UNIVERSIDADE FEDERAL DO AMAZONAS
FACULDADE DE TECNOLOGIA
ENGENHARIA DA COMPUTAÇÃO

PAULO HENRIQUE ARAÚJO MUNHOZ – 21104569

Lista I – Microcontroladores

MANAUS – AM
2019

PAULO HENRIQUE ARAÚJO MUNHOZ – 21104569

Lista I - Microcontroladores

Primeiro Relatório da Disciplina Microcontroladores, apresentado ao Curso de Engenharia da Computação.

PROFESSOR: Prof. Thiago Brito Bezerra

MANAUS - AM
2019

Dedicamos este relatório aos nossos familiares que nos proporcionaram meios de ingressar e nos manter na universidade e aos amigos do grupo de trabalho.

AGRADECIMENTOS

Ao professor, pelo tempo dedicado e incentivo dado aos alunos na matéria ministrada, que permitiu a confecção deste relatório de qualidade técnica.

Aos colegas de curso e turma pelo apoio e amizade.

E a todos aqueles que, embora aqui não mencionados, contribuíram para a realização deste trabalho, o meu muito obrigado.

"É melhor você tentar algo, vê-lo não funcionar e aprender com isso, do que não fazer nada." MARK ZUCKERBERG.

RESUMO

Ao trabalhar neste relatório pode-se abordar com microcontrolador que é um pequeno computador num único circuito integrado, possuindo um núcleo, memória e periféricos programáveis de entradas e saídas. Sua maior aplicação se destaca de forma embarcada termo este que em contrataste dos microprocessadores são utilizados como computadores pessoais, no entanto os embarcados se destacam por aplicações de uso geral, e se encontram cada vez mais emersos em nosso dia-a-dia, como controle remoto, micro-ondas, máquina de lavar e entre outros. O seu consumo de energia relativamente baixo, normalmente, na casa de miliwatts, possui a habilidade para entrar em modo de espera aguardando por uma interrupção ou evento externo, tornando-se ideais para aplicações onde a exigência de baixo consumo de energia é um fator decisivo para o sucesso do projeto. Também possui sinal misto, integrando tanto componentes analógicos quanto sistemas digitais; As simulações deste trabalho foram feitas com o Proteus 8.8, programa este para projeção de circuitos impressos (PCB) e simulações de circuitos eletrônicos.

Palavras-chaves: Microcontrolador. Circuito integrado. Sistemas digitais. Sistemas analógicos. Circuitos eletrônicos

ABSTRACT

Working on this report you can approach with microcontroller which is a small computer in a single integrated circuit, having a core, memory and programmable peripherals of inputs and outputs. Its greater application is highlighted in an embedded term that in contracting of the microprocessors are used as personal computers, however the embedded ones stand out for applications of general use, and are increasingly emerge in our day to day, as control remote, microwave, washing machine and among others. Its relatively low power consumption, usually in the house of milliwatts, has the ability to go into standby mode waiting for an outage or external event, making it ideal for applications where the requirement of low power consumption is a deciding factor for the success of the project. It also has a mixed signal, integrating both analog components and digital systems; The simulations of this work were done with Proteus 8.8, this program for the projection of printed circuits (PCB) and simulations of electronic circuits.

Keywords: Microcontroller. Integrated circuit. Digital systems. Analog systems. Electronic circuits

1 SUMÁRIO

1 INTRODUÇÃO

O objetivo geral deste trabalho é o entendimento do cenário de sistemas embarcados, voltada para o entendimento do microcontrolador PIC16F628A, a qual foi apresentado na disciplina pelo professor, foram abordados exercícios para estimulação do primeiro contato com este microcontrolador em linguagem de baixo nível Assembly a qual será relatado nos próximos capítulos a seguir, incluindo a resolução e codificação dos mesmos.

.

2 TEORIA

Para uma melhor compreensão da teoria, é importante considerar que as bases teóricas utilizadas neste trabalho foram obtidas tanto em livros e vídeo-aulas, quanto em livros clássicos sobre microcontrolador apresentados na bibliografia do curso , um destes livros é o Desbravando o PIC 16F628A, este que utilizamos em todos os projetos requisitados pelo professor da disciplina à qual este trabalho é apresentado.

2.1 CONCEITOS PRELIMINARES

O PIC 16F628A foi escolhido devido as suas características:

- Microcontrolador de 18 pinos, o que facilita a montagem de hardwares experimentais;
- Até 16 portas configuráveis como entrada ou saída e 2 osciladores internos (4MHz e 37 kHz);
- 10 interrupções disponíveis (Timers, Externa, Mudança de Estado, EEPROM, USART, CCP e Comparador);
- Memória de programação FLASH com 2.048 words, que permite a gravação do programa diversas vezes no mesmo chip sem a necessidade de apaga-lo por meio de luz ultravioleta, como acontece nos microcontroladores de janela;
- Memória EEPROM (não volátil) interna com 128 bytes;
- Recursos adicionais avançados: módulo CCP, Comparador interno e USART;
- Programação com 14 bit e 35 instruções.

A grande vantagem da família PIC é que todos os modelos possuem um set de instruções bem parecido, assim como mantêm muitas semelhanças entre suas características básicas. Desta forma, ao conhecermos e estudarmos o PIC16F628A, estaremos nos familiarizando com todos os microcontroladores da Microchip (principalmente os de 14 e 16 bits), o que tornará a migração para outros modelos muito mais simples.

2.2 A PINAGEM E SUAS NOMENCLATURAS

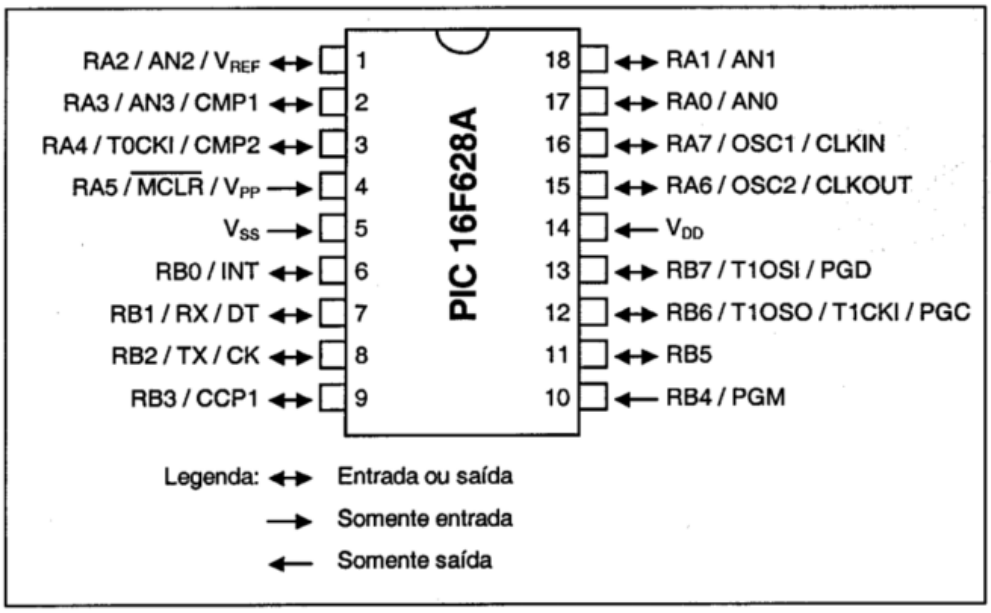

Figura 1- PIC16F628A.

O PIC16F628A possui um total de 16 I/O separados em dois grupos denominados PORTAS. Dessa forma, temo a Porta A e a Porta B. Para facilitarmos o entendimento e a comparação com os datasheets originais, usaremos os termos provenientes do inglês PORTA (porta A) e PORTB (porta B).

O PORTA A possui oito pinos que podem ser configurados como entrada ou saída, e seus nomes são definidos como RA0, RA1, RA2, RA3, RA5, RA6 e RA7. Para termos disponibilidade do pino RA5, perderemos o MCLR externo. Da mesma forma, para disponibilizarmos RA6 e RA7 não poderemos utilizar esses pinos para ligação de um oscilador externo. Por este motivo, poderemos utilizar um dos dois osciladores existentes. O pino RA4 também pode ser utilizado para incremento externo do TMR0.Alguns outros pinos do PORTA ainda possuem funções sobrecarregadas em relação aos dois comparadores existentes. O pino RA2 pode ainda ser utilizado como uma saída de tensão programável (V_{ref}) com 16 níveis diferentes.

O PORT B também possui oito pinos configuráveis como entrada e saída, sendo seus nomes RB0, RB1, RB2, RB3, RB4, RB5, RB5, RB6 e RB7. O RB0 pode ser utilizado também para gerar interrupção externa, assim como os pinos de RB4 a RB7 podem gerar a interrupção por mudança de estado. Os pinos RB1 e RB2 também são utilizados para comunicação serial (USART). Já o pino RB3 é utilizado no módulo de CCP, para saída do PWM. O pino RB6 pode ainda ser utilizado para incremento do TMR1 e, juntamente como RB7, para programação do microcontrolador.

Para que o microcontrolador possa funcionar, é necessária também a sua alimentação: são os pinos V_{ss}(GND) e V_{DD}(+5V_{cc}). A tensão de alimentação nominal dos PICs é de 5 V_{cc}, mas

o ranger de variação desta tensão depende do modelo estudado. No nosso caso o PIC16F628A, ele vai de 2.0 a 5.5 V_{cc}.

O oscilador externo deve ser ligado aos pinos OSC1 e OSC2. Temos ainda o pino denominador MCLR (barrado), que se refere ao Master Clear externo. Sempre que esse pino for colocado em nível baixo (GND), o programa será resetado e o processamento paralisado. Ao ser colocado em nível alto (+5V), a execução do programa será retomada do ponto inicial. Para entender melhor o significado das nomenclaturas utilizadas na identificação dos pinos, a tabela seguinte descreve os detalhes de cada uma delas.

Número do Pino	Função	Tipo Entrada	Tipo Saída	Descrição
17	RA0	ST	CMOS	I/O digital bidirecional.
	AN0	AN	-	Entrada analógica para os comparadores.
18	RA1	ST	CMOS	I/O digital bidirecional.
	AN1	AN	-	Entrada analógica para os comparadores.
1	RA2	ST	CMOS	I/O digital bidirecional.
	AN2	AN	-	Entrada analógica para os comparadores.
	V_{REF}	-	AN	Saída da tensão de referência programável.
2	RA3	ST	CMOS	I/O digital bidirecional.
	AN3	AN	-	Entrada analógica para os comparadores.
	CMP1	-	CMOS	Saída do comparador 1.
3	RA4	ST	OD	I/O digital bidirecional.
	T0CKI	ST	-	Entrada externa do contador TMR0.
	CMP2	-	OD	Saída do comparador 2.
4	RA5	ST	-	Entrada digital.
	MCLR	ST	-	Master Clear (reset) externo. O PIC só funciona quando este pino encontra-se em nível alto.
	V_{PP}		-	Entrada para tensão de programação (13V).
15	RA6	ST	CMOS	I/O digital bidirecional.
	OSC2	-	XTAL	Saída para cristal externo.
	CLKOUT	-	CMOS	Saída com onda quadrada em ¼ da freqüência imposta em OSC1 quando em modo RC. Essa freqüência equivale aos ciclos de máquina internos.

Número do Pino	Função	Tipo Entrada	Tipo Saída	Descrição
16	RA7	ST	CMOS	I/O digital bidirecional.
	OSC1	XTAL	-	Entrada para cristal externo.
	CLKIN	ST	-	Entrada para osciladores externos (híbridos ou RC).
6	RB0	TTL	CMOS	I/O digital bidirecional com pull-up interno.
	INT	ST	-	Entrada para interrupção externa.
7	RB1	TTL	CMOS	I/O digital bidirecional com pull-up interno.
	RX	ST	-	Recepção para comunicação USART assíncrona.
	DT	ST	CMOS	Via de dados para comunicação USART síncrona.
8	RB2	TTL	CMOS	I/O digital bidirecional com pull-up interno.
	TX	-	CMOS	Transmissão para comunicação USART assíncrona.
	CK	ST	CMOS	Via de clock para comunicação USART síncrona.
9	RB3	TTL	CMOS	I/O digital bidirecional com pull-up interno.
	CCP1	ST	CMOS	I/O para o Capture, Compare e PWM.
10	RB4	TTL	CMOS	I/O digital bidirecional com pull-up interno. Interrupção por mudança de estado.
	PGM	ST	-	Entrada para programação em baixa tensão (5V).
11	RB5	TTL	CMOS	I/O digital bidirecional com pull-up interno. Interrupção por mudança de estado.
12	RB6	TTL	CMOS	I/O digital bidirecional com pull-up interno. Interrupção por mudança de estado.
	T1OSO	-	XTAL	Saída para cristal externo para TMR1.
	T1CKI	ST	-	Entrada externa do contador TMR1.
	PGC	ST	-	Clock da programação serial (ICSP).
13	RB7	TTL	CMOS	I/O digital bidirecional com pull-up interno. Interrupção por mudança de estado.
	T1OSI	XTAL	-	Entrada para cristal externo para TMR1.
	PGD	ST	CMOS	Data da programação serial (ICSP).
5	V_{SS}	P	-	GND.
14	V_{DD}	P	-	Alimentação positiva.

Tabela 1 – Descrição das Nomenclaturas

Legenda: P = Power(alimentação)
- = Não Utilizado
TTL = Entrada tipo TTL
ST = Entrada tipo Schmitt Trigger
CMOS = Saída Tipo CMOS
OD = Saída tipo Dreno Aberto (Open Drain)
NA = Entrada/Saída Analógica

CARACTERÍSTICAS ELÉTRICAS E OUTRAS

Temperatura de trabalho ..... -40°C até +125°C
Temperatura de armazenamento ..... -65°C até +150°C
Tensão de trabalho ..... 3.0V a +5.5V
Voltagem máxima no pino V_{DD} (em relação ao V_{SS}) ..... -0.3V até +6.5V
Voltagem máxima no pino MCLR (em relação ao V_{SS}) ..... -0.3V até +14V
Voltagem máxima nos demais pinos (em relação ao V_{SS}) ..... -0.3V até (V_{DD} + 0.3V)
Dissipação máxima de energia ..... 800 mW
Corrente máxima de saída no pino V_{SS} ..... 300 mA
Corrente máxima de entrada no pino V_{DD} ..... 250 mA
Corrente máxima de entrada de um pino (quando em V_{SS}) ..... 25 mA
Corrente máxima de saída de um pino (quando em V_{DD}) ..... 25 mA
Corrente máxima de entrada em PORTA + PORTB ..... 200 mA
Corrente máxima de saída em PORTA + PORTB ..... 200 mA

2.3 MAPAS DE MEMÓRIAS

As memórias de programação e dados do PIC16F628A

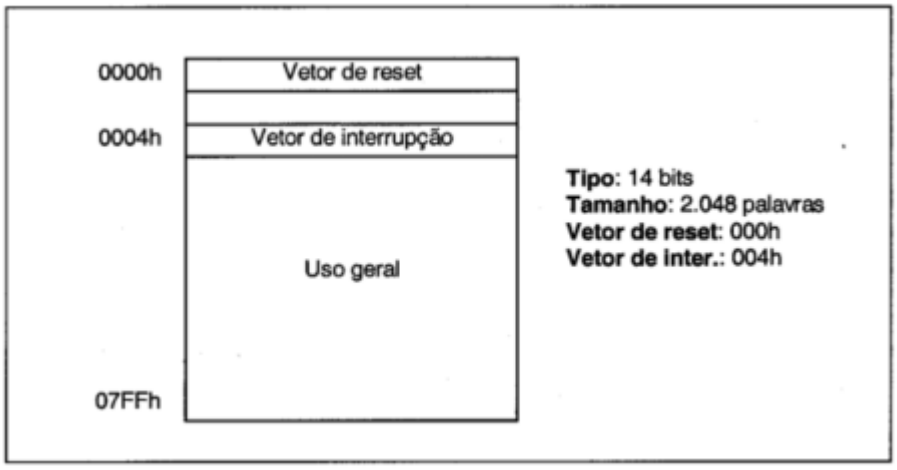

Figura 2- Memória do Programa.

