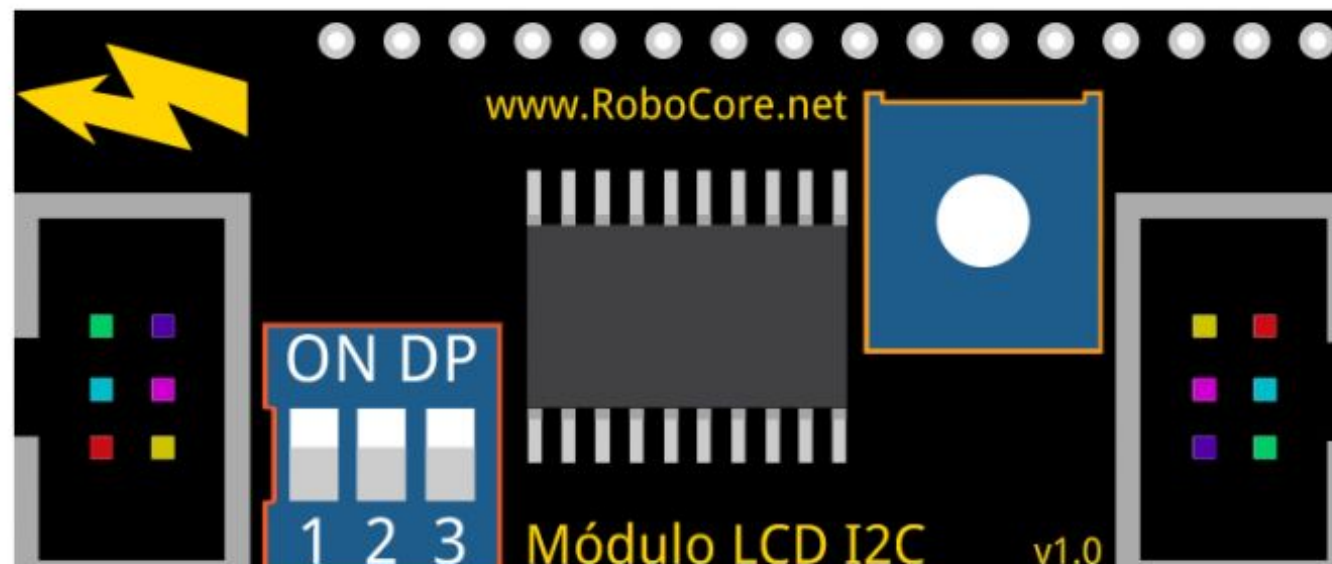


PROTOCOLO I2C (INTER INTEGRATED CIRCUIT)



