INFORME PRÁCTICA 5: Buses de comunicación I (introducción y I2C)

El objetivo de la practica es comprender el funcionamiento de los buses sistemas de comunicación entre periféricos; estos elementos pueden ser internos o externos al procesador.

Código 1

```
#include <Arduino.h>

#include <Wire.h>


void setup()
{
  Wire.begin();
  Serial.begin(115200);
  while (!Serial);
  Serial.println("\nI2C Scanner");
}


void loop()
{
  byte error, address;
  int nDevices;

  Serial.println("Scanning...");

  nDevices = 0;
  for(address = 1; address < 127; address++ )
  {
    Wire.beginTransmission(address);
    error = Wire.endTransmission();

    if (error == 0)
    {
      Serial.print("I2C device found at address 0x");
      if (address<16)
        Serial.print("0");
      Serial.print(address,HEX);
      Serial.println("  !");

      nDevices++;
    }
    else if (error==4)
    {
      Serial.print("Unknown error at address 0x");
      if (address<16)
        Serial.print("0");
      Serial.println(address,HEX);
    }
  }
```

```
  if (nDevices == 0)
    Serial.println("No I2C devices found\n");
  else
    Serial.println("done\n");

  delay(5000);
}
```

Este código nos muestra por la pantalla del terminal la dirección de cada esclavo conectado al microprocesador. En este caso, nos muestra la dirección de una pantalla LCD con el bus I2C.

Inclusión de librerías

```
#include <Arduino.h>
#include <Wire.h>
```

- *Arduino.h*: Proporciona las definiciones y funciones básicas de Arduino.
- *Wire.h*: Proporciona las funciones necesarias para la comunicación I2C

Función *setup()*

```
void setup() {
  Wire.begin();
  Serial.begin(115200);
  while (!Serial);
  Serial.println("\nI2C Scanner");
}
```

- *Wire.begin()*: Inicializa la biblioteca Wire para la comunicación I2C.
- *Serial.begin(115200)*: Inicializa la comunicación serial a 115200 baudios.
- *while (!Serial);*: Espera hasta que el monitor serial esté listo.
- *Serial.println("\nI2C Scanner");*: Imprime el mensaje "I2C Scanner" en el monitor serial.

Función *loop()*

EXPLICACIÓN POR PARTES:

Declaración de variables

```
byte error, address;
int nDevices;
```

- *error*: Almacena el código de error de la transmisión I2C.
- *address*: Almacena las direcciones I2C.
- *nDevices*: Cuenta el número de dispositivos encontrados.

Inicio del escaneo

```
Serial.println("Scanning...");
nDevices = 0;
```

+ Imprime "Scanning..." en el monitor serial.
+ Inicializa el contador de dispositivos nDevices a 0.

Escaneo de direcciones del bus I2C

```
for(address = 1; address < 127; address++ ) {
  Wire.beginTransmission(address);
  error = Wire.endTransmission();
```

+ Utiliza un bucle for para iterar sobre todas las posibles direcciones I2C (de 1 a 126).
+ *Wire.beginTransmission(address)*: Inicia una transmisión I2C hacia la dirección address.
+ *error = Wire.endTransmission()*: Finaliza la transmisión y captura el código de error.

Procesamiento del resultado del escaneo

```
if (error == 0) {
  Serial.print("I2C device found at address 0x");
  if (address < 16)
    Serial.print("0");
  Serial.print(address, HEX);
  Serial.println("  !");
  nDevices++;
}
```

+ Si *error* es igual a 0, significa que el dispositivo a reconocido la dirección.
+ Imprime la dirección del dispositivo en formato hexadecimal y se incrementa el contador nDevices

```
else if (error == 4) {
  Serial.print("Unknown error at address 0x");
  if (address < 16)
    Serial.print("0");
  Serial.println(address, HEX);
}
```

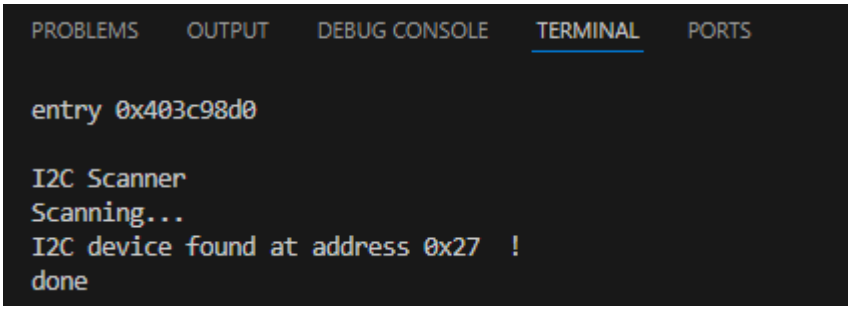
+ Si *error* es igual a 4, quiere decir que no reconoce ninguna conexión.
+ Imprime un mensaje indicando un error desconocido en esa dirección.

