Pines de Entrada/salida

> Página extraída de Federico Coca Guia de Trabajo de Microbit CC-BY-SA

En MicroPython, cada pin en la BBC micro:bit está representado por un objeto llamado pinN, donde N es el número del pin.

Por ejemplo, para usar el pin etiquetado 0 (cero), puedes usar el objeto llamado pin0 en tu script. El pin del logo V2 utiliza pin_logo.

Estos objetos tienen varios métodos asociados dependiendo de lo que el pin específico es capaz de hacer, por ejemplo, leer, escribir o tocar.

Quizá lo mas sencillo que podemos hacer es comprobar que los pines 0, 1 y 2 del borde de placa son táctiles. Haremos como ejemplo que al tocar cualquiera de ellos la micro:bit sonria y si no se toca ningúno que esté triste. Le hacemos cosquillas a la micro:bit. El programa es:

```
from microbit import *
"""
pin0.set_touch_mode(pin0.CAPACITIVE)
pin1.set_touch_mode(pin0.CAPACITIVE)
pin2.set_touch_mode(pin0.CAPACITIVE)
"""
while True:
    if (pin0.is_touched() or pin1.is_touched() or pin2.is_touched()):
        display.show(Image.HAPPY)
    else:
        display.show(Image.SAD)
```

En la animación vemos el funcionamiento del programa.

pinout.png

Autor Federico Coca Fuente : Guía de Trabajo de Microbit Licencia CC-BY-SA

Copyright 2026

Si descargamos firmware en una placa para probar el programa debemos saber que no basta con tocar alguno de los pines con una mano, hay que tocarlo simultaneamente con la otra mano en GND para cerrar el circuito eléctrico.

En la última versión de micro:bit V2 es posible cambiar el comportamiento predeterminado de la patilla, de modo que no sea necesario tocar GND. En los programas siguientes el código que hace esto está comentado por lo que si queremos probarlo debemos eliminar esos comentarios. Recordemos que por defecto los pines del conector de borde son sensores táctiles resistivos mientras que el pin logo V2 es capacitivo.

Pines digitales

Podemos utilizar los pines 0, 1 y 2 del borde de placa en modo digital tanto para leer su valor como para escribir o establecer su valor. Esto se representa con un "1" lógico (sin las comillas) si están activados o los queremos activar y un "0" lógico si están desactivados o los queremos desactivar.

Si queremos escribir en ellos los pines estarán actuando como salidas y tenemos que invocar al método `write` para hacerlo. Las sentencias, para un pin genérico "N" son:

```
pinN.write_digital(1) #Salida en estado alto
pinN.write_digital(0) #Salida en estado bajo
```

También podemos conectar, por ejemplo un interruptor o botón pulsador al pin (veremos como hacerlo en la siguiente actividad) y comprobar si el interruptor está abierto (0) o cerrado (1). En este caso los pines estarán configurados como entradas y la lectura de su estado se obtiene invocando el método `read`. Las sentencias, para un pin genérico "N" son:

```
pinN.read_digital() #Devuelve el estado 0 o 1 del pin N
```

Nunca se conecta nada a los pines con un voltaje superior a 3v porque se puede dañar la micro:bit.

Pines analógicos

Podemos utilizar los pines 0, 1 y 2 del borde de placa en modo analógico tanto para leer su valor como para escribir o establecer su valor. Esto significa que en lugar de estar activos o inactivos (0 o 1), varían su valor entre 0 y 1023.

Si queremos escribir en ellos los pines estarán actuando como salidas y tenemos que invocar al método `write` para hacerlo. La sentencia, para un pin genérico "N" es:

```
pinN.write_analog(valor) #valor puede estar entre 0 y 1023
```

Si conectamos sensores o actuadores analógicos a los pines podemos leer su valor invocando a `read`. La sentencia, para un pin genérico "N" es:

```
pinN.read_analog(valor) #valor puede estar entre 0 y 1023
```

> Página extraída de Federico Coca Guia de Trabajo de Microbit CC-BY-SA

Revision #2
Created 2024-09-26 13:19:06 CEST by Juanjo Mejías
Updated 2024-09-29 10:55:32 CEST by Juanjo Mejías