

Migración de proyectos en C para Raspberry PI4 a ESP32-C3 RUST-V1

Programación de Nodos y Sensores para IoT

9 de enero de 2026

María del Cos
Isaac Pascual

ÍNDICE

01 Introducción

05 Práctica 3

02 Adaptación de las
Prácticas

06 Practica 4

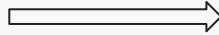
03 Practica 1

04 Práctica 2

01

Introducción

ESP32-C3: Migración



Comparativa

Raspberry Pi

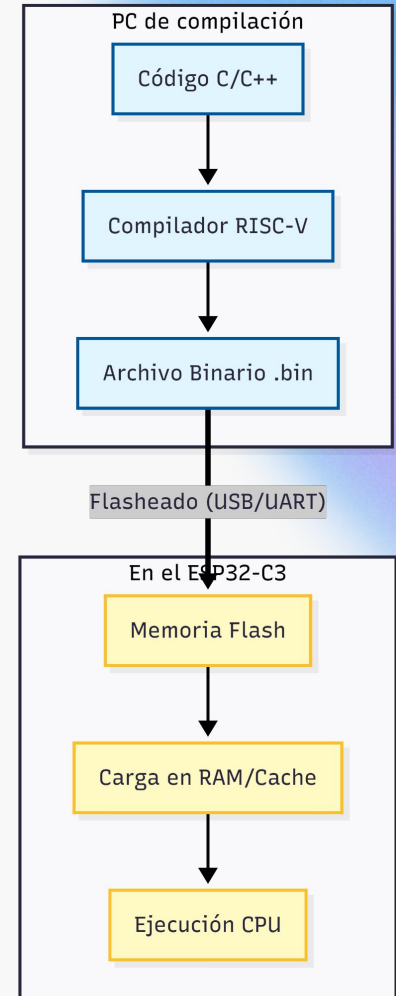
- Mini-PC con Linux
- Alto consumo (2,7 - 3,5 W)
- Ideal para multimedia y servidores

ESP32-C3

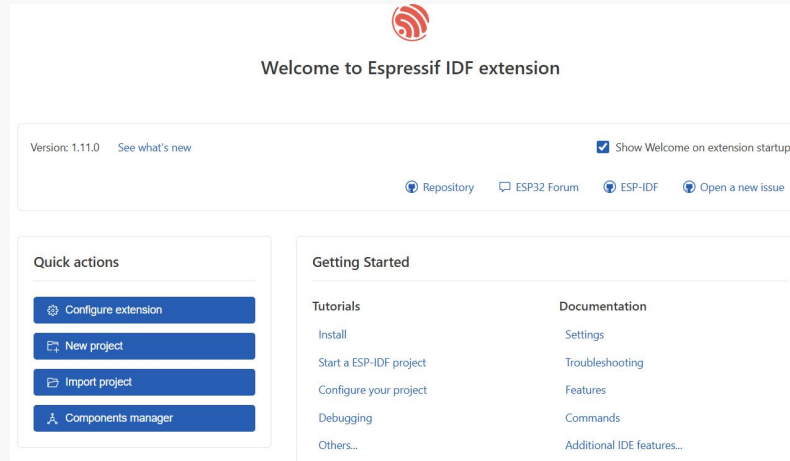
- Microcontrolador
- Bajo consumo (0,07 - 0,36 W)
- Ideal para IoT y control en tiempo real

Arquitectura

- Usamos un procesador **RISC-V de un solo núcleo**. Esto significa que físicamente solo puede hacer una cosa a la vez.
- Es un sistema **compilado**.
- Al no tener un Sistema Operativo pesado debajo, nuestro código tiene el control total del hardware (**Bare-metal**), apoyándose en **FreeRTOS** para simular la multitarea.

