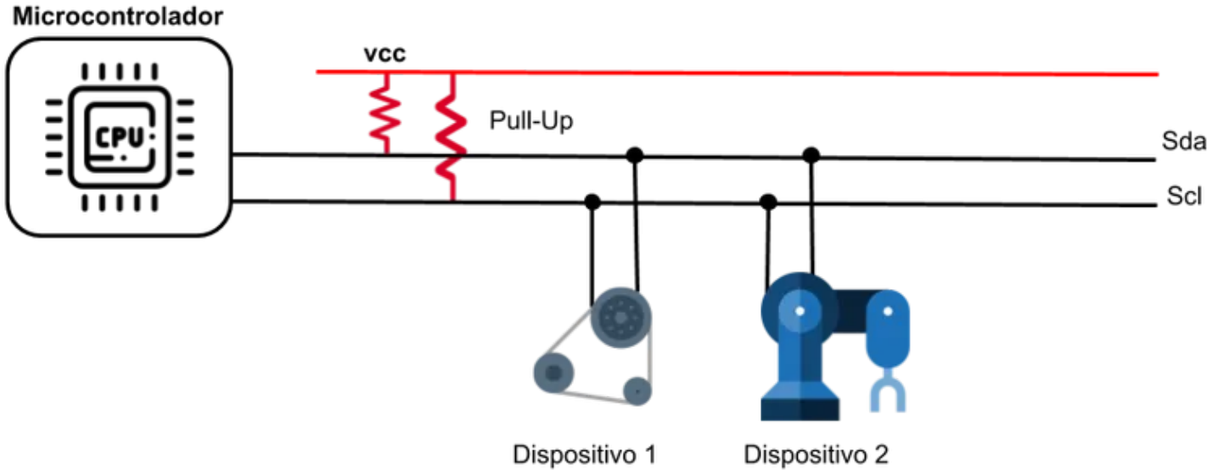
Descobrindo o endereço I2C de dispositivos com Arduino

Quando compramos equipamentos de lojas desconhecidas quase sempre nos deparamos com o problema de não termos disponível um datasheet ou algum tipo de documentação que auxilie na utilização dos mesmos. Sabendo disso, neste tutorial nós iremos aprender a **como identificar o endereço I2C de dispositivos** com um Arduino.

Protocolo I2C

O protocolo I2C, ou *Inter-Integrated Circuit* é um protocolo de comunicação desenvolvido pela Philips, que teve como objetivo inicial, interconectar dispositivos de baixa velocidade e curta distância. A comunicação i2C funciona por meio de um barramento onde apenas um dispositivo, denominado mestre, é responsável por requisitar informações dos dispositivos conectados. A conexão I2C é feita através de dois fios, sendo eles:

- **sda**: Transmitir dados entre receptor e transmissor via barramento.
- **scl**: Temporizar e sincronizar unidades conectadas ao sistema.



Protocolo I2C

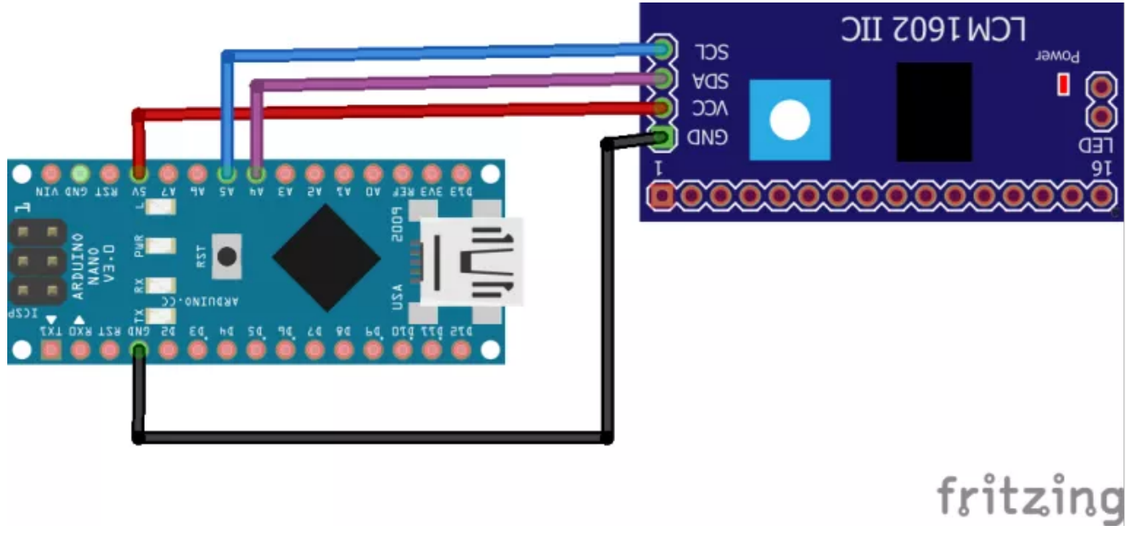
Como a comunicação I2C é feita via um barramento, cada dispositivo conectado ao sistema **deve** possuir um endereço único. Este código é composto por um valor de 7 bits, disponibilizando um total de 127 endereços onde apenas o intervalo 0x8 até o intervalo 0x77 está disponível para utilização.