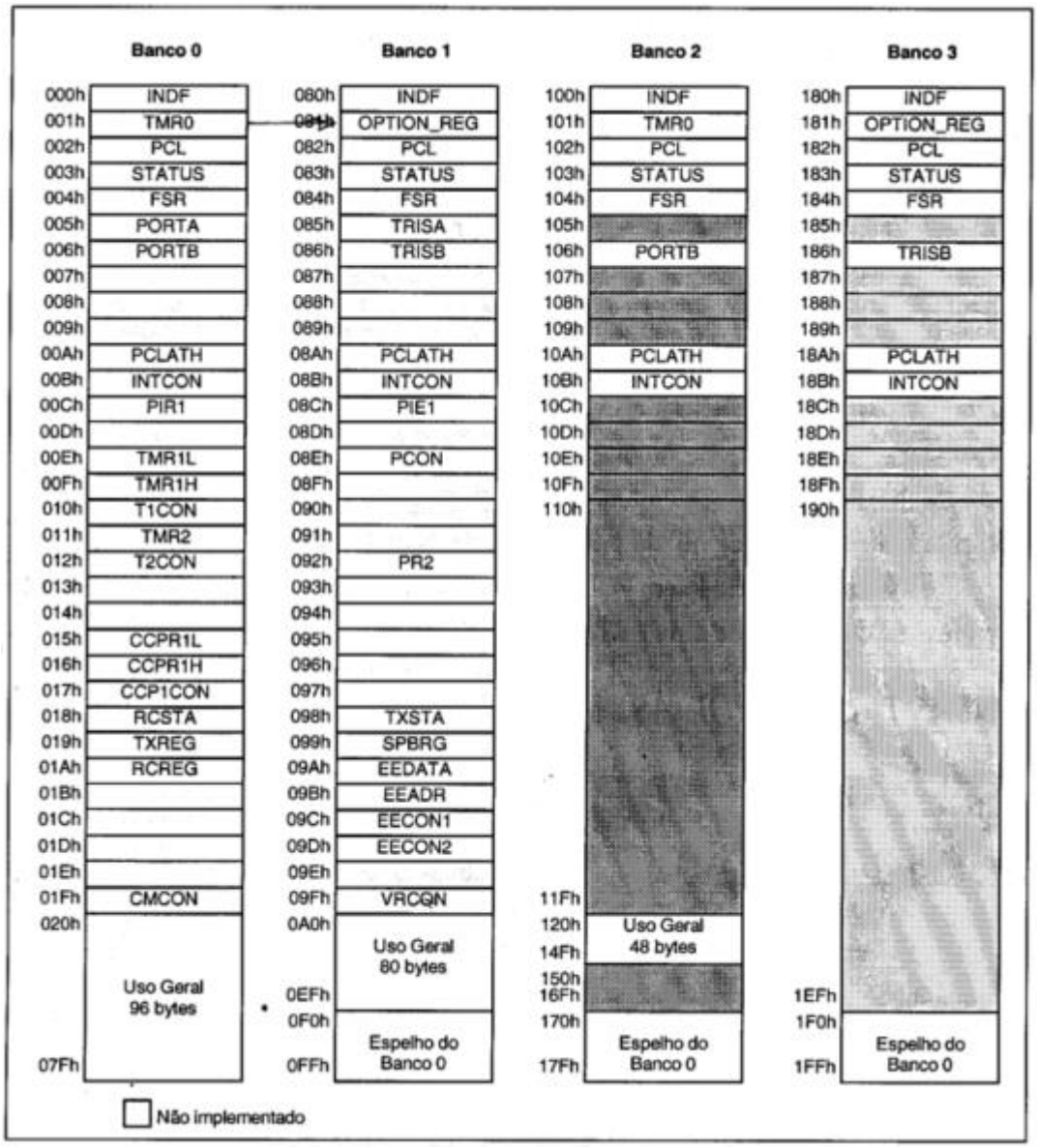

Figura 4 – Memória de Dados.

3 PRÁTICAS

3.1 Prática 1 – Esteira transportadoras de Bola de Futebol.

Questão 1: Uma esteira transporta bolas de futebol diretamente para o baú de um caminhão. Esse baú possui um sensor de nível máximo SA para informar quando ele estiver cheio. O processo funciona de tal forma que o caminhão precisa estacionar abaixo da grande esteira. A presença do veículo é detectada pelo sensor SC. Existe também um sensor de presença SB sobre a esteira que informa se há bolas depositadas sobre ela.

Um alarme deve ser acionado sempre que o nível alto do sensor SA for detectado, para o motorista retirar o caminhão, dando a oportunidade para que outro veículo reinicie o processo.

Para essa prática podem ser utilizados os seguintes materiais:

TABELA 3. 1 – MATERIAIS UTILIZADOS DA PRÁTICA 1.

MATERIAL	QUANTIDADE	PREÇOS	VENDEDOR
PIC16F628A	1 unidade	R$12,51	Baú da Eletrônica
Capacitor Cerâmico 22nF /50V	2 unidades	R$0,10	Baú da Eletrônica
Cristal de Quartzo / 4Mhz	1 unidade	R$0,71	Baú da Eletrônica
Sensor Ultrasônico	3 unidades	R$10,99	Mercado Livre
Diodo 1N4001	1 unidade	R$0.09	Baú da Eletrônica
Alarme DC 12 V /95Db	1 unidade	R$2,86	AliExpress
Transistor 2N4401	1 unidade	R$0,17	Baú da Eletrônica
Resistor 1k	1 unidade	R$0,07	Baú da Eletrônica
Placa de Fibra 8x12 cm	1 unidade	R$6,29	Baú da Eletrônica
TOTAL		R$55,86	

A montagem da Prática 1 está representada na Figura - 3.1.

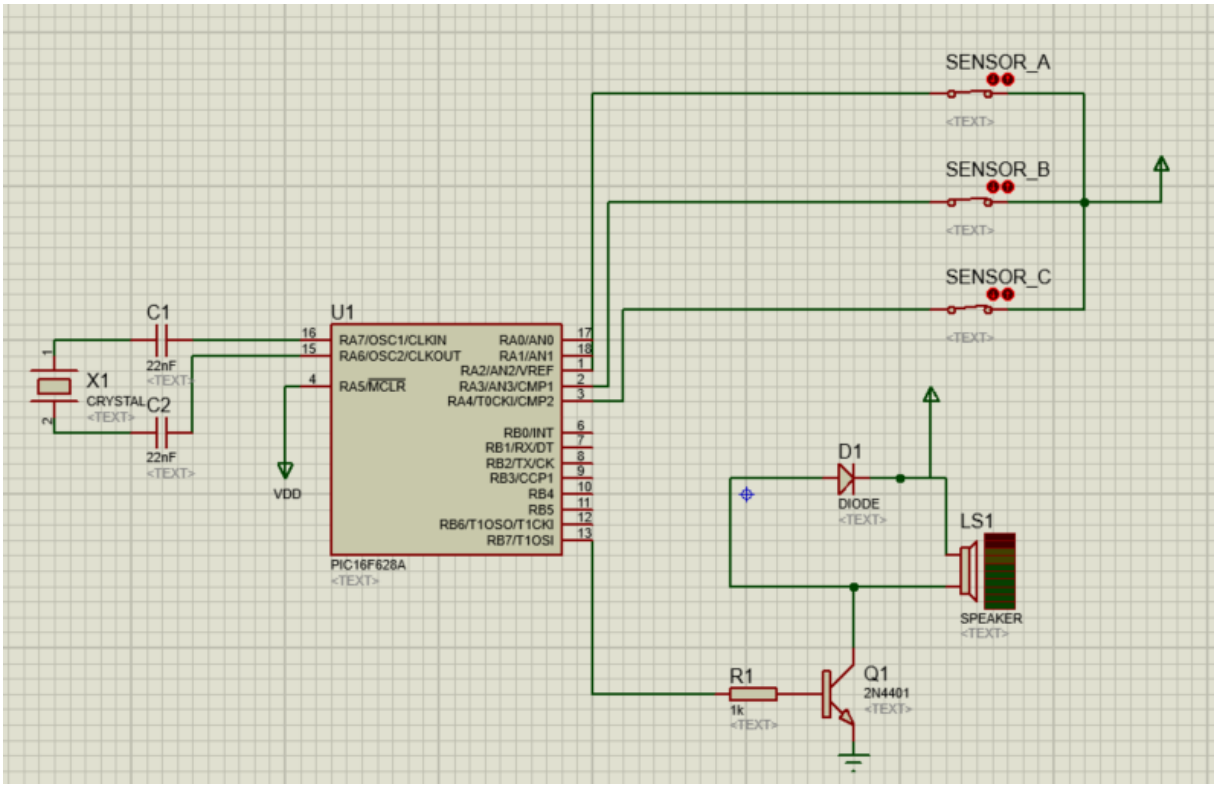

Figura 3.1 – Montagem dos componentes da Prática 1 no Proteus.

Foram adicionados 2 capacitores de 22 nF (C1 e C2) em série com o Crystal (oscilador de quartzo) nas entradas 16 e 15 que tem como entradas a OSC1 e OSC2 como foi descriminado no relatório. O sensor A é responsável por informar quando o baú estiver cheio. O sensor SB analisa se há bolas depositadas na esteira e SC se há presença do caminhão no local. Os botões AS, SB e SC simulam os sensores ultrasônicos. Quando SA estiver cheio, o Speaker acionará, simulando o Alarme, o circuito abaixo é uma configuração de transistor que funciona como chave e um Diodo que não permite passagem de corrente reversa, caso estivéssemos trabalhando com um sinal AC. No entanto se quiséssemos que escutássemos um bipe variando, teríamos que ter uma saída senoidal que será abordado em interrupções.

3.2 Pratica 2 – Contagem de Peças Defeituosas

Questão 2: Faça um sistema de contagem de peças defeituosas que ao ligar o sistema, uma esteira deverá ser ativada. A cada 10 peças defeituosas detectadas, ativar um sinal de alarme por 10 segundos. A cada 20 peças defeituosas detectadas, o sinal de alarme será de 20 segundos e a linha de produção deverá parar.

Incluir um botão de reinício do sistema.

Para essa prática podem ser utilizados os seguintes materiais:

TABELA 3.2 - MATERIAIS UTILIZADOS DA PRÁTICA 2.

MATERIAL	**QUANTIDADE**	**PREÇOS**	**VENDEDOR**
PIC16F628A	1 unidade	R$12,51	Baú da Eletrônica
Capacitor Cerâmico 22nF /50V	2 unidades	R$0,10	Baú da Eletrônica
Cristal de Quartzo / 4Mhz	1 unidade	R$0,71	Baú da Eletrônica
Sensor Ultrasônico	1 unidades	R$10,99	Mercado Livre
Motor de Passo 9 Kgf.cm	1 unidade	R$99,45	Baú da Eletrônica
Drive Motor de Passo	1 unidade	R$10,07	Baú da Eletrônica
Transistor 2N4401	1 unidade	R$0,17	Baú da Eletrônica
Resistor 10k	1 unidade	R$0,07	Baú da Eletrônica
Resistor 220	2 unidades	R$0,14	Baú da Eletrônica
Botão 8x8 mm	2 unidade	R$0,41	Baú da Eletrônica
Placa de Fibra 8x12 cm	1 unidade	R$6,29	Baú da Eletrônica
TOTAL		R$141,56	

A montagem da Prática 2 está representada na Figura - 3.2.

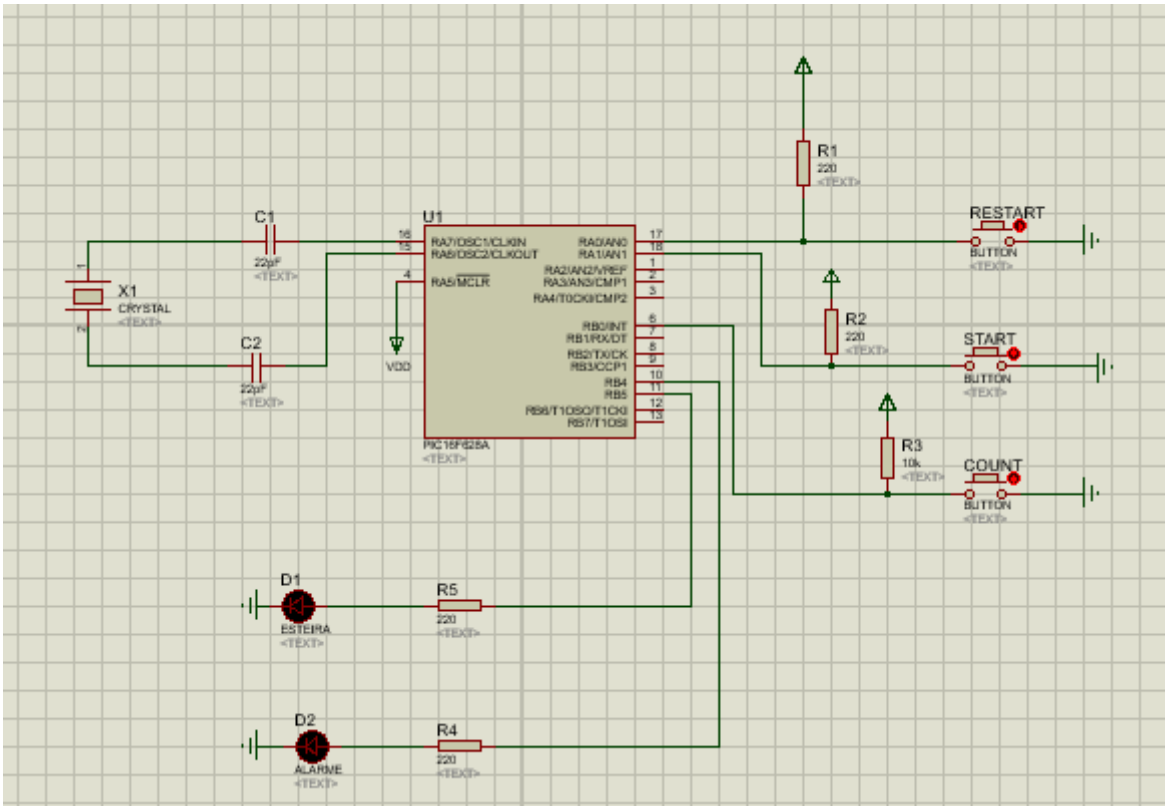

Figura 3.2 – Montagem dos componentes da Prática 2 no Proteus.

Os botões de START e RESTART, simulam o início e o reinicio do sistema, o botão COUNT simula o sensor ultrassônico, o mesmo irá verificar pela altura as possíveis peças defeituosas, pois em muitas fabricas uma peça de uma produção pode possuir um tamanho a altura única, e se por ventura tal altura não corresponder ao de limite permitido essa peça é dita como defeituosa. O LED D1 simula a esteira sendo ligada e o ALARME D2 simula o possível alarme que acionará os devidos eventos.

3.3 Prática 3 – Fábrica de Sucos

Questão 3: Uma fábrica de sucos bolou um processo para preencher automaticamente suas garrafas, cujo esquema é apresentado abaixo: O tanque armazena grande quantidade de suco. A válvula V abre sempre que o sensor SO2 detectar que há uma garrafa debaixo dela. O motor M da esteira se movimenta toda vez que o sensor SO1 detectar a presença de garrafas sobre ela. Quando o sensor SO2 detectar a presença de garrafas, o motor M deve parar, voltando a funcionar somente se o sensor SS determinar que o líquido depositado na garrafa atingiu o nível máximo. Se o sensor SL detectar que o nível de suco está mínimo no tanque, a válvula V deve fechar, o motor M deve parar e a bomba B1 preencherá o tanque com suco até o nível máximo, que será detectado por SH, quando então B1 sairá de operação. Inclua contatos que permitam ligar e desligar manualmente a bomba e o motor da esteira para efeitos de teste. Desenvolva um código que realize a automação desse processo.

Para essa prática podem ser utilizados os seguintes materiais:

TABELA 3.3 - MATERIAIS UTILIZADOS DA PRÁTICA 3.

