Trabalho prático de Arquitetura de Computadores

Maio/2025

Trabalho prático de Arquitetura de Computadores

Curso: Ciências de Dados & Inteligência Artificial

Nome: Alexandre Albuquerque

N.º de Est: 1708170

Curso: Ciências de Dados & Inteligência Artificial

Nome: Manuel Fuele Kiangebeni

N.º de Est: 1708164

Professor: Dr. Luís Figueiredo

Maio/2025

índice

[1. O problema 3](#_Toc197375875)

[2. Introdução 5](#_Toc197375876)

[3. Componentes Utilizados 7](#_Toc197375877)

[4. Solução 9](#_Toc197375878)

[4.1. Algoritmo 9](#_Toc197375879)

[4.2. Esquema 10](#_Toc197375880)

[4.3. Código 11](#_Toc197375881)

[5. Desafios 13](#_Toc197375882)

[6. Sugestão 15](#_Toc197375883)

[7. Conclusão 17](#_Toc197375884)

[Referências 19](#_Toc197375885)

1. O problema

Usando o sonar HC-SR04 controlado com funções de interrupt, controlar o número de leds acesos de um conjunto de 8 usando a seguinte escala:

a. Distâncias inferiores a 5cm: 0 leds acesos;

b. Distâncias entre 5 cm e 100cm: número de leds acesos proporcional à distância (5cm acende um led, 100 ou mais centímetros acendem os 8 leds).

1. Introdução

Este relatório é resultado do trabalho prático da cadeira de Arquitetura de Computadores, onde apresentamos a solução obtida para o problema, os desafios e a conclusão alcançada.

No capitulo 3 iremos apresentar o material utilizado para a realização deste trabalho.

No capítulo 4 vamos apresentar a solução encontrada para a resolução do problema. No seu subcapítulo vamos primeiro apresentar o algoritmo utilizado para a elaboração do código e o seu respetivo esquema.

No capítulo 5 abordamos o tema dos desafios encontrados ao longo da resolução do problema. Iremos falar sobre as dificuldades e como foram ultrapassadas.

No capitulo 6 vamos fornecer uma breve sugestão de melhoria.

Para o capítulo 7, iremos oferecer uma breve conclusão sobre a solução do problema.

1. Componentes Utilizados
2. HC-SR04 – Sensor Ultrassónico;
3. ESP32 Node;
4. Resistências:

* 8 de 470 Ω
* 1 de 1kΩ
* 1 de 2kΩ

1. 8 LEDs;
2. 13 Jumpers
3. 1 Breadboard
4. 1 Cabo Micro USB
5. Solução

A solução é composta por duas partes, o algoritmo – os caminhos seguidos na perspetiva do ESP32 e o código – logica e estruturas de programação seguidas na perspetiva do programador.

## Algoritmo

**Objetivo:**

Utilizar o sonar HC-SR04, para controlar o número de leds acesos recorrendo a funções interrupt.

**Algoritmo da Função distanceRead:**

1. Se o numLEDs for igual a zero;
   1. Desligar todos os LEDs;
2. Se não se o numLEDs for maior ou igual a oito;
   1. numLEDs igual a oito;
   2. Ligar todos os LEDs;
3. Se não;
   1. Ligar numLEDs LEDs proporcionalmente à distância;
   2. Desligar numLEDs LEDs proporcionalmente à distância.

**Algoritmo Setup:**

1. Estabelecer a ligação entre o microcontrolador e o PC a uma velocidade de 115.200bps;
2. Configurar o array pins de 8 caracteres;
   1. Configurar pino 13 como OUTPUT no array pins;
   2. Configurar pino 12 como OUTPUT no array pins;
   3. Configurar pino 14 como OUTPUT no array pins;
   4. Configurar pino 27 como OUTPUT no array pins;
   5. Configurar pino 26 como OUTPUT no array pins;
   6. Configurar pino 25 como OUTPUT no array pins;
   7. Configurar pino 33 como OUTPUT no array pins;
   8. Configurar pino 32 como OUTPUT no array pins;
   9. Desligar todos os LEDs;
3. Configurar pino 2 como OUTPUT;
4. Configurar pino 15 como INPUT;
5. Configurar pino echoPin à função distanceRead, no modo FALLING;
6. Ler startTime.

