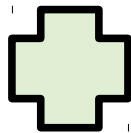


MICRO:CANSAT

Electrónica de CANSAT con MICRO:BIT



Por: *Pedro Ruiz Fernández*
Versión 29/06/2020

Licencia



Introducción

Se trata de implementar la electrónica de un minisatélite cansat con la placa micro:bit en python, aunque también se puede realizar en makecode.

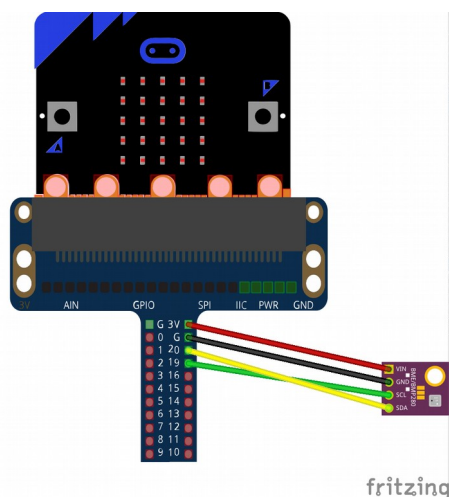
Materiales necesarios

Item	Cantidad	Descripción
1	1	Placa micro:bit
2	1	Shield para obtener pinout completo de micro:bit
3	1	Sensor bmp-280
4	4	Cables dupont hembra-hembra

Conexionado

Se conecta micro:bit al shield y en el shield se conecta el sensor bmp280 a los siguientes pines del shield, es un conexión tipo I2C.

Pin Shield	Pin bmp280
gnd	gnd
5V	vcc
19 (pin scl del shield)	scl
20 (pin sda del shield)	sda



Programación de micro:bit (python)

Vamos a necesitar el [programa principal](#) y una librería para tener las órdenes de lectura del bmp280, que es un sensor que nos da temperatura(°C), presión (Pa) y altitud (m).

Como entorno de programación he utilizado <https://python.microbit.org/v/2.0>, en el mismo he puesto el siguiente código principal:




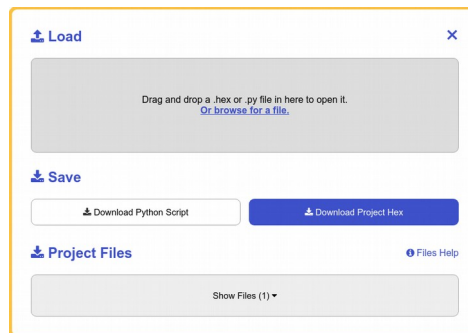
```
1 from microbit import * #importa la libreria con las órdenes propias de microbit
2 import bmp280 #importa la libreria con las instrucciones para el sensor bmp280
3 #uart.init(9600,tx=pin14,rx=pin15)
4 uart.init(9600) #inicializa una comunicación serie a 9600 baudios
5 b = bmp280.BMP280() #crea un elemento b del tipo bmp280
6 graba=False #variable booleana para controlar el envío de datos o no
7 buffer="tiempo"+"temperatura"+"presion"+"altitud+"\r\n" #variable tipo string que contiene lo que se envia por puerto serie
8 uart.write(buffer) #envia a puerto serie la variable buffer
9
10 while True: #bucle infinito
11     if button_a.is_pressed(): #si el botón "a" está presionado
12         graba = not graba #cambia el estado de graba por su contrario de False a True o de True a False
13         sleep(200) #se espera 200 ms
14     if button_b.is_pressed(): #si el botón "b" está presionado
15         display.show(Image.SNAKE) #enseña dibujo de una serpiente
16         break #sale del bucle (lo interrumpe)
17     if graba == True: # si graba es cierto
18         display.show(Image.YES) # enseña dibujo de verificado
19         buffer="soto,"+str(running_time())+","+str(b.Temperature())+","+str(b.Pressure())+","+str(b.Altitude())+",fin\r\n"
20         #al buffer añade texto soto, tiempo, temperatura, presion, altitud, fin seperados por coma y al final con retorno de carro
21         uart.write(buffer) #envia a puerto serie la variable buffer
22         sleep(500) #se espera 500 ms
23     else:
24         display.show (Image.NO) #enseña dibujo de cruz
```