Resumen del escaneo

```
if (nDevices == 0)
  Serial.println("No I2C devices found\n");
else
  Serial.println("done\n");

delay(5000);
```

+ Al final del escaneo, se imprime un mensaje indicando si se encontraron los dispositivos o no.
+ Se esperan 5 segundos para volver a iniciar otro escaneo.



Como se puede ver en la imagen, la dirección de nuestro periferico en formato hexadecimal corresponde a 0x27.

Ejercicio práctico 2

Para este ejercicio, he vuelto a usar el display OLED anterior. Con dirección 0x27, la he conectado al ESP32-S3 y he utilizado el siguiente código:

Código 2

```
#include <LiquidCrystal_I2C.h>

int lcdColumns = 16;
int lcdRows = 3;


LiquidCrystal_I2C lcd(0x27, lcdColumns, lcdRows);

void setup(){
  // initialize LCD
  lcd.init();
  // turn on LCD backlight
  lcd.backlight();
}

void loop(){
  lcd.setCursor(0, 0);

  lcd.print("HOLA");
  delay(10000);
  lcd.clear();

  lcd.setCursor(0,1);
  lcd.print("QUE");
  delay(8000);
  lcd.clear();

  lcd.setCursor(0,2);
  lcd.print("TAL");
  delay(6000);
  lcd.clear();

  lcd.setCursor(0,3);
  lcd.print("ESTAS");
  delay(4000);
```

```
  lcd.clear();
}
```

Inclusión de librerías

```
#include <LiquidCrystal_I2C.h>
```

+ *LiquidCrystal_I2C.h*: Esta librería permite controlar pantallas LCD a través de la interfaz I2C.

Configuración inicial

```
int lcdColumns = 16;
int lcdRows = 3;
```

+ Se definen las variables lcdColumns y lcdRows para especificar el número de columnas y filas de la pantalla LCD. En este caso, la pantalla tiene 16 columnas y 3 filas.

Configuración del LCD

```
LiquidCrystal_I2C lcd(0x27, lcdColumns, lcdRows);
```

+ Se crea un objeto lcd de la clase LiquidCrystal_I2C con la dirección I2C del LCD (0x27) y las dimensiones especificadas.

Función *setup()*

```
void setup() {
  lcd.init();
  lcd.backlight();
}
```

+ lcd.init(): Inicializa la pantalla LCD.

+ lcd.backlight(): Enciende la retroiluminación de la pantalla LCD.

Función *loop()*

```
void loop() {
  lcd.setCursor(0, 0);
  lcd.print("HOLA");
  delay(10000);
  lcd.clear();

  lcd.setCursor(0, 1);
  lcd.print("QUE");
  delay(8000);
  lcd.clear();

  lcd.setCursor(0, 2);
  lcd.print("TAL");
  delay(6000);
  lcd.clear();

  lcd.setCursor(0, 0);
  lcd.print("ESTAS");
  delay(4000);
  lcd.clear();
}
```

El *loop* esta dividido en 4 partes, que son las 4 palabaras que se muestran por el LCD. Son idénticas, solo cambian las palabras que se ven y el tiempo que estan en pantalla.

+ *lcd.setCursor(0, 1)*: Se indica la colocación del cursor. El primer numero indica la columna y el segundo la fila.
+ *lcd.print("PALABRA")*: Imprime la palabra que quieras.
+ *delay(10000)*: Indica el tiempo que esta en la pantalla. En este caso está durante 10 segundos.
+ *lcd.clear()*: Borra la pantalla. Se puede poner o no, depende de si quieres hacer texto o solo mostrar palabras.

Imagen de lo que se ve en la LCD:

Imagen del montaje