**Algoritmo Loop:**

1. Ler currentTime;
2. Se o currentTime-startTime for maior ou igual a 50 milisegundos;
   1. Desligar o trigPin;
   2. Se o currentTime-startTime for maior ou igual a 0.002 milisegundos;
      1. Ligar o trigPin;
      2. Adicionar o valor 0.002 ao startTime;
         1. Se o currentTime-startTime for maior ou igual a 0.01 milisegundos;
            1. Desligar o trigPin;
            2. Ligar o echoPin;
            3. Imprimir o valor da distância e o numLEDs;
   3. Adicionar ao startTime o valor 50.

## Esquema

Uma imagem com eletrónica, texto, circuito, Engenharia eletrónica

Os conteúdos gerados por IA poderão estar incorretos.

## Código

/////////////////////////////

//Trabalho Final Proposta 2//

/////////////////////////////

//Variáveis//

unsigned long startTime**=**0**,** currentTime**=**0**;**

bool bright**=false;**

long duration**;**

int distance**=**0**,** pins**[**8**]={**13**,**12**,**14**,**27**,**26**,**25**,**33**,**32**};**

volatile int numLEDs**=**0**;**

//Define//

#define trigPin 2 // OUTPUT

#define echoPin 15 //INPUT

//Função Interrupt//

void distanceRead**(){**

numLEDs**=(**0.08**\***distance**)+**0.6**;** // Formula para ligar leds proporcionalmente à distância

**if(**numLEDs**==**0**){**

**for(**int i**=**0**;**i**<**8**;**i**++){**

digitalWrite**(**pins**[**i**],** bright**);**

**}**

**}else** **if(**numLEDs**>=**8**){**

numLEDs**=**8**;**

**for(**int i**=**0**;**i**<**numLEDs**;**i**++){**

digitalWrite**(**pins**[**i**],** **!**bright**);**

**}**

**}else{**

**for(**int i**=**0**;**i**<**numLEDs**;**i**++){**

digitalWrite**(**pins**[**i**],** **!**bright**);**

**}**

**for(**int i**=**7**;** i**>=**numLEDs**;** i**--){**

digitalWrite**(**pins**[**i**],** bright**);**

**}**

**}**

**}**

void setup**(){**

Serial**.**begin**(**115200**);**

**for** **(**int i **=** 0**;** i **<** 8**;** i**++)** **{**

pinMode**(**pins**[**i**],** OUTPUT**);**

digitalWrite**(**pins**[**i**],** bright**);**

**}**

pinMode**(**trigPin**,** OUTPUT**);**

pinMode**(**echoPin**,** INPUT**);**

attachInterrupt**(**echoPin**,** distanceRead**,** FALLING**);**

startTime**=**millis**();**

**}**

void loop**(){**

currentTime**=**millis**();**

**if(**currentTime**-**startTime**>=**50**){**

digitalWrite**(**trigPin**,** LOW**);**

**if(**currentTime**-**startTime**>=**0.002**){** // Manter o trigger no LOW por 2 microsegundos para limpar ruidos ou valores anteriores

digitalWrite**(**trigPin**,** HIGH**);**

startTime**+=**0.002**;**

**if(**currentTime**-**startTime**>=**0.01**){** // Enviar um ultrasom por 10 microsegundos pois é o requerido para funcionar

digitalWrite**(**trigPin**,** LOW**);** // Terminar o ultrasom

duration **=** pulseIn**(**echoPin**,** HIGH**);**

distance **=** duration **\*** 0.034 **/** 2**;**

startTime**+=**0.01**;**

printf**(**"%d - %d\n"**,** distance**,** numLEDs**);**

**}**

**}**

startTime**+=**50**;**

**}**

**}**

1. Desafios
2. Calcular o número de LEDs proporcionalmente ao valor da distância

Para uma eficiência no cálculo, definimos a equação da reta tangente, y = m**x** + b, com **x** igual á distância e **y** igual ao número de LEDs.