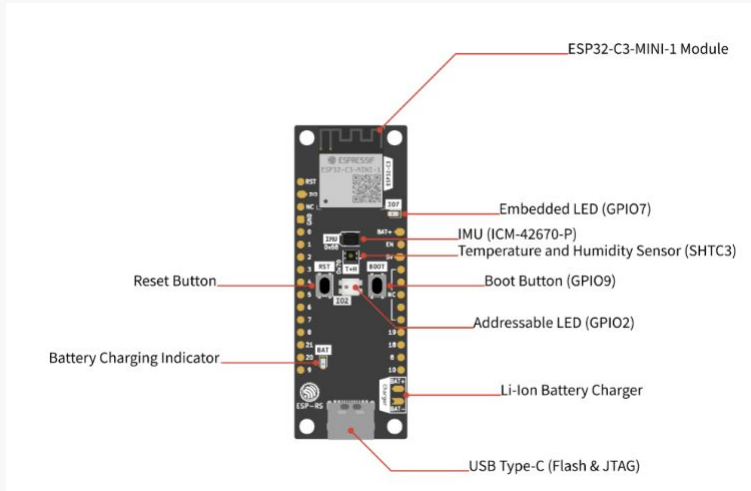


Entorno de desarrollo

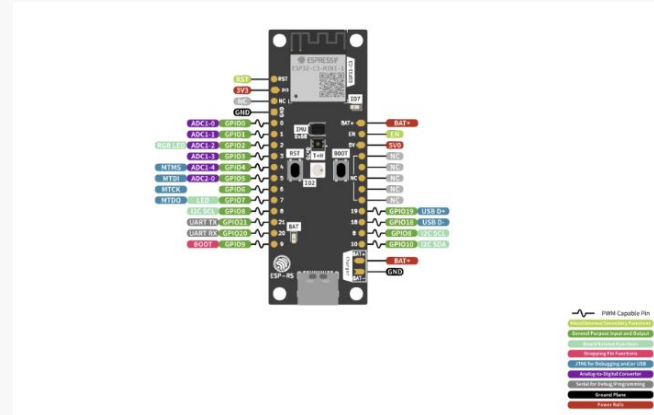


02

Adaptación de las
prácticas



- 1 solo core
- Periféricos internos
- No dispone de DAC interno



03

Práctica 1

Piano Digital

Temporizador → PWM → Altavoz

```
#define GPIO_BASE      0x60004000
#define GPIO_OUT_W1TS  (GPIO_BASE + 0x08)
#define GPIO_OUT_W1TC  (GPIO_BASE + 0x0C)
#define GPIO_ENABLE_W1TS (GPIO_BASE + 0x20)
```

**Modificación de registros GPIO
para generar PWM**

```
for (int i = 0; i < repeticiones; i++) {
    // Encender pin
    *gpio_out_w1ts = (1 << ZUMBADOR);
    ets_delay_us(mitad_periodo);

    // Apagar pin
    *gpio_out_w1tc = (1 << ZUMBADOR);
    ets_delay_us(mitad_periodo);
}
```

Table 3.3-3. Module/Peripheral Address Mapping

Target	Boundary Address		Size (KB)	Notes
	Low Address	High Address		
UART Controller 0	0x6000_0000	0x6000_0FFF	4	
Reserved	0x6000_1000	0x6000_1FFF		
SPI Controller 1	0x6000_2000	0x6000_2FFF	4	
SPI Controller 0	0x6000_3000	0x6000_3FFF	4	
GPIO	0x6000_4000	0x6000_4FFF	4	

Modificación de registros GPIO para generar PWM

Register 5.3. GPIO_OUT_WITS_REG (0x0008)



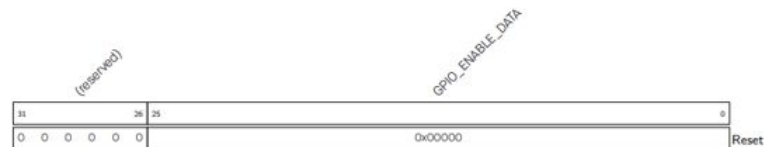
GPIO_OUT_WITS GPIO0 ~ 21 output set register. Bit0 ~ bit21 are corresponding to GPIO0 ~ 21, and bit22 ~ bit25 are invalid. If the value 1 is written to a bit here, the corresponding bit in **GPIO_OUT_REG** will be set to 1. Recommended operation: use this register to set **GPIO_OUT_REG**. (WT)