Mãos à obra – Descobrindo o endereço I2C de dispositivos utilizando o seu Arduino

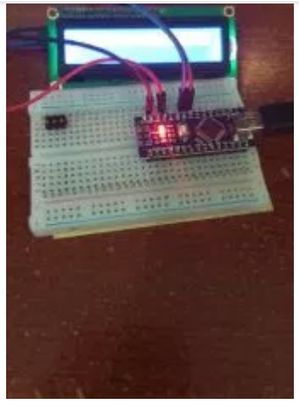
- 1x Arduino Nano
- 1x Display LCD 16×2 com Módulo I2C

Montando o projeto

Como teste do nosso código de pesquisa, iremos conectar um dispositivo I2C ao nosso Arduino Nano. Para fazer isso basta conectar o pino SDA ao pino A4 do Arduino e o pino SCL ao pino A5 do Arduino, além é claro dos pinos utilizados para alimentação. A figura abaixo ilustra o processo de ligação dos fios de comunicação e alimentação.



Veja como ficou a montagem na prática.



> **Atenção**, sempre monte o seu circuito com a alimentação do Arduino desligada. Para evitar possíveis curtos circuitos que possam vir a danificar o seu módulo ou até mesmo o microcontrolador.

Programando

Agora que temos conectado ao nosso sistema um módulo que supostamente não sabemos o seu endereço. Iremos utilizar carregar o seguinte código em nossa placa para sua identificação.

```
#include <Wire.h>
#define TEMPOLEITURA 100
#define TEMPOESPERA 3000
byte endereco;
byte codigoResultado=0;
```

```
  Wire.begin();
  while (!Serial);
  scanI2c();
}

void scanI2c(){
    for (endereco=0; endereco<128; endereco++){
    Wire.beginTransmission(endereco);
    codigoResultado = Wire.endTransmission();
    if (codigoResultado==0){
      Serial.print ("Dispositivo I2c detectado no endereço: ");
      Serial.println(endereco,HEX);
      dispositivosEncontrados++;
      delay(TEMPOESPERA);

    }
    delay(TEMPOLEITURA);
  }
  if (dispositivosEncontrados>0){
    Serial.print("Foi encontrado um total de: ");
    Serial.print(dispositivosEncontrados);
    Serial.println(" dispositivos");
  }
  else{
    Serial.println("Nenhum dispositivo foi encontrado !!!");
  }
}


void loop() {

}
```

Entendendo a fundo

– Incluindo as bibliotecas necessárias

De início, para ser possível a utilização da comunicação I2C no nosso Arduino, nós iremos precisar adicionar a biblioteca wire.h que pode ser feita através da seguinte sentença:

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

– Temporizadores de pesquisa e tempo de espera

Iremos também neste código criar dois temporizadores para definir de quanto em quantos milissegundos o nosso código deverá fazer a pesquisa em um novo endereço e por quanto tempo ele deve esperar após encontrar um dispositivo conectado ao barramento I2C. Para isso, iremos criar 2 diretivas define que serão responsáveis por armazenar estes valores, fazendo com que eles sejam facilmente parametrizáveis.

```
#define TEMPOLEITURA 100
#define TEMPOESPERA 3000
```

– Função setup

Na função setup, iremos inicializar a comunicação I2C para identificação de dispositivos e também iremos inicializar a comunicação serial, para que os dados sejam mostrados no serial monitor.

```
void setup() {
  Serial.begin(9600);
  Wire.begin();
  while (!Serial);
  scanI2c();
}
```

```
while (!Serial);
```

> Pois assim iremos garantir que a comunicação serial está de fato funcionando corretamente.