No cálculo da reta tangente, desenhamos um gráfico:

Recorremos ao seguinte cálculo para encontrar o m: **m** = (**y**2-**y**1)/(**x**2-**x**1) <=> **m** = 7/90 <=> **m** = 0.08.

A seguir calculamos o valor de b: **b** = -0.08**\*x** + **y** <=> **b** = 0.6.

Concluindo então com a fórmula: **y** = 0.08**\*x** + 0.6.

1. Utilizar o **y** para ligar o número exato de LEDs

Depois de encontrar a equação da reta tangente, precisamos de integrar o valor do **y** na lógica do problema. Para tal, foi utilizado um array “pins” que continha o número dos pinos associados a cada LED, pois, podemos utilizar um ciclo para percorrer o array e, assim, acender o número exato de LEDs.

1. Problemas com exatidão nos valores de **y**

Assim que concluído o desafio anterior, deparamo-nos com o seguinte problema:

* Para y = 0, o que significa zero LEDs acesos, a lógica do algoritmo iria para um LED aceso, pois o y igual a zero significa a posição zero no array;
* Para y >= 8, como a equação encontrada, tal como descrito no problema, está para cinco centímetros até cem centímetros, para distâncias superiores o valor de y iria ser maior que oito, ou seja, tentar acender mais LEDs do que os existentes.

Para solucionar ambos os desafios, foram criadas duas verificações. A primeira iria verificar se o y fosse zero e caso o confirmasse, os LEDs iriam ser todos apagados. Na segunda verificação, se o y tivesse valores iguais ou superiores a oito, iria ter obrigatoriamente o valor oito.

1. Não utilizar ciclos dentro da função loop

Na correção, foi mais eficaz evitar ciclos dentro da função loop, sendo estas transferidas para a função **interrupt**.

1. LEDs a piscar de forma constante

Durante a execução do código, notamos que os LEDs piscavam de forma constante, pois estávamos a desligar os LEDs no início da função loop. Para solucionar o problema, definimos dois ciclos na função interrupt:

* O primeiro ciclo, teve como objetivo ligar os LEDs proporcionalmente à distância, ou seja, percorrer as posições do array de forma crescente;
* O segundo ciclo, parecido ao anterior, teve como objetivo desligar os LEDs proporcionalmente à distância, ou seja, percorrer as posições do array de forma decrescente.

1. Sugestão

Com o objetivo de minimizar a função interrupt, foi pensado a criação de uma função apenas para executar os ciclos presentes no mesmo. Pois na função loop, seria mais eficaz chamar a função com os ciclos, do que executar os mesmos na função interrupt.

1. Conclusão

Na realização deste trabalho, tivemos a oportunidade de aplicar os conhecimentos adquiridos na aula e a importância de integração entre hardware e software no contexto de Arquitetura de Computadores.

Utilizando o sensor HC-SR04 e o microcontrolador ESP32, foi possível desenvolver um sistema funcional que controla a quantidade de LEDs acesos de forma proporcional à distância medida. Durante o desenvolvimento, enfrentamos diversos desafios, como o cálculo proporcional da distância em relação ao número de LEDs, o uso eficaz da função interrupt, e o controlo preciso dos ciclos que ligam e desligam os LEDs. Cada obstáculo contribuiu para uma aprendizagem mais profunda sobre manipulação de sinais, estruturas de controle e eficiência do código.

Em suma, o projeto proporcionou uma experiência prática valiosa, permitindo consolidar conceitos fundamentais da disciplina e desenvolver competências técnicas relevantes para o nosso percurso académico.

# Referências

Tutorials, R. N. (01 de Maio de 2025). *Complete Guide for Ultrasonic Sensor HC-SR04 with Arduino*. Obtido de Random Nerd Tutorials: https://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/