Register 5.4. GPIO_OUT_WITC_REG (0x000C)



GPIO_OUT_WITC GPIO0 ~ 21 output clear register. Bit0 ~ bit21 are corresponding to GPIO0 ~ 21, and bit22 ~ bit25 are invalid. If the value 1 is written to a bit here, the corresponding bit in **GPIO_OUT_REG** will be cleared. Recommended operation: use this register to clear **GPIO_OUT_REG**. (WT)

Register 5.5. GPIO_ENABLE_REG (0x0020)



GPIO_ENABLE_DATA GPIO output enable register for GPIO0 ~ 21. Bit0 ~ bit21 are corresponding to GPIO0 ~ 21, and bit22 ~ bit25 are invalid. (R/W/SS)

04

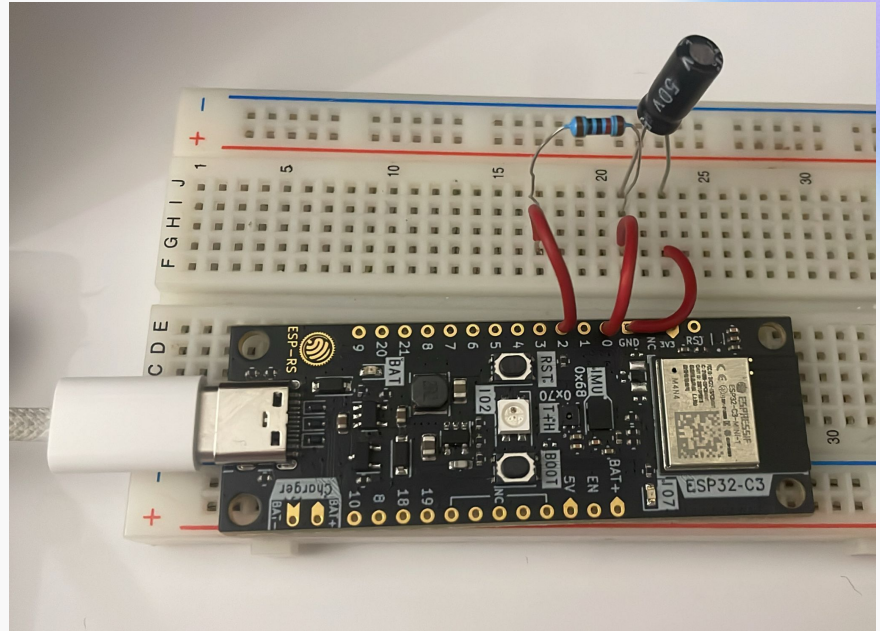
Práctica 2

Fotopletismografía

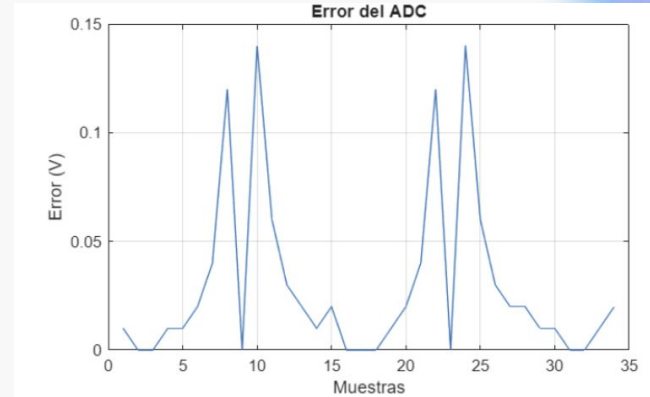