MATERIAL	QUANTIDADE	PREÇOS	VENDEDOR
PIC16F628A	1 unidade	R$12,51	Baú da Eletrônica
Capacitor Cerâmico 22nF /50V	2 unidades	R$0,10	Baú da Eletrônica
Cristal de Quartzo / 4Mhz	1 unidade	R$0,71	Baú da Eletrônica
Válvula Solenoide 1/2	1 unidade	R$40,13	Baú da Eletrônica
Sensor ultrassônico	3 unidade	R$10,99	Mercado Livre
Sensor Infravermelho	2 unidades	R$19,25	Mercado Livre
Bomba D´água 1/2	1 unidade	R$119,90	Amazon
Resistor 10k	3 unidade	R$0,07	Baú da Eletrônica
Motor 1/3 cv 3500 rpm	1 unidade	R$ 235	Mercado Livre
Resistor 220	3 unidades	R$0,14	Baú da Eletrônica
Botão 8x8 mm	2 unidade	R$0,41	Baú da Eletrônica
1 relé/ 110V e 220V	1 unidade	R$12,82	Bang good
Placa de Fibra 8x12 cm	1 unidade	R$6,29	Baú da Eletrônica
TOTAL		R$500,55	

A montagem da Prática 1 está representada na Figura - 3.3.

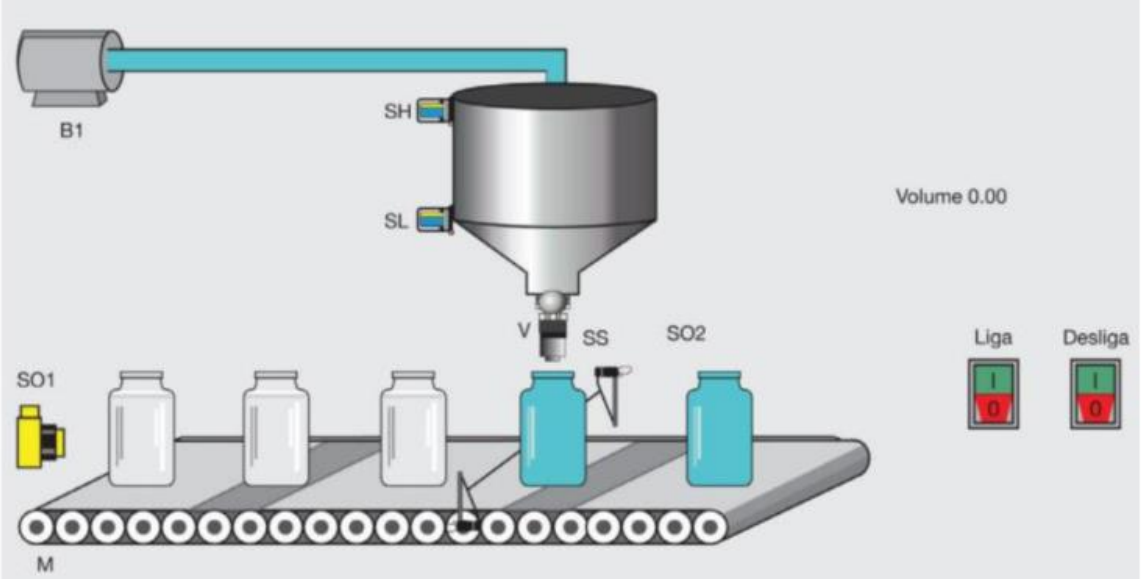

Imagem 2 – Imagem ilustrativa da Fábrica de sucos.

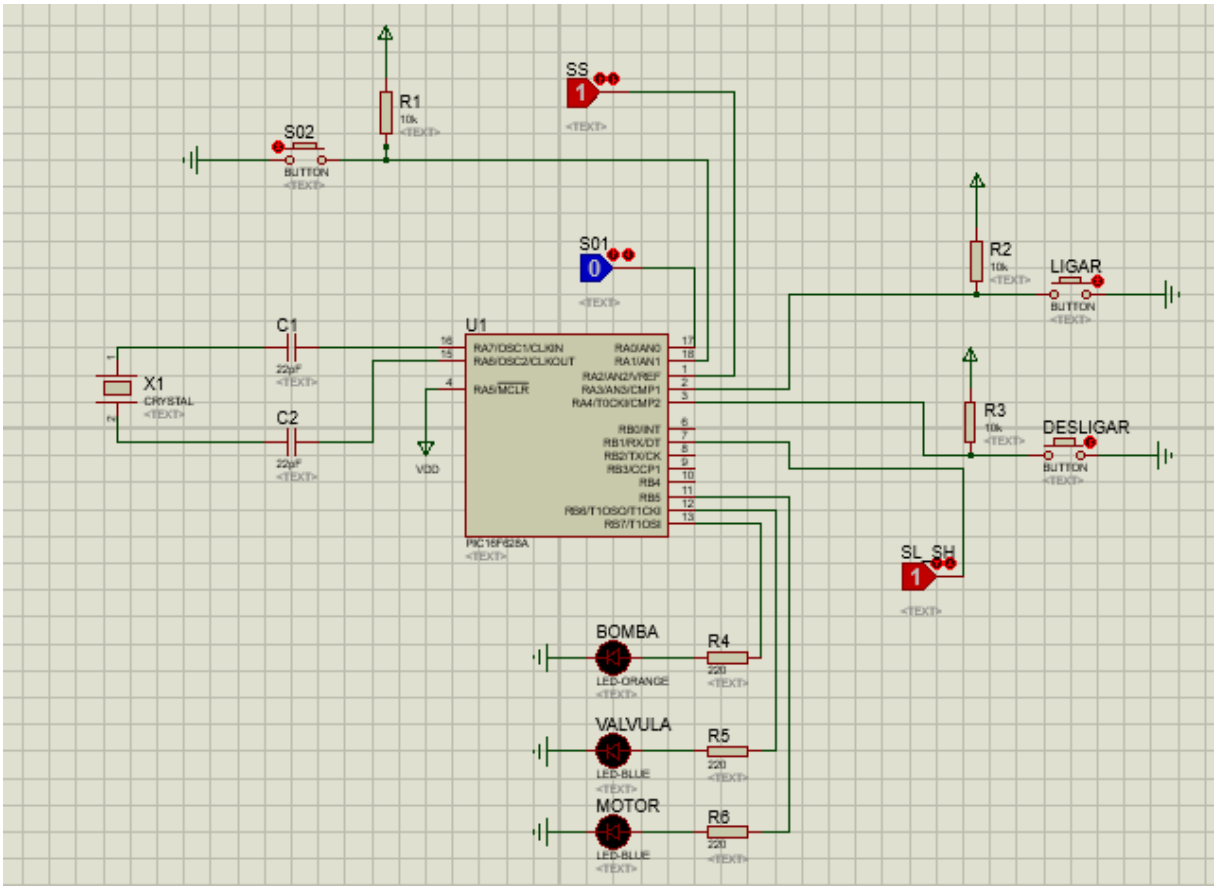

Figura 3.3 – Montagem dos componentes da Prática 3 no Proteus.

Setando para S01 para valor baixo, SL_SH para valor lógico alto e SS para alto também, temos que quando ligado o botão LIGAR temos o funcionamento do MOTOR, quando S02 é acionado o MOTOR para de funcionar e a VALVULA é acionada. E quando SS, SL_SH e S01

estiverem nível lógico baixo a VALVULA e o MOTOR irão desligar deixando apenas a BOMBA trabalhando. O botão DESLIGAR desliga todo o sistema, para fins de desenvolvimento físico teríamos como BOMBA uma bomba d´água, a VALVULA uma válvula solenoide, para S01 e S02 sensores infravermelhos, o MOTOR, motor com relé e para SS e SL sensores ultrassônicos.

3.4 Pratica 4 – Elevador Externo

Questão 4: Deseja-se comprar um elevador externo para uma indústria. O elevador consiste em um motor, capaz de realizar movimentos de ascensão e descida sobre a plataforma transportadora. Para prevenir colisões sobre o solo ou contra o topo do elevador, são utilizados sensores de nível baixo e alto. Haverá também um botão de parada que cessa os movimentos do elevador. Pede-se elaborar um programa que realize a automação desse processo.

Para essa prática podem ser utilizados os seguintes materiais:

TABELA 3.4 - MATERIAIS UTILIZADOS DA PRÁTICA 4.

MATERIAL	QUANTIDADE	PREÇOS	VENDEDOR
PIC16F628A	1 unidade	R$12,51	Baú da Eletrônica
Capacitor Cerâmico 22nF /50V	2 unidades	R$0,10	Baú da Eletrônica
Cristal de Quartzo / 4Mhz	1 unidade	R$0,71	Baú da Eletrônica
Sensor Ultrasônico	1 unidades	R$10,99	Mercado Livre
Resistor 10k	1 unidade	R$0,07	Baú da Eletrônica
Resistor 220	2 unidades	R$0,14	Baú da Eletrônica
Botão 8x8 mm	2 unidade	R$0,41	Baú da Eletrônica
Placa de Fibra 8x12 cm	1 unidade	R$6,29	Baú da Eletrônica
Sensor Caneta NA/ 5V a 60V	2 unidades	R$50	Mercado Livre
1 relé/ 110V e 220V	1 unidade	R$12,82	Bang good
Motor 1/3 cv 3500 rpm	1 unidade	R$ 235	Mercado Livre
TOTAL		R$329,69	

A montagem da Prática 4 está representada na Figura - 3.4.

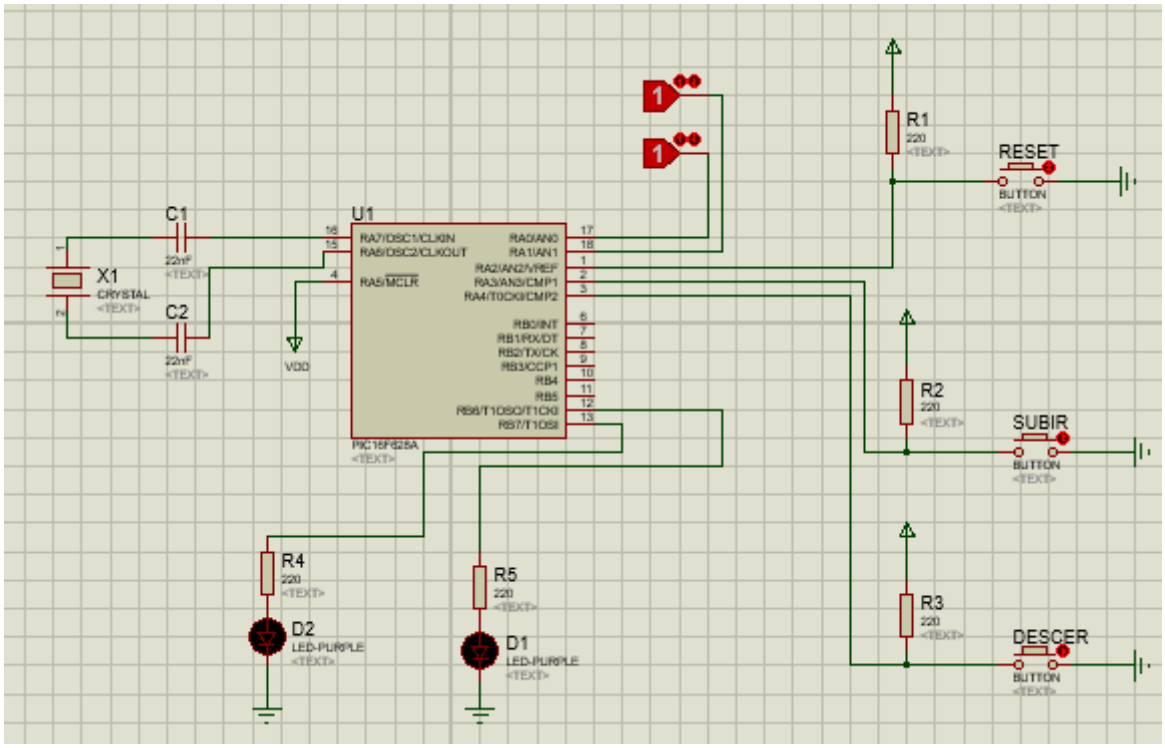

Figura 3.4 – Montagem dos componentes da Prática 4 no Proteus.

Os sensores LOGICSTATE estão setados em nível lógico alto e eles são o que temos como Sensor Caneta, são responsáveis por verificar se o elevador está no ultimo andar ou no térreo. Os botões de RESET reseta qualquer estado do elevador fazendo ele ir para o marco inicial, o botão SUBIR ativa o motor que está como LED D1 roxo que liga, e o botão DESCER liga o LED D2 roxo e desliga o D1, podemos mudar essa configuração de LEDs por um relé e um motor, fazendo assim o acionamento, no entanto devemos entrar em um detalhe quanto ao giro do motor, o adequado seria utilizar um PWM, no entanto limitei-me pois ainda será abordado nas próximas aulas.

3.5 Prática 5 – Sinal de Trânsito Convencional

Questão 5: Pede-se criar programa para controlar um sinal de trânsito convencional que tenha a lâmpada verde acesa por 10 segundos, a lâmpada amarela acesa por 5 segundos e a lâmpada vermelha acesa por 10 segundos. O sinal terá seu ciclo de operação iniciado por um botão liga. A sequência verde, amarelo e vermelho será repetida indefinidamente, até que um botão de desliga seja pressionado.

Para essa prática podem ser utilizados os seguintes materiais:

TABELA 3.5 - MATERIAIS UTILIZADOS DA PRÁTICA 5.

MATERIAL	QUANTIDADE	PREÇOS	VENDEDOR
PIC16F628A	1 unidade	R$12,51	Baú da Eletrônica
Capacitor Cerâmico 22nF /50V	2 unidades	R$0,10	Baú da Eletrônica
Cristal de Quartzo / 4Mhz	1 unidade	R$0,71	Baú da Eletrônica
Resistor 10k	2 unidade	R$0,07	Baú da Eletrônica
Resistor 220	3 unidades	R$0,14	Baú da Eletrônica
Botão 8x8 mm	2 unidade	R$0,41	Baú da Eletrônica
Semáforo	1 unidade	R$600	Mercado Livre
Placa de Fibra 8x12 cm	1 unidade	R$6,29	Baú da Eletrônica
TOTAL		R$614,8	

A montagem da Prática 5 está representada na Figura - 3.5.

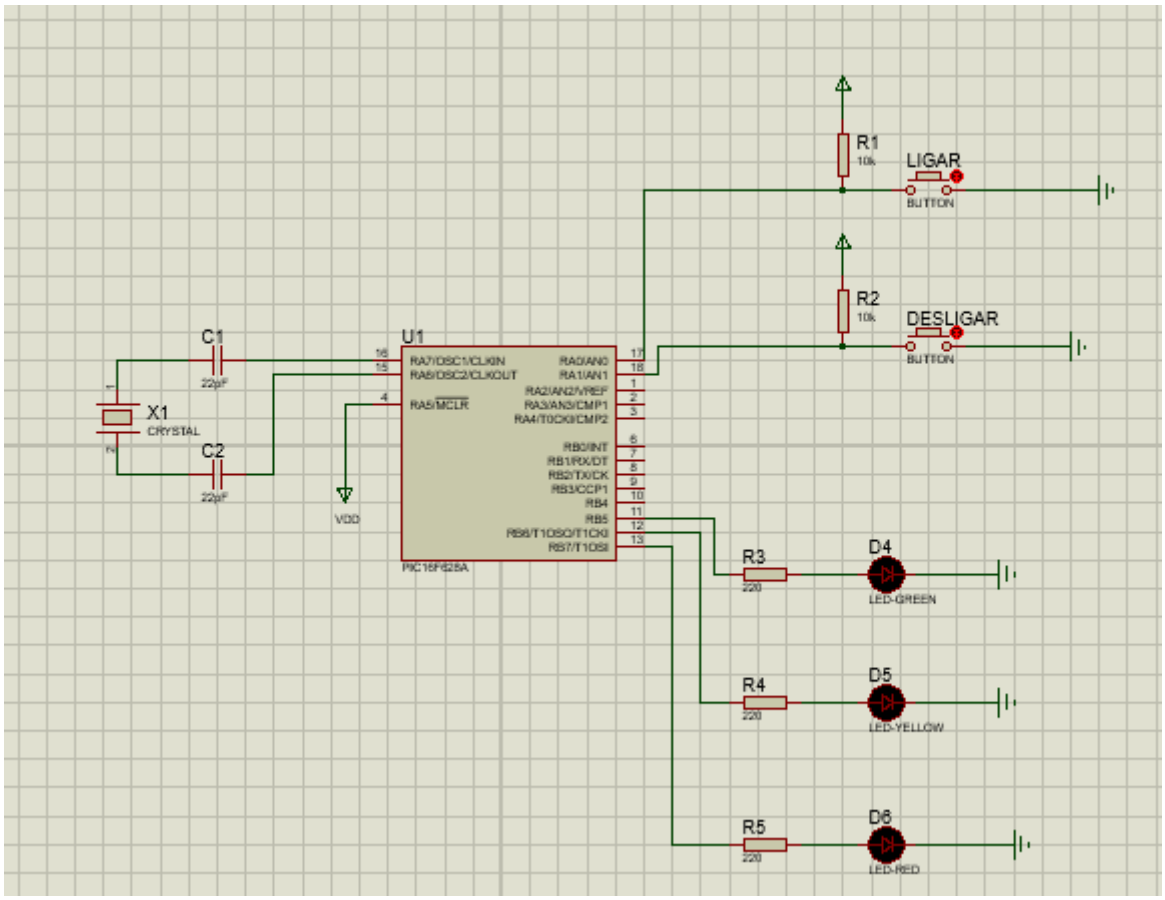

Figura 3.5 – Montagem dos componentes da Prática 5 no Proteus.