– Variáveis de controle

Para deslocarmos entre os endereços, identificarmos se o dispositivo está realmente conectado ao barramento e contar quantos dispositivos estão conectados, nós iremos utilizar três variáveis do tipo **byte**. A utilização destas variáveis com este tipo deve-se ao fato de que como estes valores nunca serão maiores que 255(,) não existe a possibilidade de overflow. Sendo assim iremos economizar cerca de 3 bytes de memória, se utilizássemos por exemplo o tipo int para armazenar.

```
byte endereco;
byte codigoResultado=0;
byte dispositivosEncontrados=0;
```

– Função scanI2c

Dentro da função scanI2c nós iremos realizar todo o processo de busca por dispositivos conectados, que será feito da seguinte forma:

1. Através do comando for, percorrer todos os endereços disponíveis no protocolo I2C, partindo do 0 até o 127.

```
for (endereco=0; endereco<128; endereco++);
```

```
1 Wire.beginTransmission(endereco);
```

3. Logo em seguida iremos encerrar a transmissão através do método **Wire.endTransmission**. Este método possui um retorno que consiste em:
   1. Igual a 0 – Conexão fechada com sucesso
   2. Diferente de 0 – Falha na transmissão ou não existe dispositivo conectado
4. Sabendo disso, iremos armazenar este valor de retorno em nossa variável **codigoResultado**

```
1 codigoResultado = Wire.endTransmission();
2
```

5. Logo em seguida verificamos se a variável codigoResultado é igual a zero, caso seja mostramos o código do dispositivo e contabilizamos a variável que mostra o total de dispositivos encontrados e aguardamos o tempo em milissegundos definido em TEMPOESPERA

```
1 if (codigoResultado==0){
2  Serial.print ("Dispositivo I2c detectado no endereço: ");
3  Serial.println(endereco,HEX);
4  dispositivosEncontrados++;
5  delay(TEMPOESPERA);
6 }
```

6. Após cada tentativa de leitura também aguardamos um intervalo de tempo definido em **TEMPOLEITURA.**

```
1 delay(TEMPOLEITURA);
2
```

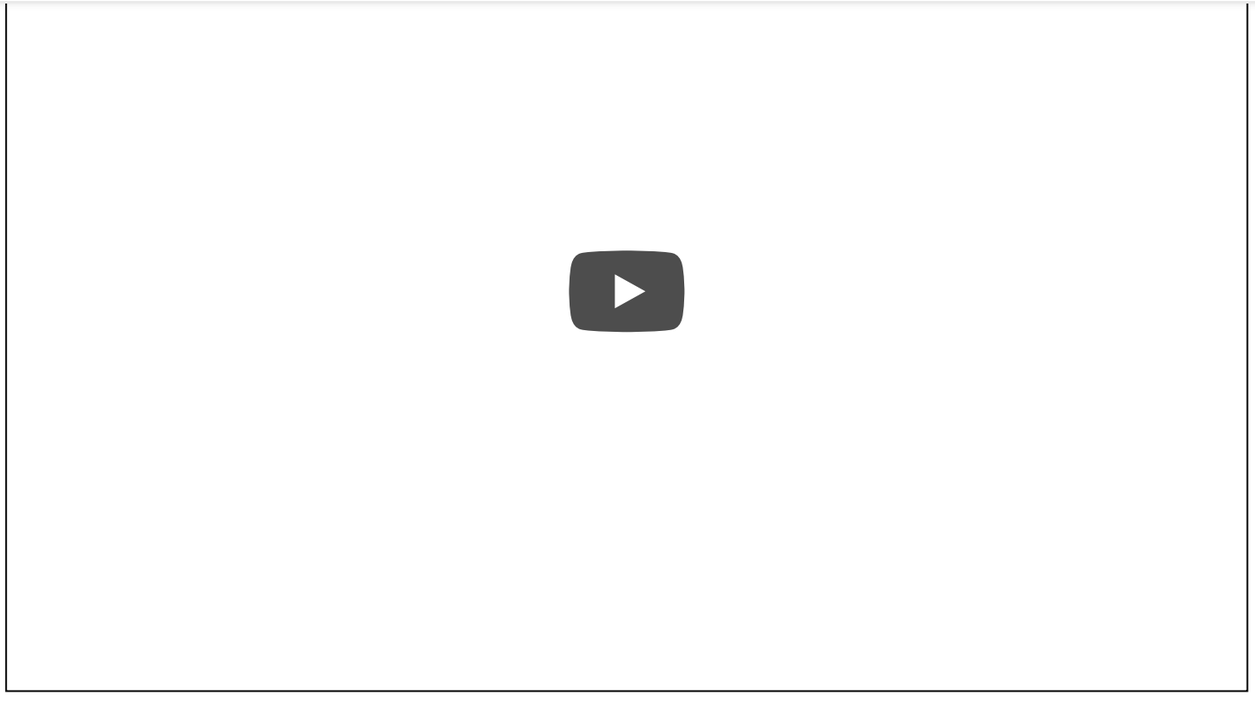
7. Ao fim do for verificamos se o número de dispositivos encontrados é maior que zero, caso seja mostramos este valor na serial, caso contrário informamos que nenhum dispositivo foi encontrado.

```
1 if (dispositivosEncontrados>0){
2   Serial.print("Foi encontrado um total de: ");
```

```
7     Serial.println("Nenhum dispositivo foi encontrado !!!");
8  }
```