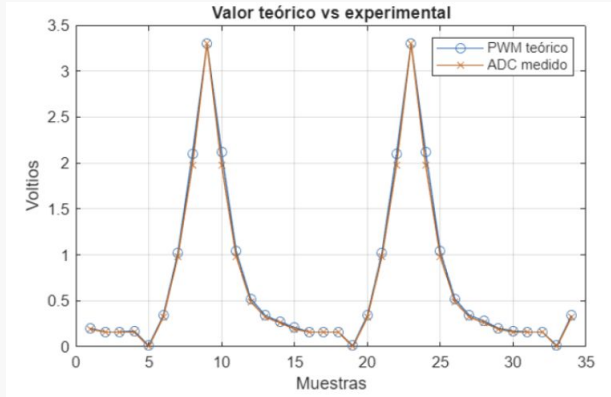
Carencia de DAC → Generación por PWM

```
void adc_init(void)
{
    adc_one_shot_unit_init_cfg_t init = {
        .unit_id = ADC_UNIT_1
    };
    adc_one_shot_new_unit(&init, &adc);

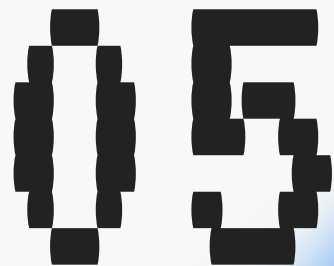
    adc_one_shot_chan_cfg_t cfg = {
        .atten = ADC_ATTEN_DB_11,
        .bitwidth = ADC_BITWIDTH_12
    };
    adc_one_shot_config_channel(adc, ADC_CH, &cfg);
}
```



Fotopletiismografía

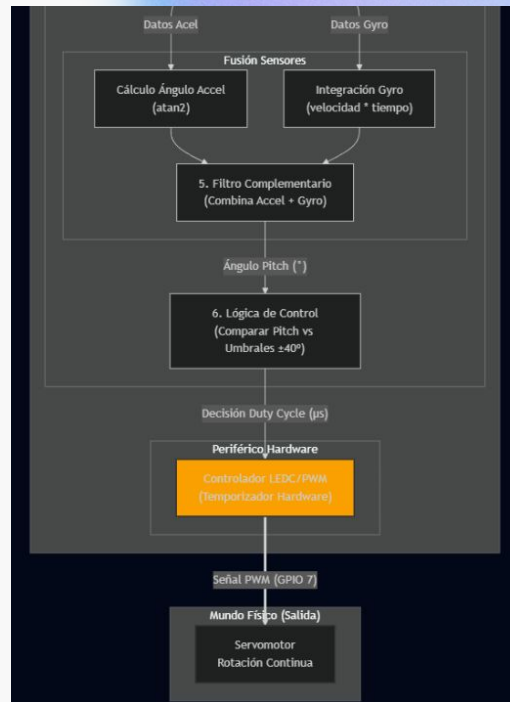
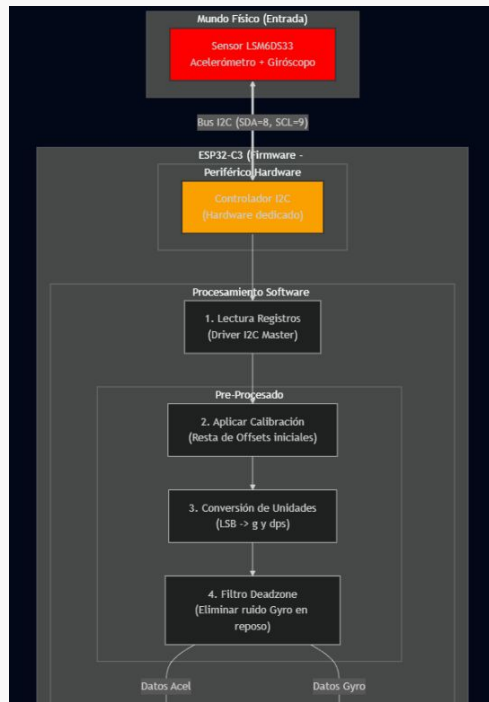
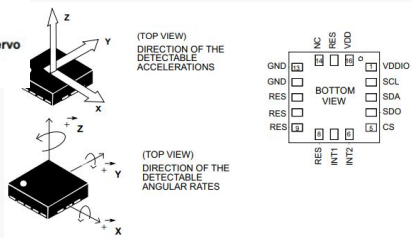
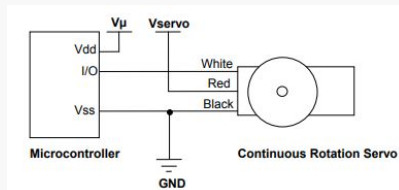
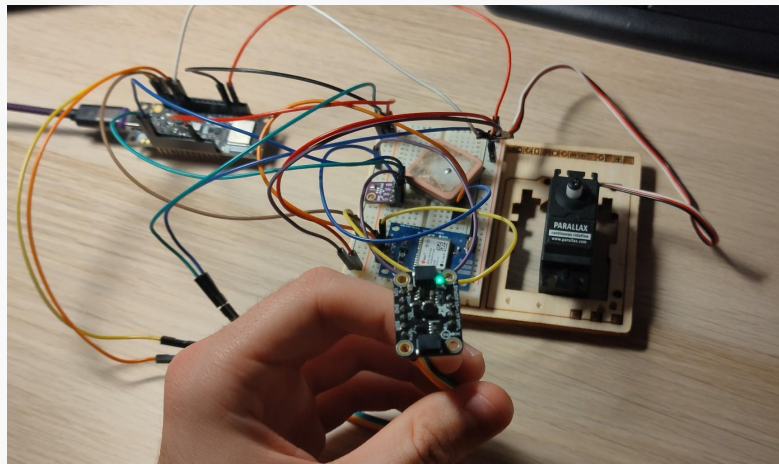


