Manual do Usuário

McMaster

Desenvolvimento de Sistemas com Microcontroladores PIC

Índice

Capítulo 1 - Introdução

Inicialmente gostaríamos de parabenizá-lo por estar adquirindo o sistema didático MCMASTER. Este sistema utiliza o microcontrolador PIC16F877A como objeto central. Junto ao microcontrolador, uma série de periféricos foram adicionados. O objetivo é disponibilizar uma placa de desenvolvimento onde o usuário possa testar seus conhecimentos em software, sem se preocupar com a montagem do hardware. Basta escrever o software. Veja todos os recursos que o sistema oferece:

- LCD alfanumérico;
- Displays de leds de 7 segmentos;
- Teclado matricial;
- Leds;
- Buzzer;
- Memória serial EEPROM (protocolo I^2C);
- Relógio de tempo real (protocolo I^2C);
- Comunicação serial padrão RS232;
- Conversão A/D;
- Leitura de jumpers;
- Sensor de temperatura;
- Aquecedor;
- Ventilador;
- Tacômetro;
- Lâmpada Incandescente;
- Gravação in-circuit.

Aliado a todos estes recursos, utilizou-se o microcontrolador PIC16F877A que é o mais completo da linha 16Fxxx. Suas principais características são:

- 8K de memória de programa;
- 368 bytes de memória de dados volátil (RAM);
- 256 bytes de memória de dados não volátil (E2PROM) ;
- 14 Interrupções;
- 33 I/Os;
- 3 Timers (2 de 8 bits 1 de 16 bits);
- 2 Capture/Compare/PWM;
- USART;
- MSSP (PSI e I2C);
- PSP;
- 8 canais de conversão A/D com 10 bits cada;
- 2 comparadores de tensão;

Fazem parte do kit de desenvolvimento:

- 1 Sistema de Treinamento em Microcontroladores PIC MCMASTER;
- 1 PIC16F877A;
- 1 Manual do Usuário;
- 1 CD-ROM;

Capítulo 2 - McMaster – Desenvolvimento de Sistemas com Microcontroladores PIC

Introdução

O McMaster é um equipamento para desenvolvimento de sistemas completo para o estudo da tecnologia de microcontroladores Microchip e em particular ao estudo do microcontrolador PIC16F877A. Na verdade, este sistema serve para muito mais que simplesmente o aperfeiçoamento dos conhecimentos da família PIC. Com ele o usuário é capaz de criar projetos completos, colocando em teste também a eficiência de seus conceitos e algoritmos.

Tudo isso é possível porque este sistema foi criado e desenvolvido pensando na didática de ensino e nos problemas mais comuns do mercado em relação ao uso de microcontroladores.

Visão Macro do Sistema

Nesta seção será abordado através de uma visão macro o conceito do sistema utilizado no McMaster.

Ele é composto de um gravador para o microcontrolador, o microcontrolador PIC central, os periféricos ligados ao microcontrolador, aos quais daremos o nome de periféricos padrão e um conector de expansão para experiências onde novos periféricos, aos quais daremos o nome de periféricos adicionais, poderão ser ligados.

Um dos periféricos padrão do MCMASTER é o módulo de comunicação serial RS232. Como o gravador também utiliza comunicação serial RS232 para se comunicar com o Mplab e no MCMASTER existe apenas uma saída serial, este recurso deve ser compartilhado para que tanto o gravador como o módulo RS232 do sistema possam utilizar a mesma saída. Desta forma, o usuário deverá escolher, através do botão localizado acima do microcontrolador, onde a serial deverá ser aplicada, no gravador ou no microcontrolador.

Todos os I/Os do microcontrolador estão disponíveis no conector de expansão para experiências. Com exceção dos pinos RB6 e RB7 que são utilizados pela gravação in-circuit, todos os outros I/Os estão ligados diretamente ao conector, ou seja, sem nenhum tipo de proteção. Apenas os pinos RB6 e RB7 foram isolados. Por este motivo, é muito importante que o usuário configure corretamente os I/Os do microcontrolador quando for utilizar o conector de expansão, pois neste caso, uma ligação errada pode danificar o microcontrolador. Se o usuário utilizar o conector de expansão apenas com placas oficiais de experiências a preocupação com a direção dos I/Os do microcontrolador não precisa ser tomada, uma vez que as placas de experiências e todo o MCMASTER foram projetados a fim de evitar que uma configuração errada do microcontrolador coloque o sistema em risco. Portanto, mesmo que um pino do microcontrolador seja configurado como saída quando o correto seria entrada a integridade do sistema está garantida. É claro que este erro pode acarretar num mau funcionamento do sistema projetado, porém nunca existirá risco ao MCMASTER e às placas de experiências, desde que as mesmas sejam oficiais e/ou homologadas pelo fabricante.

Para evitar que módulos padrão do MCMASTER venham a atrapalhar o correto funcionamento de uma eventual placa de experiências optou-se pela utilização de jumpers de configuração para que pontos importantes do circuito possam ser desabilitados e as vias do

microcontrolador possam ser utilizadas apenas pelas placas de experiências e não mais nos módulos padrão. Desta forma, foram criados 7 jumpers. Começando de cima para baixo os jumpers são:

- Comunicação TX (RC6) – Este jumper desliga o pino RC6 (TX da USART do PIC – utilizado para comunicação padrão RS232) do microcontrolador deixando-o disponível apenas no conector de expansão.
- Comunicação RX (RC7) – Da mesma forma que o jumper anterior, este jumper desliga o pino RC7 (RX da USART do PIC – utilizado para comunicação padrão RS232) do microcontrolador deixando-o disponível apenas no conector de expansão.
- Data I^2C (RC4) – Este jumper desliga o pino RC4 (via de dados para comunicação I^2C) do microcontrolador deixando-o disponível apenas no conector de expansão.
- Clock I^2C (RC3) – Este jumper desliga o pino RC3 (via de clock para comunicação I^2C) do microcontrolador deixando-o disponível apenas no conector de expansão.
- Coluna 1 (RB0) – Este jumper desliga o pino RB0 utilizado para ler os botões da coluna 1 do teclado matricial deixando-o disponível apenas no conector de expansão.
- Linha 1 / Display Milhar (RB4) – Este jumper desliga o pino RB4 utilizado para ativar a linha 1 do teclado matricial e o display do milhar deixando-o disponível apenas no conector de expansão.
- Leds Especiais (RC0, RC1 e RC2) – Este jumper desabilita os leds ligados aos pinos RC0, RC1 e RC2 utilizados pelos módulos CPP e TIMER1 do microcontrolador.

Módulos Padrão

Nesta seção serão abordados cada um dos módulos padrão do MCMASTER.

Microcontrolador

O sistema utiliza o microcontrolador PIC16F877A como centro de todo o hardware. Este microcontrolador está ligado a todos os periféricos disponíveis, possibilitando o estudo de praticamente todas as suas funções. Devido também ao grande poder de recursos deste modelo de PIC, é possível, junto aos demais recursos da placa, o desenvolvimento de projetos simples e/ou complexos, como por exemplo um controlador de temperatura com algoritmo de controle PID.

LCD alfanumérico

Nos dias de hoje, qualquer programador sabe da importância da interface com o usuário dentro de um sistema qualquer. Por isso, é muito importante o aprendizado de operação de um display do tipo LCD. No caso do MCMASTER, este display possui 2 linhas de 16 caracteres cada, sendo um padrão de mercado atual. Possui um chip de controle próprio, com o qual é realizada a interface com o microcontrolador. Com este periférico os sistemas desenvolvidos no MCMASTER poderão possuir telas explicativas, informações claras e menus de navegação.

A comunicação com o LCD é paralela com 8 vias de dados. Além destas, mais duas vias são utilizadas para controlar o LCD, uma denominada de ENABLE e a outra de RS.

A comunicação com o LCD é somente de escrita, desta forma, o pino de R/W do LCD está diretamente ligado ao terra (GND), não permitindo a leitura do mesmo.

As 8 vias de dados do LCD estão ligadas ao PORTD do microcontrolador, de RD0 (LSB) até RD7 (MSB). O pino de ENABLE está conectado ao pino RE1 do PIC e o pino RS do LCD ao pino RE0 do microcontrolador.

Assim, o esquema de ligação segue a tabela abaixo:

PIC	LCD
RD0...RD7	D0...D7
RE0	RS
RE1	ENABLE
Terra (GND)	R/W

Para maiores informações a respeito do LCD pode-se consultar o data sheet contido no CD que acompanha o MCMASTER.

Displays de leds com 7 segmentos

Como já visto, o LCD é uma ótima ferramenta de informação ao usuário, porém, muitas vezes ele ainda é inviável. Pode-se comentar alguns motivos desta inviabilidade: custos, capacidade de visualização, iluminação, etc. Por isso, em muitos projetos, os velhos e práticos displays de leds ainda são a melhor alternativa. No MCMASTER optou-se pela utilização de displays de 7 segmentos, que são numéricos, mas que permitem a visualização de diversas letras através da combinação específica destes segmentos. Optou-se também por 4 dígitos, todos com os segmentos interligados e os controles (comum) independentes, possibilitando a operação por varredura.

Atualmente, é muito comum encontrar em produtos do mercado, a combinação de ambos os tipos de display, para uma visualização mais completa e eficiente. Com o MCMASTER esta combinação também é possível.

A conexão dos displays com o microcontrolador segue a tabela abaixo:

PIC	Segmento
RD0	A
RD1	B
RD2	C
RD3	D
RD4	E
RD5	F
RD6	G
RD7	DP

E as vias de seleção de cada um dos displays, a tabela seguir:

PIC	Display
RB4	Milhar
RB5	Centena
RB6	Dezena
RB7	Unidade

Para a ativação dos displays deve-se selecionar nível lógico 1 nos pinos de seleção. Os segmentos também são ativados com nível lógico 1.

Leds

O MCMASTER possui um grupo de 8 leds que compartilha o mesmo barramento que os displays de 7 segmentos e o display LCD.

Desta forma, o seu acionamento deve ser feito via varredura sendo que os leds estão conectados ao PORTD e a seleção é feita pelo pino RA4. Da mesma forma que os displays, os leds são ativados com nível lógico 1, tanto na via de seleção (RA4) como individualmente (PORTD).

Teclado matricial

A maioria dos sistemas desenvolvidos atualmente utilizam algum tipo de teclado para a entrada de dados pelo usuário. O MCMASTER está provido de um teclado matricial de 4 linhas e 4 colunas, totalizando 16 teclas. O acionamento das linhas do teclado é feito simultaneamente com os comuns dos displays de 7 segmentos. Desta forma, ao acionar o display da unidade aciona-se também a linha 4 do teclado matricial. Junto com o display da dezena a linha 3 e assim por diante.

A tabela abaixo mostra esta relação:

Pino PIC	Estado	Teclado Matricial	Display de 7 segmentos
RB7	1	linha 4 ativada	unidade ativada
	0	linha 4 desativada	unidade desativada
RB6	1	linha 3 ativada	dezena ativada
	0	linha 3 desativada	dezena desativada
RB5	1	linha 2 ativada	centena ativada
	0	linha 2 desativada	centena desativada
RB4	1	linha 1 ativada	milhar ativada
	0	linha 1 desativada	milhar desativada

As colunas podem ser lidas através dos pinos RB0, RB1, RB2 e RB3, conforme a tabela a seguir:

Pino PIC	Estado	Teclado Matricial
RB0	1	Alguma tecla pressionada na coluna 1
	0	Nenhuma tecla pressionada na coluna 1
RB1	1	Alguma tecla pressionada na coluna 2
	0	Nenhuma tecla pressionada na coluna 2
RB2	1	Alguma tecla pressionada na coluna 3
	0	Nenhuma tecla pressionada na coluna 3
RB3	1	Alguma tecla pressionada na coluna 4
	0	Nenhuma tecla pressionada na coluna 4

Vale observar que para o correto funcionamento do teclado os jumpers relacionados com os pinos RB0 e RB4 devem estar configurados na posição ON.

Buzzer

Para chamar a atenção do usuário e confirmar certas ações, cada vez mais os sistemas estão fazendo uso de técnicas sonoras, seja através de simples bips ou por complexas melodias. Para que os usuários não fiquem sem o uso deste recurso, disponibilizou-se também um buzzer piezoelétrico com oscilação comandada diretamente pelo PIC, tornando possível a criação de sons diversos.

O microcontrolador deve então gerar (através do software) uma onda quadrada capaz de excitar o buzzer. Para gerar um simples beep pode-se utilizar uma onda quadrada com freqüência em torno de 650Hz e duração aproximada de 100ms.

O buzzer está conectado ao pino RE2 do microcontrolador.

Memória E2PROM externa

Além da memória E2PROM interna do PIC, o MCMASTER possui ainda uma memória externa do tipo serial, modelo 24LC256 com 32Kbytes disponíveis para uso. Esta memória está soquetada na placa, possibilitando a sua substituição por outros modelos compatíveis, com maior ou menor capacidade.

A comunicação com esta memória é do tipo I²C, estando diretamente ligada os pinos do PIC responsáveis por este padrão de comunicação.

Assim,

PIC	Memória
RC3	Clock (SCL) – pino 6
RC4	Data (SDA) – pino 5

Novamente os jumpers de configurações relacionados devem estar habilitados para a utilização da memória serial externa.

Como a memória serial compartilha o mesmo barramento I^2C do relógio de tempo real (PCF8583P) se faz necessária a utilização de endereços diferentes para que o barramento seja compatível com os dois periféricos. Sendo assim, escolheu-se o endereço 7h (111b) para a memória serial. Para maiores informações sobre o protocolo de comunicação da memória serial 24LC256 pode-se consultar o data sheet disponível no CD.

Relógio de tempo real (RTC)

Utilizando o mesmo barramento I^2C da memória serial, o MCMASTER possui um relógio de tempo real, modelo PCF8583P. Com este componente o usuário poderá criar sistemas que contenham informações como a hora e a data atual. O relógio utilizado é completo, ou seja, é capaz de contar dia, mês, ano (inclusive bissexto), semana, hora, minuto, segundo e milésimo de segundo. Além de poder ser configurado de formas diferentes. O data sheet deste componente está disponível no CD que acompanha o MCMASTER.

Pelo mesmo motivo já comentado na memória serial, o relógio foi configurado para utilizar o endereço 0h (000b) a fim de poder compartilhar o mesmo barramento I^2C.

Como no caso da memória, os pinos responsáveis pela comunicação são:

PIC	Relógio RTC

RC3	Clock (SCL) – pino 6
RC4	Data (SDA) – pino 5

Comunicação serial RS-232

Quem não deseja que seu projeto se comunique com um computador atualmente? Esta é outra tendência de mercado que os profissionais não podem deixar de lado. Seja para a configuração de parâmetros, para a coleta de dados ou a visualização gráfica de informações, a interligação entre o kit e o computador é um recurso que não poderia ser deixado de lado.

Optou-se pela comunicação serial, padrão RS-232 através de um conector DB-9. A interface e ajuste de tensões necessárias a este padrão, em relação à operação do PIC (TTL) é feita por um CI dedicado. Internamente, as vias de TX e RX podem ser ligadas aos pinos da USART do PIC, possibilitando o uso deste recurso.

Para habilitar este recurso é necessário que os jumpers apropriados estejam na posição ON. A ligação ao microcontrolador segue a tabela abaixo.

PIC	**COM.**
RC6	TX (saída)
RC7	RX (entrada)

Como a porta de comunicação é compartilhada com o gravador é necessário também selecionar a serial para o PIC através do botão de modo de utilização.

Faz parte também do módulo de comunicação serial, o conector DB9 fêmea que segue a pinagem abaixo:

Pino	**Função**
1	-
2	TX (saída)
3	RX (entrada)
4	-
5	Terra (GND)
6	-
7	CTS (utilizado apenas pelo gravador)
8	RTS (utilizado apenas pelo gravador)
9	-

Conversão analógica / digital (A/D)

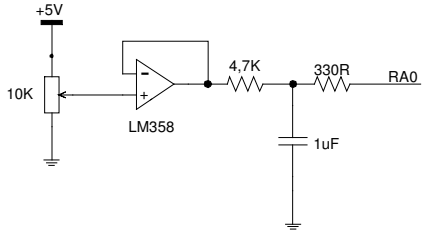
É verdade que estamos considerando o mundo cada vez mais digital, principalmente nos dias de hoje, onde vemos bilhões de informações trafegando por fibras ópticas e imagens de computador recriando o mundo real. Mas não podemos esquecer que a natureza é completamente analógica, e qualquer sistema que se baseie ou utilize informações deste meio externo precisará de um sistema de conversão para poder se comunicar. É por isso que, hoje e sempre, a conversão A/D é tão necessária.

Com o MCMASTER poderemos realizar estas conversões de duas maneiras. A primeira é através do conversor interno do PIC e a segunda é através de um pseudoconversor fundamentado no tempo de carga de um circuito RC.

Dentre os módulos padrão existem dois sistemas para trabalhar com o conversor A/D e para qualquer um deles, as duas formas de aquisição podem ser aplicadas, ou seja, tanto via A/D convencional como via RC. O primeiro sistema consiste num potenciômetro e o segundo num conjunto de jumpers que podem ser configurados como divisor resistivo ou circuito RC.

Potenciômetro

O sistema com o potenciômetro segue o esquema elétrico representado a seguir.



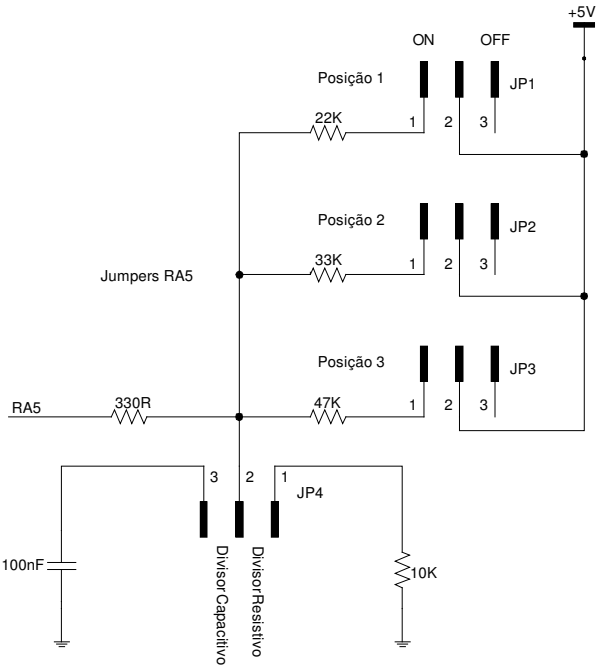
A tensão de entrada presente no pino RA0 do microcontrolador pode ser ajustada entre 0 e 5V. Caso se utilize o conversor A/D interno do PIC, o capacitor de 1uF e o resistor de 4K7 formam um filtro passa baixa útil para filtrar ruídos e deixar o sinal de entrada mais estável.

Caso se utilize o sistema de leitura via RC o conjunto de resistores e capacitores assume outra função. Neste sistema, para realizar a conversão deve-se executar as etapas a seguir:

- Inicialmente, através do software, deve-se descarregar o capacitor, colocando o pino do microcontrolador como saída em nível lógico 0. O capacitor se descarregará pelo resistor de 330R.
- Após o capacitor estar descarregado, coloca-se o pino do microcontrolador em entrada e começa-se a contar o tempo que o capacitor leva para se carregar (através do resistor de 4K7), ou seja, quanto tempo o capacitor leva para atingir nível lógico 1.
- Como tempo de carga é inversamente proporcional à tensão aplicada pelo potenciômetro, sabendo-se o tempo de carga pode-se estimar a tensão aplicada.

Jumpers

O sistema de jumpers está ligado ao pino RA5 do microcontrolador e segue o esquema elétrico representado a seguir.



Se configurarmos o sistema para divisor resistivo, basta ler com o conversor A/D do PIC a tensão presente no pino RA5 para estimar a posição do jumper.

Se configurarmos o sistema para resistor/capacitor, devemos seguir a mesma metodologia explicada no caso do potenciômetro, ou seja:

- Inicialmente descarregar o capacitor através do resistor de 330R colocando o pino do microcontrolador como saída em nível lógico 0.
- Após o capacitor estar descarregado, colocar o pino do microcontrolador em entrada e começar a contar o tempo que o capacitor leva para se carregar, ou seja, quanto tempo o capacitor leva para atingir nível lógico 1.
- Este tempo de carga é proporcional ao valor do circuito RC e portanto, pode ser utilizado para determinar a posição do jumper.

Periféricos Adicionais

A seguir serão explanados os periféricos adicionais contidos na placa de experiências EXP01 que acompanha o kit MCMASTER.

Placa de experiências EXP01

Entre outras funções a placa de experiências EXP01 possui um sistema completo para monitoramento e controle de temperatura, com um sensor e dois atuadores. Desta forma, tem-se um sensor de temperatura, um atuador de aquecimento (resistência controlada por PWM) e um atuador de resfriamento (ventilador controlado por PWM). Além disso, um sistema óptico ligado às hélices do ventilador é capaz de criar um tacógrafo, para monitoramento e controle de rotação.

Possui também uma lâmpada incandescente além de gerar uma tensão de referência estável em 2,5V que pode ser utilizada como referência para o conversor A/D.

Com tudo isso pode-se criar experimentos e projetos complexos de controle, começando em um simples controle ON/OFF até um avançado controlador PID.

Sensor de temperatura

A placa possui um circuito que utiliza um diodo de sinal como elemento sensor do medidor de temperatura ambiente. O sinal analógico proporcional à temperatura ambiente está presente no pino RA1 do microcontrolador e varia entre 0 e 5V.

Deve-se evitar que a temperatura ultrapasse 90ºC a fim de evitar que o sensor seja danificado.

Aquecedor

O aquecedor consiste numa resistência de 68Ω com 5W de dissipação. Pode ser acionada através do pino RC2 do microcontrolador. Veja que este pino pode ser configurado como PWM, e portanto, a potência de aquecimento pode ser regulada através deste recurso.

Ventilador

O sistema de ventilação consiste num cooler de PC que pode ser ativado através do pino RC1 do microcontrolador. Assim como no caso do aquecedor, este pino pode ser configurado como PWM, desta forma, pode-se modular a velocidade do ventilador utilizando este recurso do microcontrolador.

Tacômetro

Junto ao ventilador existe um sistema formado por um transmissor e um receptor de infravermelho. Este sistema é utilizado para medir a velocidade de rotação do ventilador. Quando não temos a passagem de luz, ou seja, quando a luz está interrompida por uma das palhetas do ventilador, o sistema de tacômetro apresentará na saída nível lógico 1. Quando se tem a passagem de luz, a saída do sistema de tacômetro será 0. O tacômetro está conectado ao pino RC0 (entrada de contador do TMR1) do microcontrolador.

Lâmpada incandescente

Consiste numa lâmpada incandescente de 12V que pode ser acionada através do pino RC5 do microcontrolador. Com nível lógico 1 a lâmpada acende e com nível lógico 0 a lâmpada apaga.

Tensão de referência

O circuito medidor de temperatura ambiente utiliza uma tensão de referência fixa e estável em 2,5V e como este recurso já estava presente na placa de experiências EXP01 resolveu-se também disponibilizar este recurso ao usuário. Assim, a tensão de referência de 2,5V foi conectada ao pino RA3 do PIC que pode ser configurado para utilizar este pino como entrada de referência externa do conversor A/D. Isto permite que o conversor A/D possa trabalhar em outra faixa de conversão e conseqüentemente com outra resolução.

Gravador

Para utilizar o gravador presente no MCMASTER basta selecionar corretamente a saída serial e utilizar o software de desenvolvimento Mplab da Microchip. Por se tratar de um gravador in-circuit o microcontrolador não precisa ser retirado da placa.

Ao habilitar o gravador no Mplab o software atual do PIC16F877A será paralisado e instantes após o final da gravação do novo software, o microcontrolador será automaticamente inicializado.

