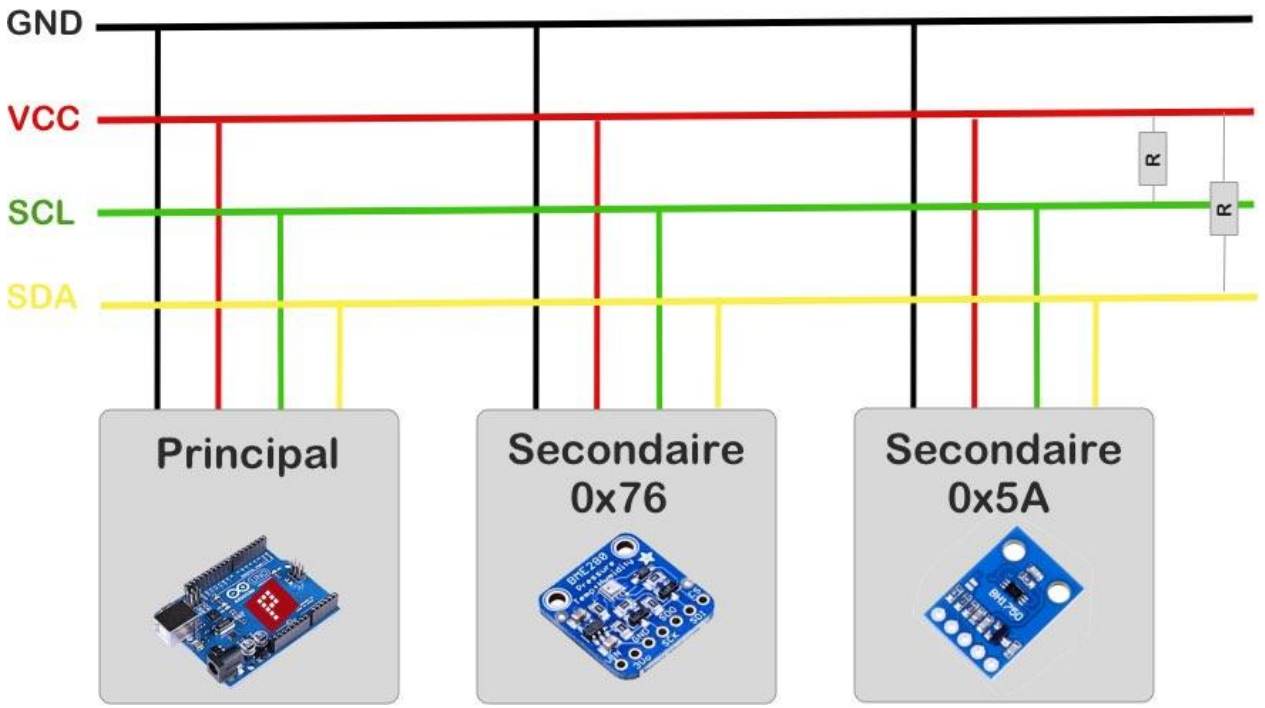
TP Capteur

MSC 2022 - CHAPUIS Clément & SUTRA Aurélien

- La norme I^2C (Inter-Integrated Circuit) a été développée par Philips en 1982.
- La version la plus récente est la 6ème
- 4 fils dont Vcc et GND pour l'alimentation, SCL pour l'horloge et SDA pour l'échange de donnée
- Un bus série transmet l'information bit par bit à des instants qui dépendent de l'horloge (synchrone). Il peut communiquer de esclave à maître et de maître à esclave (bidirectionnel) mais pas simultanément (half-duplex).
- Adresse classiquement codée sur 7 bit + 1 bit R/W =>128 appareils adressés
- La vitesse de transmission est 5Mbps



- La fonction à utiliser pour scanner le bus est : HAL_I2C_IsDeviceReady

HAL_I2C_IsDeviceReady

Function name	**HAL_StatusTypeDef HAL_I2C_IsDeviceReady (I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint32_t Trials, uint32_t Timeout)**
Function description	Checks if target device is ready for communication.

