Escola Politécnica de Pernambuco - UPE

Relatório de Sequenciais:

Sinal Inteligente

Professor orientador: Sérgio Murilo

Grupo: Andreyson Carlos

Daniel Vasconcellos

Eduardo Muller

Larissa Perboire

Recife, 2017

**Sumário:**

[**Introdução:**](#_5mhrcyq8dso6) **3**

[**2.Caracterização do problema**](#_vepqchaspnul) **4**

[**3. Objetivos e Metas:**](#_awenc7afkqed) **5**

[**Metodologia:**](#_38n9ynkhiq1n) **5**

[**Diagrama de estados:**](#_q97ml8yrmvj) **6**

[**Implementação dos circuitos:**](#_92zz1oqqt5gy) **6**

[**Montagem da placa :**](#_up60pbdmrkgl) **12**

[**Materiais utilizados:**](#_3bdkchy6fej5) **13**

[**Dificuldades:**](#_j7qjmg50te8q) **13**

[**Conclusão:**](#_4mzx2pxu9crp) **13**

[**Planos para o futuro do projeto:**](#_tfvkj46cui4v) **14**

[**Apêndice:**](#_j1sjsnpmxbp5) **14**

# Introdução:

Tendo em vista a dificuldade de trafegar pela cidade de forma rápida por conta do trânsito e a baixa acessibilidade que existe para atravessar as ruas da mesma, este projeto se propõe a implementar um sinal inteligente que fabrique ondas verde em um lado do cruzamento e vermelha no outro de acordo com o histórico de fluxo no local. Além de possuir um botão que solicita mais tempo para atravessar e emitir um alerta sonoro enquanto o sinal está fechado para que um deficiente visual saiba que pode atravessar.

Inicialmente, iríamos implementar um contador feito com flip-flop, para ativar os sinais, no entanto não conseguimos encontrar tais componentes para comprar, devido a isso implementamos no Proteus o circuito com os flip-flops, e na placa implementaríamos o microcontrolador, no entanto devido a falta de alguns materiais não conseguimos, então montamos o projeto na Protoboard.

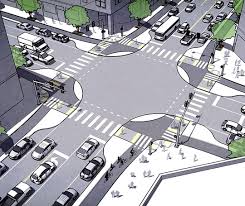


figura 1. Imagem para ilustrar o trânsito

# 2.Caracterização do problema

É fato que um grande problema presente na maioria dos centros urbanos é o congestionamento de carros, que leva as pessoas perderem muito tempo para se locomoverem, gerando estresse, além de desperdício de um tempo que poderia ser utilizado em atividades úteis.

O relatório anual com informações detalhadas sobre as cidades com os maiores congestionamentos do mundo, produzido pela empresa holandesa TomTom e divulgado em 27 de fevereiro de 2017, revela que a cidade do Recife é, no Brasil, a terceira que mais deixa o motorista parado no trânsito, e a 43ª no ranking mundial, com uma média de 37%. Ainda segundo a pesquisa, o engarrafamento faz o motorista levar 60% mais tempo durante a manhã e 65% mais no fim da tarde.

Outra dificuldade encontrada foi a circulação de deficientes visuais em cidades, posto que eles não conseguem saber se um sinal está aberto ou fechado para que possam atravessar em segurança.

No que se refere especificamente às pessoas com deficiência visual, existe “lei” (resolução nº 704 de 10 de outubro de 2017) que menciona a necessidade de implantação de dispositivos sonoros – intermitente e não estridente -, nos semáforos para orientação na travessia de ruas e avenidas com trânsito intenso, mas estas determinações acabam tendo vida curta.

Esses equipamentos não auxiliam apenas a cegos ou quem tem baixa visão, mas também são importantes para pedestres distraídos e idosos, ao passo que,somados sinalização visual e alerta sonoro, é possível melhorar a segurança de todos.

# 3. Objetivos e Metas:

Tendo em vista o exposto acima este projeto tem por objetivo:

* Construir um sinais inteligentes que em um cruzamento gere uma onda verde (ação que possibilita um tráfego maior de carros) em um lado e uma onda vermelha no outro (ação que não favorece um grande tráfego de carros).
* Emitir um alerta sonoro toda vez que o sinal fechar para que deficientes visuais saibam que podem atravessar em segurança.
* Implementar um botão que solicite mais tempo para atravessar caso o transeunte possua alguma deficiência motora.

# Metodologia:

Para atingir o objetivo determinado, nós:

* Nos reunimos para simular este projeto no software Proteus e fazer o esquemático.
* A partir da simulação descobrir quais componentes iremos precisar e comprá-los.
* Construir a placa com os componentes comprados.
* Programar o microcontrolador e integrá-lo com a placa construída.