Discentes: Lucas Pereira Ramos da Silva, Davy Araujo Sa teles

Orientador: Prof. Dr. Alexandre Jean René Serres

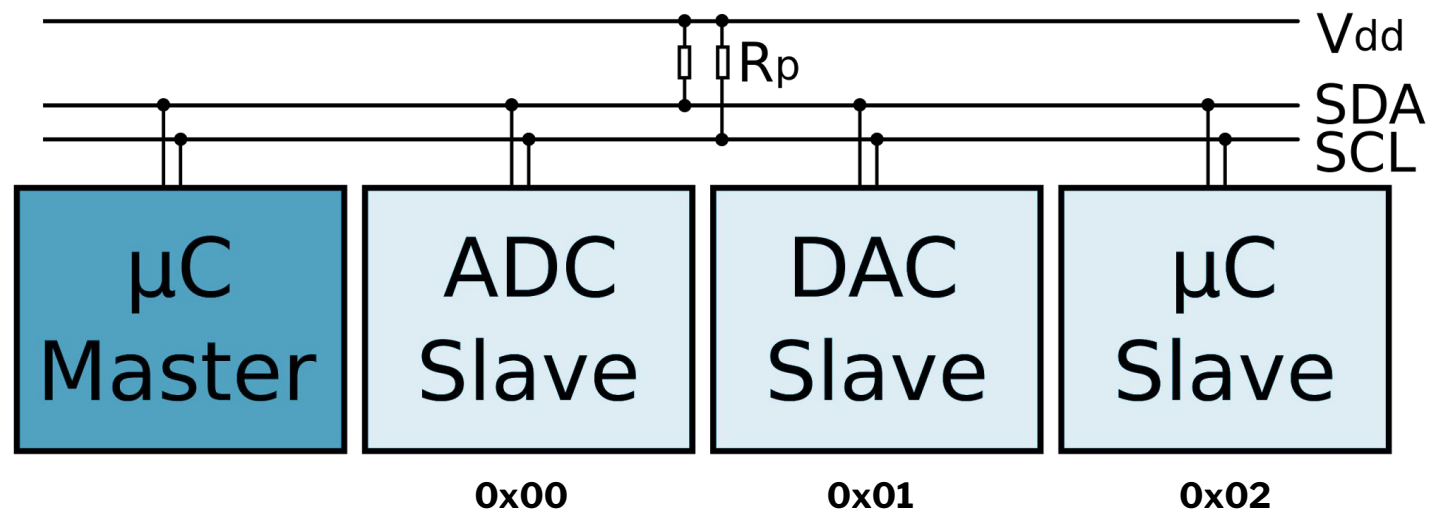


INTRODUÇÃO

- A comunicação I2C (Inter-Integrated Circuit) é um protocolo de comunicação serial que permite que dispositivos diferentes se comuniquem e troquem dados através de um barramento compartilhado.
- Desenvolvido pela Philips que é usado para conectar periféricos de baixa velocidade a uma placa mãe, a um sistema embarcado ou a um telefone celular.
- O I2C permite a comunicação entre múltiplos dispositivos por meio de apenas duas linhas de comunicação: uma linha de dados (**SDA** – Serial Data Line) e uma linha de clock (**SCL** – Serial Clock Line).

INTRODUÇÃO

FIGURA 01 - Modelo da comunicação I2C



Fonte: (Adaptado , 2023)

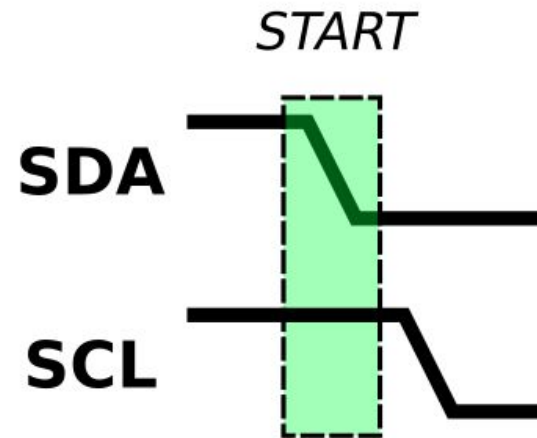
BENEFÍCIOS DO PROTOCOLO I2C

- Simplifica a comunicação entre vários dispositivos eletrônicos;
- Requer apenas duas linhas: SDA (Serial Data Line) e SCL (Serial Clock Line);
- Possibilita conectar vários dispositivos à mesma linha de comunicação;
- Útil para sistemas com diversos sensores, memórias, periféricos, etc.

ESTRUTURA DE TRANSMISSÃO

- **Início:** O dispositivo mestre gera um sinal de início, indicando o início da comunicação. A linha **SDA** muda de alto (1) para baixo (0) enquanto a linha **SCL** permanece alta;

FIGURA 02 - Início da comunicação I2C

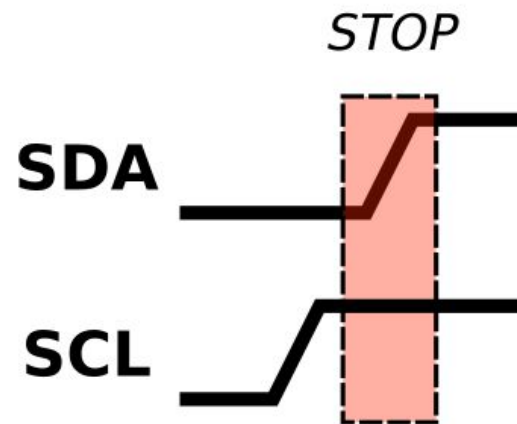


Fonte: (Adafruit , 2023)

ESTRUTURA DE TRANSMISSÃO

- **Parada:** O mestre gera um sinal de parada, indicando o fim da comunicação. A linha SDA muda de baixo (0) para alto (1) enquanto a linha SCL permanece alta.