O botão LIGAR é responsável por energizar o sistema e por sua vez ligar o LED D4 verde, depois de 10 segundos o LED D5 amarelo acende por 5 segundos e por fim o LED D6 vermelho fica acesso por 10 segundos, repetindo novamente para o LED D4 verde, LED D5

amarelo e LED D6 vermelho o mesmo ciclo, tal evento só cessará se o botão DESLIGAR for pressionado. Os LEDS D4, D5 e D6 com os resistores R3, R4 e R5 de resistências iguais, podem ser facilmente trocadas pelo semáforo completo conforme observou-se em levantamento de preços.

3.6 Prática 6 – Sinal de Trânsito com Pedestres

Questão 6: Pede-se desenvolver um programa para controlar dois sinais operando em conjunto, um para o trânsito e outro para os pedestres. As configurações para o semáforo principal são: lâmpada verde acenderá por 10 segundos, a lâmpada amarela ligará por 5 segundos e a lâmpada vermelha funcionará por 10 segundos. O sinal de pedestre funcionará de acordo com o convencional. Os sinais terão seus ciclos de operação iniciados por um botão liga. A sequência verde, amarelo e vermelho será repetida indefinidamente, até que um botão de desliga seja pressionado.

Para essa prática podem ser utilizados os seguintes materiais:

TABELA 3.6 - MATERIAIS UTILIZADOS DA PRÁTICA 6.

MATERIAL	QUANTIDADE	PREÇOS	VENDEDOR
PIC16F628A	1 unidade	R$12,51	Baú da Eletrônica
Capacitor Cerâmico 22nF /50V	2 unidades	R$0,10	Baú da Eletrônica
Cristal de Quartzo / 4Mhz	1 unidade	R$0,71	Baú da Eletrônica
Resistor 10k	2 unidade	R$0,07	Baú da Eletrônica
Resistor 220	5 unidades	R$0,14	Baú da Eletrônica
Botão 8x8 mm	2 unidade	R$0,41	Baú da Eletrônica
Semáforo comum	1 unidade	R$600	Mercado Livre
Semaforo pedestre	1 unidade	R$470	Mercado Livre
Placa de Fibra 8x12 cm	1 unidade	R$6,29	Baú da Eletrônica
TOTAL		R$1.091,37	

A montagem da Prática 6 está representada na Figura - 3.6.

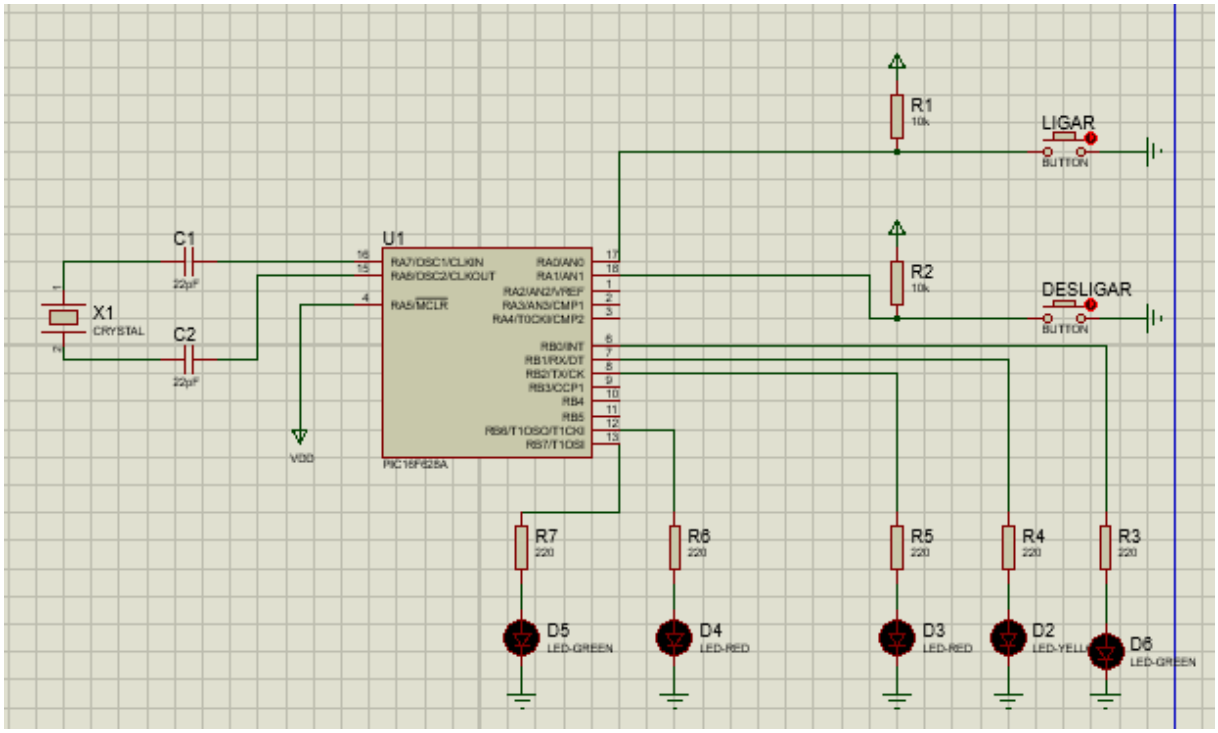

Figura 3.6 – Montagem dos componentes da Prática 6 no Proteus.

O botão LIGAR é responsável por energizar o sistema e por sua vez ligar o LED D6 acenderá por 10 segundos e simultaneamente o LED D4 vermelho também ligará, depois de passado o tempo, o LED D2 amarelo acenderá por 5 segundos, no entanto o LED D4 continuará aceso e por fim teremos o LED D3 vermelho ligando por 10 segundos e de forma síncrona o LED D5 verde acenderá. Isso se repetirá até que o botão DESLIGAR seja acionada, parando assim o sistema. O conjunto de LEDs D5 e D4 e resistores podem ser substituído na implementação física pelo semáforo de pedestres e os LEDS D3, D2 e D6 junto com os resistores podem ser substituídos pelo semáforo comum.

3.7 Prática 7 – Desligar manualmente um motor

Questão 7: Pede-se um programa que permita ligar e desligar manualmente um motor, considerando que ele deve ser automaticamente desativado após quatro acionamentos (significa que o motor será automaticamente desligado na quinta tentativa de ligação). O código prevê reset manual contador.

Para essa prática podem ser utilizados os seguintes materiais:

TABELA 3.7 - MATERIAIS UTILIZADOS DA PRÁTICA 7.

MATERIAL	QUANTIDADE	PREÇOS	VENDEDOR
PIC16F628A	1 unidade	R$12,51	Baú da Eletrônica
Capacitor Cerâmico 22nF /50V	2 unidades	R$0,10	Baú da Eletrônica
Cristal de Quartzo / 4Mhz	1 unidade	R$0,71	Baú da Eletrônica
Resistor 220	3 unidades	R$0,14	Baú da Eletrônica
Resistor 10K	1 unidade	R$0,07	Baú da Eletrônica
Botão 8x8 mm	3 unidade	R$0,41	Baú da Eletrônica
1 relé/ 110V e 220V	1 unidade	R$12,82	Bang good
Placa de Fibra 8x12 cm	1 unidade	R$6,29	Baú da Eletrônica
Motor 1/3 cv 3500 rpm	1 unidade	R$ 235	Mercado Livre
TOTAL		R$269,25	

A montagem da Prática 6 está representada na Figura - 3.7.

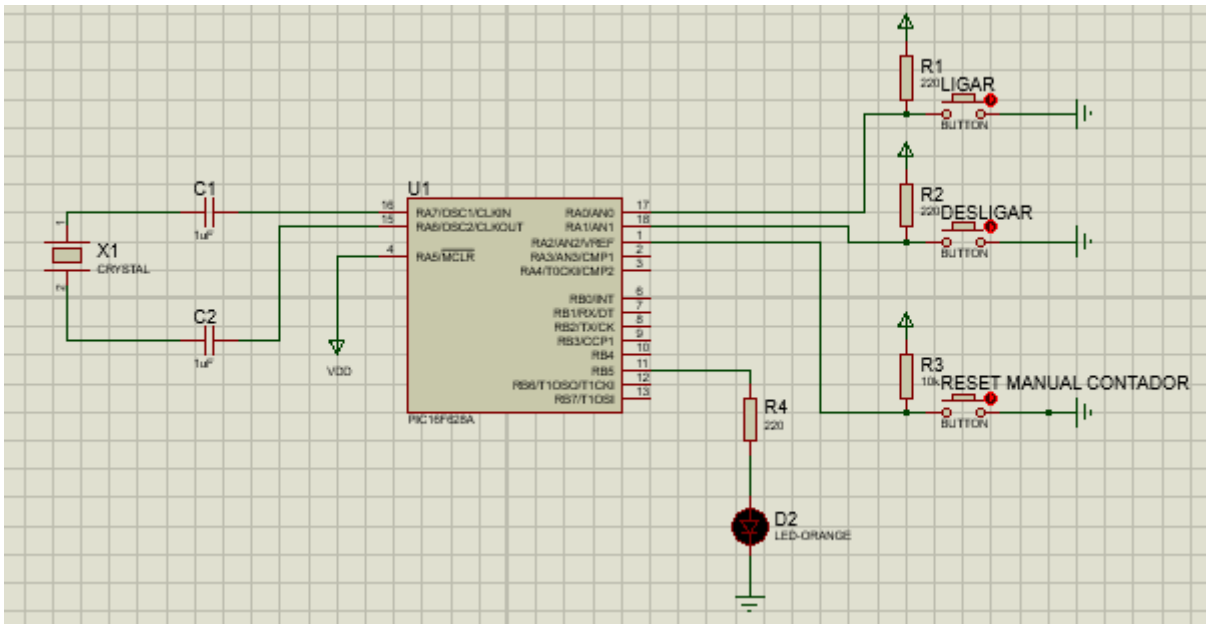

Figura 3.7 – Montagem dos componentes da Prática 7 no Proteus.

O botão LIGAR é responsável por energizar o sistema, acionando o motor LED D2 laranja, o botão DESLIGAR, desliga todo o sistema em caso de emergência (desligando o motor LED D2 laranja), o RESET MANUAL CONTADOR, reseta o número de acionamento feito em cima do botão LIGAR, vale ressaltar que após o quarto acionamento o motor que está identificado como LED D2 laranja não ligará.

3.8 Prática 8 – Acionamento Independente de 3 Motores

Questão 8: Solicita-se elaborar um programa para comandar o acionamento independente de três motores, com a restrição de poder funcionar simultaneamente um número máximo de dois motores.

Para essa prática podem ser utilizados os seguintes materiais:

TABELA 3.8 - MATERIAIS UTILIZADOS DA PRÁTICA 8.

MATERIAL	QUANTIDADE	PREÇOS	VENDEDOR
PIC16F628A	1 unidade	R$12,51	Baú da Eletrônica
Capacitor Cerâmico 22nF /50V	2 unidades	R$0,10	Baú da Eletrônica
Cristal de Quartzo / 4Mhz	1 unidade	R$0,71	Baú da Eletrônica
Resistor 220	3 unidades	R$0,14	Baú da Eletrônica
Resistor 10K	4 unidades	R$0,07	Baú da Eletrônica
Botão 8x8 mm	4 unidades	R$0,41	Baú da Eletrônica
1 relé/ 110V e 220V	3 unidade	R$12,82	Bang good
Placa de Fibra 8x12 cm	1 unidade	R$6,29	Baú da Eletrônica
Motor 1/3 cv 3500 rpm	3 unidades	R$ 235	Mercado Livre
TOTAL		R$765,51	

A montagem da Prática 8 está representada na Figura - 3.8.

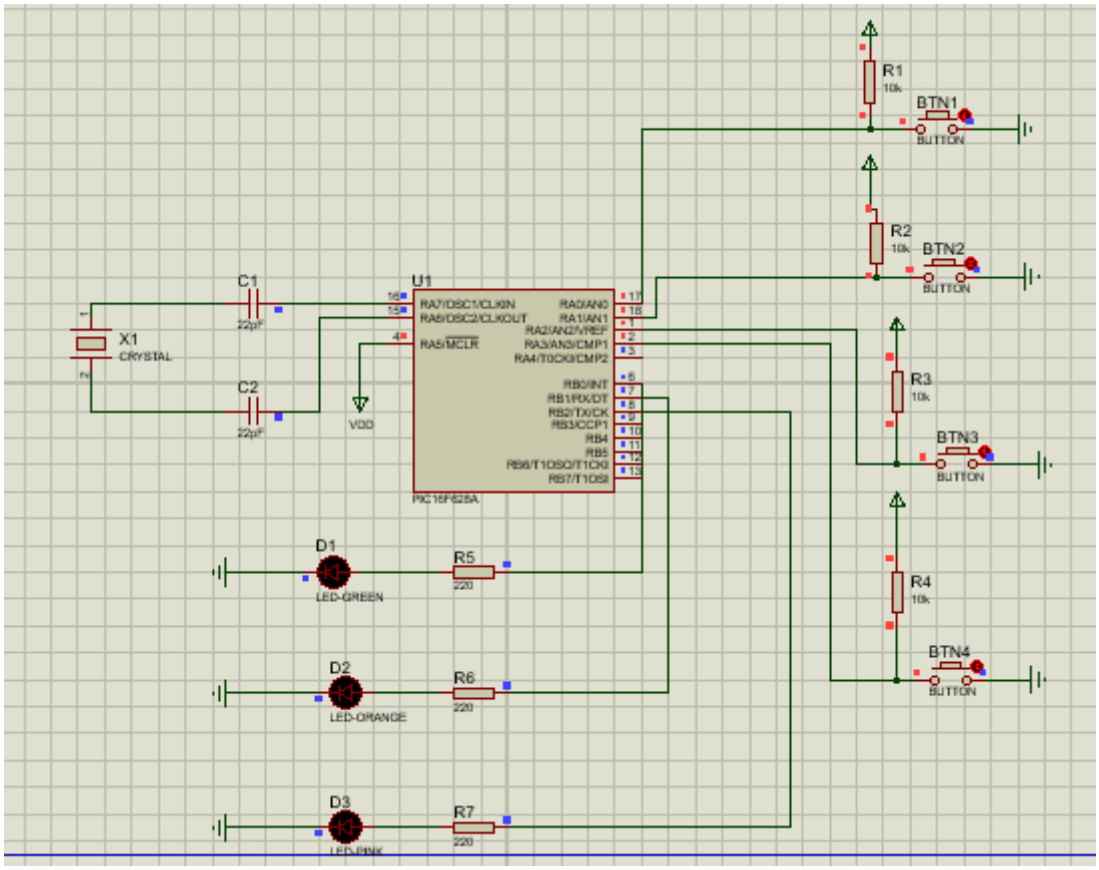

Figura 3.8 – Montagem dos componentes da Prática 8 no Proteus.