# Diagrama de estados:

O diagrama de estados está descrito segundo a figura 2, representada a seguir:

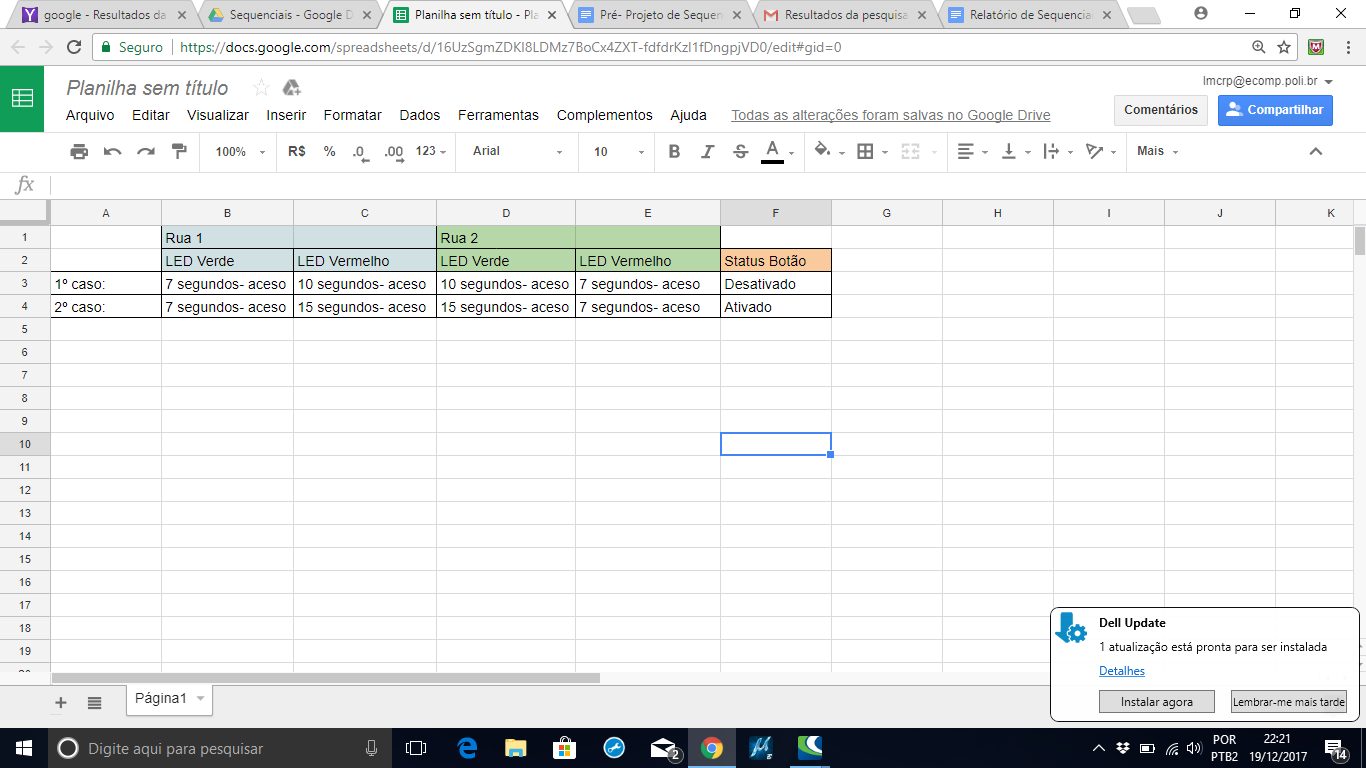


figura 2. Imagem para ilustrar o diagrama de estados

# Implementação dos circuitos:

* Implementação do projeto utilizando contador com Flip-Flops:

Os flip-flops poderiam ser utilizados como contador de 7 segundos, então a cada 7 segundos o microcontrolador receberia um input, que seria administrado pelo mesmo.

O flip-flop a ser utilizado seria o do tipo T, cuja implementação pode ser verificados a seguir, na figura 3:

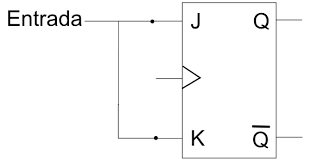


figura 3. Imagem para ilustrar o flip-flop

Tendo em vista sua montagem e a tabela verdade do Flip-Flop J-K para esta ligação, temos a tabela verdade do Flip-Flop tipo T:

|  |  |  |
| --- | --- | --- |
| **J** | **K** | **Qf** |
| 0 | 0 | Qa |
| 0 | 1 | - |
| 1 | 0 | - |
| 1 | 1 | Qa’ |

|  |  |
| --- | --- |
| **T** | **Qf** |
| 0 | Qa |
| 1 | Qa’ |

Com isso podemos montar o contador utilizando flip-flops, basta estruturá-los desta forma da forma descrita na figura 4:

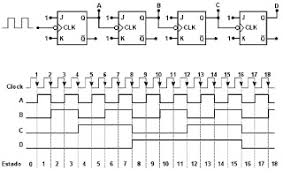


Figura 4. Imagem para ilustrar a estruturação dos flip-flops

De acordo com o diagrama do tempo apresentado acima podemos perceber que essa montagem funciona como contador.