FIGURA 03 - Parada da comunicação I2C



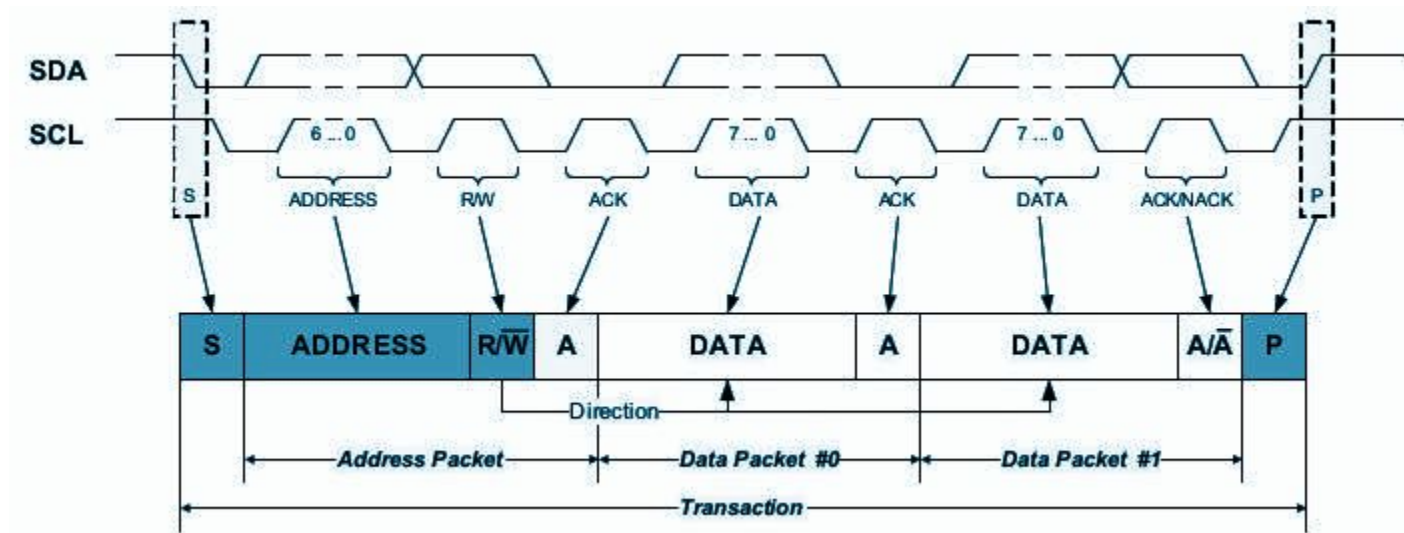
Fonte: (Adafruit , 2023)

ESTRUTURA DE TRANSMISSÃO

- **Endereço do Escravo:** O mestre envia o endereço do dispositivo escravo com o qual deseja se comunicar. O escravo que corresponde a esse endereço responde;
- **Dados:** A comunicação real ocorre aqui. O mestre ou o escravo podem enviar ou receber dados, dependendo do modo de operação;
- **ACK/NACK:** Reconhecer (ACK) e Não Reconhecer (NACK). Este é um único bit usado no protocolo I2C para indicar várias condições. Ele é enviado no SDA após a transferência de cada byte. Geralmente:
 - **ACK** = SDA BAIXO
 - **NACK** = SDA ALTO