Vídeo Tutorial

Vocês também podem ver uma explicação do código em vídeo no canal abaixo:



Desafio

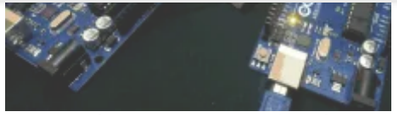
Agora que você sabe como descobrir o endereço de dispotivos I2C. Tente conectar mais de um ao barramento e veja se o código consegue detectar todos. Tente também criar uma lógica para salvar o endereço dos dispositivos identificados pelo sistema.

Este tutorial, teve como objetivo mostrar uma forma bem simples para encontrar o endereço de dispositivos conectados via I2C. Espero que tenham gostado do conteúdo apresentado, sinta-se à vontade para nos dar sugestões, críticas ou elogios. Lembre-se de deixar suas dúvidas nos comentários abaixo.

Caso queira saber mais sobre o Display LCD 16×2 com Adaptador I2C, acesse o tutorial: Display LCD 20×4 e LCD 16×2 com Adaptador I2C – Utilizando o display com Arduino

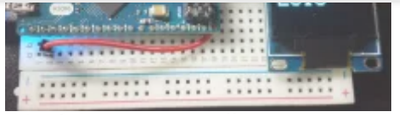
Danilo Almeida

Formado em Ciência da computação pela UFV-CAF em 2017 e atualmente cursando pós-graduação Stricto Sensu em Ciência da computação pela Universidade Federal de Viçosa, na área de arquitetura de computadores. É um entusiasta na área de sistemas embarcados e robótica.



Protocolo I2C - Comunicação entre Arduinos
4 de dezembro de 2017
Em "Comunicação"

Display LCD 20x4 e 16x2 com Adaptador I2C
22 de setembro de 2017
Em "Displays"



Display OLED 0.96" I2C com Arduino
17 de fevereiro de 2018
Em "Displays"

4 COMMENTS

Thomé Lucas REPLY

1 de janeiro de 2019, 17:34

Muito legal e muito bem explicado viu! Parabéns!

Mario Cesar REPLY

22 de junho de 2019, 01:04

Mestre parabéns pelo método fiquei muito nsentivado, não sou técnico em eletrônica formado mais Tenho um pouquinho de experiência e me esforço.

baseado nas linhas de Lissajus, porém não é conclusivo, pq não há variações nas elipses, somente se haverá um falha de capacitância na linha.

Mario Cesar REPLY

22 de junho de 2019, 01:07

Corrigindo somente haverá alterações se possuir um falha de de capacitância na linha.

Carol REPLY

30 de julho de 2019, 19:27

Estou tentando criar um sistema conectando dois sensores de temperatura infravermelho mlx60914 ao arduino. Porém, por padrão de fábrica, vieram como o mesmo endereço e, pra realizar a comunicação i2c preciso alterar pelo menos um desses endereços, você sabe como posso fazer isso?

DEIXE UMA PERGUNTA, SUGESTÃO OU ELOGIO! ESTAMOS ANSIOSOS PARA TER OUVIR!

Digite seu comentário aqui...