L			
	Blynk: Check for updates				
	Blynk: Example Builder				
	Blynk: Run USB script				
	WiFi101 / WiFiNINA Firmware Updater				
	Placa: "ESP32 Dev Module"	>			
	Upload Speed: "921600"	>			
	CPU Frequency: "240MHz (WiFi/BT)"	>			
Esta devandevá del evente	Flash Frequency: "80MHz"	>			
Esto dependeral del puerto	Flash Mode: "QIO"	>			
USB de tu PC al que hayas	Flash Size: "4MB (32Mb)"				
conectado la placa	Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)	" >			
	Core Debug Level: "Ninguno"	>			
	PSRAM: "Disabled"	>			
	Arduino Runs On: "Core 1"	>			
	Events Run On: "Core 1"	>			
	Puerto: "COM3"	>			
	Obtén información de la placa				
	Programador	>			
	Quemar Bootloader				



## **TU PRIMER PROGRAMA**

Suponiendo que solo tienes el modulo principal DA VINCI ORIGAMI ESP32 y todo lo configuraste bien, copia el siguiente código en el IDE de Arduino. Este código alternara los leds rojo, verde y azul y mostrara los datos de nivel de luz y temperatura en el monitor serial.

//Definimos los pines del sensor de luz, de temperatura y del led RGB #define luzPin 36 #define tempPin 39 #define R 25 #define G 33 #define B 26

//creamos variables para luz y temperatura float luz = 0; float temp = 0;

void setup(){ //iniciamos la comunicacion serial a 9600 baudios Serial.begin(9600); //configuramos los pines del led RGB como salida pinMode(R, OUTPUT); pinMode(G, OUTPUT); pinMode(B, OUTPUT);

}
void loop() {

//leemos el pin del sensor de temperatura temp = analogRead(tempPin); //convertimos el dato a grados centigrados temp = (temp\*125.0)/4096.0;

//este ciclo se cumplira 200 veces
for (int i = 0; i<200; i++){
 //sumamos 200 veces el valor del pin del sensor de luz
 luz = luz + analogRead(luzPin);
}</pre>

// dividimos el valor del nivel de luz entre 200 (promediamos) luz = luz/200.0;

//el valor crudo es de 0 a 4096, lo convertimos de 0 a 100 luz = (luz\*100.0)/4096.0;

//imprimimos los valores de luz y temperatura
Serial.print("LUZ= ");
//imprimimos el valor de la luz y lo configuramos a 1 decimal
Serial.print(luz,1);
Serial.print(" TEMP= ");
//imprimimos el valor de la temperatura y por defecto tendra 2 decimales
Serial.println(temp);

//encendemos el led Rojo y apagamos los demas digitalWrite(R, HIGH); digitalWrite(G, LOW); digitalWrite(B, LOW); delay(100); //esperamos 100 milisegundos

//encendemos el led Verde y apagamos los demas digitalWrite(R, LOW); digitalWrite(G, HIGH); digitalWrite(B, LOW);

delay(100); //esperamos 100 milisegundos //encendemos el led Azul y apagamos los demas digitalWrite(R, LOW); digitalWrite(G, LOW); digitalWrite(B, HIGH);

delay(100); //esperamos 100 milisegundos
}

\*FIN DEL PROGRAMA\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

