Microcontroladores: Pico

Introducción

¿Qué es un microcontrolador? en la Wikipedia podemos encontrar la siguiente definición:

> Un microcontrolador es un circuito integrado programable, capaz de ejecutar las órdenes grabadas en su memoria. Está compuesto de varios bloques funcionales que cumplen una tarea específica. Un microcontrolador incluye en su interior las tres principales unidades funcionales de una computadora: unidad central de procesamiento, memoria y periféricos de entrada/salida.

Y si la Raspberry Pi Pico es un microntrolador, ¿qué es la raspberry Pi? Esta pregunta la tenemos ya respondida en https://libros.catedu.es/link/8787#bkmrk-una-raspberry-pi-es-

La diferencia fundamental entre una Raspberry Pi 1/2/3/4/5 y una Raspberry Pi Pico 1/2 es que pertenecen a **categorías de dispositivos completamente distintas**: la primera es un ordenador completo (SBC), mientras que la segunda es un microcontrolador diseñado para tareas específicas de control electrónico:

- **Raspberry Pi** 4: Es una **computadora** de placa única (Single Board Computer). Ejecuta un sistema operativo completo (normalmente Linux/Raspberry Pi OS), tiene escritorio, navegador web y permite multitarea como un PC convencional.
- **Raspberry Pi Pico** 2 W: Es un **microcontrolador**. No tiene sistema operativo; en su lugar, ejecuta un único programa que tú le cargas (firmware). Es similar a un Arduino y está diseñada para interactuar directamente con sensores, motores y luces de forma eficiente y en tiempo real.

En la página web de Raspberry Pi podemos encontrar una sección para los **microcontroladores** de la familia. Concretamente, en la dirección https://www.raspberrypi.com/products/raspberry-pi-pico-2/ , podemos encontrar información sobre la Raspberry Pi Pico 2.

Características principales de la Raspberry Pi Pico 2

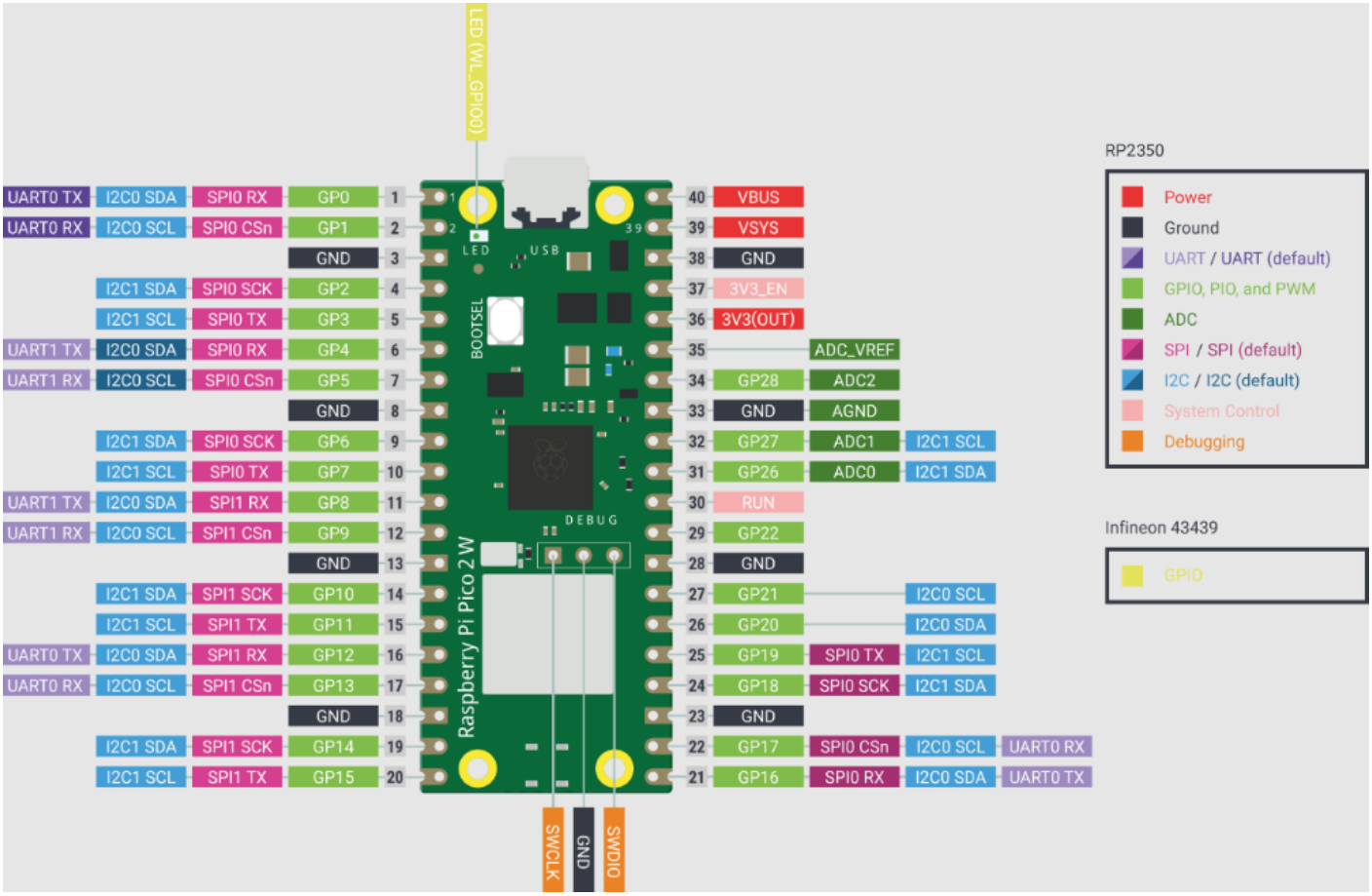
Todas las variantes Pico 2 comparten la misma base de hardware principal, porque todas usan el RP2350. Según Raspberry Pi, las características generales son Documentación oficial :

