

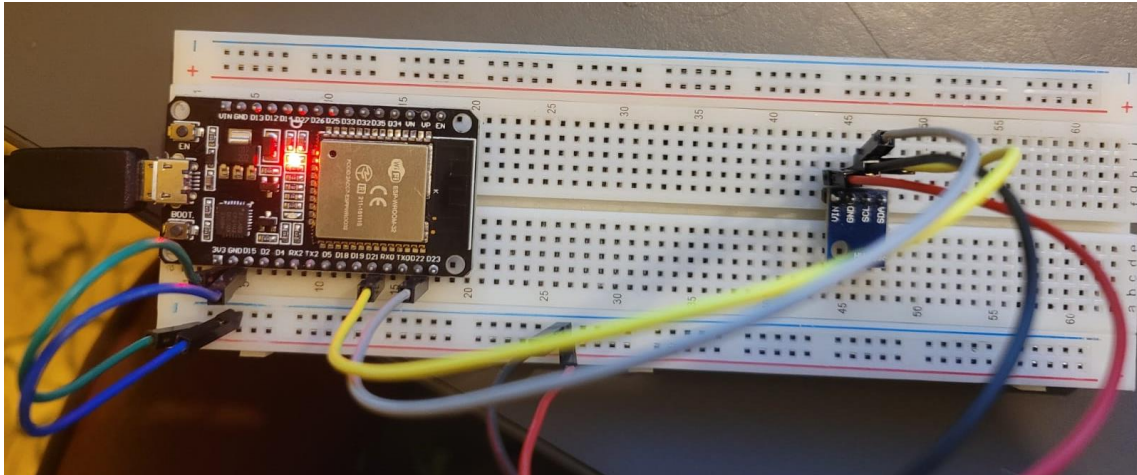
## Tarea 3

### Máquinas Digitales con Laboratorio

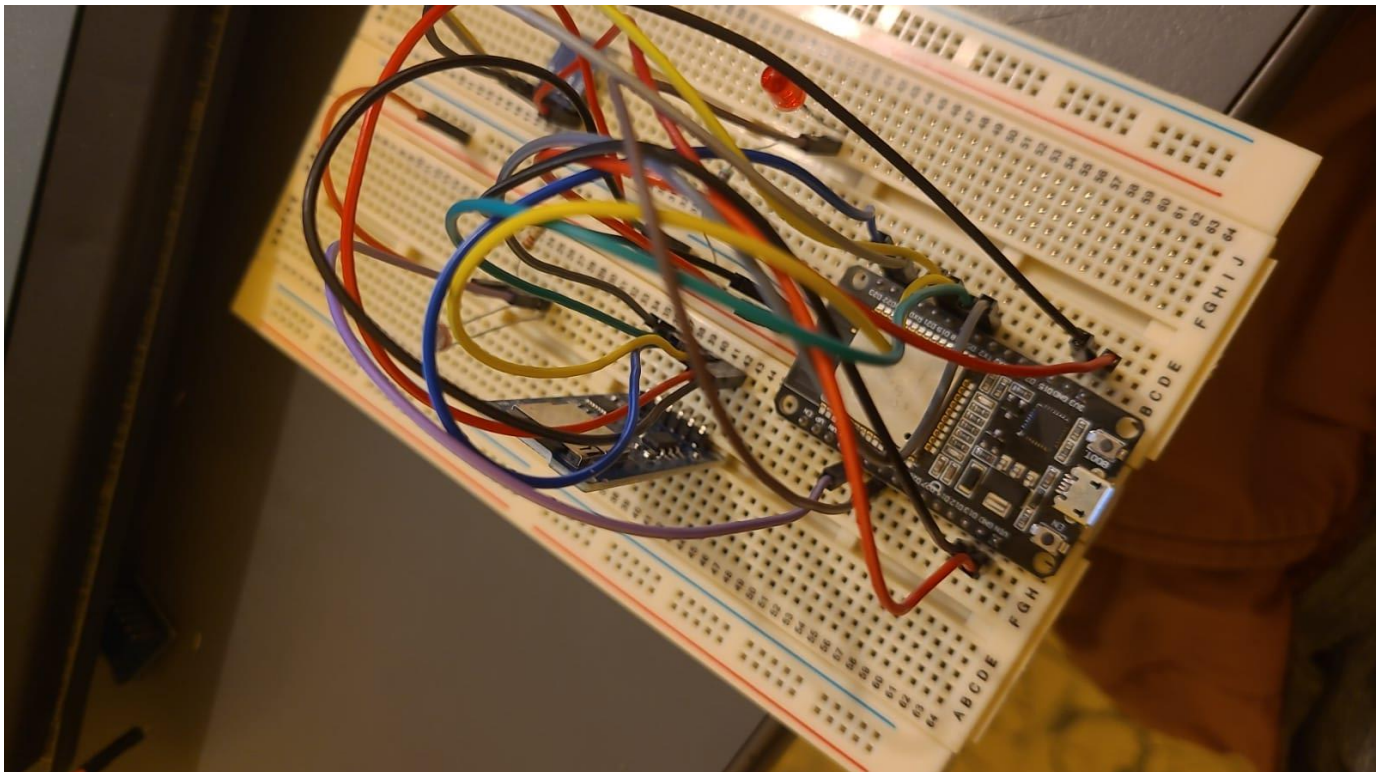
Óscar Alvarado

#### Protocolo SPI e I2C

- Foto de la conexión del sensor atmosférico



- Foto de la conexión del lector de tarjetas uSD.