Capítulo 3 - Experiência 1 - Leitura de uma tecla e acionamento de um led

Objetivo

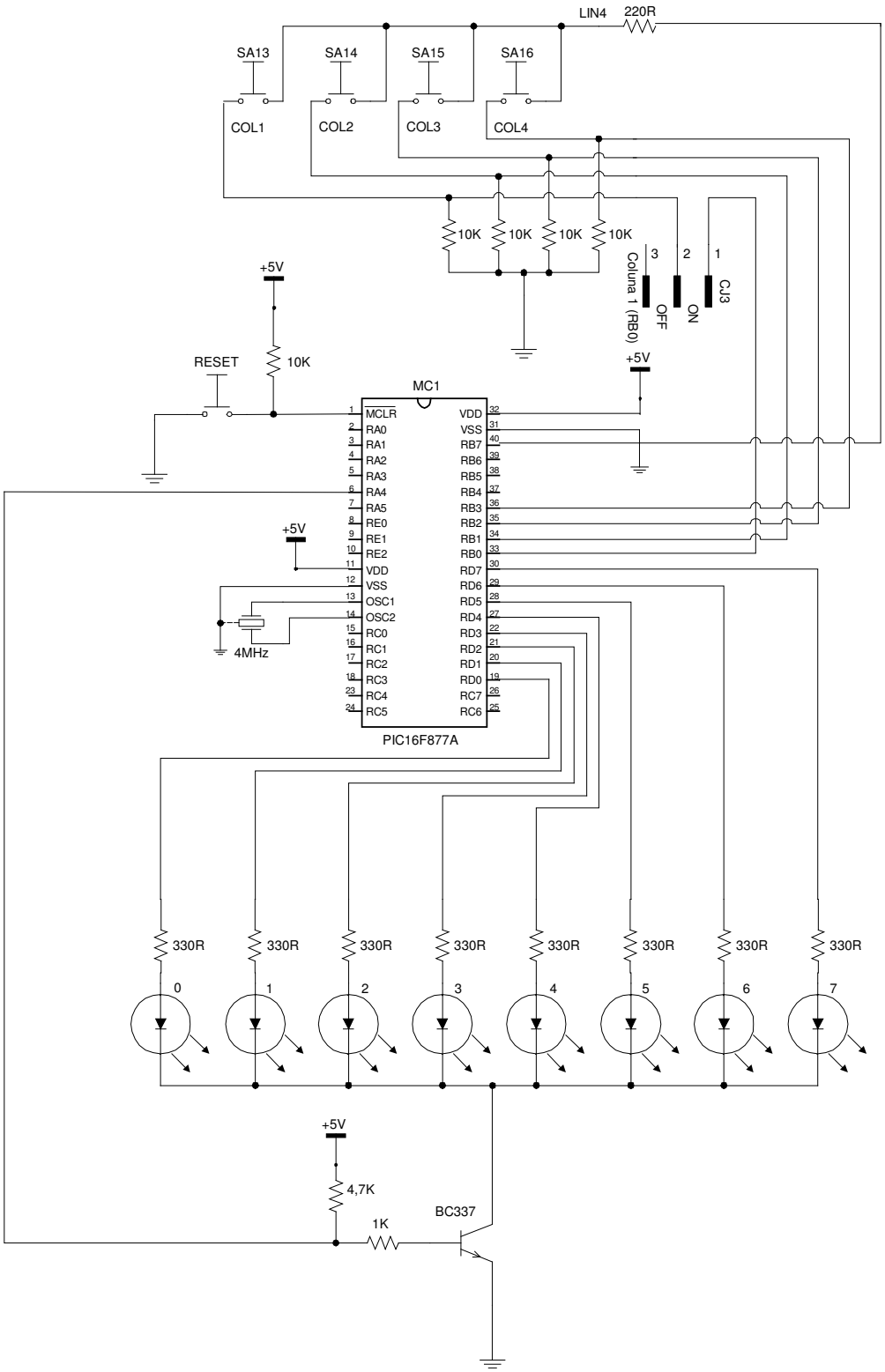
O objetivo desta experiência é ensinar ao aluno os primeiros passos sobre o microcontrolador. É apresentado o modo de configuração dos pinos de I/Os e as primeiras instruções utilizadas para testar condições nos pinos de entrada e alterações de estado nos pinos de saída, além de instruções para controle do fluxo do programa.

Descrição

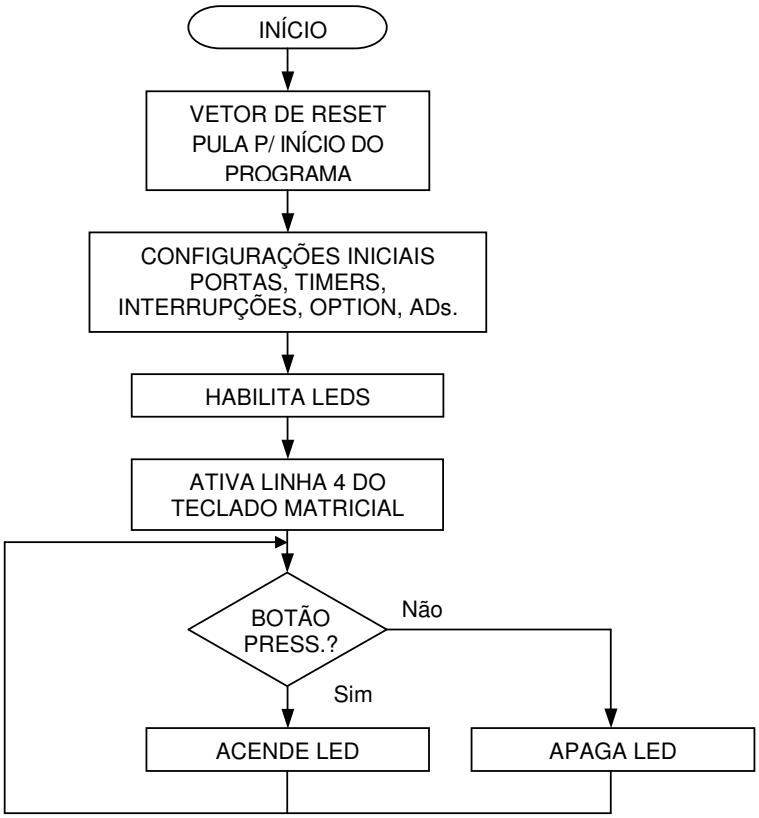
Sistema muito simples para representar o estado de um botão através de um led. Com o botão pressionado o led é ligado e com o botão solto o led é apagado.

O software inicia configurando os pinos de I/Os através dos registradores TRIS e dos registradores de periféricos pertinentes. Em seguida, o software habilita a linha 4 do teclado matricial e o grupo de leds ligados ao PORTD. A partir daí, o software entra num loop infinito onde o botão da linha 1 coluna 4 é testado e seu estado reproduzido no led 0 ligado ao pino RD0.

Esquema Elétrico



Fluxograma


INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURAÇÕES INICIAIS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION, ADs.
HABILITA LEDS
ATIVA LINHA 4 DO
TECLADO MATRICIAL
BOTÃO
PRESS.?
Não
Sim
ACENDE LED
APAGA LED

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *        EXPERIÊNCIA 1 - LEITURA DE UMA TECLA E ACIONAMENTO DE UM LED     *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                             DESCRIÇÃO GERAL                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; SISTEMA MUITO SIMPLES PARA REPRESENTAR O ESTADO DE UM BOTÃO ATRAVÉS DE
; UM LED. COM O BOTÃO DA COLUNA 1 LINHA 4 PRESSIONADO O LED LIGADO AO PINO
; RD0 PERMANECE LIGADO. SE O BOTÃO FOR SOLTO O LED APAGA.

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        CONFIGURAÇÕES PARA GRAVAÇÃO                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_OFF & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              VARIÁVEIS                                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DOS NOMES E ENDEREÇOS DE TODAS AS VARIÁVEIS UTILIZADAS
; PELO SISTEMA

   CBLOCK  0x20                  ; ENDEREÇO INICIAL DA MEMÓRIA DE
                                 ; USUÁRIO

           W_TEMP                ; REGISTRADORES TEMPORÁRIOS PARA
           STATUS_TEMP           ; INTERRUPÇÕES
                                 ; ESTAS VARIÁVEIS NEM SERÃO UTI-
                                 ; LIZADAS
   ENDC                          ; FIM DO BLOCO DE MEMÓRIA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>       ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     DEFINIÇÃO DOS BANCOS DE RAM                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                             FLAGS INTERNOS                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS FLAGS UTILIZADOS PELO SISTEMA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              CONSTANTES                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODAS AS CONSTANTES UTILIZADAS PELO SISTEMA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               ENTRADAS                                  *
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS PINOS QUE SERÃO UTILIZADOS COMO ENTRADA
; RECOMENDAMOS TAMBÉM COMENTAR O SIGNIFICADO DE SEUS ESTADOS (0 E 1)

#DEFINE    BOTAO   PORTB,0         ; PORTA DO BOTÃO
                                   ; 1 -> PRESSIONADO
                                   ; 0 -> LIBERADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                SAÍDAS                                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS PINOS QUE SERÃO UTILIZADOS COMO SAÍDA
; RECOMENDAMOS TAMBÉM COMENTAR O SIGNIFICADO DE SEUS ESTADOS (0 E 1)

#DEFINE    LED     PORTD,0         ; PORTA DO LED
                                   ; 0 -> APAGADO
                                   ; 1 -> ACESO

#DEFINE    C_LEDS  PORTA,4         ; PINO PARA ATIVAR GRUPO DE 8 LEDS
                                   ; 1 -> LEDS ATIVADOS
                                   ; 0 -> LEDS DESATIVADOS

#DEFINE    LINHA_4 PORTB,7         ; PINO PARA ATIVAR LINHA 4 DO TECLADO
                                   ;  MATRICIAL
                                   ; 1 -> LINHA ATIVADA
                                   ; 0 -> LINHA DESATIVADA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           VETOR DE RESET                                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   ORG     0x00                    ; ENDEREÇO INICIAL DE PROCESSAMENTO
   GOTO    INICIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                         INÍCIO DA INTERRUPÇÃO                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; AS INTERRUPÇÕES NÃO SERÃO UTILIZADAS, POR ISSO PODEMOS SUBSTITUIR TODO O
; SISTEMA EXISTENTE NO ARQUIVO MODELO PELO APRESENTADO ABAIXO ESTE SISTEMA
; NÃO É OBRIGATÓRIO, MAS PODE EVITAR PROBLEMAS FUTUROS

   ORG     0x04                    ; ENDEREÇO INICIAL DA INTERRUPÇÃO
   RETFIE                          ; RETORNA DA INTERRUPÇÃO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                         INICIO DO PROGRAMA                                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

INICIO
   CLRF    PORTA                   ; LIMPA O PORTA
   CLRF    PORTB                   ; LIMPA O PORTB
   CLRF    PORTC                   ; LIMPA O PORTC
   CLRF    PORTD                   ; LIMPA O PORTD
   CLRF    PORTE                   ; LIMPA O PORTE


   BANK1                           ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                   ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                   ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011001'
   MOVWF   TRISC                   ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                   ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                   ; CONFIGURA I/O DO PORTE
```

```
   MOVLW   B'00000111'
   MOVWF   CMCON                ; DESLIGA COMPARADORES ANALÓGICOS

   MOVLW   B'00000111'
   MOVWF   ADCON1               ; DESLIGA CONVERSORES A/D

   MOVLW   B'10000000'
   MOVWF   OPTION_REG           ; PRESCALER 1:2 NO TMR0
                                ; PULL-UPS DESABILITADOS
                                ; AS DEMAIS CONFG. SÃO IRRELEVANTES
   MOVLW   B'00000000'
   MOVWF   INTCON               ; TODAS AS INTERRUPÇÕES DESLIGADAS

   BANK0                        ; RETORNA PARA O BANCO 0

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INICIALIZAÇÃO DO HARDWARE                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   BSF     C_LEDS               ; ATIVA LEDS LIGADOS AO PORTD

   BSF     LINHA_4              ; ATIVA LINHA 4 DO TECLADO MATRICIAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      INICIALIZAÇÃO DAS VARIÁVEIS                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *


; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          ROTINA PRINCIPAL                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

MAIN
   BTFSS   BOTAO                ; O BOTÃO ESTÁ PRESSIONADO?
   GOTO    BOTAO_LIB            ; NÃO, ENTÃO TRATA BOTÃO LIBERADO
   GOTO    BOTAO_PRES           ; SIM, ENTÃO TRATA BOTÃO PRESSIONADO

BOTAO_LIB
   BCF     LED                  ; APAGA O LED
   GOTO    MAIN                 ; RETORNA AO LOOP PRINCIPAL

BOTAO_PRES
   BSF     LED                  ; ACENDE O LED
   GOTO    MAIN                 ; RETORNA AO LOOP PRINCIPAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          FIM DO PROGRAMA                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                          ; OBRIGATÓRIO
```

Dicas e Comentários

Veja que os pinos do microcontrolador, tanto os de entrada como os de saída são declarados através de DEFINES no início do software o que facilita futuras alterações na pinagem do hardware.

Repare também que o exemplo é extremamente simples e nenhum tipo de tratamento de debounce para a tecla foi utilizado.

Exercícios Propostos

1. Altere a lógica do sistema, ou seja, com o botão pressionado o led deve permanecer apagado e com o botão liberado o led deve permanecer acesso.
2. Altere o software a fim de trocar a tecla ativa, passando por exemplo a utilizar a tecla da linha 4 coluna 2.
3. Altere o software para ligar/desligar outro led, por exemplo, o led ligado ao pino RD3.

Capítulo 4 - Experiência 2 – Contador Simplificado

Objetivo

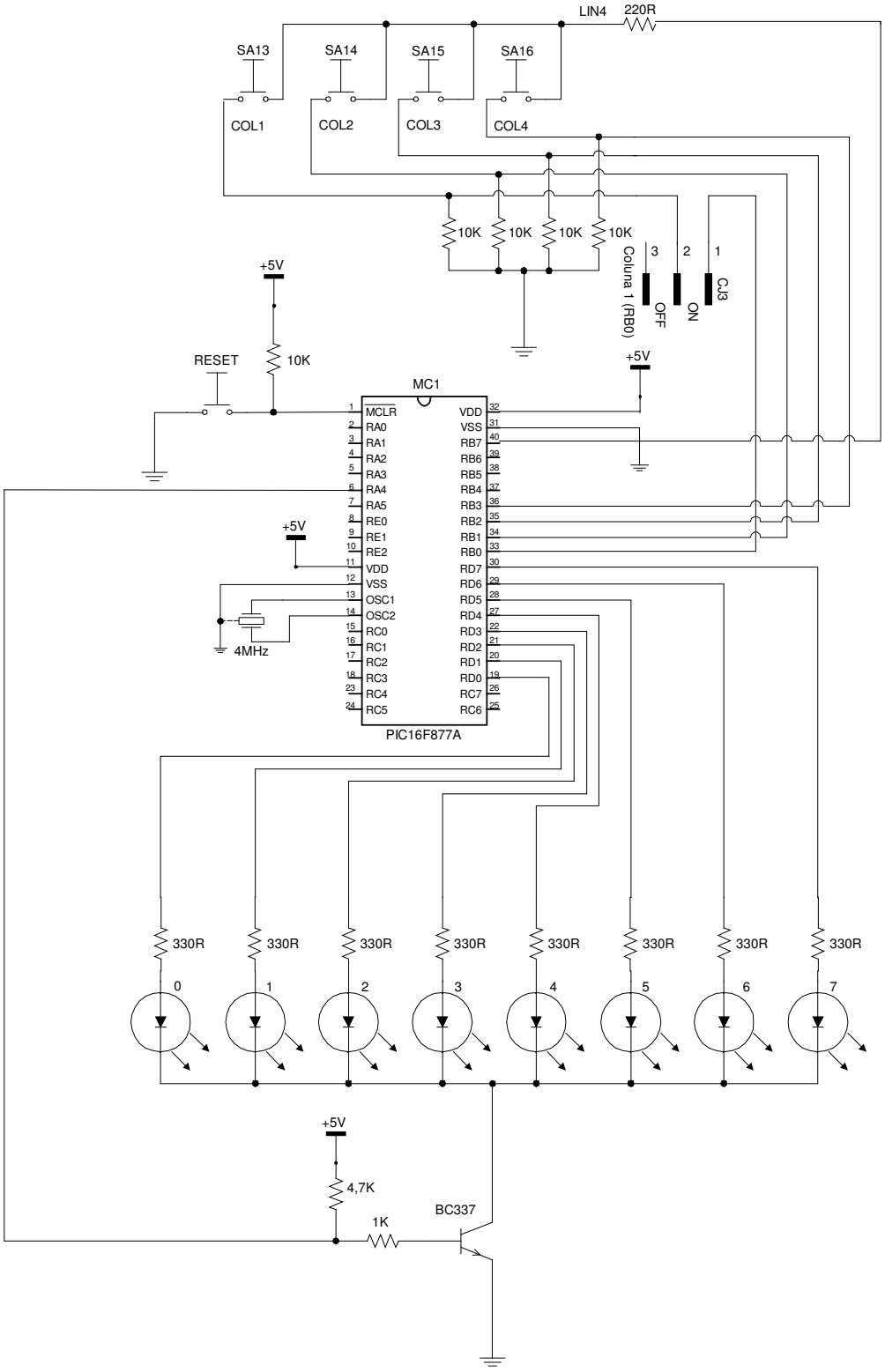
O objetivo desta experiência é ensinar os recursos de software comumente utilizados para tratamento de debounce de teclas e a manipulação de variáveis declaradas na RAM do microcontrolador.

Descrição

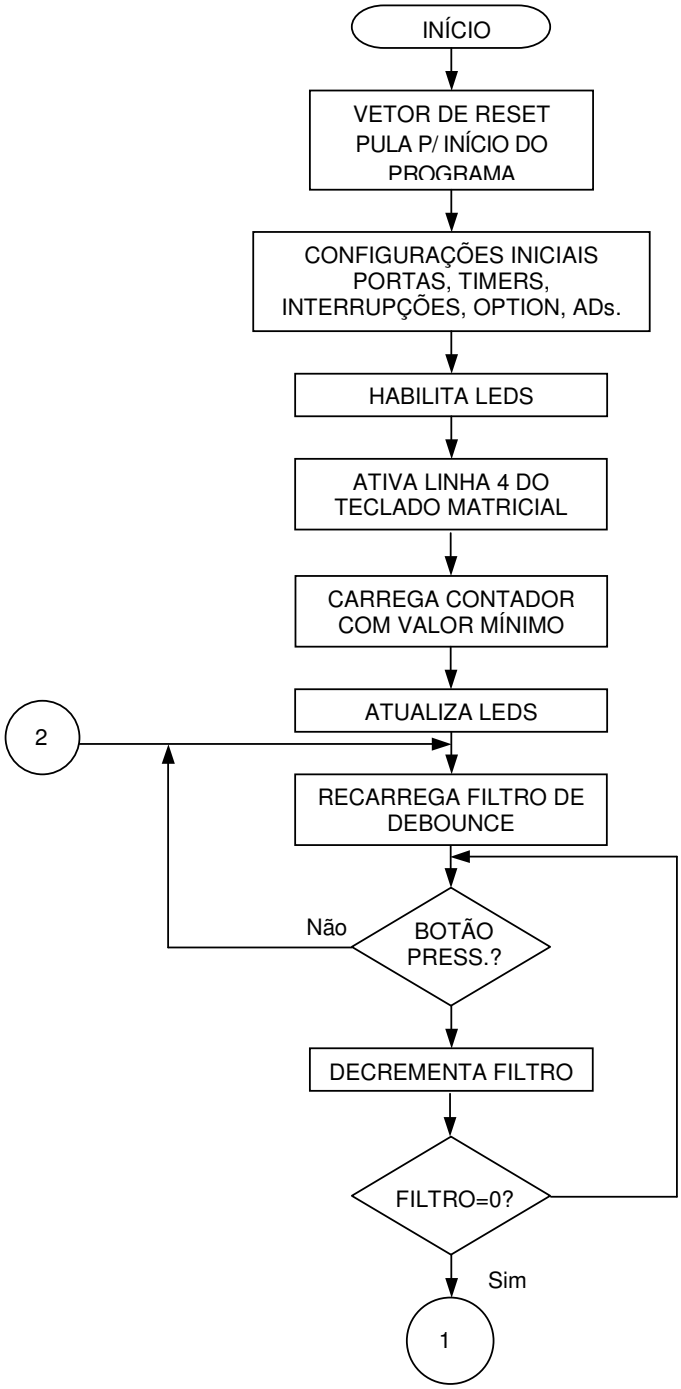
O software faz uso do grupo de leds para representar de forma binária o valor da variável "CONTADOR" declarada na RAM do microcontrolador. Utilizando o botão da linha 4 coluna 1 altera-se o valor da variável através de instruções de incremento e decremento. O valor está limitado por constantes declaradas no início do código. Como apenas um botão é utilizado, a variável é incrementada até o valor máximo e em seguida decrementada até o valor mínimo, permanecendo neste looping indefinidamente. Foi utilizado um flag para alterar o sentido da contagem sempre que um dos extremos é atingido.

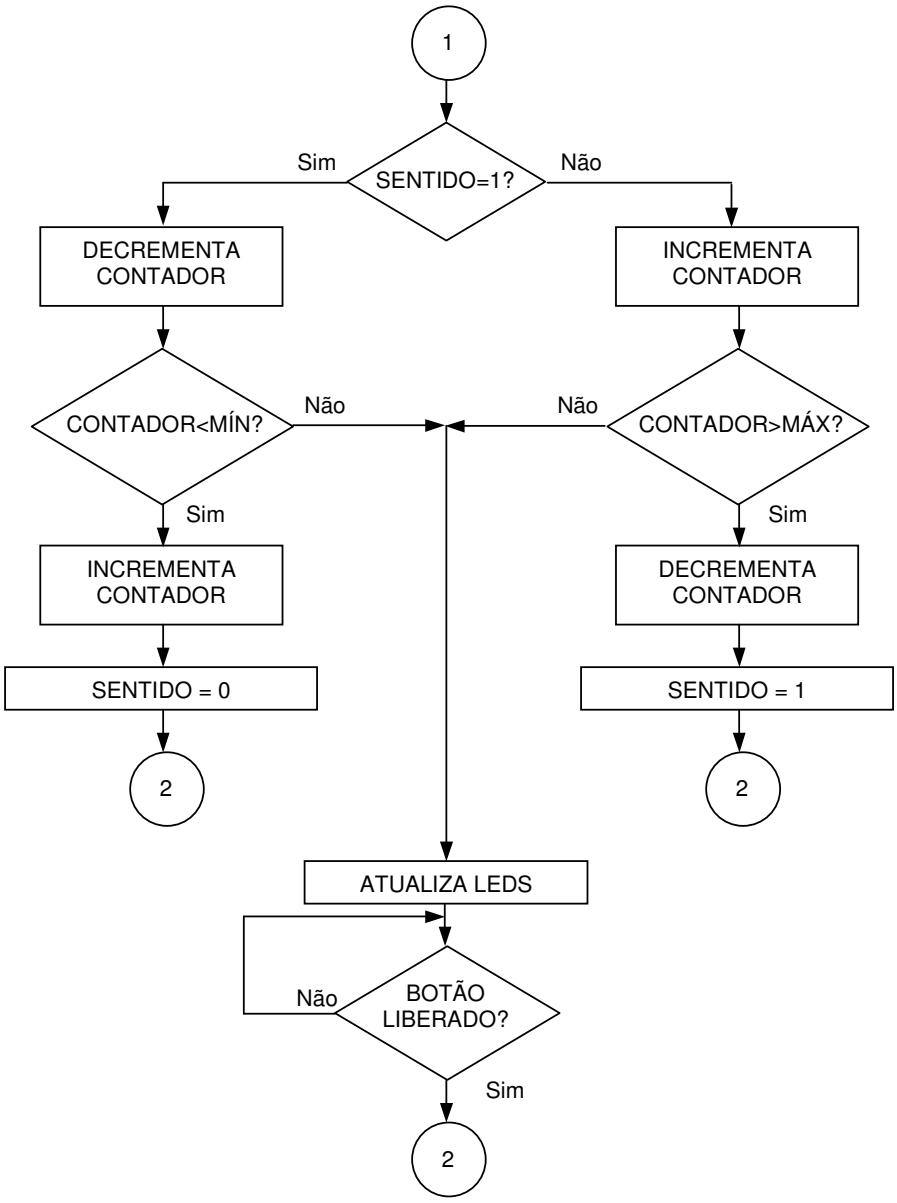
Foi feito o tratamento de debounce da tecla que consiste em testar repetidas vezes se a tecla foi realmente pressionada para somente depois executar a ação correspondente. Sempre que a tecla estiver solta o contador de debounce (variável "FILTRO") é inicializado e sempre que a tecla for pressionada o valor da variável "FILTRO" é decrementado, de forma que a tecla somente é considerada pressionada quando o valor de "FILTRO" for igual a zero.

Esquema Elétrico



Fluxograma


INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURAÇÕES INICIAIS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION, ADs.
HABILITA LEDS
ATIVA LINHA 4 DO
TECLADO MATRICIAL
CARREGA CONTADOR
COM VALOR MÍNIMO
ATUALIZA LEDS
2
RECARREGA FILTRO DE
DEBOUNCE
Não
BOTÃO
PRESS.?
DECREMENTA FILTRO
FILTRO=0?
Sim
1


1
SENTIDO=1?
Sim
Não
DECREMENTA CONTADOR
INCREMENTA CONTADOR
CONTADOR<MÍN?
CONTADOR>MÁX?
Não
Não
Sim
Sim
INCREMENTA CONTADOR
DECREMENTA CONTADOR
SENTIDO = 0
SENTIDO = 1
2
2
ATUALIZA LEDS
BOTÃO LIBERADO?
Não
Sim
2

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *                  EXPERIÊNCIA 2 - CONTADOR SIMPLIFICADO                  *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            DESCRIÇÃO GERAL                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  SISTEMA MUITO SIMPLES PARA INCREMENTAR UM CONTADOR ATÉ UM DETERMINADO
;  VALOR (MAX) E DEPOIS DECREMENTAR ATÉ OUTRO (MIN). O VALOR DO CONTADOR É
;  MOSTRADO NOS 8 LEDS LIGADOS AO PORTD. DEVE-SE UTILIZAR O PRIMEIRO BOTÃO
;  DA LINHA 4 PARA INCREMENTAR E DECREMENTAR O VALOR DO CONTADOR.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      CONFIGURAÇÕES PARA GRAVAÇÃO                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_OFF & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            VARIÁVEIS                                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DOS NOMES E ENDEREÇOS DE TODAS AS VARIÁVEIS UTILIZADAS
; PELO SISTEMA

   CBLOCK  0x20                    ; ENDEREÇO INICIAL DA MEMÓRIA DE
                                   ; USUÁRIO

           W_TEMP                  ; REGISTRADORES TEMPORÁRIOS PARA
           STATUS_TEMP             ; INTERRUPÇÕES
                                   ; ESTAS VARIÁVEIS NEM SERÃO UTI-
                                   ; LIZADAS
           CONTADOR                ; ARMAZENA O VALOR DA CONTAGEM
           FLAGS                   ; ARMAZENA OS FLAGS DE CONTROLE
           FILTRO                  ; FILTRAGEM PARA O BOTÃO

   ENDC                            ; FIM DO BLOCO DE MEMÓRIA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>         ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     DEFINIÇÃO DOS BANCOS DE RAM                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          FLAGS INTERNOS                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS FLAGS UTILIZADOS PELO SISTEMA

#DEFINE    SENTIDO FLAGS,0                 ; FLAG DE SENTIDO
```

```
                                ; 0 -> SOMANDO
                                ; 1 -> SUBTRAINDO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              CONSTANTES                                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODAS AS CONSTANTES UTILIZADAS PELO SISTEMA

MIN               EQU     .10     ; VALOR MÍNIMO PARA O CONTADOR
MAX               EQU     .30     ; VALOR MÁXIMO PARA O CONTADOR
T_FILTRO          EQU     .230    ; FILTRO PARA BOTÃO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               ENTRADAS                                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS PINOS QUE SERÃO UTILIZADOS COMO ENTRADA
; RECOMENDAMOS TAMBÉM COMENTAR O SIGNIFICADO DE SEUS ESTADOS (0 E 1)

#DEFINE    BOTAO   PORTB,0      ; PORTA DO BOTÃO
                                ; 1 -> PRESSIONADO
                                ; 0 -> LIBERADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               SAÍDAS                                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS PINOS QUE SERÃO UTILIZADOS COMO SAÍDA
; RECOMENDAMOS TAMBÉM COMENTAR O SIGNIFICADO DE SEUS ESTADOS (0 E 1)

#DEFINE    C_LEDS  PORTA,4      ; PINO PARA ATIVAR GRUPO DE 8 LEDS
                                ; 1 -> LEDS ATIVADOS
                                ; 0 -> LEDS DESATIVADOS

#DEFINE    LINHA_4 PORTB,7      ; PINO PARA ATIVAR LINHA 4 DO TECLADO
                                ;  MATRICIAL
                                ; 1 -> LINHA ATIVADA
                                ; 0 -> LINHA DESATIVADA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            VETOR DE RESET                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   ORG     0x00                 ; ENDEREÇO INICIAL DE PROCESSAMENTO
   GOTO    INICIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INÍCIO DA INTERRUPÇÃO                                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; AS INTERRUPÇÕES NÃO SERÃO UTILIZADAS, POR ISSO PODEMOS SUBSTITUIR
; TODO O SISTEMA EXISTENTE NO ARQUIVO MODELO PELO APRESENTADO ABAIXO
; ESTE SISTEMA NÃO É OBRIGATÓRIO, MAS PODE EVITAR PROBLEMAS FUTUROS

   ORG     0x04                 ; ENDEREÇO INICIAL DA INTERRUPÇÃO
   RETFIE                       ; RETORNA DA INTERRUPÇÃO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                         INICIO DO PROGRAMA                                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

INICIO
   CLRF    PORTA                ; LIMPA O PORTA
   CLRF    PORTB                ; LIMPA O PORTB
   CLRF    PORTC                ; LIMPA O PORTC
   CLRF    PORTD                ; LIMPA O PORTD
   CLRF    PORTE                ; LIMPA O PORTE


   BANK1                        ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                ; CONFIGURA I/O DO PORTB
```

```
   MOVLW   B'10011001'
   MOVWF   TRISC                   ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                   ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                   ; CONFIGURA I/O DO PORTE

   MOVLW   B'00000111'
   MOVWF   CMCON                   ; DESLIGA COMPARADORES ANALÓGICOS

   MOVLW   B'00000111'
   MOVWF   ADCON1                  ; DESLIGA CONVERSORES A/D

   MOVLW   B'10000000'
   MOVWF   OPTION_REG              ; PRESCALER 1:2 NO TMR0
                                   ; PULL-UPS DESABILITADOS
                                   ; AS DEMAIS CONFG. SÃO IRRELEVANTES
   MOVLW   B'00000000'
   MOVWF   INTCON                  ; TODAS AS INTERRUPÇÕES DESLIGADAS

   BANK0                           ; RETORNA PARA O BANCO 0

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       INICIALIZAÇÃO DO HARDWARE                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   BSF     C_LEDS                  ; ATIVA LEDS LIGADOS AO PORTD

   BSF     LINHA_4                 ; ATIVA LINHA 4 DO TECLADO MATRICIAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      INICIALIZAÇÃO DAS VARIÁVEIS                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   MOVLW   MIN
   MOVWF   CONTADOR                ; INICIA CONTADOR = V_INICIAL
   MOVWF   PORTD                   ; INICIA SAIDA = V_INICIAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          ROTINA PRINCIPAL                               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

MAIN
   MOVLW   T_FILTRO
   MOVWF   FILTRO                  ; INICIALIZA FILTRO = T_FILTRO

CHECA_BT
   BTFSC   BOTAO                   ; O BOTÃO ESTÁ PRESSIONADO?
   GOTO    MAIN                    ; NÃO, ENTÃO CONTINUA ESPERANDO
                                   ; SIM
   DECFSZ  FILTRO,F                ; DECREMENTA O FILTRO DO BOTÃO
                                   ; TERMINOU?
   GOTO    CHECA_BT                ; NÃO, CONTINUA ESPERANDO
                                   ; SIM

TRATA_BT
   BTFSS   SENTIDO                 ; DEVE SOMAR (SENTIDO=0)?
   GOTO    SOMA                    ; SIM
                                   ; NÃO
SUBTRAI
   DECF    CONTADOR,F              ; DECREMENTA O CONTADOR

   MOVLW   MIN                     ; MOVE O VALOR MÍNIMO PARA W
   SUBWF   CONTADOR,W              ; SUBTRAI O VALOR DE W (MIN) DE CONTADOR
   BTFSC   STATUS,C                ; TESTA CARRY. RESULTADO NEGATIVO?
   GOTO    ATUALIZA                ; NÃO, ENTÃO CONTA >= MIN
                                   ; SIM, ENTÃO CONTA < MIN

   INCF    CONTADOR,F              ; INCREMENTA CONTADOR NOVAMENTE
```

```
                                ; POIS PASSOU DO LIMITE
   BCF     SENTIDO              ; MUDA SENTIDO PARA SOMA
   GOTO    MAIN                 ; VOLTA AO LOOP PRINCIPAL

SOMA
   INCF    CONTADOR,F           ; INCREMENTA O CONTADOR

   MOVLW   MAX                  ; MOVE O VALOR MÁXIMO PARA W
   SUBWF   CONTADOR,W           ; SUBTRAI O VALOR DE W (MIN) DE CONTADOR
   BTFSS   STATUS,C             ; TESTA CARRY. RESULTADO NEGATIVO?
   GOTO    ATUALIZA             ; SIM, ENTÃO CONTA < MAX
                                ; NÃO, ENTÃO CONTA >= MAX

   BSF     SENTIDO              ; MUDA SENTIDO PARA SUBTRAÇÃO
   GOTO    MAIN                 ; VOLTA AO LOOP PRINCIPAL

ATUALIZA
   MOVF    CONTADOR,W           ; COLOCA CONTADOR EM W
   MOVWF   PORTD                ; ATUALIZA O PORTD PARA
                                ; VISUALIZARMOS O VALOR DE CONTADOR

   BTFSS   BOTAO                ; O BOTÃO CONTINUA PRESSIONADO?
   GOTO    $-1                  ; SIM, ENTÃO ESPERA LIBERAÇÃO PARA
                                ; QUE O CONTADOR NÃO DISPARE
   GOTO    MAIN                 ; NÃO, VOLTA AO LOOP PRINCIPAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          FIM DO PROGRAMA                                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                          ; OBRIGATÓRIO
```