ESTRUTURA DE TRANSMISSÃO

FIGURA 04 - Estrutura de transmissão

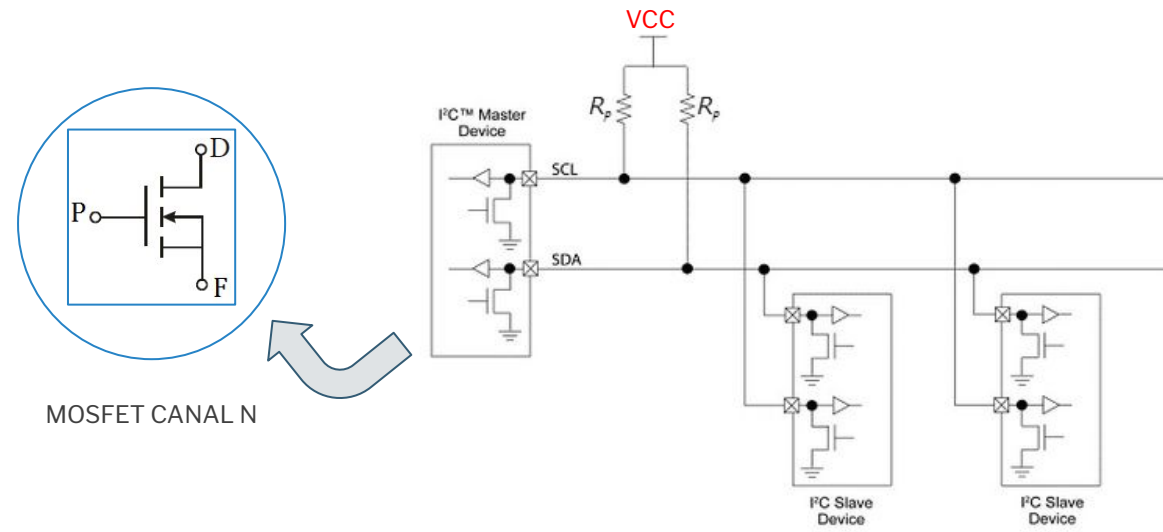


Fonte: (Adaptado , 2023)

RESISTOR PULL-UP

- Esses são os resistores exigidos pelo barramento I2C. Existem dois. Um entre **SDA** e **VCC**. Um entre **SCL** e **VCC**.

FIGURA 05 - Parada da comunicação I2C



Fonte: (PRÓPRIO AUTOR , 2023)

OPERAÇÃO

- Cada dispositivo escravo no barramento I²C possui um endereço exclusivo. Isso permite que o mestre se comunique com um dispositivo específico;
- O endereço pode ser de 7 ou 10 bits, dependendo do chip e do protocolo. Bits menos significativos podem ser usados para seleção de modos ou registradores internos.;
- O I²C oferece diversos modos de operação:
 - Transmissão: O mestre envia dados para o escravo.
 - Recepção: O mestre recebe dados do escravo.
 - Escrita: O mestre escreve dados em um registrador específico no escravo.
 - Leitura: O mestre lê dados de um registrador específico no escravo.
- Esses métodos permitem uma ampla gama de interações entre dispositivos.

EXEMPLO

FIGURA 06 - Código do mestre

```
1.  #include <Wire.h>
2.
3.  bool estado_LED;
4.
5.  void setup() {
6.      Wire.begin();
7.  }
8.
9.  void loop() {
10.     Wire.beginTransmission(0x08);
11.     wire.write(estado_LED);
12.     Wire.endTransmission();
13.
14.     estado_LED = !estadoLED;
15.
16.     delay(1000);
17. }
```

Fonte: (GITHUB, 2023)

FIGURA 07 - Código do escravo

```
1.  #include <Wire.h>
2.
3.  void setup() {
4.      Wire.begin(0x08);
5.      Wire.onReceive(receiveEvent);
6.      pinMode(4,OUTPUT);
7.  }
8.
9.  void loop() {
10.     delay(100);
11.  }
12.
13.
14.  void receiveEvent(int leitura) {
15.
16.      bool estado = Wire.read();    // receive byte as an integer
17.
18.      if (estado == 1){
19.          digitalWrite(4,HIGH);
20.      }
21.      else{
22.          digitalWrite(4,LOW);
23.      }
24.  }
```

Fonte: (GITHUB, 2023)

O CI MAX2112

- O MAX2112 possui 14 registradores:
 - 12 registradores de escrita;
 - 2 registradores de leitura.