- Responder: ¿Por qué es adecuada la decisión de usar el protocolo I2C para comunicar al sensor atmosférico y el protocolo SPI para comunicar al lector de tarjetas uSD?  
El protocolo I2C es el indicado para el sensor atmosférico porque como incluye dos mediciones, tenemos que tener dos direcciones distintas, de manera que este tipo de protocolo queda muy bien porque justo es para eso. Para el caso de SPI para el lector de tarjetas es buena idea usar dicho protocolo porque se tiene una conexión Full – dúplex, de modo que nosotros podremos enviar y recibir datos sin problema alguno.

## Sensor atmosférico

- Captura del monitor serie mostrando la dirección del sensor atmosférico y explicar a alto nivel cómo funciona ese programa. Se usa el ejemplo "Wire Scan".

```
rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0030,len:1344
load:0x40078000,len:13924
ho 0 tail 12 room 4
load:0x40080400,len:3600
entry 0x400805f0
E (129) esp_core_dump_<[V] Incorrect size of core dump image: 0
Scanning for I2C devices ...
I2C device found at address 0x77
Scanning for I2C devices ...
I2C device found at address 0x77
```

Este programa funciona por fuerza bruta, manda una señal a todas las direcciones entre 0x01 y 0x7f en hexadecimal y si alguna dirección le contesta, nos muestra como que lo encontró.

- Captura/Video del monitor serie y serial plotter con las lecturas de temperatura y presión. Los datos deben almacenarse en una variable llamada "payload", construida con la función "sprintf". El monitor serie debe mostrar los valores a dos columnas para que puedan interpretarse correctamente

por el serial plotter.

The screenshot shows the Arduino IDE environment. The main window displays the 'sensor\_atmos.ino' sketch, which is designed to read data from an Adafruit BMP085 sensor. The code includes comments in Spanish and uses the `Adafruit_BMP085` library. The `setup()` function initializes the serial port at 115200 baud. The `loop()` function reads the sensor data, formats it into a string payload, and prints it to the serial monitor. The Serial Monitor at the bottom shows the output of the sketch, displaying pressure and temperature values. To the left, the Serial Plotter is open, showing a graph of the data being received from the serial port. The plot shows a steady increase in values over time, with a 'STOP' button and an 'Interpolate' checkbox visible. The background shows a web browser with a page titled 'Maquinas Digitales 2023-2' and a search bar containing 'ESP32 101'.

- Push con el del sketch "sensor\_atmosférico" que haga lo del punto anterior.

The screenshot shows a GitHub repository page for the 'sensor\_atmos.ino' file. The repository is named 'Maquinas-Digitales' and is located in the 'Tareas / Tarea3 / sensor\_atmos' directory. The file is owned by 'OscarAlvaradoM' and has a latest commit 6 hours ago. The file content is displayed in a dark theme, showing the same C++ code as seen in the previous screenshot. The code is 24 lines long and 708 bytes in size. The repository has 1 contributor.

## Lector de tarjetas

- Captura del resultado de la prueba con el programa "SD\_Test", que muestre que el micro puede crear, leer y borrar objetos en la tarjeta uSD.

The screenshot displays the Arduino IDE 2.0.4 interface. The top menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar shows icons for checking, running, and uploading code, along with a dropdown menu currently set to 'ESP32 Dev Module'. The left sidebar contains icons for the Explorer, Sources, and Serial Monitor. The main editor window shows the 'SD\_Test.ino' file with the following code:

```

14  */
15  #include "FS.h"
16  #include "SD.h"
17  #include "SPI.h"
18
19  void listDir(fs::FS &fs, const char * dirname, uint8_t levels){
20      Serial.printf("Listing directory: %s\n", dirname);