Dicas e Comentários

O tempo do debounce pode ser calculado pelo tempo de execução da varredura da tecla multiplicado pelo valor inicial da variável de filtro. No exemplo estudado o tempo de debounce é de 1,15ms e foi calculado seguindo a equação abaixo:

Tempo de Debounce = ciclo de máquina X número de ciclos de máquina na varredura da tecla X valor inicial da variável “FILTRO”

Tempo de Debounce = 1us x 5 x 230

Tempo de Debounce = 1150us = 1,15ms

Não existe uma regra para calcular o tempo ideal que deva ser utilizado no debounce de teclas. Isso irá depender do projeto, do tipo de tecla em utilização, do ambiente de trabalho onde o sistema irá operar, etc. Na prática, tempos longos da ordem de grandeza de dezenas de milisegundos podem ser necessários.

Exercícios Propostos

1. Alterar os limites da contagem.
2. Retirar o tratamento de debounce do software e verificar os efeitos práticos.

Capítulo 5 - Experiência 3 – Pisca - Pisca

Objetivo

O objetivo desta experiência é ensinar ao aluno com criar rotinas de delays além de apresentar uma técnica simples utilizada para inverter o estado de um bit.

Descrição

O software desta experiência utiliza um dos displays de 7 segmentos para implementar um pisca-pisca sendo que a freqüência das piscadas é controlada através do uso de uma rotina de delay.

A rotina de delay é genérica é pode ser utilizada para gerar delays entre 1ms e 256ms. Na realidade, a rotina recebe um argumento passado pelo WORK para determinar o atraso que deve ser gerado, sempre em múltiplos de 1ms. Como o argumento é de 8 bits existem 256 possíveis delays, indo de 1ms até 256ms. Basta portanto, carregar o WORK com o delay desejado e chamar a rotina.

O pisca-pisca é visualizado no display de 7 segmentos na posição da unidade. Sempre que o delay é finalizado o PORTD deve ser invertido e para inverter o estado destes bits utilizou-se a operação lógica boleana XOR. Conforme a tabela verdade apresentada a seguir

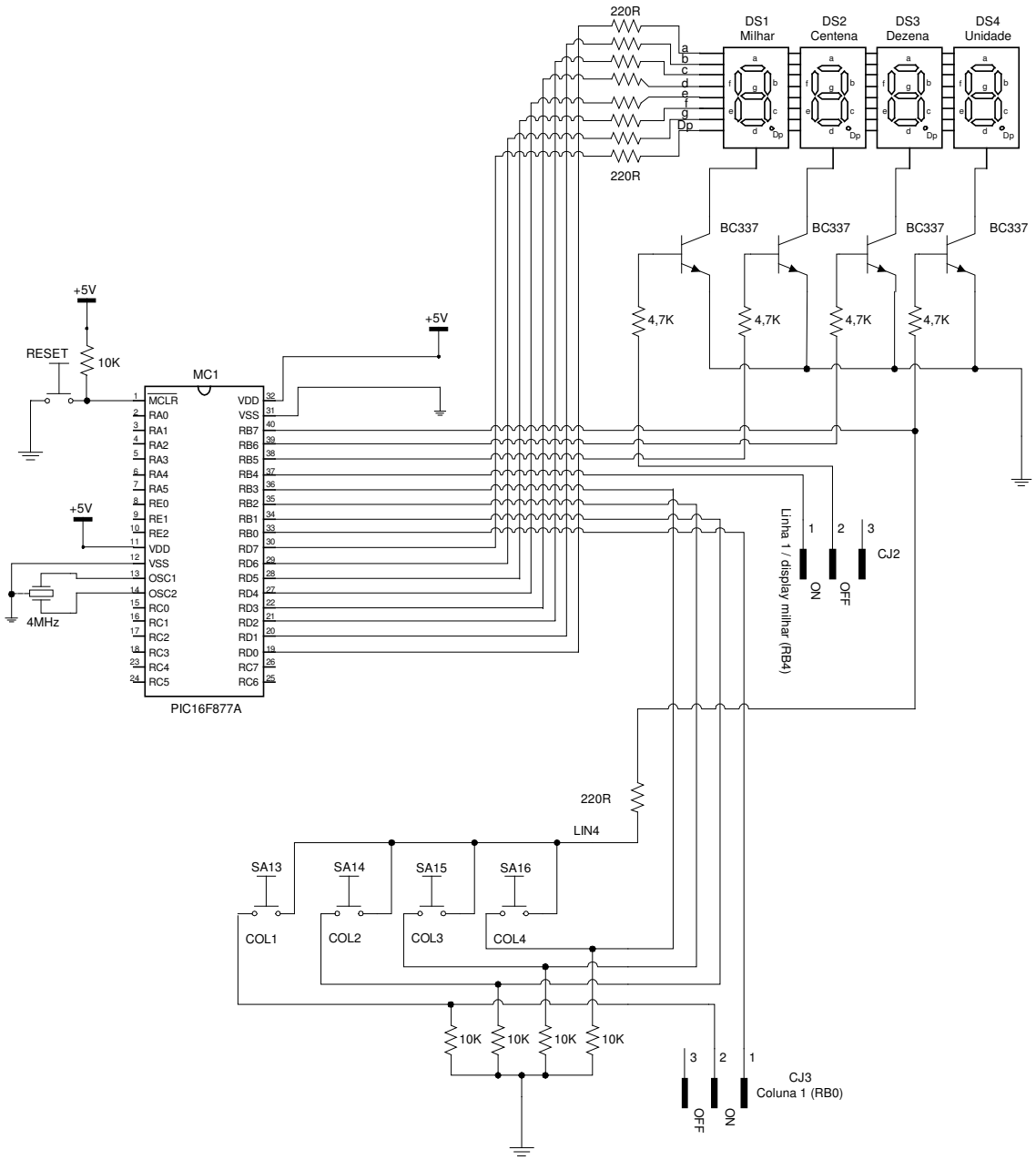
A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

pode-se verificar que sempre que a operação é realizada quando os bits da coluna **A** estão em 1 o resultado fica invertido em relação à coluna **B** e sempre que os bits da coluna **A** estão em 0 o resultado se mantém em relação à coluna **B**. Assim, sempre que ser desejar inverter um bit, basta fazer uma operação XOR entre um bit em 1 e o bit que se deseja inverter.

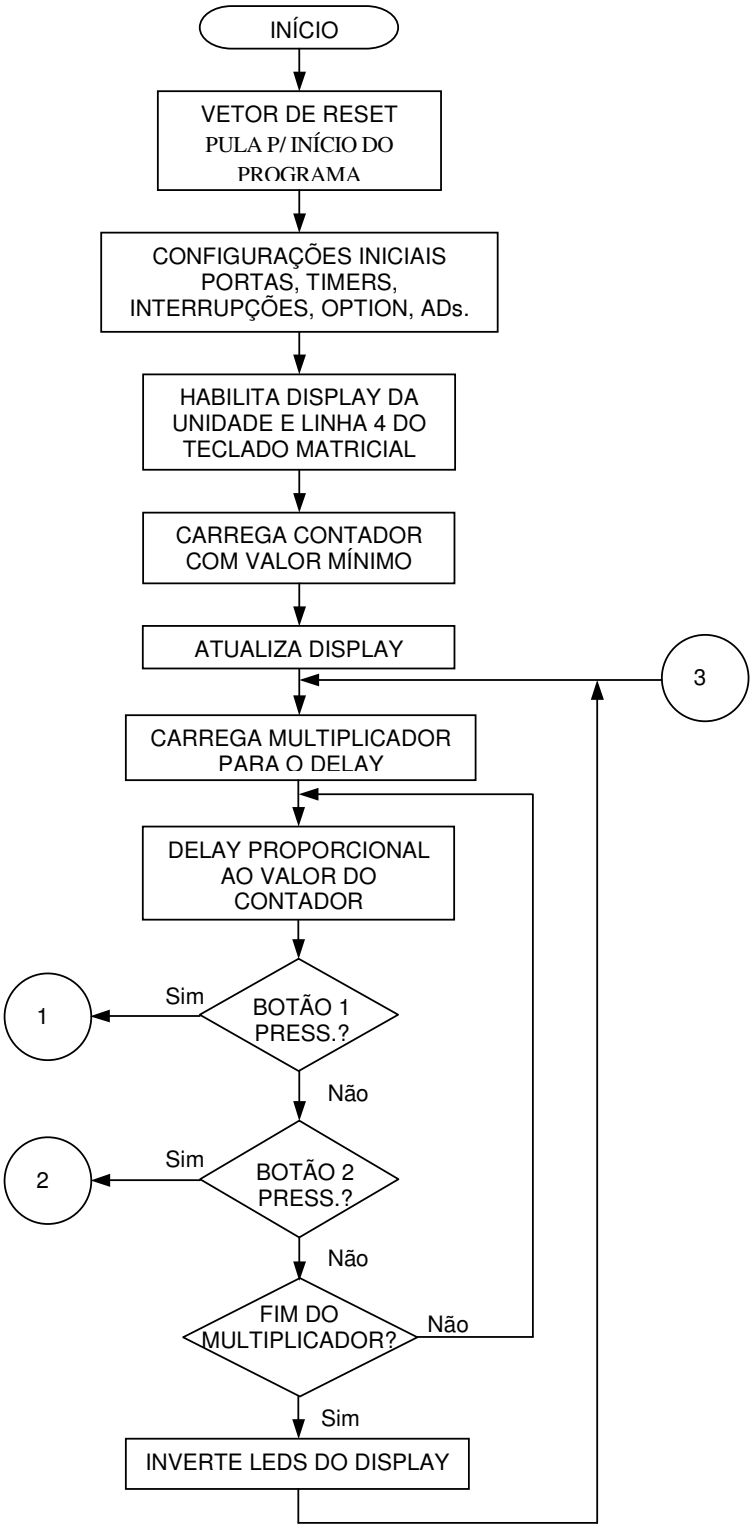
Esta é uma técnica simples de inverter o estado de um bit sem testar o estado original.

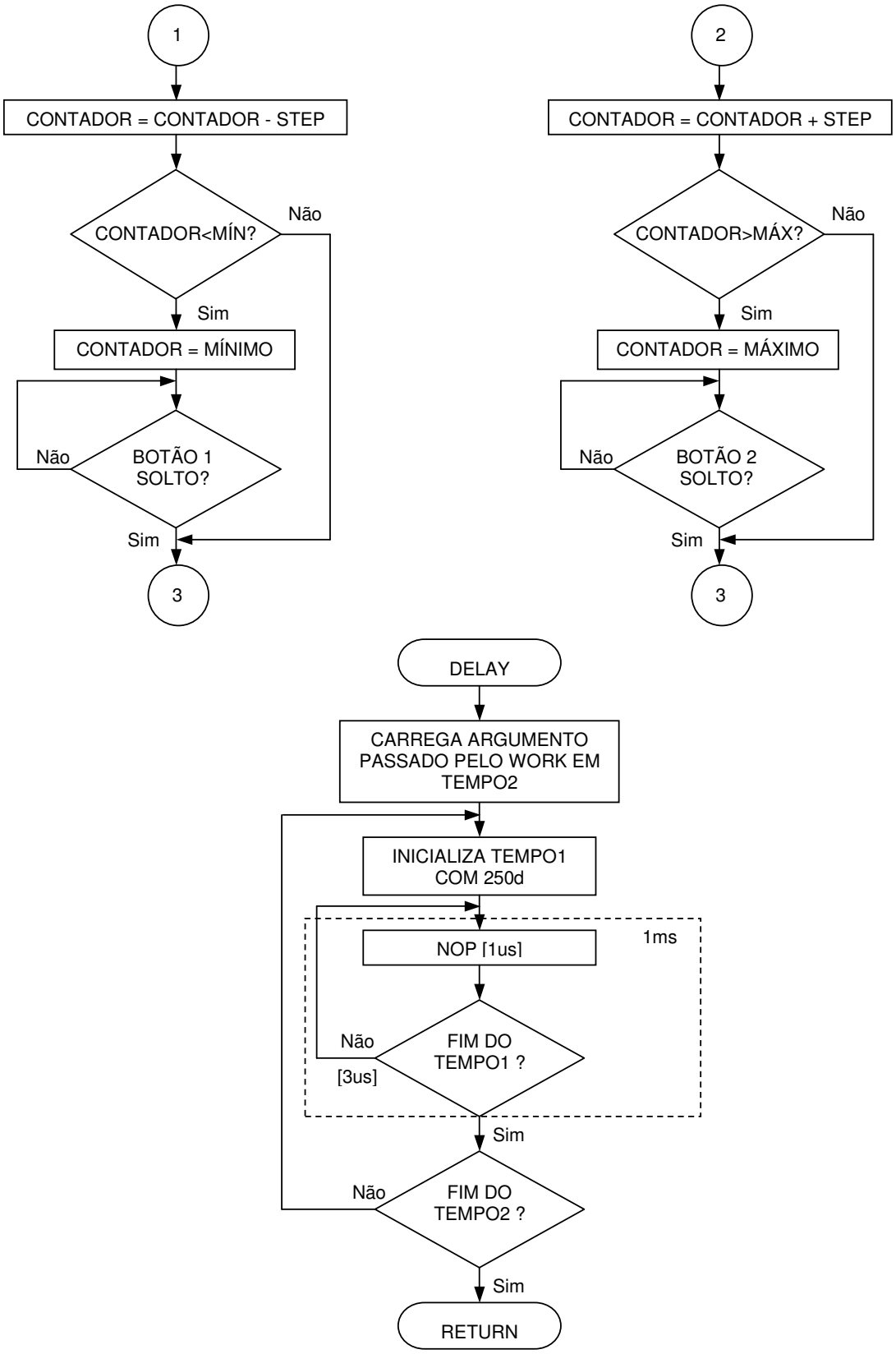
No software, pode-se utilizar as teclas da linha 4 colunas 1 e 2 para alterar o tempo do delay e conseqüentemente a freqüência das piscadas do display. A tecla da coluna 1 incrementa o valor do delay enquanto a tecla da coluna 2 decrementa.