FIGURA 08 - Registradores do MAX2112

REG NUMBER	REGISTER NAME	READ/ WRITE	REG ADDRESS	MSB								LSB	
				DATA BYTE									
				D[7]	D[6]	D[5]	D[4]	D[3]	D[2]	D[1]	D[0]		
1	N-Divider MSB	Write	0x00	FRAC 1	N[14]	N[13]	N[12]	N[11]	N[10]	N[9]	N[8]		
2	N-Divider LSB	Write	0x01	N[7]	N[6]	N[5]	N[4]	N[3]	N[2]	N[1]	N[0]		
3	Charge Pump	Write	0x02	CPMP[1] 0	CPMP[0] 0	CPLIN[1] 0	CPLIN[0] 1	F[19]	F[18]	F[17]	F[16]		
4	F-Divider MSB	Write	0x03	F[15]	F[14]	F[13]	F[12]	F[11]	F[10]	F[9]	F[8]		
5	F-Divider LSB	Write	0x04	F[7]	F[6]	F[5]	F[4]	F[3]	F[2]	F[1]	F[0]		

Fonte: (DATASHEET , 2023)

O CI MAX2112

- O MAX2112 possui 14 registradores:

FIGURA 08 - Registradores do MAX2112

6	XTAL Buffer and Reference Divider	Write	0x05	XD[2]	XD[1]	XD[0]	R[4]	R[3]	R[2]	R[1]	R[0]
7	PLL	Write	0x06	D24	CPS	ICP	X	X	X	X	X
8	VCO	Write	0x07	VCO[4]	VCO[3]	VCO[2]	VCO[1]	VCO[0]	VAS	ADL	ADE
9	Lowpass Filter	Write	0x08	10010111							
10	Control	Write	0x09	STBY	X	PWDN 0	X	BBG[3]	BBG[2]	BBG[1]	BBG[0]
11	Shutdown	Write	0x0A	X	PLL 0	DIV 0	VCO 0	BB 0	RFMIX 0	RFVGA 0	FE 0
12	Test	Write	0x0B	CPTST[2] 0	CPTST[1] 0	CPTST[0] 0	X	TURBO 1	LD MUX[2] 0	LD MUX[1] 0	LD MUX[0] 0
13	Status Byte-1	Read	0x0C	POR	VASA	VASE	LD	X	X	X	X
14	Status Byte-2	Read	0x0D	VCOSBR[4]	VCOSBR[3]	VCOSBR[2]	VCOSBR[1]	VCOSBR[0]	ADC[2]	ADC[1]	ADC[0]

X = Don't care.

