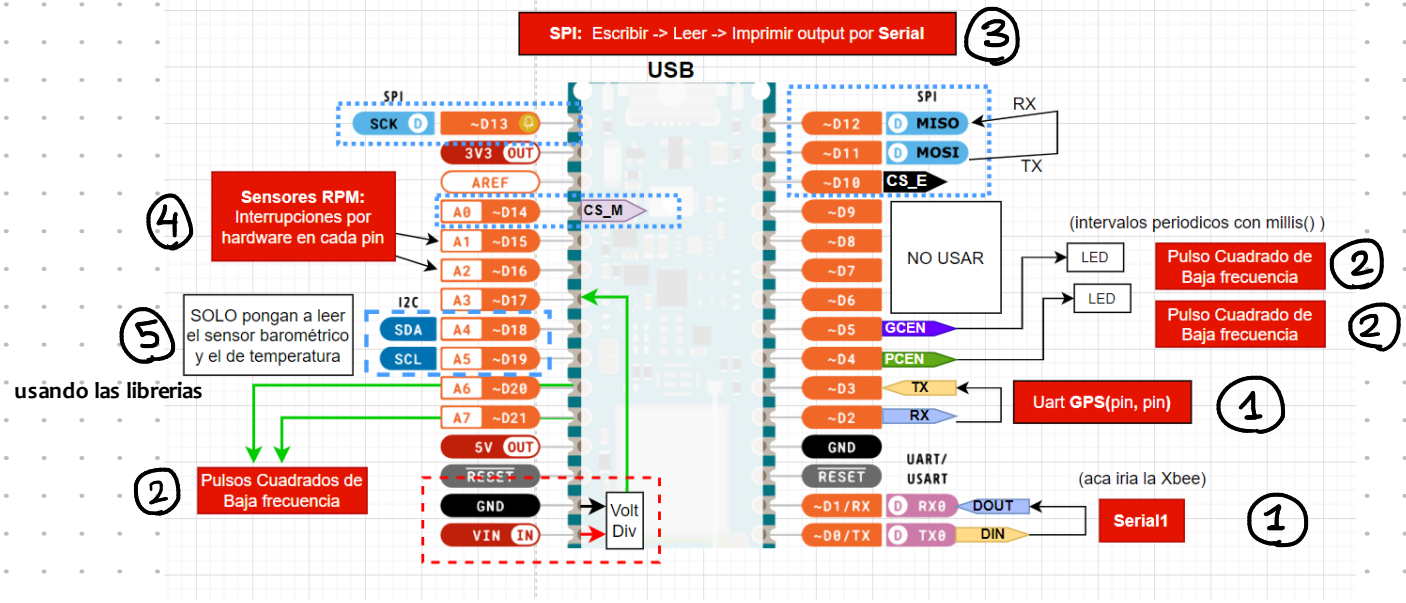


# CODIGO DE PRUEBAS BASICO PARA PCB\_LAB\_TEST



**OBJETIVO:** Probar el funcionamiento **SIMULTANEO** de todos los sistemas básicos de Comunicación o GPIO de la Placa para estar seguros de que no se interfieren entre sí y pueden coexistir funcionalmente

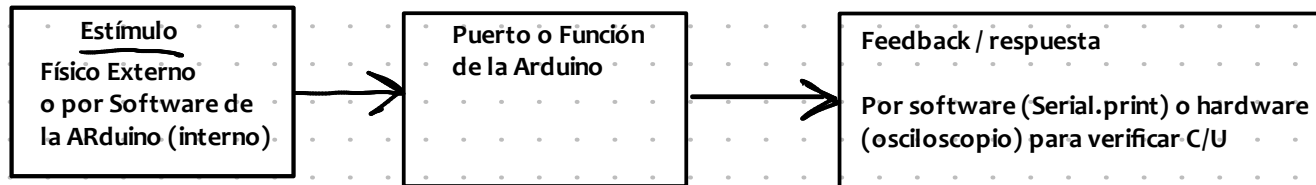
**SOLAMENTE** Para validar las conexiones de pines de este esquemático

**NO VAMOS** a hacer código funcional con lógica de CANSAT

## Lista completa de sistemas/modulos a probar

- |   |                                 |                              |
|---|---------------------------------|------------------------------|
| 1. UART                                 | (2 puertos)                     |                              |
| 2. Salidas Digitales de Baja Frecuencia | (4 pines)                       |                              |
| 3. SPI                                  | (2 instancias)                  | Implica 2 pines CS distintos |
| 4. Interrupciones de hardware           | (2 pines)                       | Implica 2 ISR distintas      |
| 5. I2C                                  | (leer 2 sensores de la Arduino) | baro.y temp                  |

La LOGICA / IDEA es la MISMA para todas las pruebas.