Funcionamiento del programa principal:

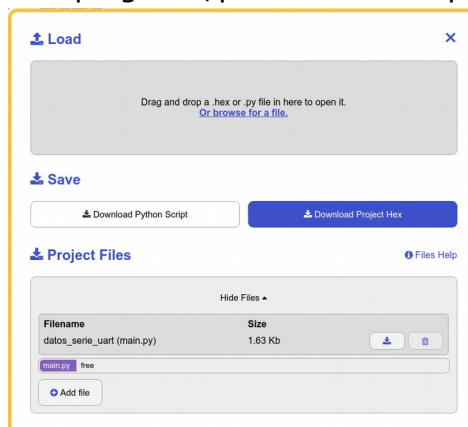
Al pulsar el botón “a” cambia el estado de la variable booleana “graba” por su contrario, y cuando es true muestra en la matriz de leds un símbolo de verificado y envia por puerto serie una cadena de texto compuesta por “soto, tiempo, temperatura, presion, altitud, fin con retorno de carro”. Los dos primer valores ya veremos que son para realizar el control de los datos enviados. Si la variable “graba” vale False se muestra un símbolo de una cruz en la matriz de leds. Si pulsamos el botón “b” se muestra en la matriz de leds el símbolo de una serpiente y sale del bucle principal interrumpiendo el programa.

¿Como introducir la librería del bmp280 y cargar todo el programa junto con la librería en microbit?.

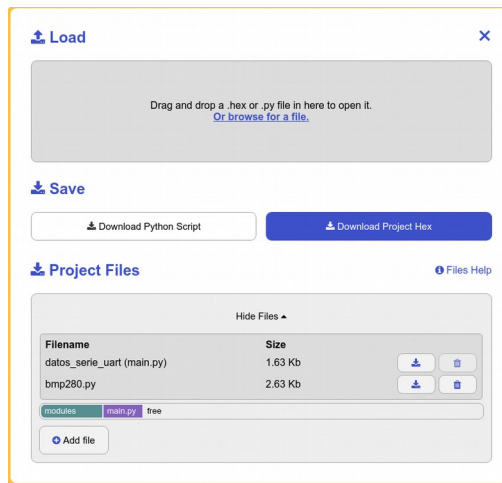
Pulsamos el botón  e incorporamos la librería [bmp280.py](#) previamente descargada en tu PC a los ficheros del programa, para ello pulsamos en la zona “Show Files”



Apareciendo los ficheros que tengo en el programa, por ahora sólo el principal:



Le damos al botón “Add File”, te sale una ventana para elegir y añadir el fichero [bmp280.py](#) previamente descargado a nuestro ordenador.



Ahora sólo queda descargar el fichero hexadecimal del proyecto en [Download Project Hex](#), te aparece una ventana para descargar el fichero en una carpeta en nuestro caso en la de microbit.

Programación de la estación base con Processing

En la estación base tendremos instalado el editor [Processing](#), en el tenemos abierto y ejecutándose un [programa](#) que recoge los datos enviados por puerto serie y comprueba que se reciben correctos, para ello hace tres comprobaciones con la información: que esté integrada por 6 campos, que el primer campo sea el texto "soto" y que el último sea el texto "fin". Con la información que cumple estos requisitos, la guardamos en tiempo real un fichero llamado "datos.csv", el fichero se encuentra en la misma carpeta del programa de estación base de processing, y además se representa en pantalla. Posteriormente este fichero se puede abrir y procesar en una hoja de cálculo o bien graficar en tiempo real con aplicaciones como [kst](#).