O botão BTN1 aciona o motor1 LED D1 verde, o botão BTN2 aciona o motor2 LED D2 laranja, o botão BTN3 aciona o motor3 LED D3 rosa e por ultimo o botão BTN4 reseta meu sistema. Quando ligado dois motores, o próximo botão não conseguirá acionar um terceiro motor.

3.9 Prática 9 – Ligar e Desligar 3 motores

Questão 9: Elabore um programa que permita ligar e desligar três motores, cada motor terá sua chave liga e d e s l i g a. Esses motores poderão funcionar simultaneamente, no entanto, eles só poderão ser ligados em ordem crescente (primeiro M1, segundo M2 e por último M3) e desligados em ordem decrescente (primeiro M3, segundo M2 e por último M1).

Para essa prática podem ser utilizados os seguintes materiais:

TABELA 3.9 - MATERIAIS UTILIZADOS DA PRÁTICA 9.

MATERIAL	QUANTIDADE	PREÇOS	VENDEDOR
PIC16F628A	1 unidade	R$12,51	Baú da Eletrônica
Capacitor Cerâmico 22nF /50V	2 unidades	R$0,10	Baú da Eletrônica
Cristal de Quartzo / 4Mhz	1 unidade	R$0,71	Baú da Eletrônica
Resistor 220	3 unidades	R$0,14	Baú da Eletrônica
Resistor 10K	3 unidades	R$0,07	Baú da Eletrônica
Botão 8x8 mm	3 unidade	R$0,41	Baú da Eletrônica
1 relé/ 110V e 220V	3 unidade	R$12,82	Bang good
Placa de Fibra 8x12 cm	1 unidade	R$6,29	Baú da Eletrônica
Motor 1/3 cv 3500 rpm	3 unidades	R$ 235	Mercado Livre
TOTAL		R$765,03	

A montagem da Prática 9 está representada na Figura - 3.9.

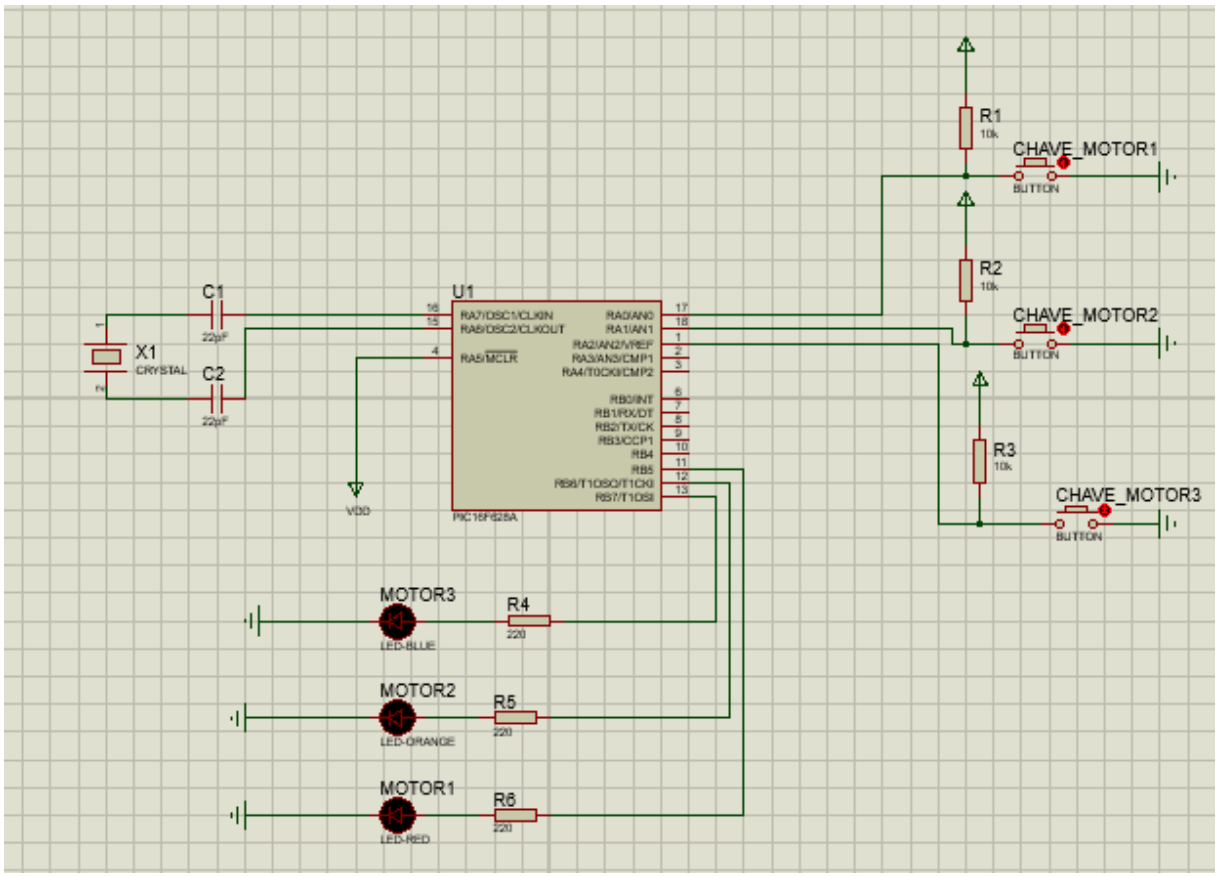

Figura 3.9 – Montagem dos componentes da Prática 9 no Proteus.

A CHAVE_MOTOR1 aciona o MOTOR1, a CHAVE_MOTOR2 aciona o MOTOR2 e a CHAVE_MOTOR3 aciona o MOTOR3. Estes motores só serão acionados de forma crescente e desligados de forma decrescente, sendo assim não serão ligados ou desligados sem essa ordem.

3.10 Prática 10 - Motores

Questão 10: Pede-se elaborar programa que permita acionar e desligar manualmente um motor M1 somente por uma vez. Na segunda tentativa de acionamento M1 será automaticamente desligado, permitindo então que seja ligado por no máximo duas vezes um segundo motor M2. Na terceira tentativa de ligar M2, ele será desligado automaticamente, possibilitando a repetição de todo o ciclo de operação.

Para essa prática podem ser utilizados os seguintes materiais:

TABELA 3.10 - MATERIAIS UTILIZADOS DA PRÁTICA 10.

MATERIAL	QUANTIDADE	PREÇOS	VENDEDOR
PIC16F628A	1 unidade	R$12,51	Baú da Eletrônica
Capacitor Cerâmico 22nF /50V	2 unidades	R$0,10	Baú da Eletrônica
Cristal de Quartzo / 4Mhz	1 unidade	R$0,71	Baú da Eletrônica
Resistor 220	2 unidades	R$0,14	Baú da Eletrônica
Resistor 10K	4 unidades	R$0,07	Baú da Eletrônica
Botão 8x8 mm	4 unidades	R$0,41	Baú da Eletrônica
1 relé/ 110V e 220V	2 unidade	R$12,82	Bang good
Placa de Fibra 8x12 cm	1 unidade	R$6,29	Baú da Eletrônica
Motor 1/3 cv 3500 rpm	2 unidades	R$ 235	Mercado Livre
TOTAL		R$517,55	

A montagem da Prática 10 está representada na Figura - 3.10.

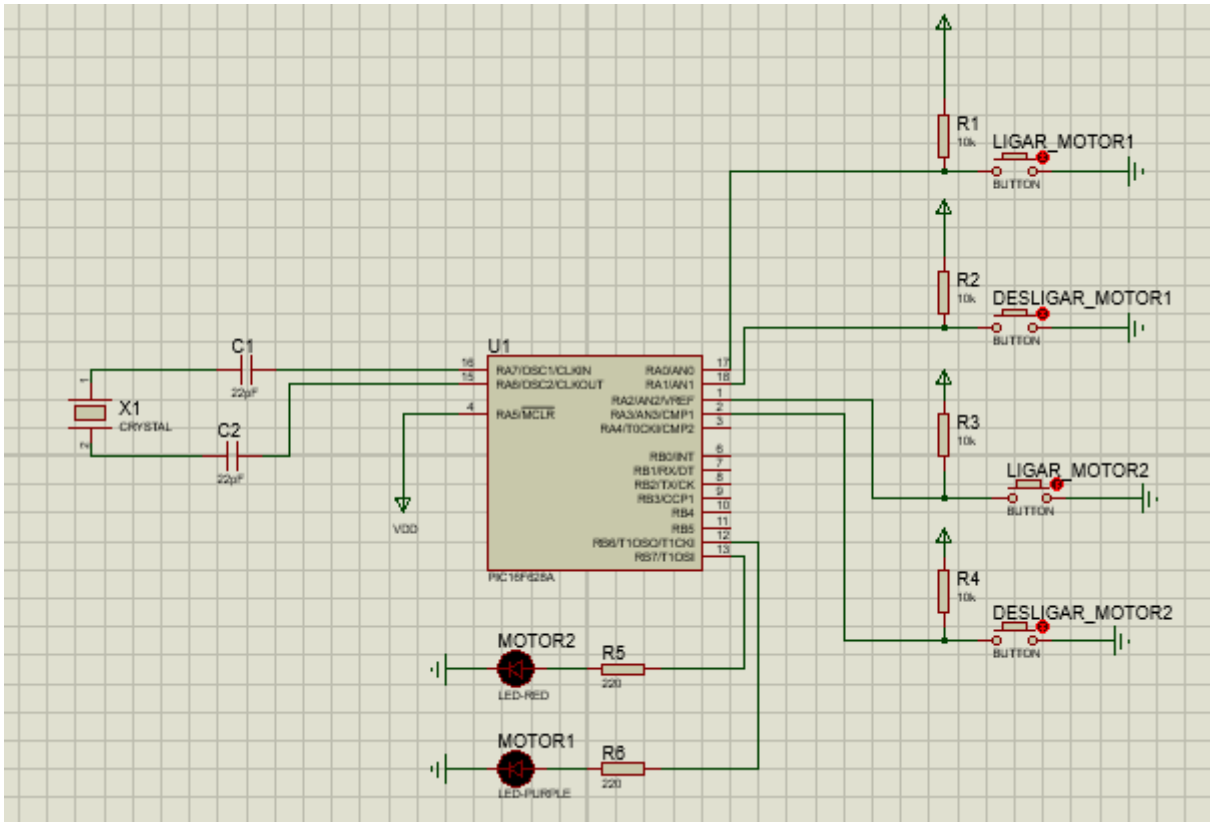

Figura 3.10 – Montagem dos componentes da Prática 10 no Proteus.

O botão LIGAR_MOTOR1 liga o MOTOR1, botão DESLIGAR_MOTOR1 desliga o MOTOR1, caso seja ligado pela segunda vez ele automaticamente irá desligar, podendo assim acionar o botão LIGAR_MOTOR2 que ligará MOTOR2, e o botão DESLIGAR_MOTOR2 desligará o MOTOR2 (o processo de ligar e desligar o motor 2 acontecerá duas vezes, na terceira não funcionará, obedecendo o loop conforme solicitado). Caso o usuário queira tentar de alguma maneira não seguir os seguinte procedimentos informado, nada funcionará, ou seja esses passos são fundamentais para o funcionamento do motor 1 e motor 2.

4 CODIFICAÇÃO

4.1 Questão I

```
;=================================================================
==
; PROJECT DESCRIPTION
;=================================================================
==

;=================================================================
==
;Uma esteira transporta bolas de futebol
;diretamente para o baú de um caminhão. Esse baú
;possui um sensor de nível máximo SA para informar
;quando ele estiver cheio. O processo funciona de tal
;forma que o caminhão precisa estacionar abaixo da
;grande esteira. A presença do veículo é detectada pelo
;sensor SC. Existe também um sensor de presença SB
;sobre a esteira que informa se há bolas depositadas
;sobre ela. Um alarme deve ser acionado sempre que o
;nível alto do sensor SA for detectado, para o motorista
;retirar o caminhão, dando a oportunidade para que
;outro veículo reinicie o processo.
;=================================================================
==
; Main.asm file generated by New Project wizard
;
; Created:   sex abr 19 2019
; Processor: PIC16F628A
; Compiler:  MPASM (Proteus)
; Author: Paulo Henrique Araújo Munhoz
;=================================================================
==

;=================================================================
==
```

```
; DEFINITIONS
;=================================================================
==

#include p16f628a.inc              ; Include register definition file

;CONFIG
; __CONFIG 0xFF61
__CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _MCLRE_ON & _BOREN_ON
& _LVP_OFF & _CPD_OFF & _CP_OFF

#DEFINE BANK0 BCF STATUS, RP0 ; SET BANK 0  OF MEMORY
#DEFINE BANK1 BSF STATUS, RP0; SET BANK 1 OF MEMORY
;=================================================================
==
; VARIABLES
;=================================================================
==

CBLOCK 0x20

ENDC

;=================================================================
==
;INPUTS
;=================================================================
==
#DEFINE  SC  PORTA,2 ; SENSOR C =  TRUCK PRESENCE SENSOR
#DEFINE  SB  PORTA,3 ; SENSOR B  =  BALL  PRESENCE  SENSOR  ON  THE
TRAY
#DEFINE  SA  PORTA,4 ; SENSOR C =  MAXIMUM  HEIGHT

;=================================================================
==
;OUTPUTS
;=================================================================
==
```

```
#DEFINE  BUZZER  PORTB,7


;=================================================================
==
; RESET and INTERRUPT VECTORS
;=================================================================
==
  ORG 0x00
   GOTO BEGIN


   ORG 0X04
   RETFIE

;=================================================================
==
; CODE SEGMENT
;=================================================================
==

BEGIN
            CLRF PORTA ; CLEAN PORTA
            CLRF PORTB; CLEAN PORTB

            BANK1 ; CHANGE FOR 1

            MOVLW B'00000111'
            MOVWF TRISA ; DEFINE RA2,RA3,RA4 AS INPUTS

            MOVLW B'00000000'
            MOVWF TRISB; DEFINE TPORTB  AS OUTPUT

            MOVLW B'10000000'
            MOVWF OPTION_REG

            MOVLW B'00000000'
            MOVWF INTCON ; ALL INTERRUPTIONS OFF
```

```
        BANK0 ;RETORN TO THE BANK 0
        MOVLW B'00000111'
        MOVWF CMCON ; FOR DEFINATION ALL OPERATION ANALOGIC

MAIN
        BCF BUZZER
        GOTO SENSORTRUCK

SENSORTRUCK
        BTFSC SC ; LOOKING FOR TRUCK
        GOTO SENSORTRUCK
        GOTO LOAD

LOAD
          BTFSC SA ; LOOKING FOR TRCUK IS FULL
          GOTO LOADBALL;
          GOTO POOP; TURN ON POOP
LOADBALL
          BTFSC SB; LOOKING FOR BALL
          GOTO LOAD; IFNOT LOKING FOR AGAIN
          GOTO LOAD; LOOKING FOR AGAIN

POOP
          BSF BUZZER ; TURN ON POOP
          BTFSC SC
          GOTO MAIN
          GOTO POOP
 END
```