Esquema Elétrico



Fluxograma


INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURAÇÕES INICIAIS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION, ADs.
HABILITA DISPLAY DA
UNIDADE E LINHA 4 DO
TECLADO MATRICIAL
CARREGA CONTADOR
COM VALOR MÍNIMO
ATUALIZA DISPLAY
3
CARREGA MULTIPLICADOR
PARA O DELAY
DELAY PROPORCIONAL
AO VALOR DO
CONTADOR
1
Sim
BOTÃO 1
PRESS.?
Não
2
Sim
BOTÃO 2
PRESS.?
Não
FIM DO
MULTIPLICADOR?
Não
Sim
INVERTE LEDS DO DISPLAY


1
CONTADOR = CONTADOR - STEP
CONTADOR<MÍN?
Não
Sim
CONTADOR = MÍNIMO
Não
BOTÃO 1 SOLTO?
Sim
3
2
CONTADOR = CONTADOR + STEP
CONTADOR>MÁX?
Não
Sim
CONTADOR = MÁXIMO
Não
BOTÃO 2 SOLTO?
Sim
3
DELAY
CARREGA ARGUMENTO PASSADO PELO WORK EM TEMPO2
INICIALIZA TEMPO1 COM 250d
1ms
NOP [1us]
Não
[3us]
FIM DO TEMPO1 ?
Sim
Não
FIM DO TEMPO2 ?
Sim
RETURN

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER           *
; *                                                                         *
; *                        EXPERIÊNCIA 3 - PISCA-PISCA                      *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           DESCRIÇÃO GERAL                               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; PISCA-PISCA VARIÁVEL PARA DEMONSTRAR A IMPLEMENTAÇÃO DE DELAYS E A INVERSÃO
; DE PORTAS. APENAS OS BOTÕES DA LINHA 4 ESTÃO ATIVADOS SENDO QUE O DA
; COLUNA 1 É UTILIZADO PARA INCREMENTAR O TEMPO ENTRE AS PISCADAS.
; O BOTÃO DA  COLUNA 2 É UTILIZADO PARA DIMINUIR O TEMPO ENTRE AS PISCADAS.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        CONFIGURAÇÕES PARA GRAVAÇÃO                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_OFF & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                             VARIÁVEIS                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DOS NOMES E ENDEREÇOS DE TODAS AS VARIÁVEIS UTILIZADAS
; PELO SISTEMA

   CBLOCK  0x20                  ; ENDEREÇO INICIAL DA MEMÓRIA DE
                                 ; USUÁRIO

           W_TEMP                ; REGISTRADORES TEMPORÁRIOS PARA
           STATUS_TEMP           ; INTERRUPÇÕES
                                 ; ESTAS VARIÁVEIS NEM SERÃO UTI-
                                 ; LIZADAS
           CONTADOR              ; BASE DE TEMPO PARA A PISCADA
           TEMPO1                ; REGISTRADORES AUXILIARES DE TEMPO
           TEMPO2
           TEMPO3

   ENDC                          ; FIM DO BLOCO DE MEMÓRIA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>       ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    DEFINIÇÃO DOS BANCOS DE RAM                          *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            FLAGS INTERNOS                               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS FLAGS UTILIZADOS PELO SISTEMA
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              CONSTANTES                                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODAS AS CONSTANTES UTILIZADAS PELO SISTEMA

MIN               EQU     .10
MAX               EQU     .240
STEP              EQU     .5
MULTIPLO          EQU     .5

;A CONSTANTE DISPLAY REPRESENTA O SÍMBOLO QUE APARECERÁ PISCANDO NO
;DISPLAY. 1=LED LIGADO E 0=LED DESLIGADO. A RELAÇÃO ENTRE BITS E
;SEGMENTOS É A SEGUINTE: '.GFEDCBA'
;     a
;     **********
;     *        *
;   f *        * b
;     *    g   *
;     **********
;     *        *
;   e *        * c
;     *    d   *
;     **********  *.

DISPLAY           EQU     B'01110110'      ; (LETRA H)

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               ENTRADAS                                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS PINOS QUE SERÃO UTILIZADOS COMO ENTRADA
; RECOMENDAMOS TAMBÉM COMENTAR O SIGNIFICADO DE SEUS ESTADOS (0 E 1)

#DEFINE   BT1     PORTB,0         ; BOTÃO 1 - INCREMENTA
                                  ; 1 -> PRESSIONADO
                                  ; 0 -> LIBERADO

#DEFINE   BT2     PORTB,1         ; BOTÃO 2 - DECREMENTA
                                  ; 1 -> PRESSIONADO
                                  ; 0 -> LIBERADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               SAÍDAS                                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS PINOS QUE SERÃO UTILIZADOS COMO SAÍDA
; RECOMENDAMOS TAMBÉM COMENTAR O SIGNIFICADO DE SEUS ESTADOS (0 E 1)

#DEFINE   DSP_UNIDADE     PORTB,7 ; PINO DISPLAY DA UNIDADE
                                  ; 1 -> DISPLAY ATIVADO
                                  ; 0 -> DISPLAY DESATIVADO

#DEFINE   LINHA_4         PORTB,7 ; PINO PARA ATIVAR LINHA 4 DO TECLADO
                                  ;  MATRICIAL
                                  ; 1 -> LINHA ATIVADA
                                  ; 0 -> LINHA DESATIVADA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            VETOR DE RESET                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   ORG     0x00                   ; ENDEREÇO INICIAL DE PROCESSAMENTO
   GOTO    INICIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INÍCIO DA INTERRUPÇÃO                                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; AS INTERRUPÇÕES NÃO SERÃO UTILIZADAS, POR ISSO PODEMOS SUBSTITUIR
; TODO O SISTEMA EXISTENTE NO ARQUIVO MODELO PELO APRESENTADO ABAIXO
; ESTE SISTEMA NÃO É OBRIGATÓRIO, MAS PODE EVITAR PROBLEMAS FUTUROS

   ORG     0x04                   ; ENDEREÇO INICIAL DA INTERRUPÇÃO
   RETFIE                         ; RETORNA DA INTERRUPÇÃO
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              ROTINA DE DELAY                                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA AGUARDA TANTOS MILISEGUNDOS QUANTO O VALOR PASSADO
; POR W. POR EXEMPLO, SE W = .200, ELA AGUARDARÁ 200 MILISEGUNDOS.
;
; O DELAY PRINCIPAL DURA 1ms, POIS POSSUI 4 INSTRUÇÕES (4us) E É
; RODADO 250 VEZES (TEMPO1). PORTANTO 250 * 4us = 1ms.
; O DELAY PRINCIPAL É RODADO TANTAS VEZES QUANTO FOR O VALOR DE
; TEMPO2, O QUAL É INICIADO COM O VALOR PASSADO EM W.


DELAY
   MOVWF   TEMPO2                ; INICIA TEMPO 2 COM O VALOR
                                 ; PASSADO EM W
DL1
   MOVLW   .250
   MOVWF   TEMPO1

DL2                              ; ESTE DELAY DURA 1ms (4*250)
   NOP
   DECFSZ  TEMPO1,F              ; DECREMENTA TEMPO1. ACABOU?
   GOTO    DL2                   ; NÃO, CONTINUA AGUARDANDO
                                 ; SIM

   DECFSZ  TEMPO2,F              ; DECREMENTA TEMPO2. ACABOU?
   GOTO    DL1                   ; NÃO, CONTINUA AGUARDANDO
                                 ; SIM
   RETURN

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          INICIO DO PROGRAMA                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

INICIO
   CLRF    PORTA                 ; LIMPA O PORTA
   CLRF    PORTB                 ; LIMPA O PORTB
   CLRF    PORTC                 ; LIMPA O PORTC
   CLRF    PORTD                 ; LIMPA O PORTD
   CLRF    PORTE                 ; LIMPA O PORTE


   BANK1                         ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                 ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                 ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011001'
   MOVWF   TRISC                 ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                 ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                 ; CONFIGURA I/O DO PORTE

   MOVLW   B'00000111'
   MOVWF   CMCON                 ; DESLIGA COMPARADORES ANALÓGICOS

   MOVLW   B'00000111'
   MOVWF   ADCON1                ; DESLIGA CONVERSORES A/D

   MOVLW   B'10000000'
   MOVWF   OPTION_REG            ; PRESCALER 1:2 NO TMR0
                                 ; PULL-UPS DESABILITADOS
                                 ; AS DEMAIS CONFG. SÃO IRRELEVANTES
   MOVLW   B'00000000'
   MOVWF   INTCON                ; TODAS AS INTERRUPÇÕES DESLIGADAS

   BANK0                         ; RETORNA PARA O BANCO 0
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      INICIALIZAÇÃO DO HARDWARE                            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   BSF     DSP_UNIDADE             ; ATIVA DISPLAY DA UNIDADE
                                   ; ESTE PINO TAMBÉM É UTILIZADO
                                   ; PARA ATIVAR A LINHA 4 DO TECLADO
                                   ; MATRICIAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      INICIALIZAÇÃO DAS VARIÁVEIS                          *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   MOVLW   DISPLAY
   MOVWF   PORTD                   ; ACENDE O VALOR CERTO NO DISPLAY
   MOVLW   MIN
   MOVWF   CONTADOR                ; INICIA CONTADOR COM VALOR MIN.

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          ROTINA PRINCIPAL                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

MAIN
   MOVLW   MULTIPLO
   MOVWF   TEMPO3                  ; INICIA COMTADOR DE MULTIPLICAÇÃO,
                                   ; POIS OS TEMPOS GERADOS POR DELAY
                                   ; SÃO MUITO PEQUENOS, GERANDO FREQ.
                                   ; MUITO ALTAS PARA A VISUALIZAÇÃO.
MAIN1
   MOVF    CONTADOR,W              ; COLOCA CONTADOR EM W
                                   ; PARA CHAMAR A ROTINA DE DELAY
   CALL    DELAY                   ; CHAMA ROTINA DE DELAY

   BTFSC   BT1                     ; BOTÃO 1 PRESSIONADO?
   GOTO    INCREMENTA              ; SIM, DEVE INCREMENTAR
                                   ; NÃO

   BTFSC   BT2                     ; BOTÃO 2 PRESSIONADO?
   GOTO    DECREMENTA              ; SIM, DEVE DECREMENTAR
                                   ; NÃO

   DECFSZ  TEMPO3,F                ; DECREMENTA CONTADOR DE MULT. ACABOU?
   GOTO    MAIN1                   ; NÃO, CONTINUA AGUARDANDO
                                   ; SIM

   MOVLW   DISPLAY                 ; APÓS TRANSCORRIDO O TEMPO, IRÁ
                                   ; INVERTER OS LEDS CORRETOS ATRAVÉS
                                   ; DA MÁSCARA "DISPLAY" E DA OPERAÇÃO
                                   ; XOR
   XORWF   PORTD,F                 ; INVERTE LEDS -> PISCA

   GOTO    MAIN                    ; COMEÇA NOVAMENTE


DECREMENTA
   MOVLW   STEP
   SUBWF   CONTADOR,F              ; DECREMENTA O CONTADOR EM STEP

   MOVLW   MIN                     ; MOVE O VALOR MÍNIMO PARA W
   SUBWF   CONTADOR,W              ; SUBTRAI O VALOR DE W (MIN) DE CONTADOR
   BTFSC   STATUS,C                ; TESTA CARRY. RESULTADO NEGATIVO?
   GOTO    MAIN                    ; NÃO, ENTÃO CONTA >= MIN
                                   ; SIM, ENTÃO CONTA < MIN

   MOVLW   MIN
   MOVWF   CONTADOR                ; ACERTA CONTADOR NO MÍNIMO, POIS
                                   ; PASSOU DO VALOR

   BTFSS   BT2                     ; BOTÃO 2 CONTINUA PRESSIONADO?
   GOTO    $-1                     ; SIM, AGUARDA LIBERAÇÃO
                                   ; NÃO
```

```
   GOTO    MAIN                    ; VOLTA AO LOOP PRINCIPAL

INCREMENTA
   MOVLW   STEP
   ADDWF   CONTADOR,F              ; INCREMENTA O CONTADOR EM STEP

   MOVLW   MAX                     ; MOVE O VALOR MÁXIMO PARA W
   SUBWF   CONTADOR,W              ; SUBTRAI O VALOR DE W (MIN) DE CONTADOR
   BTFSS   STATUS,C                ; TESTA CARRY. RESULTADO NEGATIVO?
   GOTO    MAIN                    ; SIM, ENTÃO CONTA < MAX
                                   ; NÃO, ENTÃO CONTA >= MAX
   MOVLW   MAX
   MOVWF   CONTADOR                ; ACERTA CONTADOR NO MÁXIMO, POIS
                                   ; PASSOU DO VALOR
   BTFSS   BT1                     ; BOTÃO 1 CONTINUA PRESSIONADO?
   GOTO    $-1                     ; SIM, AGUARDA LIBERAÇÃO
                                   ; NÃO
   GOTO    MAIN                    ; VOLTA AO LOOP PRINCIPAL


; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          FIM DO PROGRAMA                                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                             ; OBRIGATÓRIO
```

Dicas e Comentários

A rotina de delay recebe o argumento passado pelo WORK para determinar o delay que deve ser gerado, no entanto, o valor máximo assumido pelo WORK é 255 e conforme comentado o delay máximo gerado é de 256ms. Como isto é possível?

Acontece que a rotina foi escrita de forma que ela é executada pelo menos uma vez. Assim, a rotina executa um delay de 1ms, decrementa o argumento e quando este é igual a zero retorna. Porém, quando o argumento passado pelo WORK é zero, ao primeiro decremento, o argumento estoura e pula para 255. Como a condição é testada após o decremento e como após o decremento o valor do argumento é 255, o delay continua a ser gerado, por mais 255 vezes e desta forma obtêm-se um delay de 256ms.

A proposta da rotina de delay foi gerar um delay fundamental em 1ms, porém, a mesma idéia pode ser utilizada para criar rotinas com delays fundamentais diferentes.

Exercícios Propostos

1. Alterar a rotina de delay para gerar um delay fundamental de 100us.
2. Alterar a rotina de delay para gerar um delay fundamental de 10ms.
3. Incluir o tratamento de debounce nas teclas

Capítulo 6 - Experiência 4 –Conversão BCD para displays de 7 segmentos

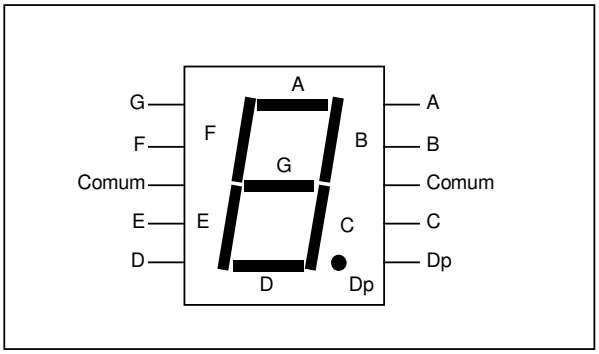
Objetivo

O objetivo desta experiência é ensinar ao aluno como criar um decodificar BCD para displays de 7 segmentos.

Descrição

Os displays utilizados no MCMASTER são conhecidos como displays de leds de 7 segmentos, pois os números são compostos por 7 traços. Estes componentes possuem ainda o ponto decimal e são considerados displays numéricos, por não possuírem traços suficientes para a exibição de todas as letras do nosso alfabeto.

Para facilitar a vida do projetista o mercado padronizou uma nomenclatura para todos os traços do display, possibilitando que tratemos cada um deles individualmente:



Desta forma, temos um pino para controlar cada um dos segmentos (A...G) e mais o ponto decimal (Dp). Os dois pinos adicionais são os comuns, que podem ser ligados a todos os catodos ou anodos dos leds internos. Por causa disso, estes displays são fornecidos em 2 tipos: catodo comum ou anodo comum. No nosso caso, os displays utilizados são do tipo catodo comum, isto é, o pino comum deve ser ligado ao terra e os segmentos devem ser ligados ao Vcc para acenderem.

Outra observação importante é que a pinagem descrita no desenho é válida para o tipo de display utilizado no MCMASTER. Existem displays de outros tamanhos que possuem uma disposição de pinos diferente.

Como cada segmento é um led individual, precisa-se de um pino do PIC para controlar cada segmento. Desta forma, são necessários 8 pinos para acionar os 7 segmentos e mais o ponto decimal.

A fim de converter o valor binário de um algarismo em um valor de 8 bits que represente o este algarismo num display de 7 segmentos fez-se uso de uma tabela de conversão. Por exemplo, para representar o algarismo “2” no display, deve-se acender os segmentos A, B, D, E e G.

Como no kit MCMASTER o PORTD está conectado aos segmentos conforme a tabela

PIC	Segmento
RD0	A
RD1	B
RD2	C
RD3	D
RD4	E
RD5	F
RD6	G
RD7	DP

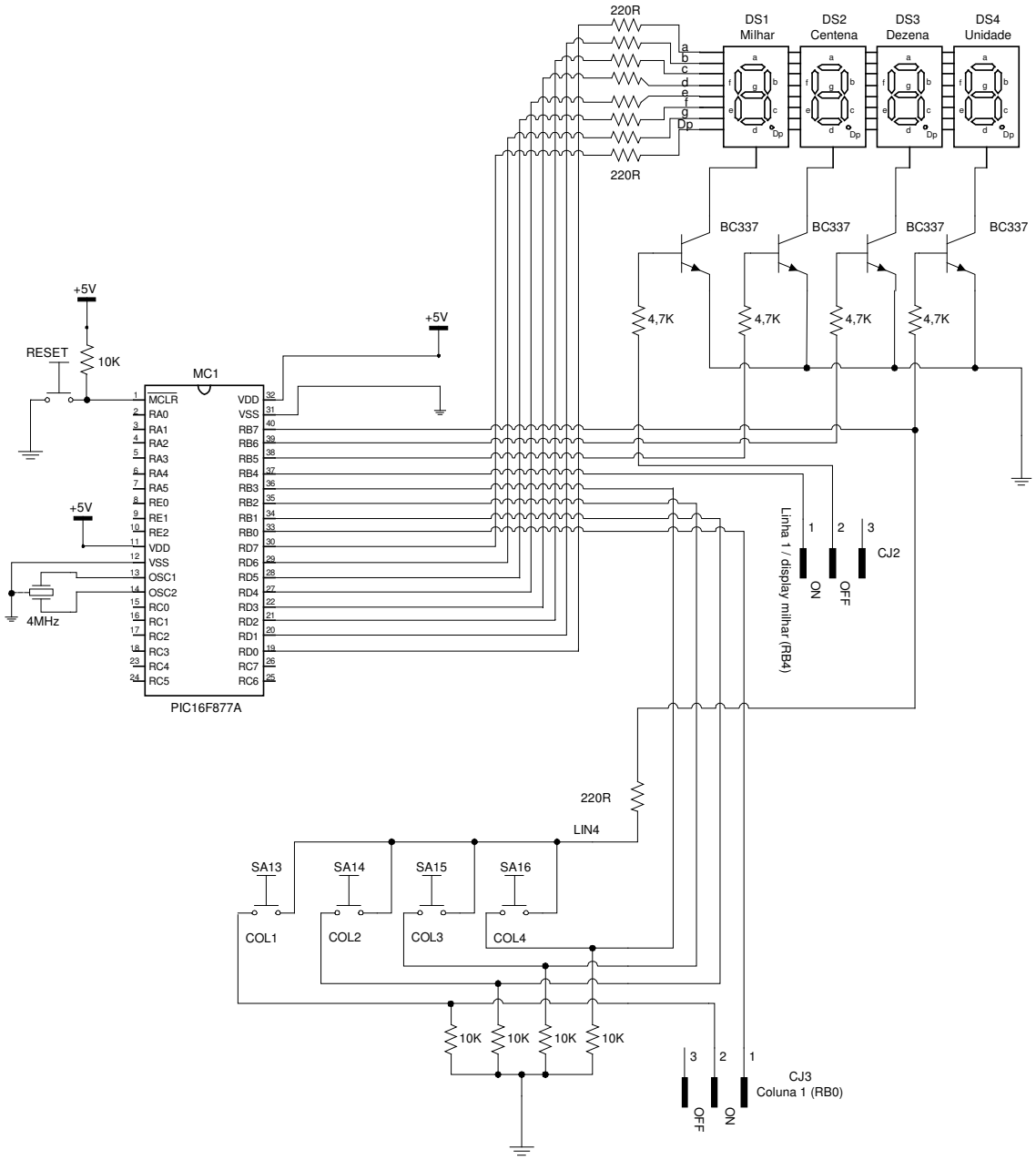
para acender o algarismo “2” no display precisa-se colocar o valor 01011011b no PORTD, ou seja, colocar em 1 os bits ligados aos segmentos que se deseja acender.

Assim, no software deve ser criada uma tabela para converter cada valor binário numa representação que posta nos segmentos (PORTD) reproduza o valor binário original.

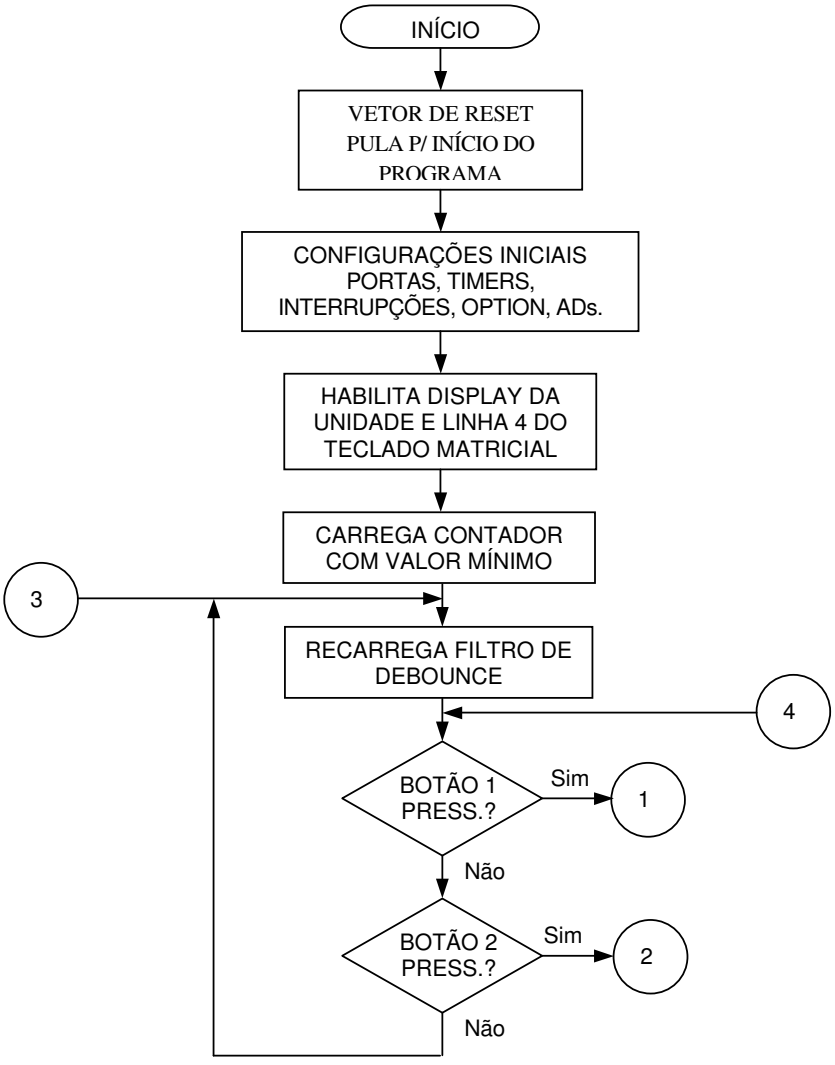
A tabela é criada utilizando-se a instrução RETLW. O valor binário que se deseja converter é adicionado ao PCL (program counter) de forma a desviar o fluxo do programa para a linha que contém a combinação de 0 e 1 que formarão o caractere no display.

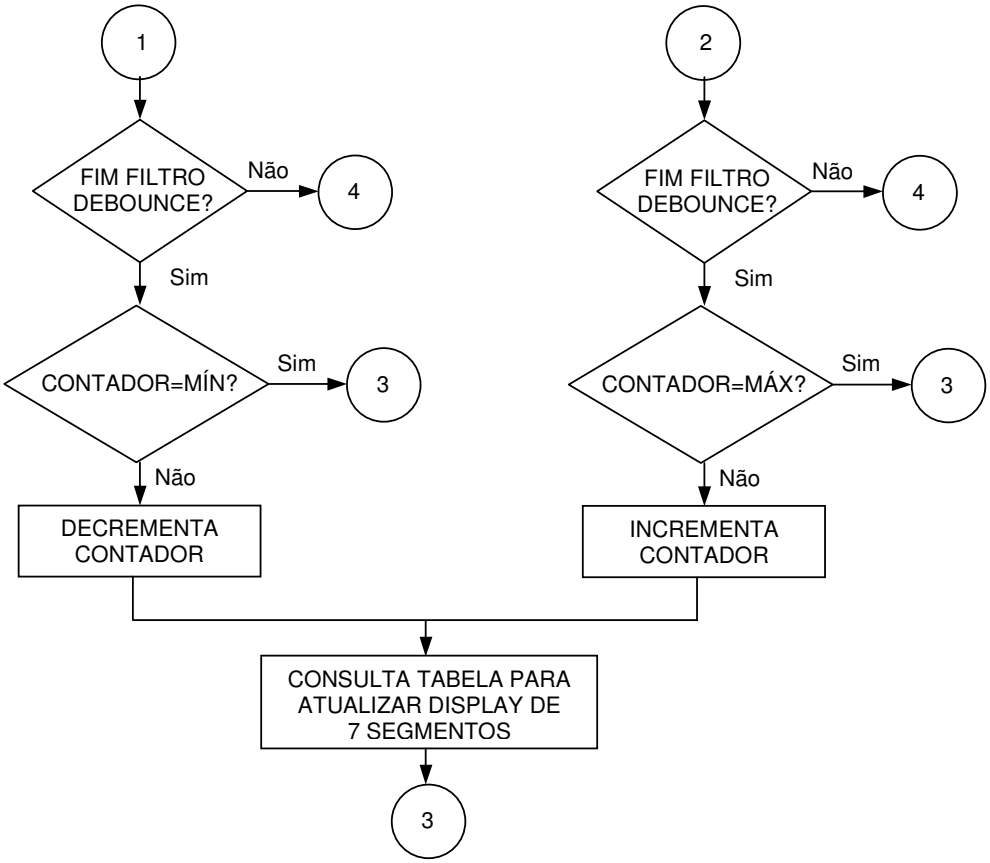
O software utiliza as teclas da linha 4 colunas 1 e 2 para incrementar e decrementar o valor da variável “CONTADOR”. Esta variável está limitada pelas constantes "MIN" e "MAX". A tabela de conversão foi utilizada a fim de visualizar o valor da variável “CONTADOR” no display de 7 segmentos.

Esquema Elétrico



Fluxograma


INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURAÇÕES INICIAIS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION, ADs.
HABILITA DISPLAY DA
UNIDADE E LINHA 4 DO
TECLADO MATRICIAL
CARREGA CONTADOR
COM VALOR MÍNIMO
3
RECARREGA FILTRO DE
DEBOUNCE
4
BOTÃO 1
PRESS.?
Sim
1
Não
BOTÃO 2
PRESS.?
Sim
2
Não


1
FIM FILTRO DEBOUNCE?
Não
4
Sim
CONTADOR=MÍN?
Sim
3
Não
DECREMENTA CONTADOR
2
FIM FILTRO DEBOUNCE?
Não
4
Sim
CONTADOR=MÁX?
Sim
3
Não
INCREMENTA CONTADOR
CONSULTA TABELA PARA ATUALIZAR DISPLAY DE 7 SEGMENTOS
3

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *        EXPERIÊNCIA 4 - CONVERSÃO BCD PARA DISPLAYS DE 7 SEGMENTOS       *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            DESCRIÇÃO GERAL                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; CONTADOR QUE UTILIZA DOIS BOTÕES PARA INCREMENTAR E DECREMENTAR O VALOR
; CONTROLADO PELA VARIÁVEL "CONTADOR". ESTA VARIÁVEL ESTÁ LIMITADA PELAS
; CONSTANTES "MIN" E "MAX". O VALOR DO CONTADOR É MOSTRADO NO DISPLAY DA
; UNIDADE.
; OS BOTÕES ATIVOS SÃO O DA LINHA 4. O BOTÃO DA COLUNA 2 PODE SER UTILIZADO
; PARA INCREMENTAR O VALOR E O DA COLUNA 1 PARA DECREMENTAR.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        CONFIGURAÇÕES PARA GRAVAÇÃO                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_OFF & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              VARIÁVEIS                                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DOS NOMES E ENDEREÇOS DE TODAS AS VARIÁVEIS UTILIZADAS
; PELO SISTEMA

   CBLOCK  0x20                    ; ENDEREÇO INICIAL DA MEMÓRIA DE
                                   ; USUÁRIO

           W_TEMP                  ; REGISTRADORES TEMPORÁRIOS PARA
           STATUS_TEMP             ; INTERRUPÇÕES
                                   ; ESTAS VARIÁVEIS NEM SERÃO UTI-
                                   ; LIZADAS
           CONTADOR                ; ARMAZENA O VALOR DA CONTAGEM
           FLAGS                   ; ARMAZENA OS FLAGS DE CONTROLE
           FILTRO1                 ; FILTRAGEM PARA O BOTÃO 1
           FILTRO2                 ; FILTRAGEM PARA O BOTÃO 2

   ENDC                            ; FIM DO BLOCO DE MEMÓRIA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>         ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     DEFINIÇÃO DOS BANCOS DE RAM                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            FLAGS INTERNOS                               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
; DEFINIÇÃO DE TODOS OS FLAGS UTILIZADOS PELO SISTEMA

#DEFINE    ST_BT1  FLAGS,0         ; STATUS DO BOTÃO 1
#DEFINE    ST_BT2  FLAGS,1         ; STATUS DO BOTÃO 2

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            CONSTANTES                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODAS AS CONSTANTES UTILIZADAS PELO SISTEMA

MIN               EQU     .0      ; VALOR MÍNIMO PARA O CONTADOR
MAX               EQU     .15     ; VALOR MÁXIMO PARA O CONTADOR
T_FILTRO          EQU     .255    ; FILTRO PARA BOTÃO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              ENTRADAS                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS PINOS QUE SERÃO UTILIZADOS COMO ENTRADA
; RECOMENDAMOS TAMBÉM COMENTAR O SIGNIFICADO DE SEUS ESTADOS (0 E 1)

#DEFINE    BOTAO1  PORTB,0         ; PORTA DO BOTÃO
                                   ; 1 -> PRESSIONADO
                                   ; 0 -> LIBERADO

#DEFINE    BOTAO2  PORTB,1         ; PORTA DO BOTÃO
                                   ; 1 -> PRESSIONADO
                                   ; 0 -> LIBERADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              SAÍDAS                                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS PINOS QUE SERÃO UTILIZADOS COMO SAÍDA
; RECOMENDAMOS TAMBÉM COMENTAR O SIGNIFICADO DE SEUS ESTADOS (0 E 1)

#DEFINE    DSP_UNIDADE     PORTB,7 ; PINO DISPLAY DA UNIDADE
                                   ; 1 -> DISPLAY ATIVADO
                                   ; 0 -> DISPLAY DESATIVADO

#DEFINE    LINHA_4         PORTB,7 ; PINO PARA ATIVAR LINHA 4 DO TECLADO
                                   ;  MATRICIAL
                                   ; 1 -> LINHA ATIVADA
                                   ; 0 -> LINHA DESATIVADA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          VETOR DE RESET                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   ORG     0x00                    ; ENDEREÇO INICIAL DE PROCESSAMENTO
   GOTO    INICIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       INÍCIO DA INTERRUPÇÃO                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; AS INTERRUPÇÕES NÃO SERÃO UTILIZADAS, POR ISSO PODEMOS SUBSTITUIR
; TODO O SISTEMA EXISTENTE NO ARQUIVO MODELO PELO APRESENTADO ABAIXO
; ESTE SISTEMA NÃO É OBRIGATÓRIO, MAS PODE EVITAR PROBLEMAS FUTUROS

   ORG     0x04                    ; ENDEREÇO INICIAL DA INTERRUPÇÃO
   RETFIE                          ; RETORNA DA INTERRUPÇÃO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                 ROTINA DE CONVERSÃO BINÁRIO -> DISPLAY                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA IRÁ RETORNAR EM W, O SIMBOLO CORRETO QUE DEVE SER
; MOSTRADO NO DISPLAY PARA CADA VALOR DE CONTADOR. O RETORNO JÁ ESTÁ
; FORMATADO PARA AS CONDIÇÕES DE LIGAÇÃO DO DISPLAY AO PORTD.
;     a
;     **********
;     *        *
;   f *        * b
;     *    g   *
;     **********
```

```
;     *        *
;   e *        * c
;     *    d   *
;     **********  *.

CONVERTE
   MOVF    CONTADOR,W            ; COLOCA CONTADOR EM W
   ANDLW   B'00001111'           ; MASCARA VALOR DE CONTADOR
                                 ; CONSIDERAR SOMENTE ATÉ 15
   ADDWF   PCL,F

;          B'.GFEDCBA'           ; POSIÇÃO CORRETA DOS SEGMENTOS
   RETLW   B'00111111'           ; 00 - RETORNA SÍMBOLO CORRETO 0
   RETLW   B'00000110'           ; 01 - RETORNA SÍMBOLO CORRETO 1
   RETLW   B'01011011'           ; 02 - RETORNA SÍMBOLO CORRETO 2
   RETLW   B'01001111'           ; 03 - RETORNA SÍMBOLO CORRETO 3
   RETLW   B'01100110'           ; 04 - RETORNA SÍMBOLO CORRETO 4
   RETLW   B'01101101'           ; 05 - RETORNA SÍMBOLO CORRETO 5
   RETLW   B'01111101'           ; 06 - RETORNA SÍMBOLO CORRETO 6
   RETLW   B'00000111'           ; 07 - RETORNA SÍMBOLO CORRETO 7
   RETLW   B'01111111'           ; 08 - RETORNA SÍMBOLO CORRETO 8
   RETLW   B'01101111'           ; 09 - RETORNA SÍMBOLO CORRETO 9
   RETLW   B'01110111'           ; 10 - RETORNA SÍMBOLO CORRETO A
   RETLW   B'01111100'           ; 11 - RETORNA SÍMBOLO CORRETO b
   RETLW   B'00111001'           ; 12 - RETORNA SÍMBOLO CORRETO C
   RETLW   B'01011110'           ; 13 - RETORNA SÍMBOLO CORRETO d
   RETLW   B'01111001'           ; 14 - RETORNA SÍMBOLO CORRETO E
   RETLW   B'01110001'           ; 15 - RETORNA SÍMBOLO CORRETO F

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          INICIO DO PROGRAMA                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

INICIO
   CLRF    PORTA                 ; LIMPA O PORTA
   CLRF    PORTB                 ; LIMPA O PORTB
   CLRF    PORTC                 ; LIMPA O PORTC
   CLRF    PORTD                 ; LIMPA O PORTD
   CLRF    PORTE                 ; LIMPA O PORTE


   BANK1                         ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                 ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                 ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011001'
   MOVWF   TRISC                 ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                 ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                 ; CONFIGURA I/O DO PORTE

   MOVLW   B'00000111'
   MOVWF   CMCON                 ; DESLIGA COMPARADORES ANALÓGICOS

   MOVLW   B'00000111'
   MOVWF   ADCON1                ; DESLIGA CONVERSORES A/D

   MOVLW   B'10000000'
   MOVWF   OPTION_REG            ; PRESCALER 1:2 NO TMR0
                                 ; PULL-UPS DESABILITADOS
                                 ; AS DEMAIS CONFG. SÃO IRRELEVANTES
   MOVLW   B'00000000'
   MOVWF   INTCON                ; TODAS AS INTERRUPÇÕES DESLIGADAS

   BANK0                         ; RETORNA PARA O BANCO 0
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INICIALIZAÇÃO DO HARDWARE                          *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   BSF     DSP_UNIDADE             ; ATIVA DISPLAY DA UNIDADE
                                   ; ESTE PINO TAMBÉM É UTILIZADO
                                   ; PARA ATIVAR A LINHA 4 DO TECLADO
                                   ; MATRICIAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      INICIALIZAÇÃO DAS VARIÁVEIS                          *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   CLRF    FLAGS                   ; LIMPA TODOS OS FLAGS
   MOVLW   MIN
   MOVWF   CONTADOR                ; INICIA CONTADOR = MIN
   GOTO    ATUALIZA                ; ATUALIZA O DISPLAY INICIALMENTE

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          ROTINA PRINCIPAL                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

MAIN
   MOVLW   T_FILTRO
   MOVWF   FILTRO1                 ; INICIALIZA FILTRO1 = T_FILTRO
   MOVWF   FILTRO2                 ; INICIALIZA FILTRO2 = T_FILTRO

CHECA_BT1
   BTFSS   BOTAO1                  ; O BOTÃO 1 ESTÁ PRESSIONADO?
   GOTO    BT1_LIB                 ; NÃO, ENTÃO TRATA COMO LIBERADO
                                   ; SIM
   DECFSZ  FILTRO1,F               ; DECREMENTA O FILTRO DO BOTÃO
                                   ; TERMINOU?
   GOTO    CHECA_BT1               ; NÃO, CONTINUA ESPERANDO
                                   ; SIM
   BTFSS   ST_BT1                  ; BOTÃO JÁ ESTAVA PRESSIONADO?
   GOTO    DEC                     ; NÃO, EXECUTA AÇÃO DO BOTÃO
   GOTO    CHECA_BT2               ; SIM, CHECA BOTÃO 2

BT1_LIB
   BCF     ST_BT1                  ; MARCA BOTÃO 1 COMO LIBERADO

CHECA_BT2
   BTFSS   BOTAO2                  ; O BOTÃO 2 ESTÁ PRESSIONADO?
   GOTO    BT2_LIB                 ; NÃO, ENTÃO TRATA COMO LIBERADO
                                   ; SIM
   DECFSZ  FILTRO2,F               ; DECREMENTA O FILTRO DO BOTÃO
                                   ; TERMINOU?
   GOTO    CHECA_BT2               ; NÃO, CONTINUA ESPERANDO
                                   ; SIM
   BTFSS   ST_BT2                  ; BOTÃO JÁ ESTAVA PRESSIONADO?
   GOTO    INC                     ; NÃO, EXECUTA AÇÃO DO BOTÃO
   GOTO    MAIN                    ; SIM, VOLTA AO LOOPING

BT2_LIB
   BCF     ST_BT2                  ; MARCA BOTÃO 2 COMO LIBERADO
   GOTO    MAIN                    ; RETORNA AO LOOPING

DEC                                ; AÇÃO DE DECREMENTAR
   BSF     ST_BT1                  ; MARCA BOTÃO 1 COMO JÁ PRESSIONADO
   MOVF    CONTADOR,W              ; COLOCA CONTADOR EM W
   XORLW   MIN                     ; APLICA XOR ENTRE CONTADOR E MIN
                                   ; PARA TESTAR IGUALDADE. SE FOREM
                                   ; IGUAIS, O RESULTADO SERÁ ZERO
   BTFSC   STATUS,Z                ; RESULTOU EM ZERO?
   GOTO    MAIN                    ; SIM, RETORNA SEM AFETAR CONT.
                                   ; NÃO
   DECF    CONTADOR,F              ; DECREMENTA O CONTADOR
   GOTO    ATUALIZA                ; ATUALIZA O DISPLAY

INC                                ; AÇÃO DE INCREMENTAR
   BSF     ST_BT2                  ; MARCA BOTÃO 2 COMO JÁ PRESSIONADO
```

```
   MOVF    CONTADOR,W              ; COLOCA CONTADOR EM W
   XORLW   MAX                     ; APLICA XOR ENTRE CONTADOR E MAX
                                   ; PARA TESTAR IGUALDADE. SE FOREM
                                   ; IGUAIS, O RESULTADO SERÁ ZERO
   BTFSC   STATUS,Z                ; RESULTOU EM ZERO?
   GOTO    MAIN                    ; SIM, RETORNA SEM AFETAR CONT.
                                   ; NÃO
   INCF    CONTADOR,F              ; INCREMENTA O CONTADOR
   GOTO    ATUALIZA                ; ATUALIZA O DISPLAY

ATUALIZA
   CALL    CONVERTE                ; CONVERTE CONTADOR NO NÚMERO DO
                                   ; DISPLAY
   MOVWF   PORTD                   ; ATUALIZA O PORTD PARA
                                   ; VISUALIZARMOS O VALOR DE CONTADOR
                                   ; NO DISPLAY
   GOTO    MAIN                    ; NÃO, VOLTA AO LOOP PRINCIPAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          FIM DO PROGRAMA                                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                             ; OBRIGATÓRIO
```

Dicas e Comentários

A tabela implementada neste exemplo apresenta 16 linhas, assim, é possível representar qualquer valor hexadecimal entre 0h e Fh. No entanto, utilizando o mesmo conceito poderíamos criar representações para praticamente todas as letras do alfabeto. E para escrever mensagens no display bastaria associar a cada letra um número binário.

O conceito da tabela não se limita apenas à utilização com displays de 7 segmentos, infinitas situações podem ser resolvidas com o uso de uma tabela. Por exemplo, pode-se criar uma tabela para converter números binários em caracteres ASCII e vice-versa.

Exercícios Propostos

1. Mantendo as 16 linhas atuais, alterar a tabela para mostrar as letras: “A”, “b”, “C”, “d”, “E”, “F”, “G”, “H”, “I”, “J”, “L”, “n”, “O”, “P”, “r” e “S”.
2. Alterar o software a fim de utilizar uma tabela que contenha todos os números e todas as possíveis letras.

Capítulo 7 - Experiência 5 – Timer de segundos

Objetivo

O uso de rotinas de delays para contagem de tempo nem sempre pode ser aplicado, uma vez que este tipo de rotina deixa o processador parado. Um recurso muito mais adequado para a contagem de tempos é a utilização do timer do microcontrolador. Este é o objetivo desta experiência. Ensinar ao aluno como configurar e utilizar o timer e a interrupção.

Descrição

Esta experiência cria um timer decrescente em segundos. O valor inicial é determinado pela constante V_INICIO e pode estar entre 1 e 15 segundos.

Os botões ativos são os da linha 4. O botão da coluna 1 dispara o timer, mostrando o tempo restante no display. O da coluna 2 paralisa o timer. O led ligado ao pino RC0 é utilizado para indicar o estado atual do timer sendo acesso se o timer estiver rodando e apagado se o timer estiver parado.

O timer utiliza como base de tempo a interrupção de TMR0 sendo que esta ocorre quando o timer estoura, ou seja, quando o valor do TMR0 pula de 0xFF para 0x00. Assim, o intervalo de tempo entre interrupções irá depender do prescaler configurado para o timer, do valor inicial com que ele é carregado e do tempo de execução de um ciclo de máquina.

No exemplo desta experiência, sempre que se entra na interrupção de TMR0, o contador do timer é carregado com 131, de forma que sempre se faça 125 contagens, pois, o timer irá contar de 131 até 256 e quando voltar a estourar será novamente carregado com 131.

Desta forma, podemos calcular o tempo entre interrupções seguindo a equação abaixo:

Tempo TMR0 = (256 – valor com que é carregado) * prescaler * ciclo de máquina

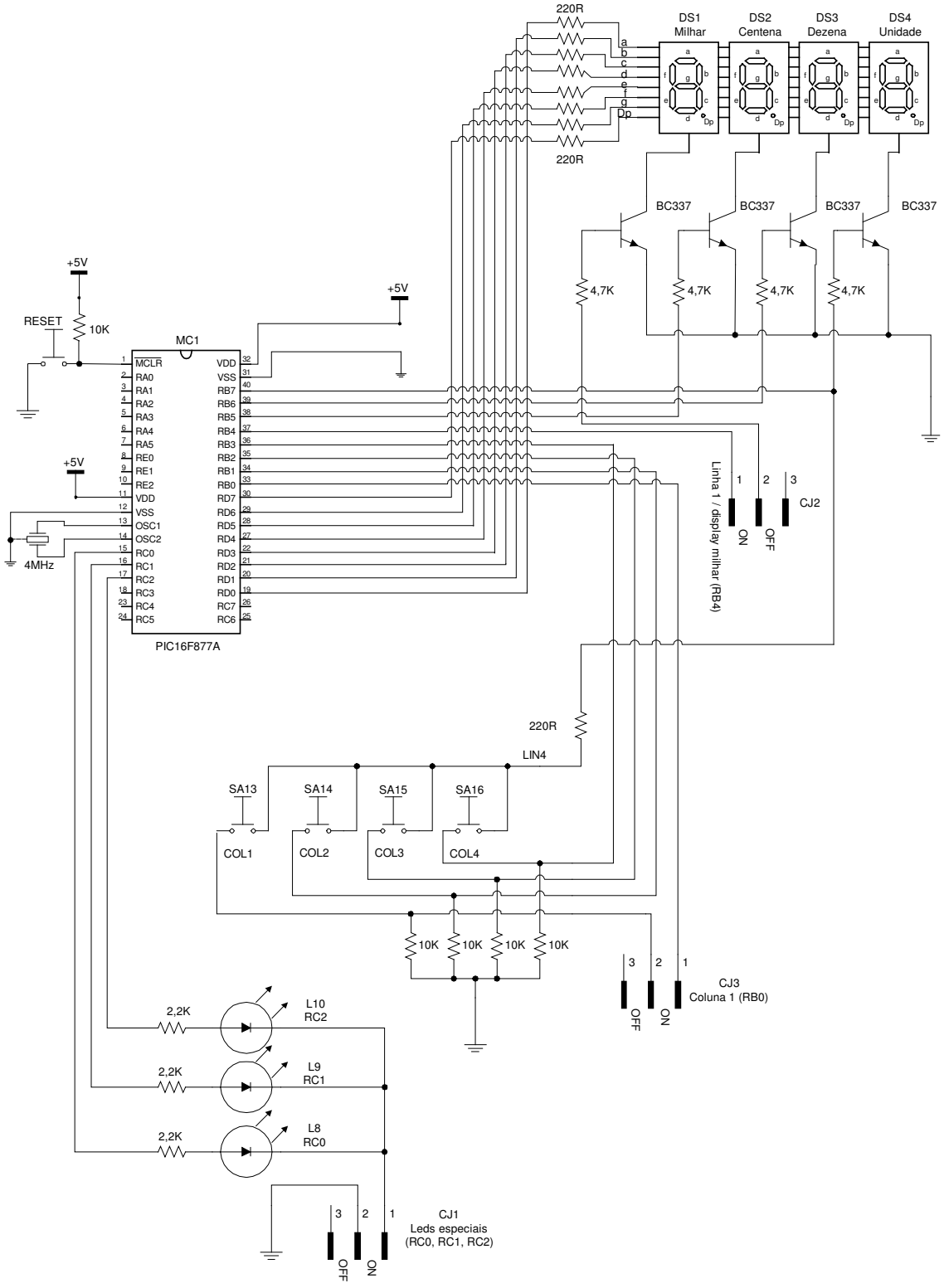
Tempo TMR0 = (256 – 131) * 64 * 1us

Tempo TMR0 = 8ms

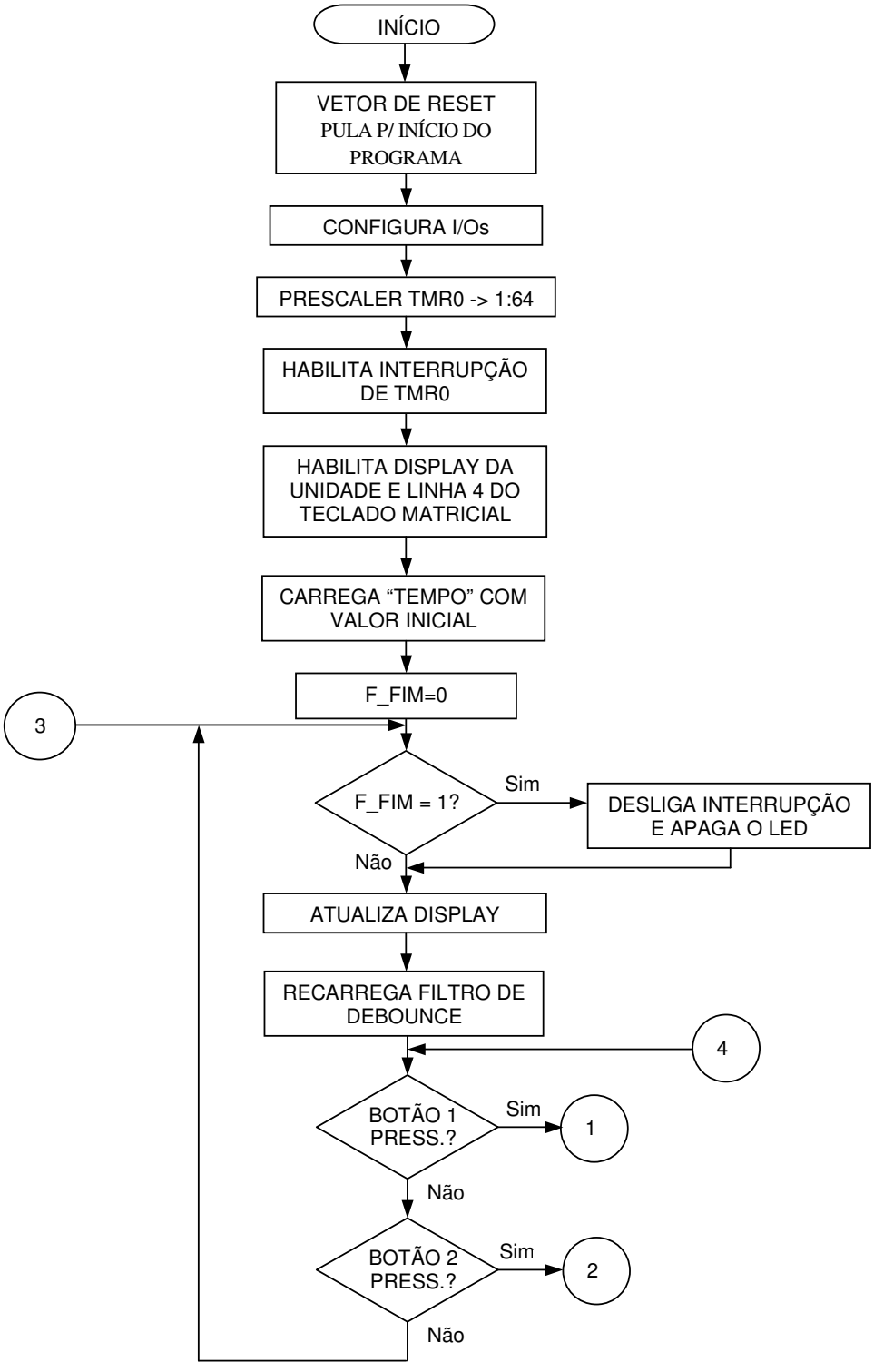
Portanto, a interrupção irá ocorrer a cada 8ms.