- Ci-contre un exemple d'utilisation de cette fonction pour scanner le bus I2c

```
32
33  #include "main.h"
34  #include "stdio.h"
35
36  I2C_HandleTypeDef hi2c1;
37  UART_HandleTypeDef huart1;
38
39  uint8_t Buffer[25] = {0};
40  uint8_t Space[] = " - ";
41  uint8_t StartMSG[] = "Starting I2C Scanning: \r\n";
42  uint8_t EndMSG[] = "Done! \r\n\r\n";
43
44  void SystemClock_Config(void);
45  static void MX_GPIO_Init(void);
46  static void MX_I2C1_Init(void);
47  static void MX_USART1_UART_Init(void);
48
49  int main(void)
50  {
51      uint8_t i = 0, ret;
52
53      HAL_Init();
54      SystemClock_Config();
55      MX_GPIO_Init();
56      MX_I2C1_Init();
57      MX_USART1_UART_Init();
58
59      HAL_Delay(1000);
60
61      /*-[ I2C Bus Scanning ]-*/
        HAL_UART_Transmit(&huart1, StartMSG, sizeof(StartMSG), 10000);
        for(i=1; i<128; i++)
        {
            ret = HAL_I2C_IsDeviceReady(&hi2c1, (uint16_t)(i<<1), 3, 5);
            if (ret != HAL_OK) /* No ACK Received At That Address */
            {
                HAL_UART_Transmit(&huart1, Space, sizeof(Space), 10000);
            }
            else if(ret == HAL_OK)
            {
                sprintf(Buffer, "0x%X", i);
                HAL_UART_Transmit(&huart1, Buffer, sizeof(Buffer), 10000);
            }
        }
        HAL_UART_Transmit(&huart1, EndMSG, sizeof(EndMSG), 10000);
        /*--[ Scanning Done ]--*/

        while (1)
        {
        }
  }
```

Capteur MPU-9250

- Le registre WHOAMI permet de vérifier l'identité du capteur utilisé.
- La valeur attendue est une ID d'appareil sur 8 bit.
- HAL_I2C_Mem_Read permet de lire la valeur du registre WHOAMI

Name: WHOAMI

Serial IF: Read Only

Reset value: 0x68

BIT	NAME	FUNCTION
[7:0]	WHOAMI	Register to indicate to user which device is being accessed.

This register is used to verify the identity of the device. The contents of *WHO_AM_I* is an 8-bit device ID. The default value of the register is 0x71.

HAL_I2C_Mem_Read

Function name	**HAL_StatusTypeDef HAL_I2C_Mem_Read (I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint16_t MemAddress, uint16_t MemAddSize, uint8_t * pData, uint16_t Size, uint32_t Timeout)**
Function description	Read an amount of data in blocking mode from a specific memory address.

Température

- Pour lire la température mesurée par le capteur, il faut lire les registres TEMP_OUT_H et TEMP_OUT_L
- Pour convertir les données du capteur en °C, on fait le calcul suivant : TEMP_degC=((TEMP_OUT– RoomTemp_Offset)/Temp_Sensitivity) + 21degC
- Roomtemps représente la temperature ambiante de la pièce (21°C), sans unité. Temp sensitivity est la sensibilité du capteur en LSB/°C

Parameter	Conditions	MIN	TYP	MAX	Units
Supply Ramp Time	Monotonic ramp. Ramp rate is 10% to 90% of the final value	0.1		100	ms
Operating Range	Ambient	-40		85	°C
Sensitivity	Untrimmed		333.87		LSB/°C
Room Temp Offset	21°C		0		LSB

4.23 Registers 65 and 66 – Temperature Measurement

Name: TEMP_OUT_H

Serial IF: SyncR

Reset value: 0x00 (if sensor disabled)

BIT	NAME	FUNCTION
[7:0]	D[7:0]	High byte of the temperature sensor output

Name: TEMP_OUT_L

Serial IF: SyncR

Reset value: 0x00 (if sensor disabled)

BIT	NAME	FUNCTION
[7:0]	D[7:0]	Low byte of the temperature sensor output: **TEMP_degC** = ((TEMP_OUT – RoomTemp_Offset)/Temp_Sensitivity) + 21degC Where Temp_degC is the temperature in degrees C measured by the temperature sensor. TEMP_OUT is the actual output of the temperature sensor.

Accélération

- Les registres à lire pour connaitre l'accélération sont accel_xout, accel_yout et accel_zout. Chacun est codé sur 16 bits
- Le passage des données brutes en g se fait à partir de l'échelle choisie. Par exemple, pour une échelle de +-2, on a 0x0000 qui correspond à -2g et 0xFFFF qui correspond à 2g
- D'après la doc : « When the device is placed on a flat surface, it will measure 0g on the X- and Y-axes and +1g on the Z-axis. »

4.7 Register 28 – Accelerometer Configuration

Serial IF: R/W

Reset value: 0x00

BIT	NAME	FUNCTION
[7]	ax_st_en	X Accel self-test
[6]	ay_st_en	Y Accel self-test
[5]	az_st_en	Z Accel self-test
[4:3]	ACCEL_FS_SEL[1:0]	Accel Full Scale Select: ±2g (00), ±4g (01), ±8g (10), ±16g (11)
[2:0]	-	Reserved