4.2 Questão 2

```
;=================================================================
==
; PROJECT DESCRIPTION
;=================================================================
==
```

```
;==========================================================================
;Faça um sistema de contagem de peças
;defeituosas que ao ligar o sistema, uma esteira deverá
;ser ativada. A cada 10 peças defeituosas detectadas,
;ativar um sinal de alarme por 10 segundos. A cada 20
;peças defeituosas detectadas, o sinal de alarme será
;de 20 segundos e a linha de produção deverá parar.
;Incluir um botão de reinício do sistema.

;==========================================================================

;==========================================================================
; Main.asm file generated by New Project wizard
;
; Created:   dom abr 21 2019
; Processor: PIC16F628A
; Compiler:  MPASM (Proteus)
; Author: Paulo Henrique Araújo Munhoz
;==========================================================================

;==========================================================================
; DEFINITIONS
;==========================================================================

#include p16f628a.inc             ; Include register definition file

;CONFIG
; __CONFIG 0xFF61
__CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _MCLRE_ON & _BOREN_ON
& _LVP_OFF & _CPD_OFF & _CP_OFF

#DEFINE BANK0 BCF STATUS, RP0; SET  BANK 0 OF THE MEMORY
#DEFINE BANK1 BSF STATUS, RP0; SET  BANK 1 OF THE MEMORY
```

```
;=================================================================
==
; VARIABLES
;=================================================================
==

  CBLOCK 0x20               ; INITIAL MEMORY ADDRESS

   PARTS
   D
   d1
   d2
   W_TEMP
   STATUS_TEMP


  ENDC                      ; END MEMORY BLOCK

;=================================================================
==
; INPUTS
;=================================================================
==
#DEFINE          RESET              PORTA,0        ; BUTTON RESET
#DEFINE          TURNON     PORTA,1              ; BUTTON TURN ON
#DEFINE          ISSUE              PORTB,0        ; BUTTON COUNT ISSUES


;=================================================================
==
; OUTPUTS
;=================================================================
==

#DEFINE          LED        PORTB,4
#DEFINE          MAT        PORTB,5    ; RUNNING MACHINE

;=================================================================
==
```

```
; RESET and INTERRUPT VECTORS
;=================================================================
==
  ORG 0x00

  GOTO BEGIN


  ORG 0X04

  ;-------------------SAVE THE STATUS--------------------

  MOVWF     W_TEMP
  SWAPF     STATUS,W

  BANK0
  MOVWF     STATUS_TEMP
  ;-----------------END STATUS SAVING------------

  BTFSS     INTCON, INTF              ; IF THERE EXTERNAL INTERRUPTION
  GOTO          EXIT_ISR                        ;  IF  NOT,  CHANGE  FOR
INTERRUPTION OUT
  GOTO          ISSUES20                    ; GO  ISSUE



  COMF              PORTB, 0

EXIT_ISR
  SWAPF STATUS_TEMP, W
  MOVWF     STATUS
  SWAPF     W_TEMP, F
  SWAPF     W_TEMP,W


  RETFIE

;=================================================================
==
```

```
; CODE SEGMENT
;=================================================================
==

BEGIN
  CLRF      PORTA              ; CLEAN PORTA
  CLRF      PORTB              ; CLEAN PORTB

  BANK1                  ; CHANGE FOR THE BANK1

  MOVLW B'00000011'
  MOVWF     TRISA            ; DEFINE RA0 AND RA1 LIKE INPUTS AND ANOTHES
OUTPUTS

  MOVLW B'00000001'
  MOVWF     TRISB            ; DEFINE ALL PORTB LIKE OUTPUTS

  MOVLW B'10000000'
  MOVWF OPTION_REG           ; PRESCALER 1:2 NO
                                        ; PULL-UPS OFF
                                        ; THE OTHERS SETTINGS ARE
IRRELEVANT
  BSF OPTION_REG,6                  ; SET UPS EXTERNAL INTERRUPTIONS FOR
RISE EDGE
  MOVLW B'00000001'
  MOVWF     INTCON                  ; ALL INTERRUPTIONS OFF

  BANK0                             ; RETURN BANK0
  MOVLW     H'90'
  MOVLW     INTCON
  BCF       PORTB,4
  MOVLW     B'00000111'
  MOVWF     CMCON                   ; DEFINE MODE OF OPERATION ANALOG
COMPARATOR

  movlw     D'10'                   ; MOVE VALOR FOR W
  movwf     PARTS                       ; INITIALIZE VARIABLE OF TIME
```

```
MAIN
  BTFSS     TURNON
  GOTO          FBEON
  GOTO          MAIN

FBEON
  BSF       MAT
  BTFSS     ISSUE
  GOTO              FDEBOUNCE
  BTFSS     RESET
  GOTO              FSTOP
  GOTO              FBEON

FDEBOUNCE
  BTFSC     ISSUE
  GOTO              DECREMENTS
  GOTO              FDEBOUNCE

DECREMENTS
  DECFSZ    PARTS                   ; DECREMENT TIME UNTIL 0
  GOTO              FBEON
  GOTO      ALERT10

ALERT10
    movlw   D'10'           ; MOVE THE VALUE TO W
    movwf   PARTS
    BSF         LED
    CALL    DELAY10S

    GOTO    ISSUES20

ISSUES20
  BCF       LED
  BTFSC     ISSUE
  GOTO              ISSUES20
  GOTO              FDEBOUNCE20

FDEBOUNCE20
  BTFSC     ISSUE
```

```
  GOTO          DECREMENTS20
  GOTO          FDEBOUNCE20

DECREMENTS20
  DECFSZ   PARTS                ; DECREMENT TIME UNTIL 0
  GOTO          ISSUES20
  GOTO     ALERT20



ALERT20
   BSF          LED          ; TURN ON 20S ALERT
   CALL     DELAY10S
   CALL     DELAY10S
   BCF          MAT
   GOTO     FSTOP

FSTOP
  BCF       LED
  BCF       MAT
  GOTO          BEGIN

DELAY10S
  CALL          DELAY500MS
  CALL          DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
```

```
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
  CALL            DELAY500MS
RETURN


DELAY500MS:

  MOVLW   D'200'          ; MOVE THE VALUE TO W
  MOVWF   d1                  ; INITIALIZE THE TIME0 VARIABLE
                                  ; 4 MACHINE CYCLES
aux1:
  MOVLW   D'250'          ; MOVE THE VALUE TO W
  MOVWF   d2              ; INITIALIZE THE TIME1 VARIABLE

                                  ; 2 MACHINE CYCLES
aux2:
  NOP                     ; SPEND 1 MACHINE CYCLE
  NOP
  NOP
  NOP
  NOP



  BTFSS   RESET
  GOTO            FSTOP


  DECFSZ  d2                  ; DECREMENT TIME1 UNTIL 0
  GOTO            aux2            ; GO TO LABEL AUX2

                          ; 250 x 10 MACHINE CYCLES = 2500 CYCLES

  DECFSZ  d1                  ; DECREMENT TIME0 UNTIL 0
  GOTO            aux1        ; GO TO LABEL AUX1

                                  ; 3 MACHINE CYCLES
```

```
                                        ; 2500 x 200 = 500000

  RETURN                                ; RETURN AFTER THE SUB ROUTINE CALL



END
```

4.3 Questão 3

```
;=================================================================
==
; PROJECT DESCRIPTION
;=================================================================
==

;=================================================================
==
;  Uma fábrica de sucos bolou um processo
; para preencher automaticamente suas garrafas, cujo
; esquema é apresentado abaixo:O tanque armazena grande quantidade
; de suco. A válvula V abre sempre que o sensor SO2 detectar que
; há uma garrafa debaixo dela. O motor M da esteira se movimenta
; toda vez que o sensor SO1 detectar a presença de garrafas sobre
; ela. Quando o sensor SO2 detectar a presença de garrafas, o
; motor M deve parar, voltando a funcionar somente se o sensor SS
; determinar que o líquido depositado na garrafa atingiu o nível
; máximo. Se o sensor SL detectar que o nível de suco está mínimo
; no tanque, a válvula V deve fechar, o motor M deve parar e a
; bomba B1 preencherá o tanque com suco até o nível máximo, que
; será detectado por SH, quando então B1 sairá de operação. Inclua
; contatos que permitam ligar e desligar manualmente a bomba e o
; motor da esteira para efeitos de teste. Desenvolva um código que
; realize a automação desse processo.

;=================================================================
==
```

```
;=================================================================
==
; Main.asm file generated by New Project wizard
;
; Created:   dom abr 21 2019
; Processor: PIC16F628A
; Compiler:  MPASM (Proteus)
; Author: Paulo Henrique Araújo Munhoz
;=================================================================
==

;=================================================================
==
; DEFINITIONS
;=================================================================
==

#include p16f628a.inc             ; Include register definition file

;CONFIG
; __CONFIG 0xFF61
__CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _MCLRE_ON & _BOREN_ON
& _LVP_OFF & _CPD_OFF & _CP_OFF


#DEFINE    BANK0    BCF   STATUS, RP0      ; DEFINE THE MEMORY BANK 0
#DEFINE    BANK1    BSF   STATUS, RP0      ; DEFINE THE MEMORY BANK 1


;=================================================================
==
; VARIABLES
;=================================================================
==
  CBLOCK 0x20
  d1
  d2
```

```
  ENDC

;=================================================================
==
; INPUTS
;=================================================================
==
#DEFINE    S01             PORTA,0    ; SYSTEM TURN ON
#DEFINE    S02             PORTA,1    ; SYSTEM TURN OFF
#DEFINE    SS              PORTA,2      ; IF THE RUNNING MACHINE HAS SOME
BOTTLE
#DEFINE    TURNON          PORTA,3    ; IF THE BOTTLE IS IN POSITION
#DEFINE    TURNOFF         PORTA,4    ; VERIFY IF THE BOTTLE IS FULL
#DEFINE    SH              PORTB,0    ; VERIFY IF THE TANK IS EMPTY - SS(0), SH(1)
#DEFINE    SLSH            PORTB,1    ; VERIFY IF THE TANK IS FULL
;=================================================================
==
; OUTPUTS
;=================================================================
==
#DEFINE    MAT                    PORTB,5    ; RUNNING MACHINE
#DEFINE    V               PORTB,6    ; FILL THE BOTTLE
#DEFINE    B1              PORTB,7    ; FILL THE TANK

;=================================================================
==
; RESET and INTERRUPT VECTORS
;=================================================================
==
  ORG 0x00
  GOTO FBEGIN


  ORG 0X04
  RETFIE

;=================================================================
==
; CODE SEGMENT
```

```
;=================================================================
==

FBEGIN
  CLRF                  PORTA    ; CLEAN THE PORTA
  CLRF                  PORTB    ; CLEAN THE PORTB

  BANK1          ; SWITCH TO BANK 1

  MOVLW    B'00011111'
  MOVWF                 TRISA    ; DEFINY RA0, RA1 AND RA2 AS INPUT AND THE
OTHERS AS OUTPUTS

  MOVLW    B'00000001'
  MOVWF                 TRISB    ; DEFINE THE PORTB AS OUTPUT

  MOVLW    B'10000000'
  MOVWF    OPTION_REG  ; PRESCALER 1:2 NO
              ; PULL-UPS OFF
              ;THE OTHERS CPNFG. WAS IRRELEVANTS

  MOVLW    B'00000000'
  MOVWF                 INTCON   ; ALL THE INTERRUPTIONS SHOULD BE OFF

  BANK0          ; RETURN TO BANK 0
  MOVLW    B'00000111'
  MOVWF                 CMCON      ; DEFINE THE OPERATION MODE OF THE
ANALOG COMPARATOR

MAIN

  BTFSS       TURNON
  GOTO        FROUTINE

  GOTO        MAIN

FROUTINE
  BTFSS       TURNOFF
  GOTO        FTURNOFF
```

```
  BTFSS     S01
  GOTO      FVALVE

  GOTO      FROUTINE


 FVALVE
  BTFSS     TURNOFF
  GOTO      FTURNOFF
  BSF       MAT
  BTFSC     S02
  GOTO      FVALVE
  GOTO      FFILL

 FFILL
  BSF       V
  BCF       MAT
  GOTO      FFULL

FFULL
  BTFSC     SS
  GOTO      FFULL
  BCF       V
  CALL      DELAY500MS
  BSF       MAT
  GOTO      FTANK

FTANK
  BTFSC     SLSH
  GOTO      FDONE
  BSF       B1
  BCF       MAT
  GOTO      FTANK

FDONE
  BCF       B1
  GOTO      FROUTINE

FTURNOFF
```

```
  BCF         MAT
  BCF         V
  BCF         B1
  GOTO        MAIN

DELAY500MS:

  MOVLW     D'200'   ; MOVE THE VALUE TO W
  MOVWF     d1       ; INITIALIZE THE VARIABLE TIME0
                              ;      4 MACHINE CYCLES
aux1:
  MOVLW     D'250'   ; MOVE THE VALUE TO W
  MOVWF     d2       ; INITIALIZE THE VARIABLE TIME1

               ; 2 MACHINE CYCLES
aux2:
  NOP             ; SPEND 1 MACHINE CYCLES
  NOP
  NOP
  NOP
  NOP
  NOP
  NOP
  NOP
  NOP



  DECFSZ   d2       ; DECREMENT TIME1 UNTIL 0
  GOTO     aux2     ; GO TO LABEL AUX2

               ; 250 x 10 MACHINE CYCLES = 2500 CYCLES

  DECFSZ   d1       ; DECREMENT TIME0 UNTIL 0
  GOTO     aux1     ; GO TO LABEL AUX1

               ; 3 MACHINE CYCLES

               ; 2500 x 200 = 500000
```

```
  RETURN          ; RETURN AFTER THE SUB ROUTINE CALL



  END
```

4.4 Questão 4

```
;=================================================================
==
; PROJECT DESCRIPTION
;=================================================================
==

;=================================================================
==
;Deseja-se comprar um elevador externo
;para uma indústria.
;O elevador consiste em um
;motor, capaz de realizar movimentos de ascensão e
;descida sobre a plataforma transportadora. Para
;prevenir colisões sobre o solo ou contra o topo do
;elevador, são utilizados sensores de nível baixo e alto.
;Haverá também um botão de parada que cessa os
;movimentos do elevador. Pede-se elaborar um
;programa que realize a automação desse processo.
;=================================================================
==

;=================================================================
==
; Main.asm file generated by New Project wizard
;
; Created:   dom abr 21 2019
; Processor: PIC16F628A
; Compiler:  MPASM (Proteus)
; Author: Paulo Henrique Araújo Munhoz
;=================================================================
==
```

```
;=================================================================
==
; DEFINITIONS
;=================================================================
==

#include p16f628a.inc             ; Include register definition file

;CONFIG
; __CONFIG 0xFF61
__CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _MCLRE_ON & _BOREN_ON
& _LVP_OFF & _CPD_OFF & _CP_OFF

#DEFINE    BANK0      BCF   STATUS, RP0    ; DEFINE THE MEMORY BANK 0
#DEFINE    BANK1      BSF   STATUS, RP0    ; DEFINE THE MEMORY BANK 1


;=================================================================
==
; VARIABLES
;=================================================================
==
  CBLOCK 0x20


  ENDC

;=================================================================
==
; INPUTS
;=================================================================
==
#DEFINE    DOWNSENSOR        PORTA,0    ; RAISE LEVEL SENSOR
#DEFINE    UPSENSOR          PORTA,1    ; DOWN LEVEL SENSOR
#DEFINE    STOP              PORTA,2    ; STOP THE SYSTEM
#DEFINE    DOWN              PORTA,4    ; MAXIMUM LEVEL
#DEFINE    UP                      PORTA,3    ; BALLS ON THE MAT
```

```
;=================================================================
==
; OUTPUTS
;=================================================================
==
#DEFINE    MOTORS        PORTB,6
#DEFINE    MOTORD        PORTB,7

;=================================================================
==
; RESET and INTERRUPT VECTORS
;=================================================================
==
  ORG 0x00
  GOTO FBEGIN


  ORG 0X04
  RETFIE

;=================================================================
==
; CODE SEGMENT
;=================================================================
==

FBEGIN
  CLRF        PORTA    ; CLEAN THE PORTA
  CLRF        PORTB    ; CLEAN THE PORTB

  BANK1          ; SWITCH TO BANK 1

  MOVLW    B'00011111'
  MOVWF    TRISA      ; DEFINE RA0, RA1 AND RA2 AS INPUTS AND THE OTHERS
AS OUTPUTS

  MOVLW    B'00000000'
  MOVWF    TRISB    ; DEFINE EVERY PORTB AS OUTPUT
```

```
  MOVLW    B'10000000'
  MOVWF    OPTION_REG  ; PRESCALER 1:2 NO
                ; PULL-UPS OFF
                     ;THE OTHERS CPNFG. WAS IRRELEVANTS

  MOVLW    B'00000000'
  MOVWF    INTCON      ; ALL THE INTERPRETATIONS OFF

  BANK0           ; GO BACK TO BANK 0
  MOVLW    B'00000111'
  MOVWF      CMCON          ; DEFINE THE OPERATION MODE OF THE ANALOG
COMPARATOR

MAIN

BTFSS          STOP
GOTO           FSTOPBUTTON
BTFSS          UPSENSOR
CALL                    FHIGH
BTFSC          DOWNSENSOR
GOTO           FROUTINE
CALL                    FLOWER

FROUTINE
  BTFSS        UP
  GOTO         FGOHIGH
  BTFSS        DOWN
  GOTO         FGODOWN

  GOTO      MAIN


FGOHIGH
  BTFSS        UPSENSOR
  GOTO         FHIGH
  BCF          MOTORD
  BSF          MOTORS
  GOTO         MAIN
```

```
FGODOWN
  BTFSS       DOWNSENSOR
  GOTO        FLOWER
  BCF         MOTORS
  BSF         MOTORD
  GOTO        MAIN


FHIGH
  BCF         MOTORS

  RETURN


 FLOWER
  BCF         MOTORD
  RETURN


FSTOPBUTTON
  BCF               MOTORS
  BCF               MOTORD
  GOTO        FBEGIN

   END
```