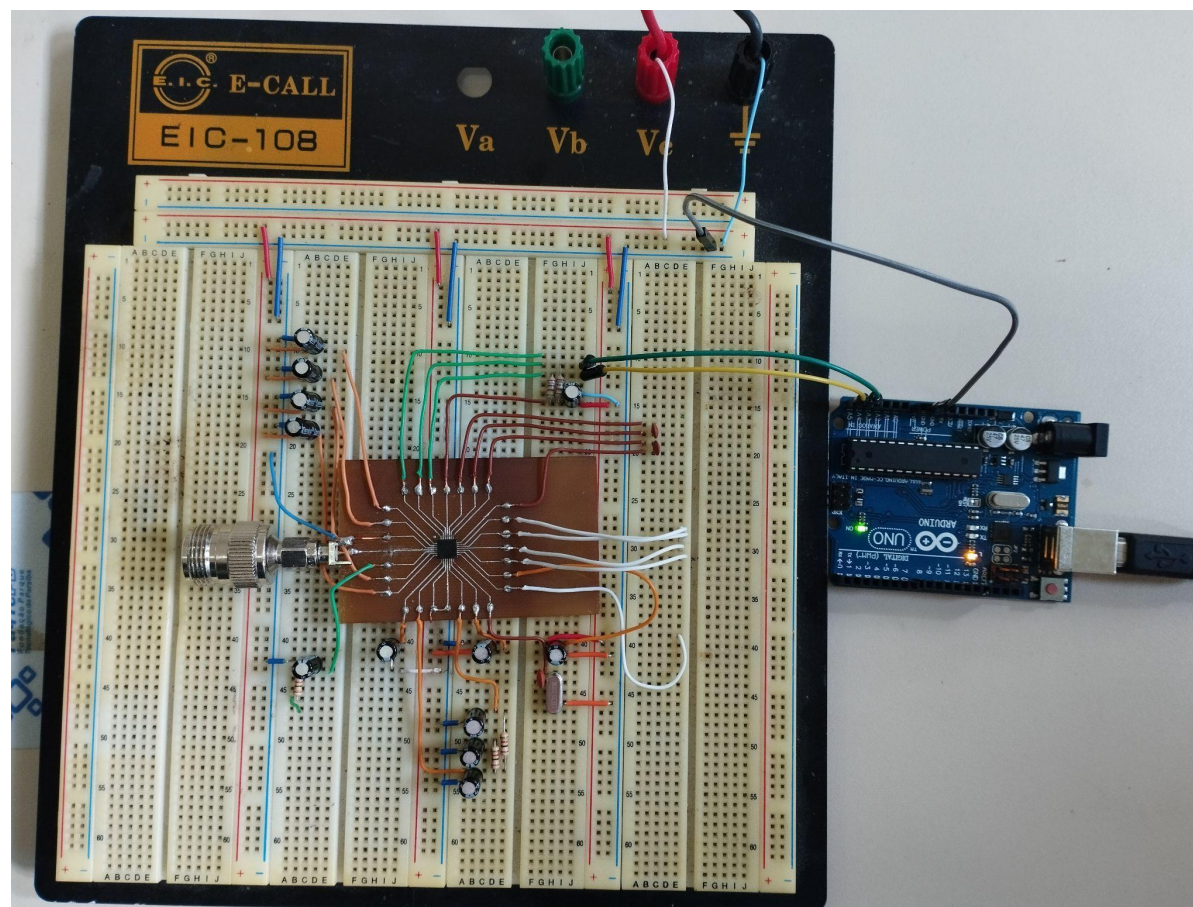
0 = Set to 0 for factory-tested operation.

1 = Set to 1 for factory-tested operation.

Fonte: (DATASHEET , 2023)