Com base no que foi exposto, iremos desenvolvermos um contador de 7,10 e 15 segundos, iremos utilizar este tempo curto para fins demonstrativos, o primeiro representa o tempo que o sinal ficará fechado em uma onda verde, o segundo o tempo fechado quando o botão é acionado e o último é o tempo aberto.

Para isso, fizemos a seguinte montagem, representada na figura 5:

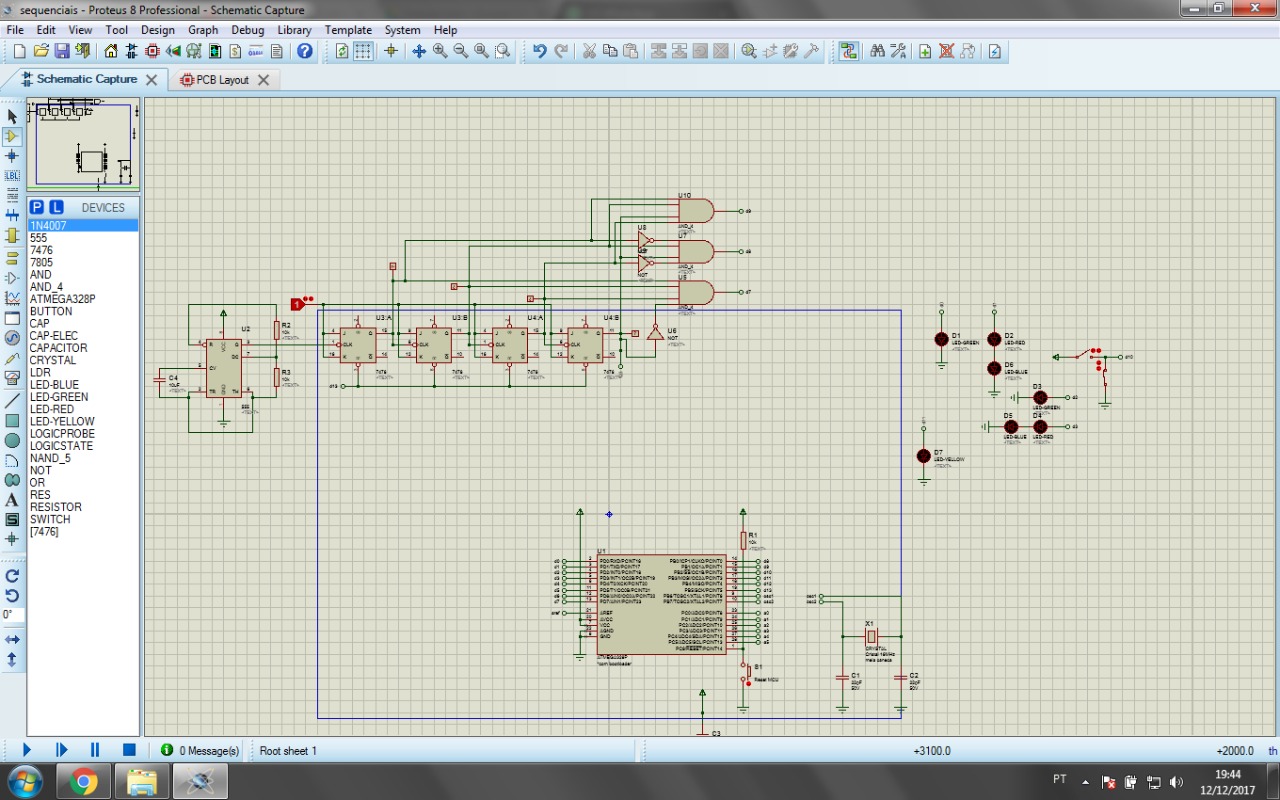


Figura 5. Imagem para ilustrar o contador com flip-flops.

A primeira AND indica a contagem de 15 segundos, pois o número 15 escrito em binário é 1111, logo quando o contador chegar em 15 as saídas Q0,Q1,Q2 e Q3 serão 1111, quando as mesmas são ligadas em um AND a saída vai ser 1, indicando assim a contagem de 15 segundos.

A segunda AND indica a contagem de 10 segundos, pois o número 10 escrito em binário é 0101, logo quando o contador chegar em 10 as saídas Q0, Q1,Q2 e Q3 serão respectivamente 0101, ao ligarmos as saídas Q0 e Q2 em um inversor e em seguida em uma AND junto com as saídas Q1 e Q3, ao chegar a 10 segundos a saída será 1, indicando assim a contagem de 10 segundos.