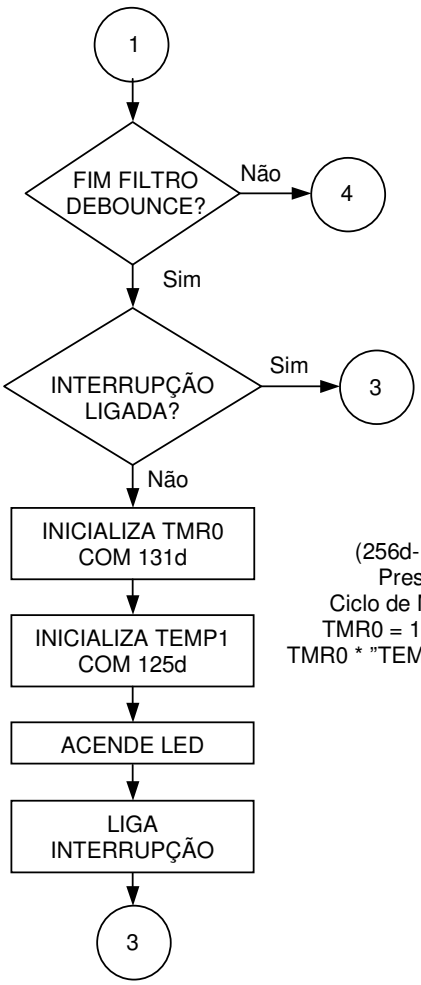
A fim de criar a base de tempo de 1 segundo, foi acrescentado um contador auxiliar que conta o número de ocorrências da interrupção de TMR0. Com este contador auxiliar pode-se estender a base de tempo sempre em múltiplos de 8ms. Caso este contador auxiliar seja configurado para contar 125 interrupções de TMR0, pode-se obter a base de tempo de 1 segundo, fundamental para a criação do timer da experiência.

Esquema Elétrico

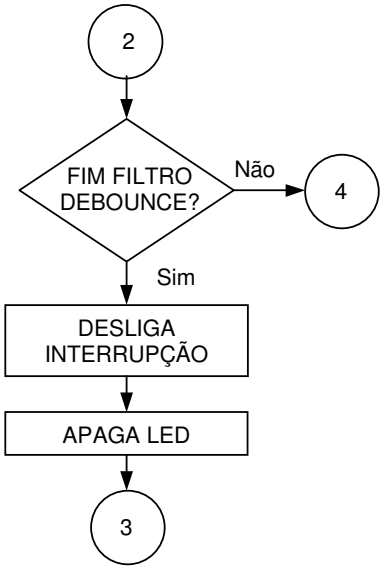


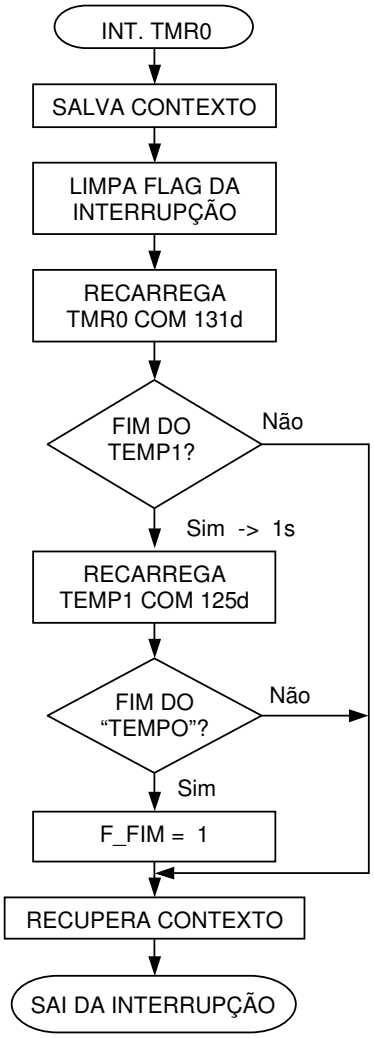
Fluxograma


INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURA I/Os
PRESCALER TMR0 -> 1:64
HABILITA INTERRUPÇÃO
DE TMR0
HABILITA DISPLAY DA
UNIDADE E LINHA 4 DO
TECLADO MATRICIAL
CARREGA "TEMPO" COM
VALOR INICIAL
F_FIM=0
3
F_FIM = 1?
Sim
DESLIGA INTERRUPÇÃO
E APAGA O LED
Não
ATUALIZA DISPLAY
RECARREGA FILTRO DE
DEBOUNCE
4
BOTÃO 1
PRESS.?
Sim
1
Não
BOTÃO 2
PRESS.?
Sim
2
Não



(256d-125d)=131d
Prescale 1:64
Ciclo de Máquina = 1us
TMR0 = 125*64us = 8ms
TMR0 * "TEMP1" = 8ms * 125 =




INT. TMR0
SALVA CONTEXTO
LIMPA FLAG DA INTERRUPÇÃO
RECARREGA TMR0 COM 131d
FIM DO TEMP1?
Não
Sim -> 1s
RECARREGA TEMP1 COM 125d
FIM DO "TEMPO"?
Não
Sim
F_FIM = 1
RECUPERA CONTEXTO
SAI DA INTERRUPÇÃO

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *                     EXPERIÊNCIA 5 - TIMER DE SEGUNDOS                   *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                             DESCRIÇÃO GERAL                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; TIMER DECRESCENTE EM SEGUNDOS. O VALOR INICIAL É DETERMINADO PELA CONSTANTE
; V_INICIO E PODE ESTAR ENTRE 1 E 15 SEGUNDOS.
; OS BOTÕES ATIVOS SÃO O DA LINHA 4. O BOTÃO DA COLUNA 1 DISPARA O TIMER,
; MOSTRANDO O TEMPO RESTANTE NO DISPLAY. O DA COLUANA 2 PARALIZA O TIMER.
; O LED (RC0) É UTILIZADO PARA INDICAR O ESTADO ATUAL DO TIMER:
; ACESO=RODANDO E APAGADO=PARADO
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                         CONFIGURAÇÕES PARA GRAVAÇÃO                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_OFF & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               VARIÁVEIS                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DOS NOMES E ENDEREÇOS DE TODAS AS VARIÁVEIS UTILIZADAS
; PELO SISTEMA

   CBLOCK  0x20                    ; ENDEREÇO INICIAL DA MEMÓRIA DE
                                   ; USUÁRIO

           W_TEMP                  ; REGISTRADORES TEMPORÁRIOS PARA
           STATUS_TEMP             ; INTERRUPÇÕES
                                   ; ESTAS VARIÁVEIS NEM SERÃO UTI-
                                   ; LIZADAS

           TEMPO                   ; ARMAZENA O VALOR DO TEMPO
           FLAGS                   ; ARMAZENA OS FLAGS DE CONTROLE
           TEMP1                   ; REGISTRADORES AUXILIARES
           TEMP2
           FILTRO1                 ; FILTROS DOS BOTÕES
           FILTRO2

   ENDC                            ; FIM DO BLOCO DE MEMÓRIA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>         ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      DEFINIÇÃO DOS BANCOS DE RAM                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            FLAGS INTERNOS                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS FLAGS UTILIZADOS PELO SISTEMA

#DEFINE    F_FIM   FLAGS,0         ; FLAG DE FIM DE TEMPO
#DEFINE    ST_BT1  FLAGS,1         ; STATUS DO BOTÃO 1
#DEFINE    ST_BT2  FLAGS,2         ; STATUS DO BOTÃO 2

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                             CONSTANTES                                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODAS AS CONSTANTES UTILIZADAS PELO SISTEMA

V_INICIO   EQU     .15             ; VALOR INICIAL DO TIMER (1 A 15 SEG.)
T_FILTRO   EQU     .255            ; VALOR DO FILTRO DOS BOTÕES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              ENTRADAS                                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS PINOS QUE SERÃO UTILIZADOS COMO ENTRADA
; RECOMENDAMOS TAMBÉM COMENTAR O SIGNIFICADO DE SEUS ESTADOS (0 E 1)

#DEFINE    BOTAO1  PORTB,0         ; PORTA DO BOTÃO
                                   ; 1 -> PRESSIONADO
                                   ; 0 -> LIBERADO

#DEFINE    BOTAO2  PORTB,1         ; PORTA DO BOTÃO
                                   ; 1 -> PRESSIONADO
                                   ; 0 -> LIBERADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               SAÍDAS                                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS PINOS QUE SERÃO UTILIZADOS COMO SAÍDA
; RECOMENDAMOS TAMBÉM COMENTAR O SIGNIFICADO DE SEUS ESTADOS (0 E 1)

#DEFINE    DSP_UNIDADE     PORTB,7 ; PINO DISPLAY DA UNIDADE
                                   ; 1 -> DISPLAY ATIVADO
                                   ; 0 -> DISPLAY DESATIVADO

#DEFINE    LINHA_4         PORTB,7 ; PINO PARA ATIVAR LINHA 4 DO TECLADO
                                   ;  MATRICIAL
                                   ; 1 -> LINHA ATIVADA
                                   ; 0 -> LINHA DESATIVADA

#DEFINE    LED             PORTC,0 ; LED
                                   ; 0 -> DESLIGADO
                                   ; 1 -> LIGADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           VETOR DE RESET                                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   ORG     0x00                    ; ENDEREÇO INICIAL DE PROCESSAMENTO
   GOTO    INICIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INÍCIO DA INTERRUPÇÃO                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; UTILIZAREMOS SOMENTE A INTERRUÇÃO DE TMR0, MAS EFETUAREMOS O TESTE
; PARA TERMOS CERTEZA DE QUE NENHUM PROBLEMA ACONTECEU. É NECESSÁRIO
; SALVAR E RECUPERAR OS VALOR DE W E STATUS.

   ORG     0x04                    ; ENDEREÇO INICIAL DA INTERRUPÇÃO
   MOVWF   W_TEMP                  ; SALVA W EM W_TEMP
   SWAPF   STATUS,W
   MOVWF   STATUS_TEMP             ; SALVA STATUS EM STATUS_TEMP

   BTFSS   INTCON,T0IF             ; É INTERRUPÇÃO DE TMR0?
   GOTO    SAI_INT                 ; NÃO, SAI SE AÇÃO
                                   ; SIM
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    TRATAMENTO DA INTERRUPÇÃO DE TMR0                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA IRÁ CONTAR O TEMPO, E QUANDO PASSAR 1 SEGUNDO, A VARIÁVEL
; "TEMPO" SERÁ DECREMENTADA.
; 1 SEGUNDO = 64us (PRESCALER) X 125 (TMR0) X 125 (TEMP1)

   BCF     INTCON,T0IF             ; LIMPA FLAG DA INT.
   MOVLW   .256-.125
   MOVWF   TMR0                    ; REINICIA TMR0
   DECFSZ  TEMP1,F                 ; DECREMENTA CONTADOR AUXILIAR. ACABOU?
   GOTO    SAI_INT                 ; NÃO, SAI SEM AÇÃO
                                   ; SIM
   MOVLW   .125
   MOVWF   TEMP1                   ; REINICIALIZA TEMPO AUXILIAR
   BTFSC   F_FIM                   ; JÁ CHEGOU AO FIM?
   GOTO    SAI_INT                 ; SIM, ENTÃO NÃO DECREMENTA O TEMPO
                                   ; NÃO
   DECFSZ  TEMPO,F                 ; DECREMENTA TEMPO. ACABOU?
   GOTO    SAI_INT                 ; NÃO, SAI DA INTERRUÇãO
                                   ; SIM
   BSF     F_FIM                   ; MARCA FIM DO TEMPO
   GOTO    SAI_INT                 ; SAI DA INTERRUPÇÃO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           FIM DA INTERRUPÇÃO                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

SAI_INT
   SWAPF   STATUS_TEMP,W
   MOVWF   STATUS                  ; RECUPERA STATUS
   SWAPF   W_TEMP,F
   SWAPF   W_TEMP,W                ; RECUPERA W
   RETFIE                          ; RETORNA DA INTERRUPÇÃO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  ROTINA DE CONVERSÃO BINÁRIO -> DISPLAY                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA IRÁ RETORNAR EM W, O SIMBOLO CORRETO QUE DEVE SER
; MOSTRADO NO DISPLAY PARA CADA VALOR DE CONTADOR. O RETORNO JÁ ESTÁ
; FORMATADO PARA AS CONDIÇÕES DE LIGAÇÃO DO DISPLAY AO PORTD.
;     a
;     **********
;     *        *
;   f *        * b
;     *    g   *
;     **********
;     *        *
;   e *        * c
;     *    d   *
;     **********  *.

CONVERTE
   MOVF    TEMPO,W                 ; COLOCA TEMPO EM W
   ANDLW   B'00001111'             ; MASCARA VALOR DE TEMPO
                                   ; CONSIDERAR SOMENTE ATÉ 15
   ADDWF   PCL,F

;          B'.GFEDCBA'             ; POSIÇÃO CORRETA DOS SEGMENTOS
   RETLW   B'00111111'             ; 00 - RETORNA SÍMBOLO CORRETO 0
   RETLW   B'00000110'             ; 01 - RETORNA SÍMBOLO CORRETO 1
   RETLW   B'01011011'             ; 02 - RETORNA SÍMBOLO CORRETO 2
   RETLW   B'01001111'             ; 03 - RETORNA SÍMBOLO CORRETO 3
   RETLW   B'01100110'             ; 04 - RETORNA SÍMBOLO CORRETO 4
   RETLW   B'01101101'             ; 05 - RETORNA SÍMBOLO CORRETO 5
   RETLW   B'01111101'             ; 06 - RETORNA SÍMBOLO CORRETO 6
   RETLW   B'00000111'             ; 07 - RETORNA SÍMBOLO CORRETO 7
   RETLW   B'01111111'             ; 08 - RETORNA SÍMBOLO CORRETO 8
   RETLW   B'01101111'             ; 09 - RETORNA SÍMBOLO CORRETO 9
   RETLW   B'01110111'             ; 10 - RETORNA SÍMBOLO CORRETO A
   RETLW   B'01111100'             ; 11 - RETORNA SÍMBOLO CORRETO b
```

```
   RETLW   B'00111001'            ; 12 - RETORNA SÍMBOLO CORRETO C
   RETLW   B'01011110'            ; 13 - RETORNA SÍMBOLO CORRETO d
   RETLW   B'01111001'            ; 14 - RETORNA SÍMBOLO CORRETO E
   RETLW   B'01110001'            ; 15 - RETORNA SÍMBOLO CORRETO F

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      ROTINA DE ATUALIZAÇÃO DO DISPLAY                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA CONVERTE O VALOR DE TEMPO ATRAVÉS DA ROTINA CONVERTE
; E ATUALIZA O PORTB PARA ACENDER O DISPLAY CORRETAMENTE

ATUALIZA
   CALL    CONVERTE               ; CONVERTE CONTADOR NO NÚMERO DO
                                  ; DISPLAY
   MOVWF   PORTD                  ; ATUALIZA O PORTD PARA
                                  ; VISUALIZARMOS O VALOR DE CONTADOR
                                  ; NO DISPLAY
   RETURN                         ; NÃO, RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      ROTINA DE DESLIGAR O TIMER                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA EXECUTA AS AÇÕES NECESSÁRIAS PARA DESLIGAR O TIMER

DESL_TIMER
   BCF     INTCON,GIE             ; DESLIGA CHAVE GERAL DE INT.
   BCF     LED                    ; APAGA O LED
   RETURN                         ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      ROTINA DE LIGAR O TIMER                            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA EXECUTA AS AÇÕES NECESSÁRIAS PARA LIGAR O TIMER

LIGA_TIMER
   BTFSC   INTCON,GIE             ; TIMER JÁ ESTA LIGADO?
   RETURN                         ; SIM, RETORNA DIRETO
                                  ; NÃO
   BCF     INTCON,T0IF            ; LIMPA FLAG DE INT. DE TMR0
   MOVLW   .256-.125
   MOVWF   TMR0                   ; INICIA TMR0 CORRETAMENTE
   MOVLW   .125
   MOVWF   TEMP1                  ; INICIA TEMP1 CORRETAMENTE
   BSF     INTCON,GIE             ; LIGA CHAVE GERAL DE INTERRUPÇÕES
   BSF     LED                    ; ACENDE O LED
   RETURN                         ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          INICIO DO PROGRAMA                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

INICIO
   CLRF    PORTA                  ; LIMPA O PORTA
   CLRF    PORTB                  ; LIMPA O PORTB
   CLRF    PORTC                  ; LIMPA O PORTC
   CLRF    PORTD                  ; LIMPA O PORTD
   CLRF    PORTE                  ; LIMPA O PORTE


   BANK1                          ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                  ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                  ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011000'
   MOVWF   TRISC                  ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                  ; CONFIGURA I/O DO PORTD
```

```
   MOVLW   B'00000000'
   MOVWF   TRISE                   ; CONFIGURA I/O DO PORTE

   MOVLW   B'00000111'
   MOVWF   CMCON                   ; DESLIGA COMPARADORES ANALÓGICOS

   MOVLW   B'00000111'
   MOVWF   ADCON1                  ; DESLIGA CONVERSORES A/D

   MOVLW   B'10000101'
   MOVWF   OPTION_REG              ; PRESCALER 1:64 NO TMR0
                                   ; PULL-UPS DESABILITADOS
                                   ; AS DEMAIS CONFG. SÃO IRRELEVANTES
   MOVLW   B'00100000'
   MOVWF   INTCON                  ; HABILITA INTERRUPÇÃO DE TMR0

   BANK0                           ; RETORNA PARA O BANCO 0

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INICIALIZAÇÃO DO HARDWARE                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   BSF     DSP_UNIDADE             ; ATIVA DISPLAY DA UNIDADE

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       INICIALIZAÇÃO DAS VARIÁVEIS                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   CLRF    FLAGS                   ; LIMPA TODOS OS FLAGS
   MOVLW   V_INICIO
   MOVWF   TEMPO                   ; INICIA TEMPO = V_INICIO
   CALL    ATUALIZA                ; ATUALIZA O DISPLAY INICIALMENTE

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          ROTINA PRINCIPAL                               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

MAIN
   BTFSC   F_FIM                   ; CHEGOU AO FIM?
   CALL    DESL_TIMER              ; SIM, ENTÃO DESLIGA O TIMER
                                   ; NÃO
   CALL    ATUALIZA                ; ATUALIZA O DISPLAY
   MOVLW   T_FILTRO
   MOVWF   FILTRO1                 ; INICIALIZA FILTRO1 = T_FILTRO
   MOVWF   FILTRO2                 ; INICIALIZA FILTRO2 = T_FILTRO

CHECA_BT1
   BTFSS   BOTAO1                  ; O BOTÃO 1 ESTÁ PRESSIONADO?
   GOTO    BT1_LIB                 ; NÃO, ENTÃO TRATA COMO LIBERADO
                                   ; SIM
   DECFSZ  FILTRO1,F               ; DECREMENTA O FILTRO DO BOTÃO
                                   ; TERMINOU?
   GOTO    CHECA_BT1               ; NÃO, CONTINUA ESPERANDO
                                   ; SIM
   BTFSS   ST_BT1                  ; BOTÃO JÁ ESTAVA PRESSIONADO?
   GOTO    ACAO_BT1                ; NÃO, EXECUTA AÇÃO DO BOTÃO
   GOTO    CHECA_BT2               ; SIM, CHECA BOTÃO 2

BT1_LIB
   BCF     ST_BT1                  ; MARCA BOTÃO 1 COMO LIBERADO

CHECA_BT2
   BTFSS   BOTAO2                  ; O BOTÃO 2 ESTÁ PRESSIONADO?
   GOTO    BT2_LIB                 ; NÃO, ENTÃO TRATA COMO LIBERADO
                                   ; SIM
   DECFSZ  FILTRO2,F               ; DECREMENTA O FILTRO DO BOTÃO
                                   ; TERMINOU?
   GOTO    CHECA_BT2               ; NÃO, CONTINUA ESPERANDO
                                   ; SIM
   BTFSS   ST_BT2                  ; BOTÃO JÁ ESTAVA PRESSIONADO?
   GOTO    ACAO_BT2                ; NÃO, EXECUTA AÇÃO DO BOTÃO
   GOTO    MAIN                    ; SIM, VOLTA AO LOOPING
```

```
BT2_LIB
   BCF     ST_BT2                  ; MARCA BOTÃO 2 COMO LIBERADO
   GOTO    MAIN                    ; RETORNA AO LOOPING

ACAO_BT1                           ; AÇÃO PARA O BOTÃO 1
   BSF     ST_BT1                  ; MARCA BOTÃO 1 COMO JÁ PRESSIONADO
   CALL    LIGA_TIMER              ; LIGA O TIMER
   GOTO    MAIN