- Microcontrolador: RP2350
- CPU: elección entre:
  - 2 núcleos Arm Cortex-M33
  - 2 núcleos Hazard3 RISC-V
- Frecuencia: hasta 150 MHz
- SRAM: 520 KB
- Flash integrada: 4 MB
- GPIO: 26 pines multipropósito
- USB: USB 1.1 con soporte host y device
- PIO: 12 máquinas de estado PIO
- Periféricos:
  - 2 × UART
  - 2 × SPI
  - 2 × I2C
  - 3 × ADC de 12 bits
  - PWM
- Voltaje de entrada: 1.8 V a 5.5 V
- Temperatura operativa: -20 °C a +85 °C
- Compatibilidad: compatible en hardware y software con la generación Pico anterior

Además, el RP2350 añade funciones de seguridad importantes, como:

- Arm TrustZone
- arranque firmado
- almacenamiento OTP
- aceleración SHA-256
- protecciones frente a ataques por fallos

El pinout de la raspberry pi pico 2 w es el siguiente:



Versiones

Variante	Wi-Fi	Bluetooth	Headers presoldados
Raspberry Pi Pico 2	No	No	No
Raspberry Pi Pico 2 with headers	No	No	Si
Raspberry Pi Pico 2 W	Si	Si	No
Raspberry Pi Pico 2 W with headers	Si	Si	Si

Algunos proyectos

Con Arduino IDE versión 2.x podemos trabajar con la placa. Si usas Linux es probable que para que te detecte la placa tengas que ejecutar el comando `sudo usermod -a -G dialout $USER` para añadir al usuario al grupo dialout (deberás reiniciar la sesión). Además, con independencia del sistema

operativo que utilices deberás añadir a las placas disponibles la Raspberry Pi Pico. Para ello, dirígete a file > Preferences > Additional boards manager URLs y añade la url https://github.com/earlephilhower/arduino-pico/releases/download/global/package_rp2040_index.json Reinicia el IDE, conecta la microcontroladora y debería estar lista para funcionar.