4.5 Questão 5

```
;=================================================================
==
; PROJECT DESCRIPTION
;=================================================================
==

;=================================================================
==
; Pede-se criar programa para controlar um sinal
;de trânsito convencional que tenha a lâmpada verde acesa
```

```
;por 10 segundos, a lâmpada amarela acesa por 5 segundos e
;a lâmpada vermelha acesa por 10 segundos. O sinal terá seu
;ciclo de operação iniciado por um botão liga. A sequência
;verde, amarelo e vermelho será repetida indefinidamente, até
;que um botão de desliga seja pressionado.

;=================================================================
==
; Main.asm file generated by New Project wizard
;
; Created:   dom abr 21 2019
; Processor: PIC16F628A
; Compiler:  MPASM (Proteus)
; Author: Paulo Henrique Araújo Munhoz
;=================================================================
==

;=================================================================
==
; DEFINITIONS
;=================================================================
==

#include p16f628a.inc             ; Include register definition file
;CONFIG
; __CONFIG 0xFF61
__CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _MCLRE_ON & _BOREN_ON
& _LVP_OFF & _CPD_OFF & _CP_OFF

#DEFINE    BANK0   BCF   STATUS, RP0   ; DEFINE THE MEMORY BANK 0
#DEFINE    BANK1   BSF   STATUS, RP0   ; DEFINE THE MEMORY BANK 1


;=================================================================
==
; VARIABLES
;=================================================================
==
  CBLOCK 0x20
```

```
   d1
   d2


  ENDC

;=====================================================================
==
; INPUTS
;=====================================================================
==
#DEFINE    START          PORTA,0   ; IF THE CAR IS NEAR
#DEFINE    TURNOFF        PORTA,1   ; TURN OFF SYSTEM


;=====================================================================
==
; OUTPUTS
;=====================================================================
==
#DEFINE    GREENLED           PORTB,5
#DEFINE    YELLOWLED          PORTB,6
#DEFINE    REDLED             PORTB,7
;=====================================================================
==
; RESET and INTERRUPT VECTORS
;=====================================================================
==
  ORG    0x00
  GOTO   FBEGIN


  ORG    0X04
  RETFIE

;=====================================================================
==
; CODE SEGMENT
```

```
;=================================================================
=

FBEGIN:
  CLRF    PORTA      ; CLEAN THE PORTA
  CLRF    PORTB      ; CLEAN THE PORTB

  BANK1              ; SWITCH TO BANK 1

  MOVLW  B'00000011'
  MOVWF  TRISA       ; DEFINE RA0, RA1 AND RA2 AS INPUTS AND THE OTHERS
AS OUTPUTS

  MOVLW  B'00000000'
  MOVWF  TRISB       ; DEFINE EVERY PORTB AS OUTPUT

  MOVLW  B'10000000'
  MOVWF  OPTION_REG   ; PRESCALER 1:2 NO
                ; PULL-UPS TURNED OFF
                ; THE OTHERS CPNFG. WAS IRRELEVANTS

  MOVLW  B'00000000'
  MOVWF  INTCON      ; ALL THE INTERPRETATIONS TURNED OFF

  BANK0              ; RETURN TO THE BANK 0
  MOVLW  B'00000111'
  MOVWF   CMCON            ; DEFINE THE OPERATION MODE OF THE ANALOG
COMPARATOR

MAIN:
  BTFSS       START
  CALL        FTRAFFICLIGHT
  BTFSS       TURNOFF
  CALL        FSTOP
  GOTO        MAIN



 FTRAFFICLIGHT
```

```
  BSF         GREENLED
  CALL    DELAY10S
  BCF         GREENLED
  BSF         YELLOWLED
  CALL    DELAY5S
  BCF         YELLOWLED
  BSF         REDLED
  CALL    DELAY10S
  BCF         REDLED
  RETURN



FSTOP:
  BCF         GREENLED
  BCF         YELLOWLED
  BCF         REDLED
  GOTO    FBEGIN

DELAY10S
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
```

```
  CALL  DELAY500MS
  RETURN

DELAY5S
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  CALL  DELAY500MS
  RETURN

DELAY500MS:

  MOVLW  D'200'  ; MOVE THE VALUE TO W
  MOVWF  d1      ; INITIALIZE THE TIME 0
              ; 4 MACHINE CYCLES
aux1:
  MOVLW  D'250'  ; MOVE THE VALUE TO W
  MOVWF  d2      ; INITIALIZE THE TIME 1

              ; 2 MACHINE CYCLES
aux2:
     NOP        ; SPEND 1 MACHINE CYCLE
     NOP
     NOP
     NOP
     NOP



BTFSS          TURNOFF
GOTO                FSTOP

 DECFSZ   d2      ; DECREMENT THE TIME 1 UNTIL 0
 GOTO   aux2      ; GO TO LABEL AUX 2
```

```
            ; 250 x 10 MACHINE CYCLES = 2500 CYCLES

 DECFSZ   d1      ; DECREMENT THE TIME 0 UNTIL 0
 GOTO   aux1      ; GO TO LABEL AUX 1

            ; 3 MACHINE CYCLES

            ; 2500 x 200 = 500000

RETURN          ; RETURN AFTER THE SUB ROUTINE CALL

END
```

4.6 Questão 6

```
;=================================================================
==
; PROJECT DESCRIPTION
;=================================================================
==

;=================================================================
==
; Pede-se desenvolver um programa para
;controlar dois sinais operando em conjunto, um para o
;trânsito e outro para os pedestres. As configurações para o
;semáforo principal são: lâmpada verde acenderá por 10
;segundos, a lâmpada amarela ligará por 5 segundos e a
;lâmpada vermelha funcionará por 10 segundos. O sinal de
;pedestre funcionará de acordo com o convencional. Os sinais
;terão seus ciclos de operação iniciados por um botão liga. A
;sequência verde, amarelo e vermelho será repetida
;indefinidamente, até que um botão de desliga seja
;pressionado.

;=================================================================
==
; Main.asm file generated by New Project wizard
;
```

```
; Created:   dom abr 21 2019
; Processor: PIC16F628A
; Compiler:  MPASM (Proteus)
; Author: Paulo Henrique Araújo Munhoz
;=====================================================================
==

;=====================================================================
==
; DEFINITIONS
;=====================================================================
==

#include p16f628a.inc             ; Include register definition file

;CONFIG
; __CONFIG 0xFF61
__CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _MCLRE_ON & _BOREN_ON
& _LVP_OFF & _CPD_OFF & _CP_OFF

#DEFINE    BANK0    BCF   STATUS, RP0    ; DEFINE THE MEMORY BANK 0
#DEFINE    BANK1    BSF   STATUS, RP0    ; DEFINE THE MEMORY BANK 1



;=====================================================================
==
; VARIABLES
;=====================================================================
==
  CBLOCK 0x20

   d1
   d2


  ENDC

;=====================================================================
==
```

```
; INPUTS
;=================================================================
==
#DEFINE    BEGIN                    PORTA,0  ; CAR IS NEAR
#DEFINE    TURNOFF        PORTA,1  ; TURN OFF


;=================================================================
==
; OUTPUTS
;=================================================================
==
#DEFINE    GREENLED     PORTB,0
#DEFINE    YELLOWLED    PORTB,1
#DEFINE    REDLED         PORTB,2
#DEFINE    REDLEDP        PORTB,6
#DEFINE    GREENLEDP    PORTB,7
;=================================================================
==
; RESET and INTERRUPT VECTORS
;=================================================================
==
  ORG      0x00
  GOTO     FBEGIN


  ORG      0X04
  RETFIE

;=================================================================
==
; CODE SEGMENT
;=================================================================
=

FBEGIN:
  CLRF     PORTA         ; CLEAN THE PORTA
  CLRF     PORTB         ; CLEAN THE PORTB
```

```
  BANK1               ; SWITCH TO BANK 1

  MOVLW   B'00000011'
  MOVWF   TRISA       ; DEFINE RA0, RA1 AND RA2 AS INPUTS AND THE OTHERS
AS OUTPUTS

  MOVLW   B'00000000'
  MOVWF   TRISB       ; DEFINE EVERY PORTB AS OUTPUTS

  MOVLW   B'10000000'
  MOVWF   OPTION_REG    ; PRESCALER 1:2 NO
                        ; PULL-UPS TURNED OFF
                        ; THE OTHERS CPNFG. WAS IRRELEVANTS

  MOVLW   B'00000000'
  MOVWF   INTCON      ; ALL THE INTERPRETATIONS ARE TURNED OFF

  BANK0               ; RETURN TO THE BANK 0
  MOVLW   B'00000111'
  MOVWF    CMCON          ; DEFINE THE OPERATION MODE OF THE ANALOG
COMPARATOR

MAIN:

  BTFSS       BEGIN
  GOTO        FTRAFFICLIGHT
  BTFSS       TURNOFF
  GOTO        FSTOP
  GOTO              MAIN



 FTRAFFICLIGHT
  BSF         GREENLED
  BSF         REDLEDP
  CALL        DELAY10S
  BCF         GREENLED
  BSF         YELLOWLED
  CALL        DELAY5S
```

```
  BCF     YELLOWLED
  BSF     REDLED
  BCF     REDLEDP
  BSF     GREENLEDP
  CALL    DELAY10S
  BCF     REDLED
  BCF     GREENLEDP
  GOTO    FTRAFFICLIGHT


FSTOP:
  BCF     GREENLED
  BCF     YELLOWLED
  BCF     REDLED
  BCF     GREENLEDP
  BCF     REDLEDP
  GOTO    FBEGIN

DELAY10S
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
```

```
    RETURN

DELAY5S
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    CALL  DELAY500MS
    RETURN

DELAY500MS:

  MOVLW D'200'     ; MOVE THE VALUE TO W
  MOVWF d1         ; INITIALIZE THE TIME 0
                   ; 4 MACHINE CYCLES
aux1:
  MOVLW D'250'     ; MOVE THE VALUE TO W
  MOVWF d2         ; INITIALIZE THE TIME 1

                   ; 2 MACHINE CYCLES
aux2:
    NOP            ; SPEND 1 MACHINE CYCLE
    NOP
    NOP
    NOP
    NOP



BTFSS   TURNOFF
GOTO    FSTOP

 DECFSZ     d2    ; DECREMENT THE TIME 1 UNTIL 0
 GOTO    aux2     ; GO TO LABEL AUX2
```

```
            ; 250 x 10 MACHINE CYCLES = 2500 CYCLES

 DECFSZ     d1    ; DECREMENT THE TIME 0 UNTIL 0
 GOTO    aux1     ; GO TO LABEL AUX1

            ; 3 MACHINE CYCLES

            ; 2500 x 200 = 500000

  RETURN         ; RETURN AFTER THE SUB ROUTINE CALL



  END
```

4.7 Questão 7

```
;=================================================================
==
; PROJECT DESCRIPTION
;=================================================================
==

;=================================================================
==
;  Pede-se um programa que permita ligar e
;desligar manualmente um motor, considerando que ele
;deve ser automaticamente desativado após quatro
;acionamentos (significa que o motor será
;automaticamente desligado na quinta tentativa de
;ligação). O código prevê reset manual contador.


;=================================================================
==
; Main.asm file generated by New Project wizard
;
; Created:   dom abr 21 2019
; Processor: PIC16F628A
```

```
; Compiler:  MPASM (Proteus)
; Author: Paulo Henrique Araújo Munhoz
;=================================================================
==

;=================================================================
==
; DEFINITIONS
;=================================================================
==

#include p16f628a.inc             ; Include register definition file
;CONFIG
; __CONFIG 0xFF61
__CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _MCLRE_ON & _BOREN_ON
& _LVP_OFF & _CPD_OFF & _CP_OFF

#DEFINE     BANK0       BCF STATUS, RP0   ; DEFINE THE MEMORY BANK 0
#DEFINE     BANK1       BSF STATUS, RP0   ; DEFINE THE MEMORY BANK 1


;=================================================================
==
; VARIABLES
;=================================================================
==
  CBLOCK 0x20   ; ADDRESS OF THE USER MEMORY

   ACTIONS

   W_TEMP
   STATUS_TEMP


  ENDC        ; END OF MEMORY BLOCK

;=================================================================
==
; INPUTS
```

```
;=================================================================
==
#DEFINE    TURNON        PORTA,0  ; TURN ON THE MOTOR
#DEFINE    TURNOFF       PORTA,1  ; TURN OFF THE MOTOR
#DEFINE    RESET         PORTA,2  ; RESET THE COUNTER


;=================================================================
==
; OUTPUTS
;=================================================================
==

#DEFINE    MOTOR      PORTB,5


;=================================================================
==
; RESET and INTERRUPT VECTORS
;=================================================================
==
  ORG 0x00

  GOTO                FBEGIN


  ORG 0X04

  ;-------------------SAVE THE STATUS--------------------

  MOVWF    W_TEMP
  SWAPF      STATUS,W

  BANK0
  MOVWF    STATUS_TEMP
  ;-----------------END OF STATUS SAVING----------------

  BTFSS   INTCON, INTF     ; IF EXTERNAL INTERRUPTION
  GOTO    EXIT_ISR         ; IF NO, CHANGE THE INTERRUPTION OUTPUTS
```

```
EXIT_ISR
  SWAPF     STATUS_TEMP, W
  MOVWF     STATUS
  SWAPF     W_TEMP, F
  SWAPF     W_TEMP,W


  RETFIE

;=================================================================
==
; CODE SEGMENT
;=================================================================
==

FBEGIN
  CLRF    PORTA       ; CLEAN THE PORTA
  CLRF    PORTB       ; CLEAN THE PORTB

  BANK1               ; SWITCH TO BANK 1

  MOVLW   B'00000111'
  MOVWF   TRISA       ; DEFINE RA0 AND RA1 AS INPUTS AND THE OTHERS AS
OUTPUTS

  MOVLW   B'00000000'
  MOVWF   TRISB       ; DEFINE EVERY PORTB AS OUTPUT

  MOVLW   B'10000000'
  MOVWF   OPTION_REG    ; PRESCALER 1:2 NO
                        ; PULL-UPS TURNED OFF
                        ; THE OTHERS CONFG. WAS IRRELEVANTS
  BSF       OPTION_REG,6    ; CONFIGURE EXTERNAL INTERRUPTION BY RISE
EDGE
  MOVLW   B'00000001'
  MOVWF   INTCON      ; ALL THE INTERPRETATIONS TURNED OFF

  BANK0               ; RETURN TO THE BANK 0
```

```
  MOVLW   H'90'
  MOVLW   INTCON
  BCF     PORTB,1
  MOVLW   B'00000111'
  MOVWF   CMCON          ; DEFINE THE OPERATION MODE OF THE ANALOG
COMPARATOR

  movlw   D'5'       ; MOVE THE VALUE TO W
  movwf   ACTIONS    ; INITIALIZE THE TIME 1


MAIN

  BTFSS   TURNON
  CALL    DEBOUNCE
  BTFSS   TURNOFF
  CALL    FTURNOFF
  GOTO    MAIN


DEBOUNCE
  BTFSC   TURNON
  GOTO    FDECREMENT
  GOTO    DEBOUNCE

FDECREMENT
  DECFSZ  ACTIONS    ; DECREMENTS ACTIONS UNTIL 0
  GOTO    FTURNON
  GOTO    FDISABLE

FTURNON
  BSF     MOTOR
  GOTO    MAIN

FTURNOFF
  BCF             MOTOR
  GOTO    MAIN

FDISABLE
```

```
   BCF         MOTOR
   GOTO        FDISQUALIFY    ; CAN NOT TURN ON

FDISQUALIFY                   ; THE MOTOR IS NEVER TURNED ON AGAIN UNTIL THE
COUNTER WAS RESETED
  BTFSC        RESET
  GOTO         FDISQUALIFY
  GOTO         RESETCOUNTER


RESETCOUNTER              ; RESET THE COUNTER VALUES
   MOVLW    D'5'              ; MOVE THE VALUE TO W
   MOVWF    ACTIONS
   GOTO     FBEGIN


  END
```