O CI MAX2112

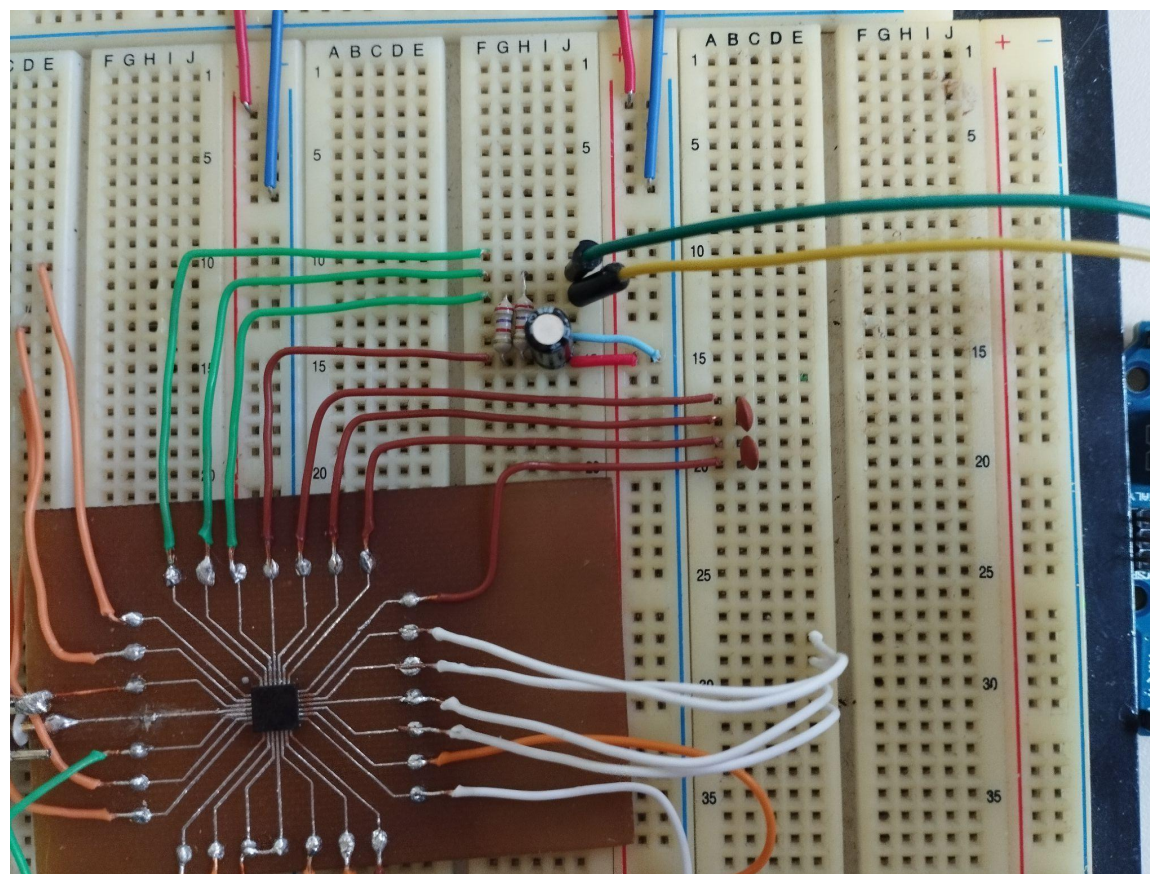
FIGURA 09 - Montagem do CI MAX 2112



Fonte: (PRÓPRIO AUTOR , 2023)

O CI MAX2112

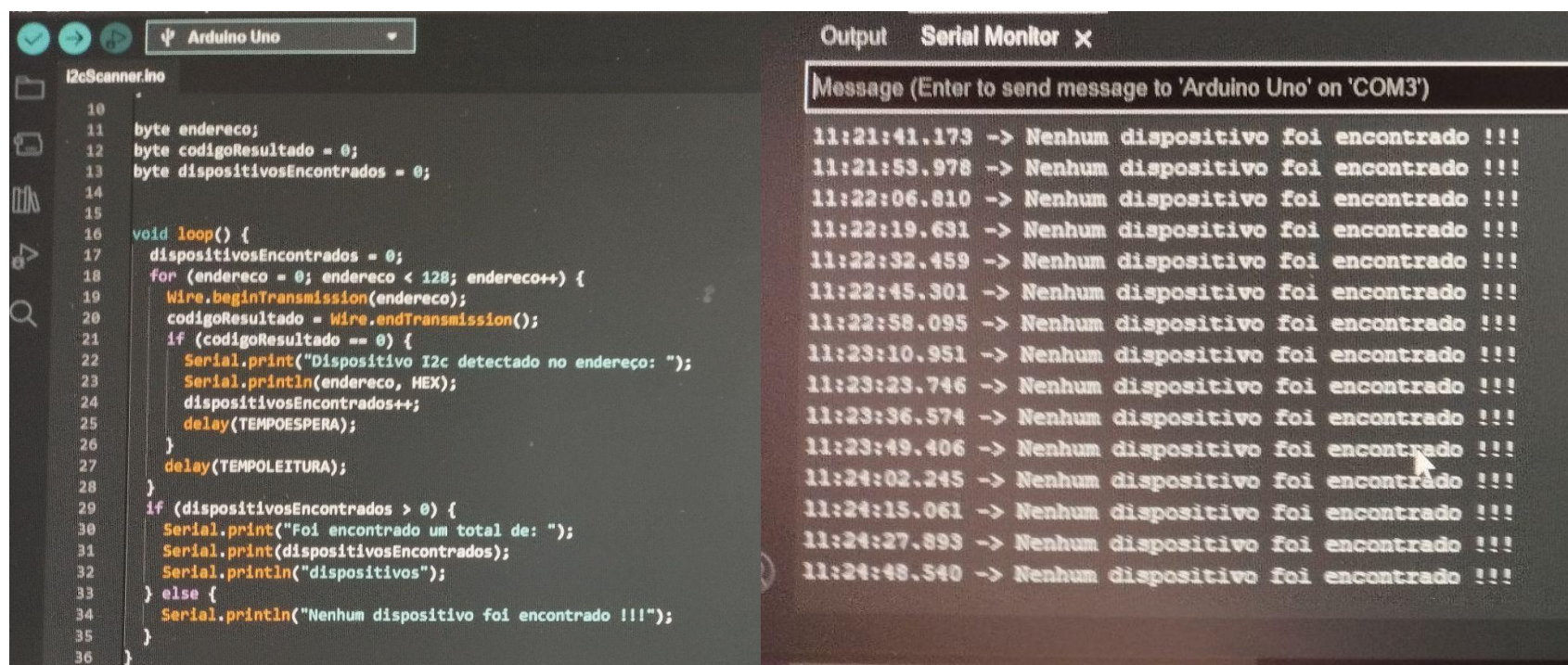
FIGURA 10 - Conexão das portas SDA e SCL