A terceira AND indica a contagem de 7 segundos, pois o número 7 escrito em binário é 1110, logo quando o contador chegar em 10 as saídas Q0, Q1, Q2 e Q3 serão respectivamente 1110 se ligarmos um inversor a saída Q3 e em seguida ligar a uma AND junto com as demais saídas, ao chegar em 7 segundos a saída da AND erá 1, indicando a contagem de 7 segundos.

O clear dado nos flip-flops é disparado pelo microcontrolador.

Para a entrada de clock no flip-flop T0 usamos um 555, como pode ser percebido na imagem seguir, representada pela figura 6:

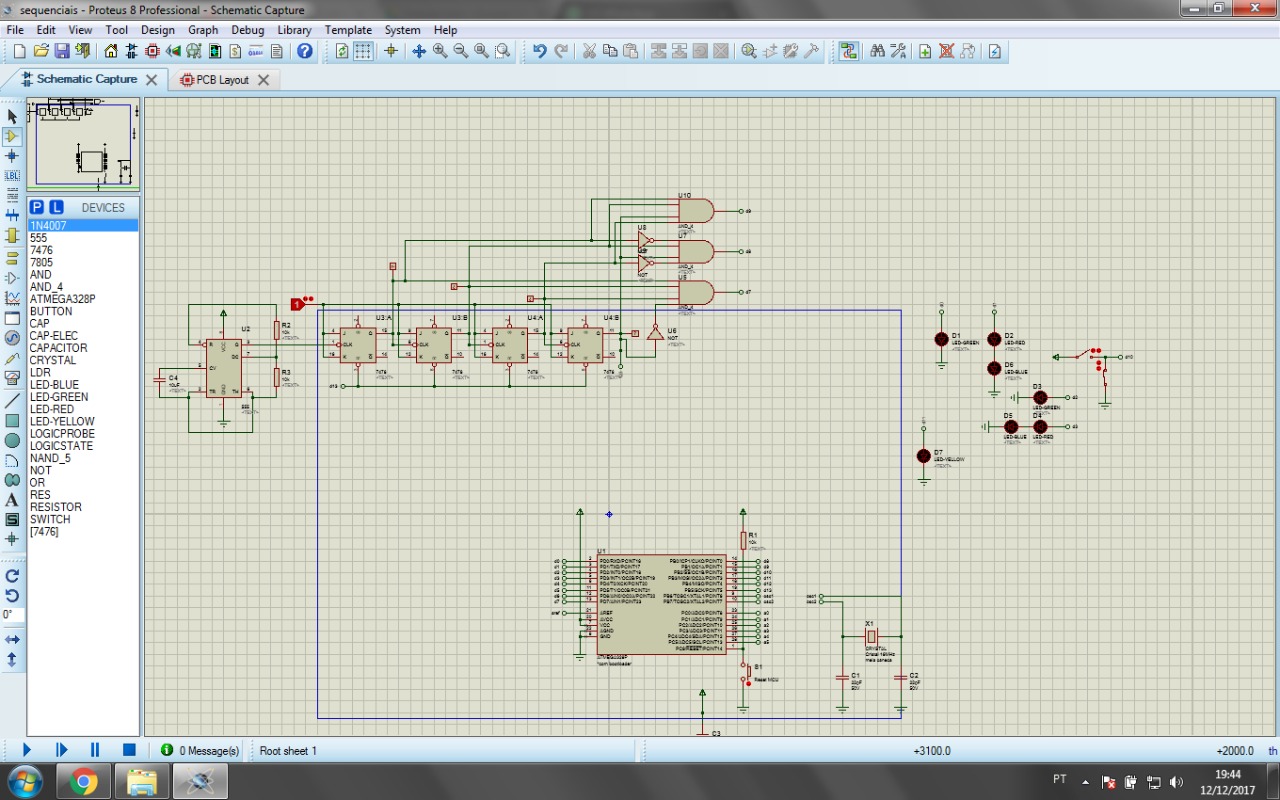


figura 6

Na figura acima podemos perceber o componente 555 dando o clock no flip-flop T0, a frequência do clock é determinada por esta equação:

Formula frequência 555

Após a montagem do contador, ligamos as saídas das ANDs ao microcontrolador e suas saídas aos LEDs, desta forma na figura 7:

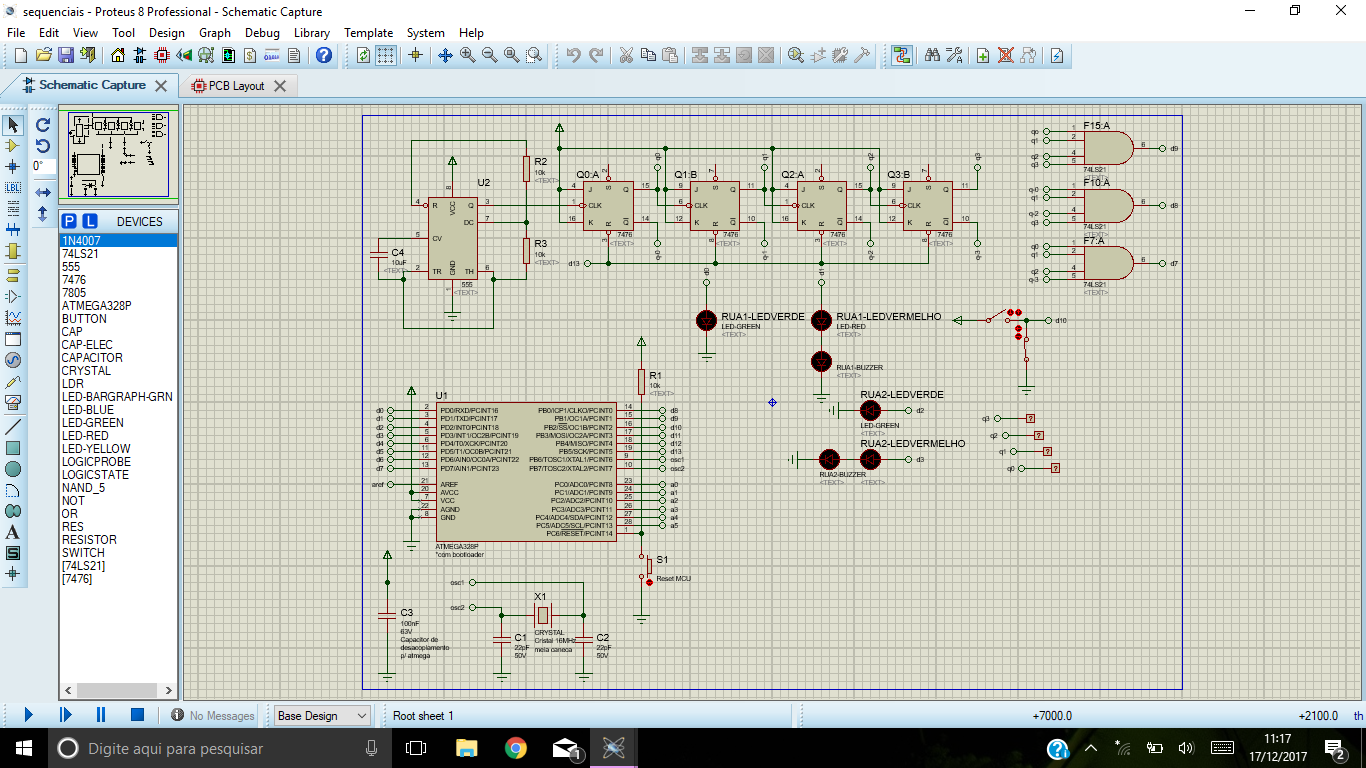


figura 7. Imagem para ilustrar a simulação do projeto

Na simulação utilizamos LEDs azul para indicar o funcionamento do buzzer uma chave para indicar o funcionamento do botão.

Código do microcontrolador, para a simulação com Flip-Flops se encontra no apêndice na primeira seção.

* Implementação do arduino:

Para montar o arduino iremos utilizar o regulador de tensão Lm7805, o microcontrolador atmega para controlar os LEDs além de acionar os buzzers e receber o input do botão, um socket para o atmega, um cristal para dar o clock no atmega, capacitores para o cristal, leds, botão e buzzers, tal implementação pode ser verificada na figura 8.