## PUERTOS SPI, UART y Serial1

**INPUT:** En cada uno mandenle un par de bytes distintos, puede ser "HOLA", chau, etc

También agreguen un par de líneas de código con lógica para recibir bytes

### Serial1

ya esta instanciada, incluida en <Arduino.h> de la Nano 33 BLE

usar con begin/available, etc normalmente

### Uart GPS

Funciona igual que Serial1 PERO

Hay que instanciarla globalmente como: Uart GPS(pin, pin) y asignarle los pines

### SPI x2

Hacer 2 (DOS) instancias con pines CS distintos



La lógica de control de estas 4 instancias la aclaro mas abajo

# PINES de SALIDA DIGITAL DE BAJA FRECUENCIA



Por CADA UNO de estos pines, hagan un INTERVALO DISTINTO  
y solamente un: **con PERÍODOS DISTINTOS**

```
d5_out = !d5_out;  
digitalWrite( d5_out )
```

## INTERRUPCIONES DE HARDWARE

**CRITICO**

Son ESPECÍFICAS para el Microcontrolador / Placa / Framework que estamos usando:

NINA B306


nRF52840

Arduino Nano 33 BLE

es el mismo micro que  
para la Sense Rev1/2, etc

```
platform = nordicnrf52  
board = nano33ble  
framework = arduino
```

INVESTIGUEN solamente lo MINIMO para tener el código andando de una ISR (Interrupt Subroutine) en nuestra placa.

- A) Cómo se implementa una ISR en nuestra placa? (qué macros o definiciones hay que usar)
- B) Cómo se instancia, cuál es la estructura de la función?
- C) Cómo colocar una ISR que se dispare por una IRQ (Interrupt) tipo "Rising-Edge" o "Falling Edge" ?
- D) SE PUEDEN USAR LOS PINES  Para estas IRQ?

NO PROFUNDICEN en teoría sobre otras cosas relacionadas, no tenemos tiempo para ver todas las implicaciones.

## IMPORTANTE

La función ISR que se ejecuta cuando llega una IRQ, debe ser **MINIMA**

SOLAMENTE incrementen un contador global distinto en cada ISR

## I2C

Solamente lean los valores de 2 (DOS) sensores de la Arduino, con esto verificamos que I2C anda bien  
Barométrico y Temperatura (creo que son integrados distintos)

## LÓGICA DE CONTROL (LOOP)

Intervalo de 1 (un) segundo (con millis)

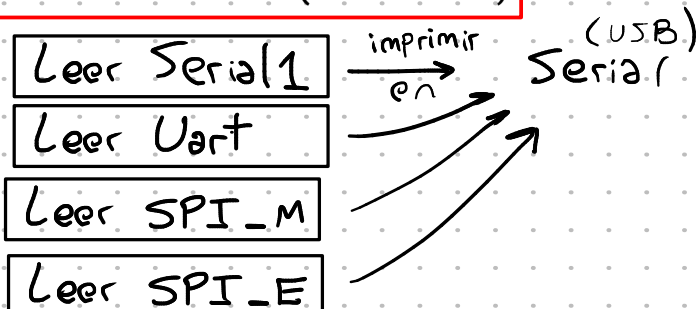
UART, Serial y las 2 Instancias de SPI ponganlas para que en un mismo intervalo de 1 (un) segundo envíen toda la información

SPI Ambos sensores ponganlos en este mismo intervalo, leer los valores e imprimirlos por serial todo acá

4 Intervalos de períodos distintos (con millis)

PINES de SALIDA DIGITAL DE BAJA FRECUENCIA

## LOOP "REAL TIME" (sin intervalos)



La lógica debería ser similar para TODOS estos objetos

```
if ( X available )  
    Serial.print ( X read )
```