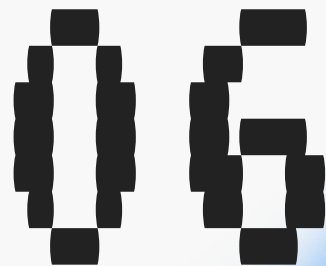
Parámetro	Valor (LSB)	Valor (V)
Offset error	21	16.9 mV
Gain error	0.5	0.40 mV
DNL máximo	31	25 mV



Práctica 3

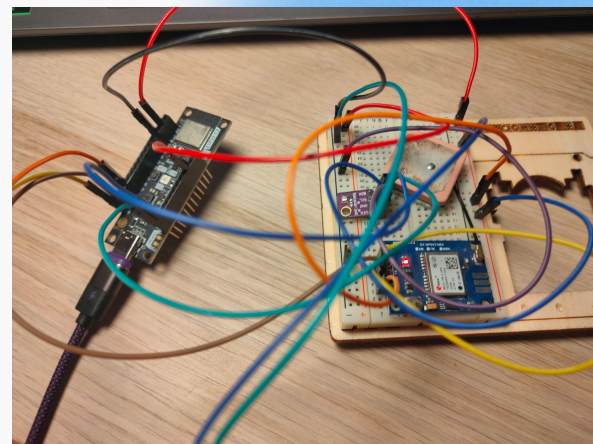
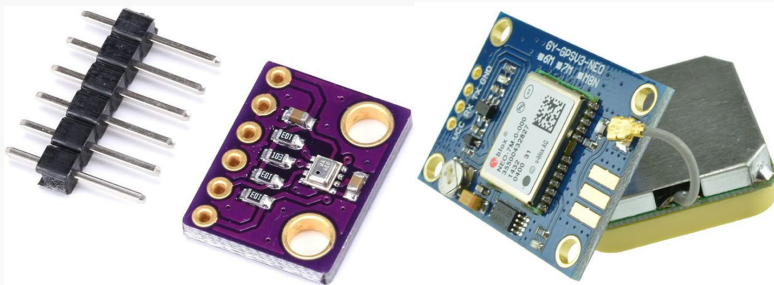
Sensores y Actuadores





Práctica 4

MQTT



```
xTaskCreate(bmp_task,      "bmp_task",      4096, NULL, 5, NULL);  
xTaskCreate(gps_uart_task, "gps_uart_task", 4096, NULL, 6, NULL);  
xTaskCreate(publisher_task, "publisher_task", 4096, NULL, 4, NULL);
```