```

Below the editor is the 'Serial Monitor' tab, which is active. It shows a message input field with the placeholder 'Message (Enter to send message to 'ESP32 Dev Module' on 'COM6')' and buttons for 'New Line' and '115200 baud'. The output area displays the following serial data:

```

FILE: test.txt  SIZE: 1048576
FILE: foo.txt  SIZE: 13
Writing file: /hello.txt
File written
Appending to file: /hello.txt
Message appended
Reading file: /hello.txt
Read from file: Hello World!
Deleting file: /foo.txt
File deleted
Renaming file /hello.txt to /foo.txt
File renamed
Reading file: /foo.txt
Read from file: Hello World!
1048576 bytes read for 2478 ms
1048576 bytes written for 2913 ms
Total space: 7423MB
Used space: 1MB

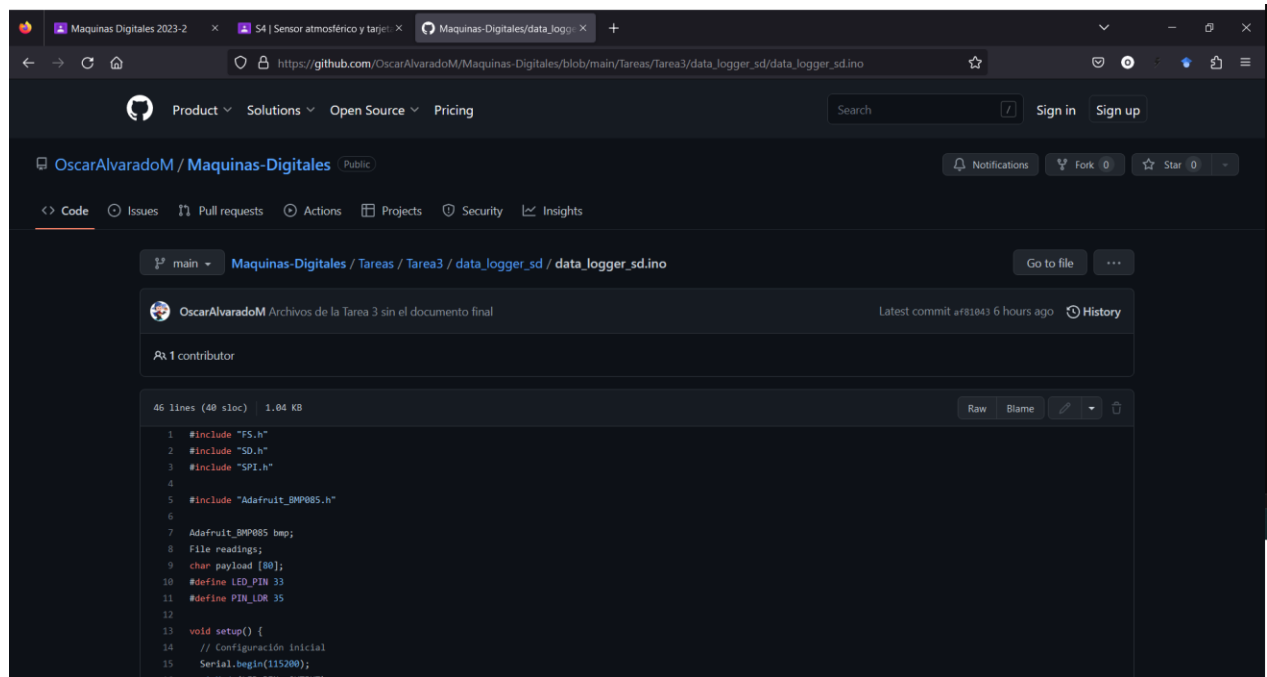
```

The status bar at the bottom indicates 'Ln 5, Col 17' and 'ESP32 Dev Module on COM6'.

- Video con el micro encendiendo un LED cada que guarda una lectura en la tarjeta.  
(Lo adjunto en el Classroom)
- Captura de la hoja en Excel/Google Sheets con los valores y gráficas individuales de las lecturas del sensor atmosférico y el LDR, las cuales se guardaron en la tarjeta SD en un archivo .csv.

</

- Push con el programa "data\_logger\_sd" que haga lo que se pide en los dos puntos anteriores.



The screenshot shows a web browser displaying a GitHub repository. The repository is named "Maquinas-Digitales" and is owned by "OscarAlvaradoM". The file "data\_logger\_sd.ino" is selected, showing its code content. The code is an Arduino sketch for an SD card logger using an Adafruit BMP085 sensor. The code includes headers for "FS.h", "SD.h", "SPI.h", and "Adafruit\_BMP085.h". It defines a file name "Adafruit\_BMP085 bmp;" and a payload array "char payload [80];". It also defines constants for "LED\_PIN" (33) and "PIN\_LDR" (35). The "setup()" function initializes the serial port at 115200 baud. The code is 46 lines long (40 SLOC) and 1.04 KB in size.

```
1 #include "FS.h"
2 #include "SD.h"
3 #include "SPI.h"
4
5 #include "Adafruit_BMP085.h"
6
7 Adafruit_BMP085 bmp;
8 File readings;
9 char payload [80];
10 #define LED_PIN 33
11 #define PIN_LDR 35
12
13 void setup() {
14   // Configuración inicial
15   Serial.begin(115200);
16   // ... (the rest of the code is truncated in the image)
```