4.8 Questão 8

```
;=================================================================
==
; PROJECT DESCRIPTION
;=================================================================
==
;Solicita-se elaborar um programa para
;comandar o acionamento independente de três
;motores, com a restrição de poder funcionar
;simultaneamente um número máximo de dois motores

;=================================================================
==
;
;=================================================================
==
; Main.asm file generated by New Project wizard
;
; Created:   dom abr 21 2019
```

```
; Processor: PIC16F628A
; Compiler:  MPASM (Proteus)
; Author: Paulo Henrique Araújo Munhoz
;=================================================================
==


;=================================================================
==
; DEFINITIONS
;=================================================================
==

#include p16f628a.inc             ; Include register definition file


;CONFIG
; __CONFIG 0xFF61
__CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _MCLRE_ON & _BOREN_ON
& _LVP_OFF & _CPD_OFF & _CP_OFF

#DEFINE     BANK0       BCF    STATUS, RP0       ; DEFINE THE MEMORY BANK 0
#DEFINE     BANK1       BSF    STATUS, RP0       ; DEFINE THE MEMORY BANK 1


;=================================================================
==
; VARIABLES
;=================================================================
==
  CBLOCK 0x20

  MOTORS


  ENDC

;=================================================================
==
```

```
; INPUTS
;=================================================================
==
#DEFINE    BMOTOR1    PORTA,0   ; DEFINE NAME AND PORTS OF THE CURRENT
BUTTONS
#DEFINE    BMOTOR2    PORTA,1
#DEFINE    BMOTOR3    PORTA,2
#DEFINE    RESET      PORTA, 3

;=================================================================
==
; OUTPUTS
;=================================================================
==
#DEFINE    LEDMOTOR1   PORTB,0
#DEFINE    LEDMOTOR2   PORTB,1
#DEFINE    LEDMOTOR3   PORTB,2
;=================================================================
==
; RESET and INTERRUPT VECTORS
;=================================================================
==
  ORG      0x00
  GOTO     FBEGIN


  ORG    0X04
  RETFIE

;=================================================================
==
; CODE SEGMENT
;=================================================================
=

FBEGIN:
  CLRF     PORTA       ; CLEAN THE PORTA
  CLRF     PORTB       ; CLEAN THE PORTB
```

```
  BANK1             ; SWITCH TO THE BANK 1

  MOVLW  B'00001111'
  MOVWF  TRISA      ; DEFINE RA0, RA1 AND RA2 AS INPUTS AND THE OTHERS
AS OUTPUTS

  MOVLW  B'00000000'
  MOVWF  TRISB      ; DEFINE EVERY PORTB AS OUTPUTS

  MOVLW  B'10000000'
  MOVWF  OPTION_REG    ; PRESCALER 1:2 NO
                ; PULL-UPS TURNED OFF
                ; THE OTHERS CONFG. ARE IRRELEVANTS

  MOVLW  B'00000000'
  MOVWF  INTCON     ; ALL THE INTERPRETATIONS ARE TURNED OFF

  BANK0             ; RETURN TO THE BANK 0
  MOVLW  B'00000111'
  MOVWF   CMCON           ; DEFINE THE OPERATION MODE OF THE ANALOG
COMPARATOR

  movlw   D'2'      ; MOVE THE VALUE TO W
  movwf    MOTORS     ; INITIALIZE THE TIME1

MAIN:
  BTFSS     BMOTOR1
  CALL      DEBOUNCE1
  BTFSS     BMOTOR2
  CALL      DEBOUNCE2
  BTFSS     BMOTOR3
  CALL      DEBOUNCE3
  GOTO      MAIN

DEBOUNCE1
  BTFSS     BMOTOR1
  GOTO      DEBOUNCE1
  BSF       LEDMOTOR1
  DECFSZ    MOTORS      ; DECREMENT ACTIVE MOTORS UNTIL 0
```

```
  RETURN
  GOTO      FDISQUALIFY   ; CAN NOT TURN ON

DEBOUNCE2
  BTFSS     BMOTOR2
  GOTO      DEBOUNCE2
  BSF       LEDMOTOR2
  DECFSZ    MOTORS        ; DECREMENT ACTIVE MOTORS UNTIL 0
  RETURN
  GOTO      FDISQUALIFY

DEBOUNCE3
  BTFSS     BMOTOR3
  GOTO      DEBOUNCE3
  BSF       LEDMOTOR3
  DECFSZ    MOTORS        ; DECREMENT ACTIVE MOTORS UNTIL 0
  RETURN
  GOTO      FDISQUALIFY

FDISQUALIFY
   BTFSS    RESET
   GOTO     FBEGIN
   GOTO     FDISQUALIFY
END
```

4.9 Questão 9

```
;=================================================================
==
; PROJECT DESCRIPTION
;=================================================================
==

;=================================================================
==
;Elabore um programa que permita ligar e
;desligar três motores, cada motor terá sua chave liga e
;desliga. Esses motores poderão funcionar
;simultaneamente, no entanto, eles só poderão ser
```

```
;ligados em ordem crescente (primeiro M1, segundo M2
;e por último M3) e desligados em ordem decrescente
;(primeiro M3, segundo M2 e por último M1).
;=================================================================
==
; Main.asm file generated by New Project wizard
;
; Created:   dom abr 21 2019
; Processor: PIC16F628A
; Compiler:  MPASM (Proteus)
; Author: Paulo Henrique Araújo Munhoz
;=================================================================
==

;=================================================================
==
; DEFINITIONS
;=================================================================
==

#include p16f628a.inc             ; Include register definition file
;CONFIG
; __CONFIG 0xFF61
__CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _MCLRE_ON & _BOREN_ON
& _LVP_OFF & _CPD_OFF & _CP_OFF

#DEFINE    BANK0    BCF   STATUS, RP0    ; DEFINE THE MEMORY BANK 0
#DEFINE    BANK1    BSF   STATUS, RP0    ; DEFINE THE MEMORY BANK 1


;=================================================================
==
; VARIABLES
;=================================================================
==
  CBLOCK 0x20

  d1
  d2
```

```
  ENDC

;=====================================================================
==
; INPUTS
;=====================================================================
==
#DEFINE    BMOTOR1    PORTA,0  ; DEFINE BUTTONS NAME AND PORTS
#DEFINE    BMOTOR2    PORTA,1
#DEFINE    BMOTOR3    PORTA,2


;=====================================================================
==
; OUTPUTS
;=====================================================================
==
#DEFINE      LEDMOTOR1     PORTB,5
#DEFINE      LEDMOTOR2     PORTB,6
#DEFINE      LEDMOTOR3     PORTB,7
;=====================================================================
==
; RESET and INTERRUPT VECTORS
;=====================================================================
==
  ORG      0x00
  GOTO        FBEGIN


  ORG      0X04
  RETFIE

;=====================================================================
==
; CODE SEGMENT
;=====================================================================
=
```

```
FBEGIN:
  CLRF      PORTA      ; CLEAN THE PORTA
  CLRF      PORTB      ; CLEAN THE PORTB

  BANK1                ; SWITCH TO BANK 1

  MOVLW   B'00000111'
  MOVWF   TRISA      ; DEFINE RA0, RA1 AND RA2 AS INPUTS AND THE OTHERS
AS OUTPUTS

  MOVLW   B'00000000'
  MOVWF   TRISB      ; DEFINE EVERY PORTB AS OUTPUT

  MOVLW   B'10000000'
  MOVWF   OPTION_REG   ; PRESCALER 1:2 NO
                       ; PULL-UPS TURNED OFF
                       ; THE OTHERS CONFG. ARE IRRELEVANTS

  MOVLW   B'00000000'
  MOVWF   INTCON     ; ALL THE INTERPRETATIONS TURNED OFF

  BANK0                ; RETURN TO THE BANK 0
  MOVLW   B'00000111'
  MOVWF   CMCON          ; DEFINE THE OPERATION MODE OF THE ANALOG
COMPARATOR


MAIN:
  BTFSC     BMOTOR1
  GOTO      MAIN
  GOTO      FTURNON1

FTURNON1

  BSF       LEDMOTOR1
  BTFSS     BMOTOR2
  GOTO      FTURNON2
  GOTO      FTURNON1
```

```
FTURNON2
  BSF       LEDMOTOR2
  BTFSC     BMOTOR3
  GOTO      FTURNON2
  BSF       LEDMOTOR3
  CALL      DELAY500MS
  GOTO      FTURNON3

DEBOUNCE3
  BTFSC     BMOTOR3
  GOTO      FTURNOFF3
  GOTO      DEBOUNCE3


FTURNON3
  BTFSC     BMOTOR3
  GOTO      FTURNON3
  GOTO      DEBOUNCE3

FTURNOFF3
  BCF       LEDMOTOR3
  GOTO      FTURNOFF2

FTURNOFF2
  BTFSC     BMOTOR2
  GOTO      FTURNOFF2
  BCF       LEDMOTOR2
  GOTO      FTURNOFF1


FTURNOFF1
  BTFSC     BMOTOR1
  GOTO      FTURNOFF1
  GOTO      DEBOUNCE1


DEBOUNCE1
  BTFSC     BMOTOR1
```

```
  GOTO      FDISABLE1
  GOTO      DEBOUNCE1

FDISABLE1
  BCF       LEDMOTOR1
  GOTO      MAIN

 DELAY500MS:

  MOVLW     D'200'     ; MOVE THE VALUE TO W
  MOVWF     d1         ; INITIALIZE THE TIME0
                ; 4 MACHINE CYCLES
aux1:
  MOVLW     D'250'   ; MOVE THE VALUE TO W
  MOVWF     d2       ; INITIALIZE THE TIME1

                ; 2 MACHINE CYCLES
aux2:
  NOP             ; SPEND 1 MACHINE CYCLE
  NOP
  NOP
  NOP
  NOP
  NOP
  NOP
  NOP
  NOP
  NOP
  NOP



  DECFSZ    d2       ; DECREMENT TIME1 UNTIL 0
  GOTO      aux2     ; GO TO LABEL AUX2

                ; 250 x 10 MACHINE CYCLES = 2500 CYCLES

  DECFSZ    d1       ; DECREMENTS TIME0 UNTIL 0
  GOTO      aux1     ; GO TO LABEL AUX1
```

```
                ; 3 MACHINE CYCLES

                ; 2500 x 200 = 500000

  RETURN        ; RETURN AFTER THE SUB ROUTINE CALL


END
```

4.10 Questão 10

```
;=================================================================
==
; PROJECT DESCRIPTION
;=================================================================
==

;=================================================================
==
;Pede-se elaborar programa que permita
;acionar e desligar manualmente um motor M1 somente
;por uma vez. Na segunda tentativa de acionamento M1
;será automaticamente desligado, permitindo então que
;seja ligado por no máximo duas vezes um segundo
;motor M2. Na terceira tentavia de ligar M2, ele será
;desligado automaticamente, possibilitando a repetição
;de todo o ciclo de operação.
;=================================================================
==
; Main.asm file generated by New Project wizard
;
; Created:   dom abr 21 2019
; Processor: PIC16F628A
; Compiler:  MPASM (Proteus)
; Author: Paulo Henrique Aaújo Munhoz
;=================================================================
==
```

```
;=================================================================
==
; DEFINITIONS
;=================================================================
==

#include p16f628a.inc             ; Include register definition file
;CONFIG
; __CONFIG 0xFF61
__CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _MCLRE_ON & _BOREN_ON
& _LVP_OFF & _CPD_OFF & _CP_OFF

#DEFINE    BANK0    BCF   STATUS, RP0    ; DEFINE THE MEMORY BANK 0
#DEFINE    BANK1    BSF   STATUS, RP0    ; DEFINE THE MEMORY BANK 1


;=================================================================
==
; VARIABLES
;=================================================================
==
  CBLOCK 0x20

  ACTIVATIONS
  d1
  d2


  ENDC

;=================================================================
==
; INPUTS
;=================================================================
==
#DEFINE    TURNON1            PORTA,0  ; DEFINE BUTTONS NAME AND PORTS
#DEFINE    TURNOFF1           PORTA,1
#DEFINE    TURNON2            PORTA,2
#DEFINE    TURNOFF2           PORTA,3
```

```
;=================================================================
==
; OUTPUTS
;=================================================================
==
#DEFINE    LEDMOTOR1          PORTB,6
#DEFINE    LEDMOTOR2          PORTB,7

;=================================================================
==
; RESET and INTERRUPT VECTORS
;=================================================================
==
  ORG     0x00
  GOTO    FBEGIN


  ORG     0X04
  RETFIE

;=================================================================
==
; CODE SEGMENT
;=================================================================
=

FBEGIN:
  CLRF    PORTA        ; CLEAN THE PORTA
  CLRF    PORTB        ; CLEAN THE PORTB

  BANK1                ; SWITCH TO BANK 1

  MOVLW   B'00001111'
  MOVWF   TRISA        ; DEFINE RA0, RA1 AND RA2 AS INPUTS AND THE OTHERS
AS OUTPUTS

  MOVLW   B'00000000'
```

```
  MOVWF    TRISB        ; DEFINE EVERY PORTB AS OUTPUT

  MOVLW    B'10000000'
  MOVWF    OPTION_REG    ; PRESCALER 1:2 NO
                ; PULL-UPS TURNED OFF
                ; THE OTHERS CONFG. ARE IRRELEVANTS

  MOVLW    B'00000000'
  MOVWF    INTCON       ; ALL THE INTERPRETATIONS TURNED OFF

  BANK0              ; RETURN TO BANK 0
  MOVLW    B'00000111'
  MOVWF    CMCON           ; DEFINE THE OPERATION MODE OF THE ANALOG
COMPARATOR

  MOVLW    D'2'            ; MOVE THE VALUE TO W
  MOVWF    ACTIVATIONS     ; INITIALIZE THE TIME0
                ; 4 MACHINE CYCLES


MAIN:
  BTFSC    TURNON1
  GOTO     MAIN
  GOTO     ENABLE1

ENABLE1
  BSF      LEDMOTOR1
  BTFSC    TURNOFF1
  GOTO     ENABLE1
  BCF      LEDMOTOR1
  GOTO     ENABLE2

ENABLE2
  BTFSC    TURNON1
  GOTO     ENABLE2
  GOTO     TURNONMOTOR2

TURNONMOTOR2
  BTFSC    TURNON2
```

```
  GOTO    TURNONMOTOR2
  BSF     LEDMOTOR2
  GOTO    DISABLE2

DISABLE2
  BTFSC   TURNOFF2
  GOTO    DISABLE2
  BCF     LEDMOTOR2
  GOTO    DISABLE22

DISABLE22
  BTFSC   TURNON2
  GOTO    DISABLE22
  BSF     LEDMOTOR2
  GOTO    FSTOP

FSTOP
  BTFSC   TURNOFF2
  GOTO    FSTOP
  BCF     LEDMOTOR2
  GOTO    MAIN

END
```

5 CONCLUSÃO

A compreensão dos conteúdos aplicados neste primeiro contato com o microcontrolador PIC16F628A foi bem desafiador, tendo assim um grau de grande importância para o estudo de microcontroladores na engenharia, especialmente nas simulações presentes neste trabalho. Foi notável a importância da noção básica sobre funcionamento, mapeamento, nomenclaturas que desde o início do relatório, bem como a compreensão do mesmo possibilitou-me e a quem desenvolveu conforme as explicações ministradas em aula a ampliação a mais quanto aos conceitos e noções sobre programação de baixo nível e também trazendo uma aproximação na parte da eletrônica em questão de conhecimento de componentes tanto em seu uso como financeiramente, pois o desenvolvimento de projeto pelo engenheiro deve possuir a sensibilidade se este pode ou não ser desenvolvido.

REFERÊNCIAS

Desbravando o PIC: ampliado e atualizado para PIC 16F628A / David José de Souza. – 6.ed. –São Paulo: Érica, 2003.

Curso de Assembly para PIC – WRKits – Professor Wagner Rambo – disponível em: <https://www.youtube.com/watch?v=4vua8Fxbs1A&list=PLZ8dBTV2_5HQd6f4IaoO50L6oToxQMFYt>