Encender y apagar un led

La placa tiene integrado un led junto al conector USB. Para encenderlo y apagarlo alternativamente copia y pega el siguiente código:

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT); // Indicamos que el LED integrado en la placa funcione como salida
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);  // Enciende el LED
  delay(1000);                       // Espera 1 segundo
  digitalWrite(LED_BUILTIN, LOW);   // Apaga el LED
  delay(1000);                       // Espera 1 segundo
}
```

Ahora compila y envía a la microcontroladora. En cuanto el código esté en ella debería empezar a parpadear el led integrado en la placa

Activar un GPIO en concreto

Para determinados proyectos puede ser que nos interese encender y apagar un GPIO concreto. En el siguiente ejemplo vamos a encender y apagar cada 3 segundos el GPIO 0 (cero). Para ello copia y pega el siguiente código:

```
void setup() {
  pinMode(0, OUTPUT);      // Configura GPIO0 como salida
}

void loop() {
```

```
  digitalWrite(0, HIGH);   // Enciende GPIO0 (pone tensión en él)
  delay(3000);
  digitalWrite(0, LOW);
  delay(3000);
}
```

En el GPIO 0 alternarás cada 3 segundos 3,3V con 0 V de tensión. La corriente por defecto será de 4mA. En caso de querer variar la corriente debrás usar la función gpio_set_drive_strength tras haber importado la bibilioteca correspondiente:

```
#include "hardware/gpio.h"

void setup() {
  pinMode(0, OUTPUT);

  // Opciones: GPIO_DRIVE_STRENGTH_2MA, _4MA, _8MA, _12MA
  gpio_set_drive_strength(0, GPIO_DRIVE_STRENGTH_12MA);

  digitalWrite(0, HIGH);
}

void loop() {
  // tu código
}
```

Leer entrada digital GPIO

Puedes hacer uso de cualquier GPIO. El código es el siguiente:

```
void setup() {
  Serial.begin(115200);
  pinMode(1, INPUT);          // GPIO1 como entrada
  // Si no tienes resistencia externa, usa la pull-up interna:
  // pinMode(1, INPUT_PULLUP);
  // o pull-down:
  // pinMode(1, INPUT_PULLDOWN);
```

```
}

void loop() {
  int valor = digitalRead(1);  // Lee GPIO1: devuelve HIGH (1) o LOW (0)
  Serial.println(valor);
  delay(100);
}
```

Para ver la salida de Serial.println dirígete a arduinoIDE > tools > Serial Monitor

Leer entrada analógica GPIO

GPIO26, 27 y 28 son los únicos que funcionan de modo analógico. El código es el siguiente:

```
void setup() {
  Serial.begin(115200);
  // No hace falta pinMode para analogRead
}

void loop() {
  int valor = analogRead(26);        // 0–4095 (12 bits)
  float tension = valor * 3.3 / 4095.0;
  Serial.print("Tensión: ");
  Serial.println(tension);
  delay(100);
}
```

Pulsación de botón

```
void setup() {
  Serial.begin(115200);
  pinMode(1, INPUT_PULLUP);  // Botón entre GPIO1 y GND
}

void loop() {
```

```
  if (digitalRead(1) == LOW) {  // LOW = pulsado (con pull-up)
    Serial.println("Botón pulsado");
  }
  delay(50);
}
```

Sensor de movimiento

```
void setup() {
  pinMode(1, INPUT);   // Señal del PIR
  pinMode(0, OUTPUT);  // LED
}

void loop() {
  if (digitalRead(1) == HIGH) {  // PIR activa HIGH al detectar
    digitalWrite(0, HIGH);       // Enciende LED
  } else {
    digitalWrite(0, LOW);
  }
  delay(100);
}
```

Detección de agua

```
void setup() {
  Serial.begin(115200);
  pinMode(1, INPUT_PULLDOWN);  // Sensor de nivel
}

void loop() {
  if (digitalRead(1) == HIGH) {
    Serial.println("Nivel de agua alto");
  }
  delay(500);
}
```

Temperatura con termistor

Analógico. Has de usar los pines indicados anteriormente. En el ejemplo se usa GPIO26.

```
void loop() {
  int raw = analogRead(26);
  float tension = raw * 3.3 / 4095.0;
  // Con divisor de tensión 10kΩ + NTC 10kΩ:
  float resistencia = 10000.0 * tension / (3.3 - tension);
  // Fórmula Steinhart-Hart simplificada:
  float tempK = 1.0 / (log(resistencia / 10000.0) / 3950.0 + 1.0 / 298.15);
  float tempC = tempK - 273.15;
  Serial.print("Temperatura: ");
  Serial.print(tempC);
  Serial.println(" °C");
  delay(1000);
}
```

Otros

Y tu, ¿qué montajes has realizado?

Revision #4
Created 2026-03-27 17:16:32 CET by Pablo Ruiz
Updated 2026-03-28 09:33:50 CET by Pablo Ruiz