ACAO_BT2                           ; AÇÃO PARA O BOTÃO 2
   BSF     ST_BT2                  ; MARCA BOTÃO 2 COMO JÁ PRESSIONADO
   CALL    DESL_TIMER              ; DESLIGA O TIMER
   GOTO    MAIN                    ; NÃO, VOLTA AO LOOP PRINCIPAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          FIM DO PROGRAMA                                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                             ; OBRIGATÓRIO
```

Dicas e Comentários

No exemplo desta experiência, sempre que ocorre a interrupção, o valor do TMR0 é carregado com 131 de forma a obter uma contagem de 125 ciclos. Porém, o contador do TMR0 não ficou parado entre o instante em que a interrupção ocorreu e o instante em que ele foi novamente carregado. Por este motivo, pode ocorrer, que no instante em que o contador é novamente carregado com 131 ele já esteja com outro valor diferente de zero e aí a contagem não seria de exatamente 125 ciclos.

Não é o caso deste exemplo, mais se o prescaler fosse de 1:1, ou seja, se o contador fosse incrementado a cada ciclo de máquina, quando a interrupção ocorresse e o contador fosse carregado, com certeza, o seu valor já seria diferente de zero. No caso do exemplo, o contador valeria 10, pois foram gastos 2 ciclos de máquina para desviar o programa para o vetor de interrupção, mais 3 ciclos para salvar o contexto, mais 2 ciclos para testar se a interrupção era de TMR0, mais 1 ciclo para limpar o bit da interrupção e mais 2 ciclos para efetivamente fazer a carga do contador. No total foram gastos 10 ciclos entre o instante em que a interrupção ocorreu e o instante em que o contador foi carregado com 131. Portanto, o tempo entre interrupções que deveria ser de 125 ciclos passou para 135 ciclos. Isto provoca um erro na contagem de tempo.

Para corrigir este problema, deve-se somar o valor ao timer e não carrega-lo com um valor e desprezar o que ele já contou. Ainda admitindo que o prescaler fosse de 1:1, se fosse somado ao timer o valor 131, após a soma, o resultado final seria 141, sendo que 10 ele próprio já teria contado e 131 que foram somados. Desse instante até a próxima interrupção ocorreriam 115 contagens (256 – 141 = 115) e o tempo total entre interrupções se manteria conforme o desejado, ou seja, a cada 125 contagens.

Exercícios Propostos

1. Alterar o prescaler para 1:16 e obter a mesma base de tempo de 1 segundo.
2. Alterar o software para obter a mesma base de tempo de 1 segundo, porém sem utilizar o prescaler, ou seja, com prescaler configurado em 1:1.
3. Alterar a base de tempo do timer de 1 segundo para 60 segundos, ou seja, alterar o timer de segundos original para um timer de minutos.

Capítulo 8 - Experiência 6 – Aceso à memória de dados EEPROM

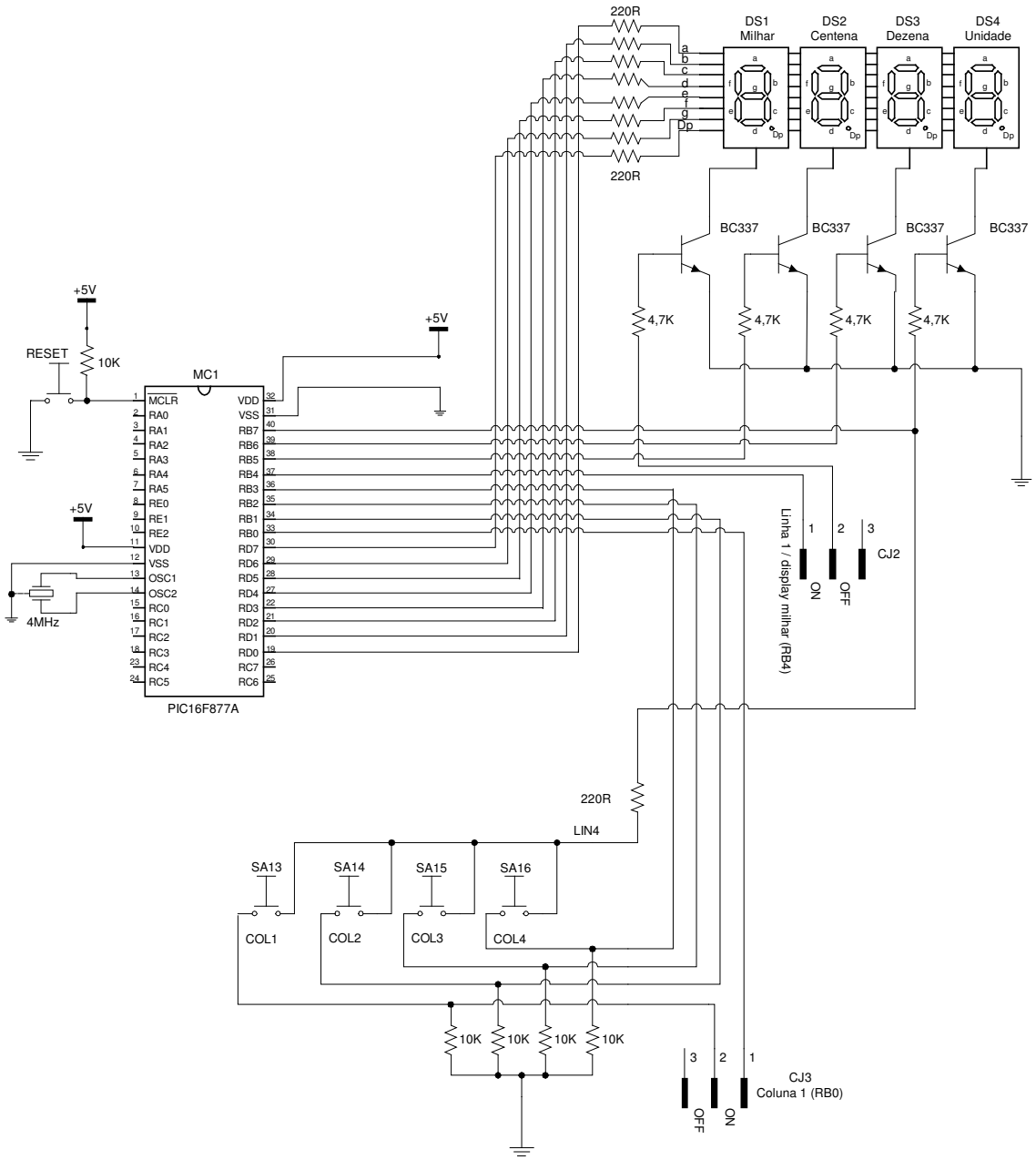
Objetivo

O objetivo desta experiência é o aprendizado da utilização da memória de dados não volátil EEPROM interna do microcontrolador.

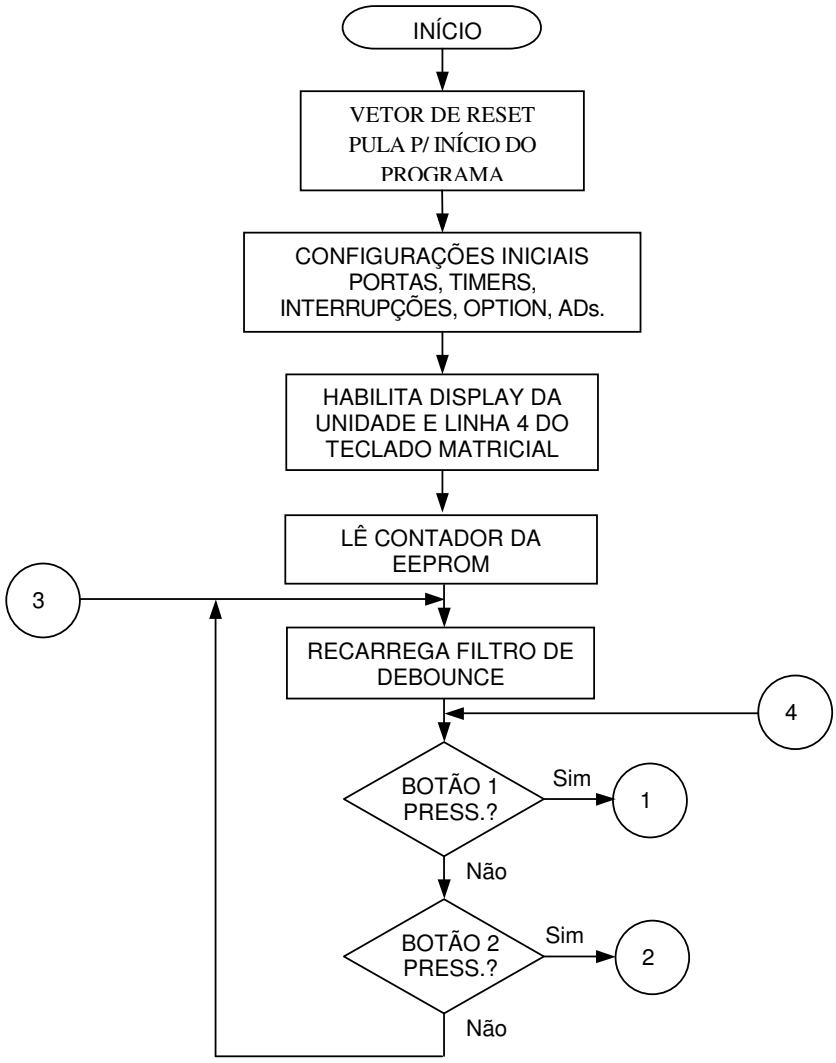
Descrição

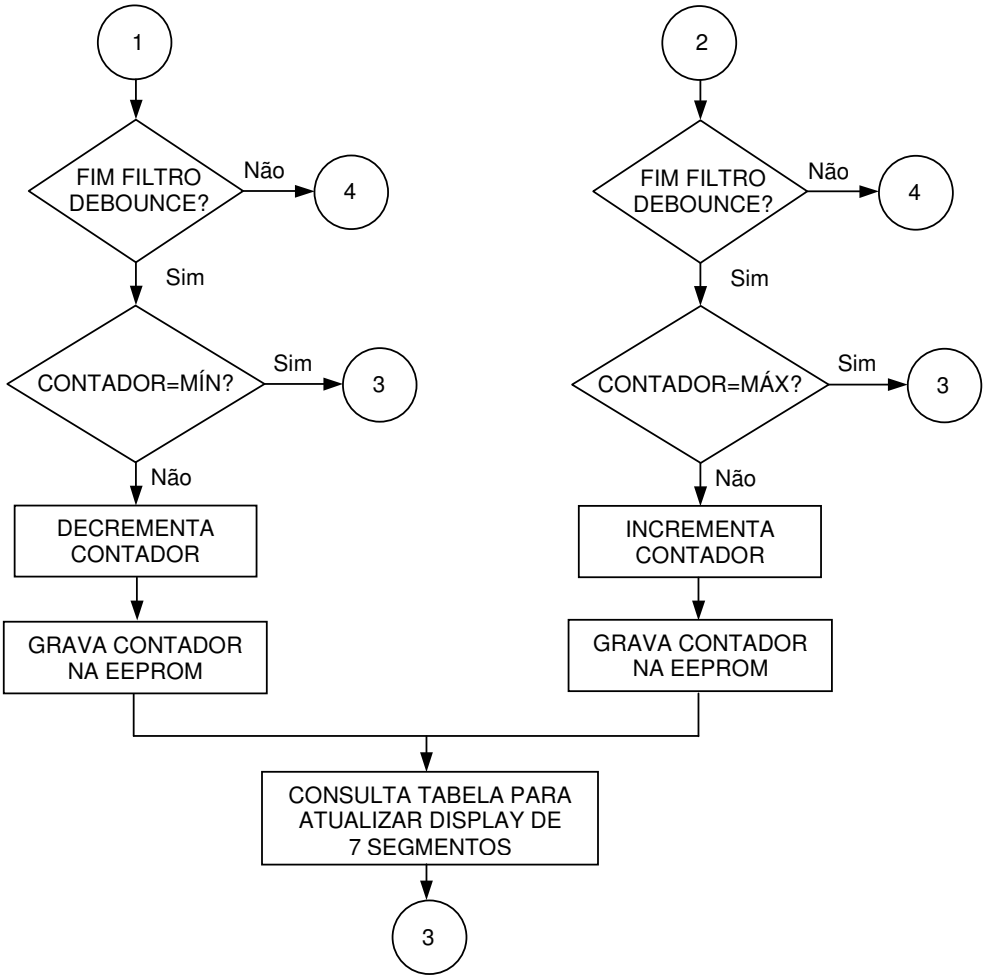
Nesta experiência foram implementadas rotinas para escrita e leitura de dados na memória EEPROM. O exemplo estudado na experiência 4 foi modificado e sempre que o valor da variável “CONTADOR” é alterado o seu valor é salvo na memória EEPROM. Ao inicializar o microcontrolador, a memória é lida e então o valor original pode ser restaurado. Assim, mesmo na queda de energia, o valor do contador é preservado.

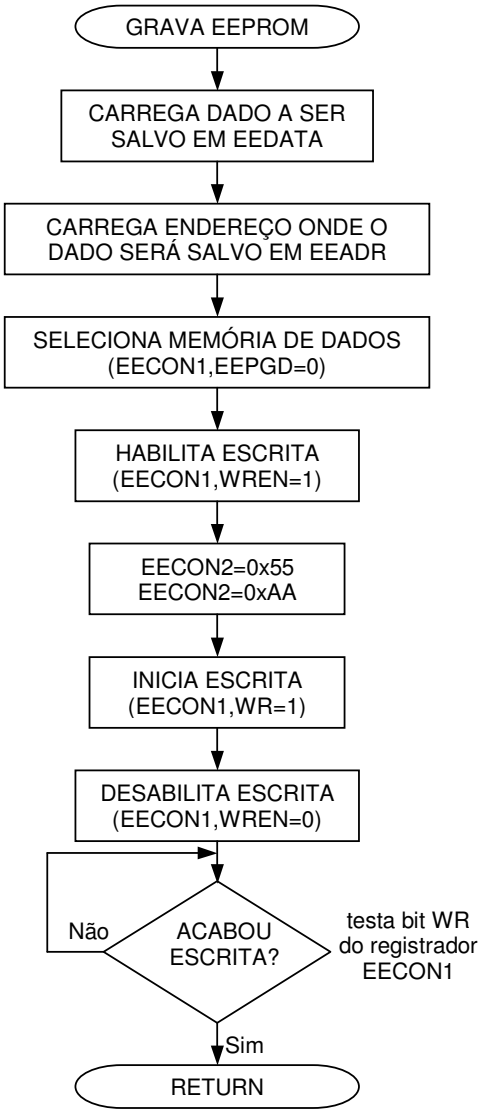
Esquema Elétrico

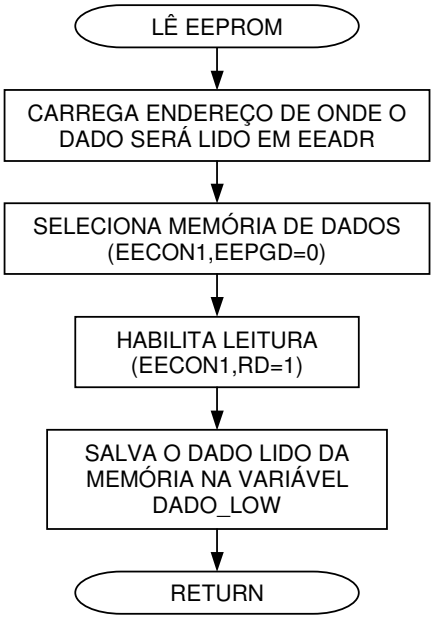


Fluxograma


INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURAÇÕES INICIAIS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION, ADs.
HABILITA DISPLAY DA
UNIDADE E LINHA 4 DO
TECLADO MATRICIAL
LÊ CONTADOR DA
EEPROM
3
RECARREGA FILTRO DE
DEBOUNCE
4
BOTÃO 1
PRESS.?
Sim
1
Não
BOTÃO 2
PRESS.?
Sim
2
Não


1
FIM FILTRO DEBOUNCE?
Não
4
Sim
CONTADOR=MÍN?
Sim
3
Não
DECREMENTA CONTADOR
GRAVA CONTADOR NA EEPROM
2
FIM FILTRO DEBOUNCE?
Não
4
Sim
CONTADOR=MÁX?
Sim
3
Não
INCREMENTA CONTADOR
GRAVA CONTADOR NA EEPROM
CONSULTA TABELA PARA ATUALIZAR DISPLAY DE 7 SEGMENTOS
3


GRAVA EEPROM
CARREGA DADO A SER SALVO EM EEDATA
CARREGA ENDEREÇO ONDE O DADO SERÁ SALVO EM EEADR
SELECIONA MEMÓRIA DE DADOS (EECON1,EEPGD=0)
HABILITA ESCRITA (EECON1,WREN=1)
EECON2=0x55
EECON2=0xAA
INICIA ESCRITA (EECON1,WR=1)
DESABILITA ESCRITA (EECON1,WREN=0)
Não
ACABOU ESCRITA?
testa bit WR do registrador EECON1
Sim
RETURN


LÊ EEPROM
CARREGA ENDEREÇO DE ONDE O DADO SERÁ LIDO EM EEADR
SELECIONA MEMÓRIA DE DADOS (EECON1,EEPGD=0)
HABILITA LEITURA (EECON1,RD=1)
SALVA O DADO LIDO DA MEMÓRIA NA VARIÁVEL DADO_LOW
RETURN

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *               EXPERIÊNCIA 6 - ACESSO À MEMÓRIA DE DADOS                 *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            DESCRIÇÃO GERAL                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; CONTADOR QUE UTILIZA DOIS BOTÕES PARA INCREMENTAR E DECREMENTAR O VALOR
; CONTROLADO PELA VARIÁVEL "CONTADOR". ESTA VARIÁVEL ESTÁ LIMITADA PELAS
; CONSTANTES "MIN" E "MAX". O VALOR DO CONTADOR É MOSTRADO NO DISPLAY DA
; UNIDADE E ARMAZENADO NA EEPROM PARA NÃO SER PERDIDO MESMO NO CASO DE RESET.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       CONFIGURAÇÕES PARA GRAVAÇÃO                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_OFF & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                             VARIÁVEIS                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DOS NOMES E ENDEREÇOS DE TODAS AS VARIÁVEIS UTILIZADAS
; PELO SISTEMA

   CBLOCK  0x20                    ; ENDEREÇO INICIAL DA MEMÓRIA DE
                                   ; USUÁRIO

           W_TEMP                  ; REGISTRADORES TEMPORÁRIOS PARA
           STATUS_TEMP             ; INTERRUPÇÕES
                                   ; ESTAS VARIÁVEIS NEM SERÃO UTI-
                                   ; LIZADAS
           CONTADOR                ; ARMAZENA O VALOR DA CONTAGEM
           FLAGS                   ; ARMAZENA OS FLAGS DE CONTROLE
           FILTRO1                 ; FILTRAGEM PARA O BOTÃO 1
           FILTRO2                 ; FILTRAGEM PARA O BOTÃO 2

           ENDERECO                ; ARMAZENA O ENDERECO PARA ACESSO À
                                   ; MEMÓRIA DE DADOS (EEPROM)

   ENDC                            ; FIM DO BLOCO DE MEMÓRIA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>         ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      DEFINIÇÃO DOS BANCOS DE RAM                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                             FLAGS INTERNOS                              *
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS FLAGS UTILIZADOS PELO SISTEMA

#DEFINE    ST_BT1  FLAGS,0                ; STATUS DO BOTÃO 1
#DEFINE    ST_BT2  FLAGS,1                ; STATUS DO BOTÃO 2

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                CONSTANTES                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODAS AS VARIÁVEIS UTILIZADAS PELO SISTEMA

MIN                EQU     .0      ; VALOR MINIMO PARA O CONTADOR
MAX                EQU     .15     ; VALOR MÁXIMO PARA O CONTADOR
T_FILTRO           EQU     .255    ; FILTRO PARA BOTÃO
POS_MEM            EQU     .0      ; ENDEREÇO DA EEPROM ONDE SERÁ
                                   ; ARMAZENADO O VALOR DO CONTADOR

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                ENTRADAS                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS PINOS QUE SERÃO UTILIZADOS COMO ENTRADA
; RECOMENDAMOS TAMBÉM COMENTAR O SIGNIFICADO DE SEUS ESTADOS (0 E 1)

#DEFINE    BOTAO1  PORTB,0         ; PORTA DO BOTÃO
                                   ; 1 -> PRESSIONADO
                                   ; 0 -> LIBERADO

#DEFINE    BOTAO2  PORTB,1         ; PORTA DO BOTÃO
                                   ; 1 -> PRESSIONADO
                                   ; 0 -> LIBERADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                SAÍDAS                                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS PINOS QUE SERÃO UTILIZADOS COMO SAÍDA
; RECOMENDAMOS TAMBÉM COMENTAR O SIGNIFICADO DE SEUS ESTADOS (0 E 1)

#DEFINE    DSP_UNIDADE     PORTB,7 ; PINO DISPLAY DA UNIDADE
                                   ; 1 -> DISPLAY ATIVADO
                                   ; 0 -> DISPLAY DESATIVADO

#DEFINE    LINHA_4         PORTB,7 ; PINO PARA ATIVAR LINHA 4 DO TECLADO
                                   ;  MATRICIAL
                                   ; 1 -> LINHA ATIVADA
                                   ; 0 -> LINHA DESATIVADA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      INICIALIZAÇÃO DA EEPROM                               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   ORG     H'2100'+POS_MEM         ; INÍCIO DA EEPROM
   DE      .5                      ; VALOR INICIAL PARA CONTADOR = 5

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          VETOR DE RESET                                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   ORG     0x00                    ; ENDEREÇO INICIAL DE PROCESSAMENTO
   GOTO    INICIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INÍCIO DA INTERRUPÇÃO                               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; AS INTERRUPÇÕES NÃO SERÃO UTILIZADAS, POR ISSO PODEMOS SUBSTITUIR
; TODO O SISTEMA EXISTENTE NO ARQUIVO MODELO PELO APRESENTADO ABAIXO
; ESTE SISTEMA NÃO É OBRIGATÓRIO, MAS PODE EVITAR PROBLEMAS FUTUROS

   ORG     0x04                    ; ENDEREÇO INICIAL DA INTERRUPÇÃO
   RETFIE                          ; RETORNA DA INTERRUPÇÃO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   ROTINA DE CONVERSÃO BINÁRIO -> DISPLAY                   *
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA IRÁ RETORNAR EM W, O SIMBOLO CORRETO QUE DEVE SER
; MOSTRADO NO DISPLAY PARA CADA VALOR DE CONTADOR. O RETORNO JÁ ESTÁ
; FORMATADO PARA AS CONDIÇÕES DE LIGAÇÃO DO DISPLAY AO PORTD.
;     a
;     **********
;     *        *
;   f *        * b
;     *    g   *
;     **********
;     *        *
;   e *        * c
;     *    d   *
;     **********  *.

CONVERTE
   MOVF    CONTADOR,W             ; COLOCA CONTADOR EM W
   ANDLW   B'00001111'            ; MASCARA VALOR DE CONTADOR
                                  ; CONSIDERAR SOMENTE ATÉ 15
   ADDWF   PCL,F

;          B'.GFEDCBA'            ; POSIÇÃO CORRETA DOS SEGMENTOS
   RETLW   B'00111111'            ; 00 - RETORNA SÍMBOLO CORRETO 0
   RETLW   B'00000110'            ; 01 - RETORNA SÍMBOLO CORRETO 1
   RETLW   B'01011011'            ; 02 - RETORNA SÍMBOLO CORRETO 2
   RETLW   B'01001111'            ; 03 - RETORNA SÍMBOLO CORRETO 3
   RETLW   B'01100110'            ; 04 - RETORNA SÍMBOLO CORRETO 4
   RETLW   B'01101101'            ; 05 - RETORNA SÍMBOLO CORRETO 5
   RETLW   B'01111101'            ; 06 - RETORNA SÍMBOLO CORRETO 6
   RETLW   B'00000111'            ; 07 - RETORNA SÍMBOLO CORRETO 7
   RETLW   B'01111111'            ; 08 - RETORNA SÍMBOLO CORRETO 8
   RETLW   B'01101111'            ; 09 - RETORNA SÍMBOLO CORRETO 9
   RETLW   B'01110111'            ; 10 - RETORNA SÍMBOLO CORRETO A
   RETLW   B'01111100'            ; 11 - RETORNA SÍMBOLO CORRETO b
   RETLW   B'00111001'            ; 12 - RETORNA SÍMBOLO CORRETO C
   RETLW   B'01011110'            ; 13 - RETORNA SÍMBOLO CORRETO d
   RETLW   B'01111001'            ; 14 - RETORNA SÍMBOLO CORRETO E
   RETLW   B'01110001'            ; 15 - RETORNA SÍMBOLO CORRETO F

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     ROTINA DE LEITURA NA E2PROM                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA LÊ O BYTE DO ENDEREÇO PASSADO PELO W E RETORNA
; O VALOR EM W.

LE_E2PROM
   MOVF    ENDERECO,W             ; RECUPERA ENDERECO A SER LIDO
   BSF     STATUS,RP1             ; COMUTA PARA BANCO 2 DA RAM
   MOVWF   EEADR                  ; SALVA O ENDEREÇO PARA LEITURA

   BSF     STATUS,RP0             ; COMUTA PARA BANCO 3 DA RAM
   BCF     EECON1,EEPGD           ; APONTA PARA MEMÓRIA DE DADOS
   BSF     EECON1,RD              ; HABILITA LEITURA

   BCF     STATUS,RP0             ; VOLTA PARA BANCO 2 DA RAM
   MOVF    EEDATA,W               ; COLOCA DADO LIDO EM W

   BCF     STATUS,RP1             ; VOLTA PARA BANCO 0 DA RAM

   RETURN                         ; RETORNA

;* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;*                  ROTINA DE ESCRITA NA E2PROM                              *
;* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA ESCREVE O DADO PASSSADO EM W NO ENDEREÇO ACERTADO
; ANTERIORMENTE NA VARIÁVEL ENDERECO

ESCR_E2PROM
   BSF     STATUS,RP1             ; COMUTA PARA BANCO 2 DA RAM
   MOVWF   EEDATA                 ; SALVA DADO PASSADO PELO W

   BCF     STATUS,RP1             ; VOLTA PARA BANCO 0 DA RAM
```

```
   MOVF    ENDERECO,W              ; CARREGA EM W O ENDERECO
   BSF     STATUS,RP1              ; COMUTA PARA BANCO 2 DA RAM
   MOVWF   EEADR                   ; SALVA ENDERECO EM EEADR

   BSF     STATUS,RP0              ; COMUTA PARA BANCO 3 DA RAM
   BCF     EECON1,EEPGD            ; APONTA PARA MEMÓRIA DE DADOS
   BSF     EECON1,WREN             ; HABILITA ESCRITA NA EEPROM

   MOVLW   0X55                    ; INICIALIZAÇÃO DA ESCRITA
   MOVWF   EECON2
   MOVLW   0XAA
   MOVWF   EECON2
   BSF     EECON1,WR               ; INICIA ESCRITA

   NOP                             ; NAO OPERA

   BCF     EECON1,WREN             ; DESABILITA ESCRITA NA EEPROM

   BTFSC   EECON1,WR               ; ACABOU ESCRITA?
   GOTO    $-1                     ; NÃO, AGUARDA

   BCF     STATUS,RP1
   BCF     STATUS,RP0              ; VOLTA AO BANCO 0 DA RAM

   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                ROTINA DE LEITURA DO VALOR DO CONTADOR                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA LÊ O VALOR DA MEMÓRIA E COLOCA O RESULTADO NA
; VARIÁVEL "CONTADOR".

LE_CONTA
   MOVLW   POS_MEM
   MOVWF   ENDERECO
   CALL    LE_E2PROM               ; EFETUA A LEITURA DA EEPROM
   MOVWF   CONTADOR                ; ATUALIZA O CONTADOR
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                ROTINA DE ESCRITA DO VALOR DO CONTADOR                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA ESCREVE O VALOR ATUAL DE CONTADOR NA MEMÓRIA EEPROM

ESCR_CONTA
   MOVLW   POS_MEM
   MOVWF   ENDERECO                ; ACERTA O ENDERECO DE LEITURA
   MOVF    CONTADOR,W              ; COLOCA CONTADOR EM W
   CALL    ESCR_E2PROM             ; EFETUA A ESCRITA NA EEPROM
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          INICIO DO PROGRAMA                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

INICIO
   CLRF    PORTA                   ; LIMPA O PORTA
   CLRF    PORTB                   ; LIMPA O PORTB
   CLRF    PORTC                   ; LIMPA O PORTC
   CLRF    PORTD                   ; LIMPA O PORTD
   CLRF    PORTE                   ; LIMPA O PORTE


   BANK1                           ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                   ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                   ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011001'
   MOVWF   TRISC                   ; CONFIGURA I/O DO PORTC
```

```
   MOVLW   B'00000000'
   MOVWF   TRISD                   ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                   ; CONFIGURA I/O DO PORTE

   MOVLW   B'00000111'
   MOVWF   CMCON                   ; DESLIGA COMPARADORES ANALÓGICOS

   MOVLW   B'00000111'
   MOVWF   ADCON1                  ; DESLIGA CONVERSORES A/D

   MOVLW   B'10000000'
   MOVWF   OPTION_REG              ; PRESCALER 1:2 NO TMR0
                                   ; PULL-UPS DESABILITADOS
                                   ; AS DEMAIS CONFG. SÃO IRRELEVANTES
   MOVLW   B'00000000'
   MOVWF   INTCON                  ; TODAS AS INTERRUPÇÕES DESLIGADAS

   BANK0                           ; RETORNA PARA O BANCO 0

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       INICIALIZAÇÃO DO HARDWARE                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   BSF     DSP_UNIDADE             ; ATIVA DISPLAY DA UNIDADE

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       INICIALIZAÇÃO DAS VARIÁVEIS                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   CLRF    FLAGS                   ; LIMPA TODOS OS FLAGS
   CALL    LE_CONTA                ; INICIALIZA CONTADOR COM VALOR
                                   ; DA EEPROM
   GOTO    ATUALIZA                ; ATUALIZA O DISPLAY INICIALMENTE

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                         ROTINA PRINCIPAL                                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

MAIN
   MOVLW   T_FILTRO

   MOVWF   FILTRO1                 ; INICIALIZA FILTRO1 = T_FILTRO
   MOVWF   FILTRO2                 ; INICIALIZA FILTRO2 = T_FILTRO

CHECA_BT1

   BTFSS   BOTAO1                  ; O BOTÃO 1 ESTÁ PRESSIONADO?
   GOTO    BT1_LIB                 ; NÃO, ENTÃO TRATA COMO LIBERADO
                                   ; SIM

   DECFSZ  FILTRO1,F               ; DECREMENTA O FILTRO DO BOTÃO
                                   ; TERMINOU?
   GOTO    CHECA_BT1               ; NÃO, CONTINUA ESPERANDO
                                   ; SIM

   BTFSS   ST_BT1                  ; BOTÃO JÁ ESTAVA PRESSIONADO?
   GOTO    DEC                     ; NÃO, EXECUTA AÇÃO DO BOTÃO
   GOTO    CHECA_BT2               ; SIM, CHECA BOTÃO 2

BT1_LIB
   BCF     ST_BT1                  ; MARCA BOTÃO 1 COMO LIBERADO

CHECA_BT2

   BTFSS   BOTAO2                  ; O BOTÃO 2 ESTÁ PRESSIONADO?
   GOTO    BT2_LIB                 ; NÃO, ENTÃO TRATA COMO LIBERADO
                                   ; SIM
   DECFSZ  FILTRO2,F               ; DECREMENTA O FILTRO DO BOTÃO
                                   ; TERMINOU?
```

```
   GOTO    CHECA_BT2              ; NÃO, CONTINUA ESPERANDO
                                  ; SIM
   BTFSS   ST_BT2                 ; BOTÃO JÁ ESTAVA PRESSIONADO?
   GOTO    INC                    ; NÃO, EXECUTA AÇÃO DO BOTÃO
   GOTO    MAIN                   ; SIM, VOLTA AO LOOPING

BT2_LIB
   BCF     ST_BT2                 ; MARCA BOTÃO 2 COMO LIBERADO
   GOTO    MAIN                   ; RETORNA AO LOOPING

DEC                               ; AÇÃO DE DECREMENTAR
   BSF     ST_BT1                 ; MARCA BOTÃO 1 COMO JÁ PRESSIONADO
   MOVF    CONTADOR,W             ; COLOCA CONTADOR EM W
   XORLW   MIN                    ; APLICA XOR ENTRE CONTADOR E MIN
                                  ; PARA TESTAR IGUALDADE. SE FOREM
                                  ; IGUAIS, O RESULTADO SERÁ ZERO
   BTFSC   STATUS,Z               ; RESULTOU EM ZERO?
   GOTO    MAIN                   ; SIM, RETORNA SEM AFETAR CONT.
                                  ; NÃO
   DECF    CONTADOR,F             ; DECREMENTA O CONTADOR
   CALL    ESCR_CONTA             ; ATUALIZA O VALOR DE CONTADOR NA
                                  ; EEPROM
   GOTO    ATUALIZA               ; ATUALIZA O DISPLAY

INC                               ; AÇÃO DE INCREMENTAR
   BSF     ST_BT2                 ; MARCA BOTÃO 2 COMO JÁ PRESSIONADO
   MOVF    CONTADOR,W             ; COLOCA CONTADOR EM W
   XORLW   MAX                    ; APLICA XOR ENTRE CONTADOR E MAX
                                  ; PARA TESTAR IGUALDADE. SE FOREM
                                  ; IGUAIS, O RESULTADO SERÁ ZERO
   BTFSC   STATUS,Z               ; RESULTOU EM ZERO?
   GOTO    MAIN                   ; SIM, RETORNA SEM AFETAR CONT.
                                  ; NÃO
   INCF    CONTADOR,F             ; INCREMENTA O CONTADOR
   CALL    ESCR_CONTA             ; ATUALIZA O VALOR DE CONTADOR NA
                                  ; EEPROM

ATUALIZA                          ; ATUALIZAÇÃO DO DISPLAY
   CALL    CONVERTE               ; CONVERTE CONTADOR NO NÚMERO DO
                                  ; DISPLAY
   MOVWF   PORTD                  ; ATUALIZA O PORTD PARA
                                  ; VISUALIZARMOS O VALOR DE CONTADOR
                                  ; NO DISPLAY
   GOTO    MAIN                   ; NÃO, VOLTA AO LOOP PRINCIPAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          FIM DO PROGRAMA                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                            ; OBRIGATÓRIO
```