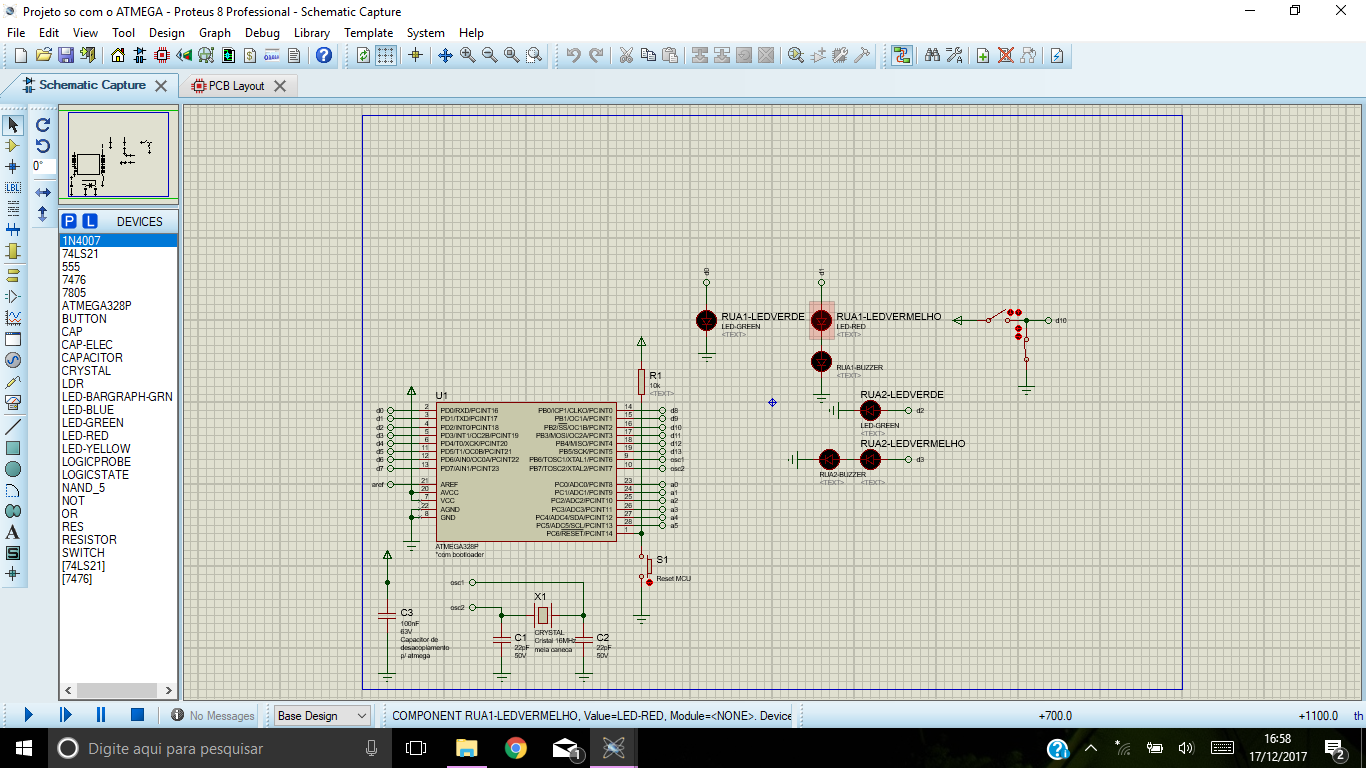


figura 8

A PCB deste circuito é necessária para entendermos a disposição dos componentes na montagem da placa, além de montarmos as ligações da melhor forma possível. Tal montagem pode ser verificada a seguir na figura 9:

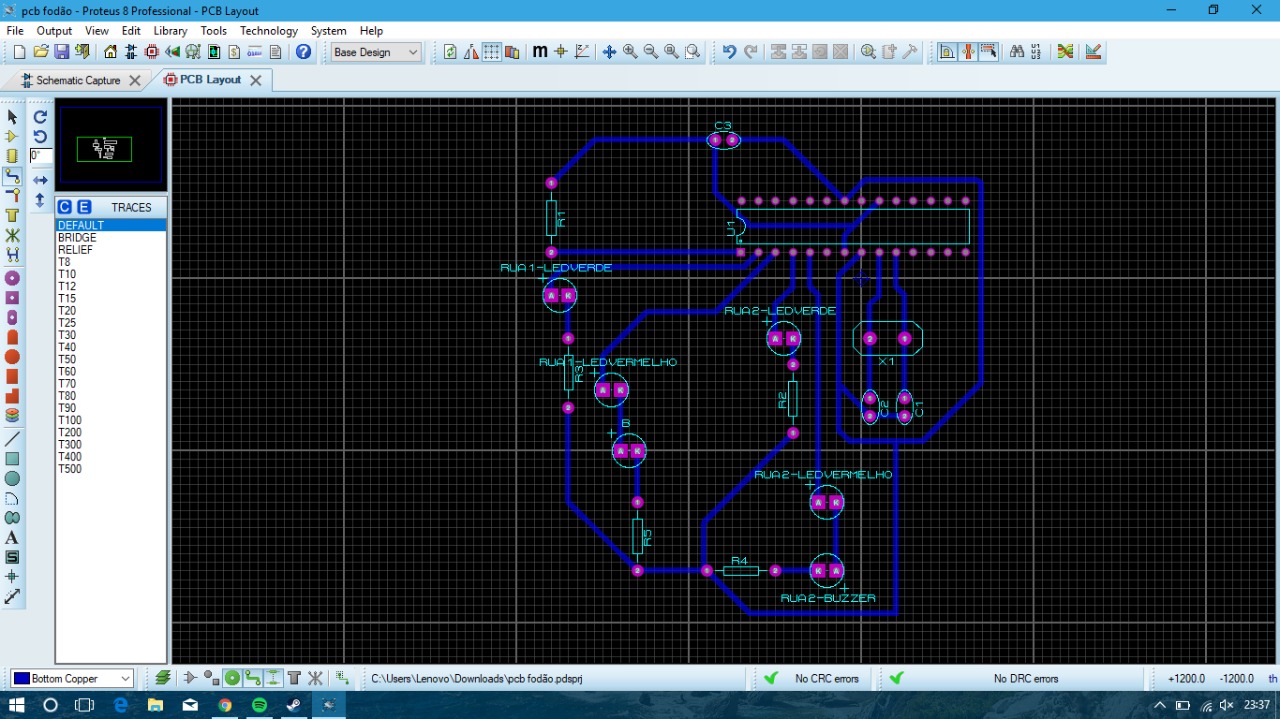


figura 9. Imagem para ilustrar a PCB

O código do microcontrolador, ficou um pouco diferente do utilizado para os flip-flops posto que agora o microcontrolador também funcionará como contador, abaixo podemos perceber o novo código na segunda seção do apêndice.

* **Montagem da Protoboard:**

Para a realização da montagem utilizamos um arduíno UNO para controlar os LEDS, receber a entrada do botão e do 555; 555 para dar clock no arduíno que irá contá-los para alterar os estados dos sinais; os leds para simular os semáforos e o botão para que possa ser solicitado mais tempo para atravessar.

Abaixo podemos perceber o resultado final da montagem figura 10:

# 

figura 10. Imagem para ilustrar a montagem na protoboard

# Montagem da placa :

A montagem da placa se deu da seguinte forma:

1. Imprimimos a PCB em uma folha fotográfica;
2. Prensamos na placa para passar o circuito na placa;
3. Mergulhamos no ácido e esperamos corroer;
4. Depois limpamos com bombril;
5. Furamos a placa no lugar onde os componentes ficarão;
6. Colocamos os componentes;
7. Soldamos os componentes.

# Materiais utilizados:

* Uma placa;
* Papel fotográfico;
* 1- Cristal 16 MHz;
* 2- Capacitores Cerâmicos 22pF;
* 2- Capacitores de 100 microF;
* 4- LEDs ( 2- vermelhos e 2- verdes);
* 4- Resistores de 330 ohms;
* 1- Buzzer;
* 1- Botão;
* 1- Atmega328p-pu;
* 1- Socket para o atmega;
* 1- Lm7805;

# Dificuldades:

A primeira dificuldade que encontramos foi a de encontrar os flip-flops para implementar o que havíamos simulado no Proteus, então em acordo com o monitor resolvemos implementar o microcontrolador na placa.

Em seguida tivemos dificuldade para montar a PCB e a placa posto que nunca criamos uma, mas esta dificuldade foi sanada com a ajuda do monitor Marcos que nos auxiliou na montagem das mesmas.

# Conclusão:

Com este projeto empregamos os conteúdos aprendidos na disciplina de circuitos sequenciais, principalmente no que diz respeito à utilização de componentes para resolução de problemas reais, além de produzir algo que pode ser muito útil para sociedade.

Além de ampliar nossos conceitos sobre simulação de circuitos, montagem de PCB e de placa, que são ferramentas essenciais para o desenvolvimento de qualquer projeto eletrônico.

# Planos para o futuro do projeto:

Nesta etapa do projeto utilizamos o botão para requisitar mais tempo de sinal fechado, mas pretendemos implementar o módulo RFID ao projeto para substituí-lo, de forma a garantir que apenas as pessoas que de fato necessitam, utilizarão este recurso.

Também pretendemos construir um protótipo utilizando um atmega, que se assemelhe consideravelmente a um sinal de verdade, para que possamos apresentá-lo ao prefeito, como uma forma de melhorar o tráfego de pessoas e carros em Recife.