Fonte: (PRÓPRIO AUTOR , 2023)

O CI MAX2112

FIGURA 11 - Código para encontrar endereços de dispositivos



The image shows a screenshot of the Arduino IDE interface. On the left, the code for `I2cScanner.ino` is displayed. The code defines variables for address, result code, and count of devices found. It then enters a loop that scans addresses from 0 to 127. For each address, it sends a transmission and checks the result. If successful, it prints the address in hexadecimal and increments the count. After the loop, it prints the total count or a message if no devices were found. On the right, the Serial Monitor shows the output of the code, displaying a series of messages indicating that no devices were found at various addresses.

```
10
11 byte endereco;
12 byte codigoResultado = 0;
13 byte dispositivosEncontrados = 0;
14
15
16 void loop() {
17     dispositivosEncontrados = 0;
18     for (endereco = 0; endereco < 128; endereco++) {
19         Wire.beginTransmission(endereco);
20         codigoResultado = Wire.endTransmission();
21         if (codigoResultado == 0) {
22             Serial.print("Dispositivo I2c detectado no endereço: ");
23             Serial.println(endereco, HEX);
24             dispositivosEncontrados++;
25             delay(TEMPOESPERA);
26         }
27         delay(TEMPOLEITURA);
28     }
29     if (dispositivosEncontrados > 0) {
30         Serial.print("Foi encontrado um total de: ");
31         Serial.print(dispositivosEncontrados);
32         Serial.println("dispositivos");
33     } else {
34         Serial.println("Nenhum dispositivo foi encontrado !!!");
35     }
36 }
```

Output Serial Monitor X

Message (Enter to send message to 'Arduino Uno' on 'COM3')

11:21:41.173 -> Nenhum dispositivo foi encontrado !!!
11:21:53.978 -> Nenhum dispositivo foi encontrado !!!
11:22:06.810 -> Nenhum dispositivo foi encontrado !!!
11:22:19.631 -> Nenhum dispositivo foi encontrado !!!
11:22:32.459 -> Nenhum dispositivo foi encontrado !!!
11:22:45.301 -> Nenhum dispositivo foi encontrado !!!
11:22:58.095 -> Nenhum dispositivo foi encontrado !!!
11:23:10.951 -> Nenhum dispositivo foi encontrado !!!
11:23:23.746 -> Nenhum dispositivo foi encontrado !!!
11:23:36.574 -> Nenhum dispositivo foi encontrado !!!
11:23:49.406 -> Nenhum dispositivo foi encontrado !!!
11:24:02.245 -> Nenhum dispositivo foi encontrado !!!
11:24:15.061 -> Nenhum dispositivo foi encontrado !!!
11:24:27.893 -> Nenhum dispositivo foi encontrado !!!
11:24:40.725 -> Nenhum dispositivo foi encontrado !!!

Fonte: (PRÓPRIO AUTOR , 2023)

REFERÊNCIAS BIBLIOGRÁFICAS

<https://learn.adafruit.com/working-with-i2c-devices/terminology>

<https://embarcados.com.br/comunicacao-i2c/>

2023

2023

2023

<https://www.topgadget.com.br/howto/eletronica/visao-geral-e-beneficios-do-protocolo-e-barramento-i2c.htm>