# Apêndice:

1- Código do microcontrolador para a simulação com Flip-Flops

|  |  |
| --- | --- |
| //defino as portas para os componentes |  |
|  | #define LEDVERDE1 0 |
|  | #define LEDVERMELHO1 1 |
|  | #define LEDVERDE2 2 |
|  | #define LEDVERMELHO2 3 |
|  | #define FLIPFLOP7 7 |
|  | #define FLIPFLOP10 8 |
|  | #define FLIPFLOP15 9 |
|  | #define BOTAO 10 |
|  | #define CLEAR 13 |
|  |  |
|  | int ledVerde1, ledVerde2; //crio as flags dos leds |
|  |  |
|  | void setup() { |
|  | // put your setup code here, to run once: |
|  |  |
|  | //defino o tipo de porta |
|  | pinMode(LEDVERDE1, OUTPUT); |
|  | pinMode(LEDVERMELHO1, OUTPUT); |
|  | pinMode(LEDVERDE2, OUTPUT); |
|  | pinMode(LEDVERMELHO2, OUTPUT); |
|  | pinMode(FLIPFLOP7, INPUT); |
|  | pinMode(FLIPFLOP10, INPUT); |
|  | pinMode(FLIPFLOP15, INPUT); |
|  | pinMode(BOTAO, INPUT); |
|  | pinMode(CLEAR, OUTPUT); |
|  |  |
|  | //Iniciar o sistema com leds acesos |
|  | delay(500); //marcos ordenou |
|  |  |
|  | //rua 1 |
|  | digitalWrite(LEDVERDE1, HIGH); //inicia com o led aceso |
|  | ledVerde1 = 1;//seta a flag para 1 |
|  | digitalWrite(LEDVERMELHO1, LOW); |
|  |  |
|  | //rua 2 |
|  | digitalWrite(LEDVERDE2, LOW); |
|  | ledVerde2 = 0; |
|  | digitalWrite(LEDVERMELHO2, HIGH); |
|  | } |
|  |  |
|  | void loop() { |
|  | // put your main code here, to run repeatedly: |
|  |  |
|  | digitalWrite(CLEAR, HIGH);//desabilita o clear |
|  |  |
|  | if (ledVerde1 == 1 && digitalRead(FLIPFLOP7)) { |
|  | //rua 1 |
|  | digitalWrite(LEDVERDE1, LOW); |
|  | ledVerde1 = 0; |
|  | digitalWrite(LEDVERMELHO1, HIGH); |
|  | //rua 2 |
|  | digitalWrite(LEDVERDE2, HIGH); |
|  | ledVerde2 = 1; |
|  | digitalWrite(LEDVERMELHO2, LOW); |
|  | digitalWrite(CLEAR, LOW); |
|  | } else { |
|  | if (digitalRead(BOTAO) && ledVerde1 == 0) { |
|  | if (digitalRead(FLIPFLOP15)) { |
|  | //rua 1 |
|  | digitalWrite(LEDVERDE1, HIGH); |
|  | ledVerde1 = 1; |
|  | digitalWrite(LEDVERMELHO1, LOW); |
|  | //rua 2 |
|  | digitalWrite(LEDVERDE2, LOW); |
|  | ledVerde2 = 0; |
|  | digitalWrite(LEDVERMELHO2, HIGH); |
|  | digitalWrite(CLEAR, LOW); |
|  | } |
|  | } else { |
|  | if (ledVerde1 == 0 && digitalRead(FLIPFLOP10) && !digitalRead(BOTAO)) { |
|  | //rua 1 |
|  | digitalWrite(LEDVERDE1, HIGH); |
|  | ledVerde1 = 1; |
|  | digitalWrite(LEDVERMELHO1, LOW); |
|  | //rua 2 |
|  | digitalWrite(LEDVERDE2, LOW); |
|  | ledVerde2 = 0; |
|  | digitalWrite(LEDVERMELHO2, HIGH); |
|  | digitalWrite(CLEAR, LOW); |
|  | } |
|  | } |
|  | } |
|  | } |

2- Código do microcontrolador sem o 555 e os flip-flops:

//defino as portas para os componentes

#define LEDVERDE1 0

#define LEDVERMELHO1 1

#define LEDVERDE2 2

#define LEDVERMELHO2 3

#define BOTAO 10

int ledVerde1, ledVerde2; //crio as flags dos leds

void setup() {

// put your setup code here, to run once:

//defino o tipo de porta

pinMode(LEDVERDE1, OUTPUT);

pinMode(LEDVERMELHO1, OUTPUT);

pinMode(LEDVERDE2, OUTPUT);

pinMode(LEDVERMELHO2, OUTPUT);

pinMode(BOTAO, INPUT);

//Iniciar o sistema com leds acesos

delay(500); //marcos ordenou

//rua 1

digitalWrite(LEDVERDE1, HIGH); //inicia com o led aceso

ledVerde1 = 1;//seta a flag para 1

digitalWrite(LEDVERMELHO1, LOW);

//rua 2

digitalWrite(LEDVERDE2, LOW);

ledVerde2 = 0;

digitalWrite(LEDVERMELHO2, HIGH);

}

void loop() {

// put your main code here, to run repeatedly:

if ((ledVerde1 == 1)) {//7segundos

delay(7000);

//rua 1

digitalWrite(LEDVERDE1, LOW);

ledVerde1 = 0;

digitalWrite(LEDVERMELHO1, HIGH);

//rua 2

digitalWrite(LEDVERDE2, HIGH);

ledVerde2 = 1;

digitalWrite(LEDVERMELHO2, LOW);

} else {

if (digitalRead(BOTAO) && ledVerde1 == 0) {

delay(15000);//15

//rua 1

digitalWrite(LEDVERDE1, HIGH);

ledVerde1 = 1;

digitalWrite(LEDVERMELHO1, LOW);

//rua 2

# 