Dicas e Comentários

A memória EEPROM do microcontrolador PIC16F877A pode ser lida a qualquer momento e infinitas vezes, isto é, não existe um limite para a quantidade de leituras que são realizadas na memória de dados. Porém, no caso da escrita, a memória EEPROM do PIC16F877A possui uma vida útil limitada. Atualmente o fabricante do componente garante uma vida útil de aproximadamente 1.000.000 de ciclos de escrita. Apesar deste número parecer extremamente alto, existem casos no qual este limite pode ser atingido com relativa facilidade. Portanto, sempre que se desenvolve um sistema que utilize a memória EEPROM e que necessite realizar um número de escritas considerável, deve-se pensar se o limite de 1.000.000 de ciclos será ou não atingido.

Exercícios Propostos

1. Alterar a posição onde o dado é armazenado na memória EEPROM.
2. Alterar o valor inicial do contador.
3. Acrescentar duas teclas, uma para salvar o dado na memória e outra para ler o dado da memória. Neste caso o contador não pode ser salvo sempre que alterado, apenas será salvo quando a tecla correta for pressionada. Quanto à leitura, deve ser realizada tanto na inicialização do software como quando forçada pela tecla.

Capítulo 9 - Experiência 7 - Dimmer

Objetivo

O objetivo desta experiência é ensinar ao aluno como regular a potência fornecida a pequenas cargas através do microcontrolador. Para isso, é proposto um dimmer para uma lâmpada incandescente onde a potência de lâmpada é regulada através do uso de um PWM criado no software.

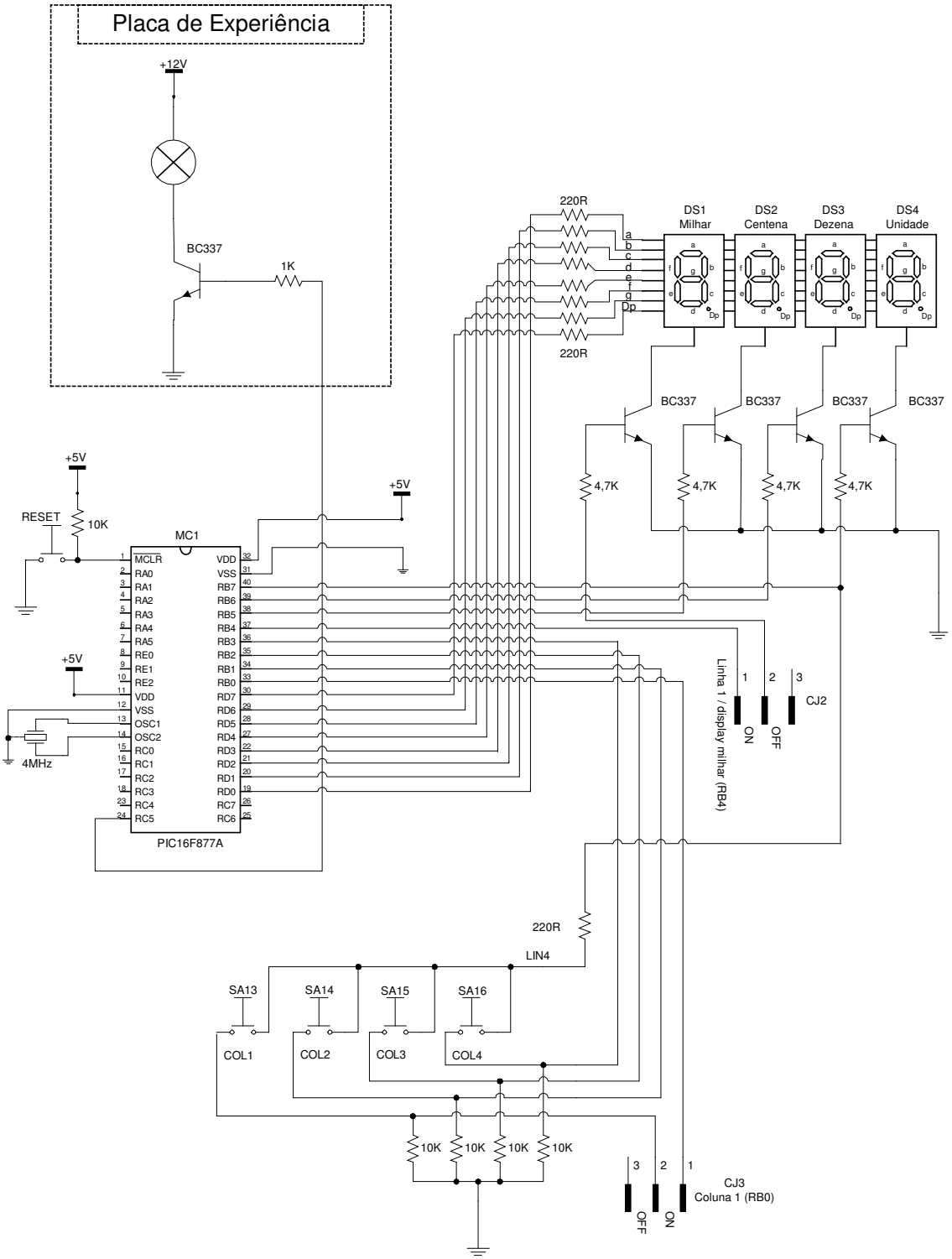
Descrição

Software que utiliza dois botões para incrementar e decrementar a intensidade da lâmpada. A interrupção de tmr0 é utilizada para controlar o PWM que aciona a lâmpada. A intensidade também é mostrada no display da unidade.

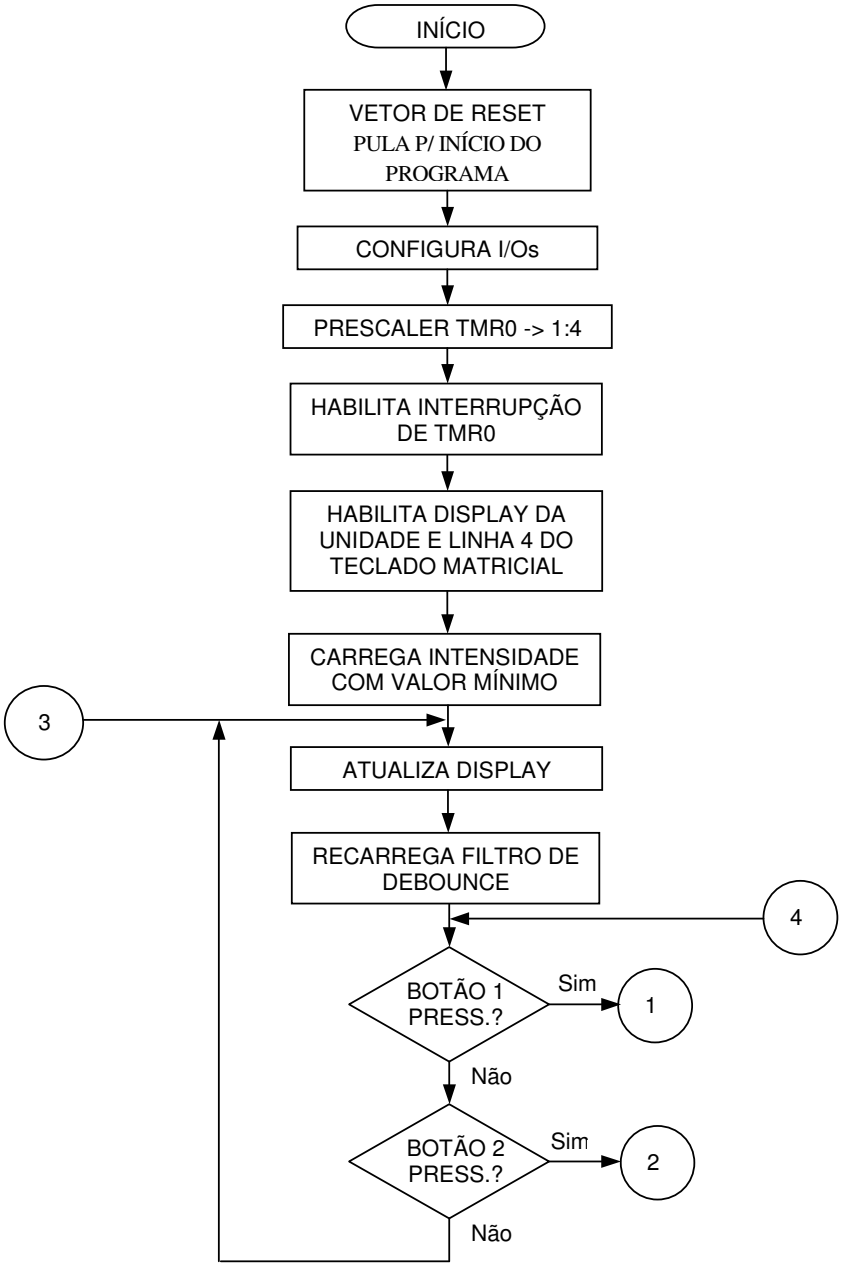
Os botões ativos são os da linha 4. O botão da coluna 1 incrementa a intensidade da lâmpada e o da coluna 2 decrementa a intensidade.

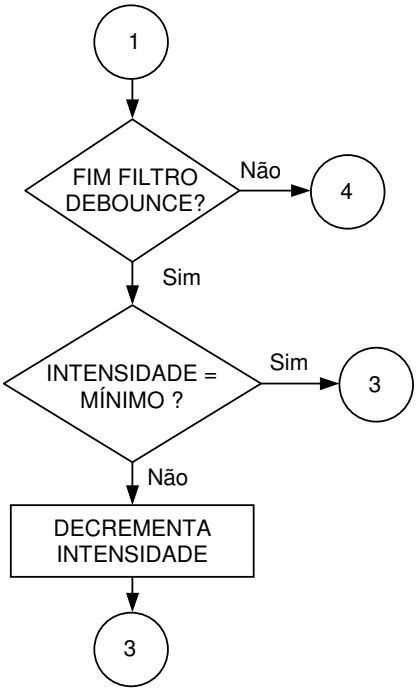
Foi criado um PWM com 4 bits de resolução de forma que existem 16 intensidades diferentes para a lâmpada. Para sua implementação, utilizou-se a interrupção de TMR0 sendo que a cada interrupção uma variável é incrementada. Esta variável é utilizada para subdividir o PWM em 16 intervalos. Comparando esta variável com o duty cycle ajustado pelo usuário, o software coloca o pino da lâmpada em nível lógico 1 ou 0, conforme a necessidade. A cada 16 interrupções o ciclo se repete e portanto o período do PWM é 16 vezes maior do que o período de uma interrupção. Este procedimento pode ser facilmente analisado pelo fluxograma da experiência.

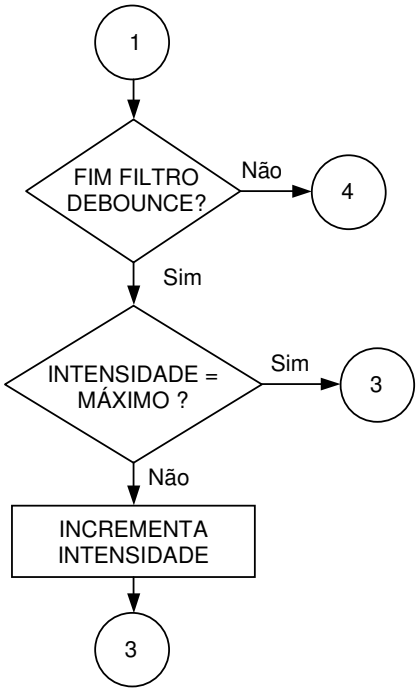
Esquema Elétrico

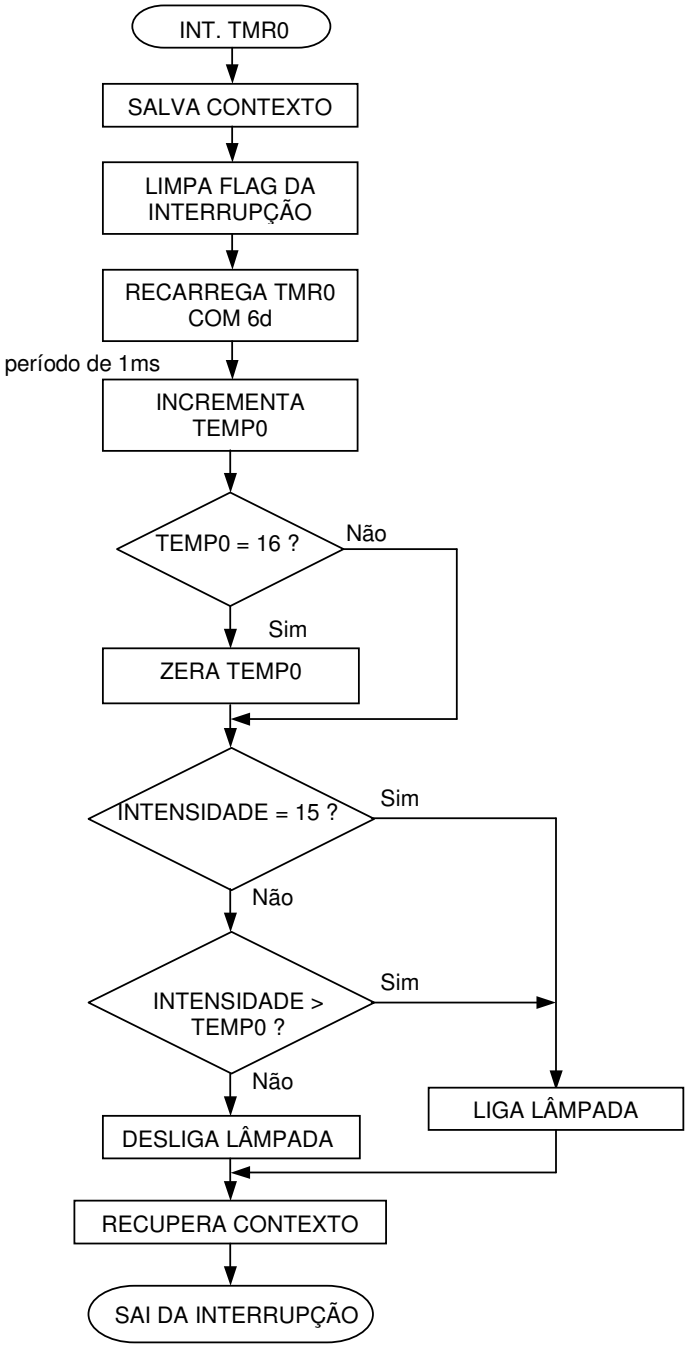


Fluxograma


INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURA I/Os
PRESCALER TMR0 -> 1:4
HABILITA INTERRUPÇÃO
DE TMR0
HABILITA DISPLAY DA
UNIDADE E LINHA 4 DO
TECLADO MATRICIAL
CARREGA INTENSIDADE
COM VALOR MÍNIMO
3
ATUALIZA DISPLAY
RECARREGA FILTRO DE
DEBOUNCE
4
BOTÃO 1
PRESS.?
Sim
1
Não
BOTÃO 2
PRESS.?
Sim
2
Não


1
FIM FILTRO DEBOUNCE?
Não
4
Sim
INTENSIDADE = MÍNIMO ?
Sim
3
Não
DECREMENTA INTENSIDADE
3


1
FIM FILTRO DEBOUNCE?
Não
4
Sim
INTENSIDADE = MÁXIMO ?
Sim
3
Não
INCREMENTA INTENSIDADE
3


INT. TMR0
SALVA CONTEXTO
LIMPA FLAG DA INTERRUPÇÃO
RECARREGA TMR0 COM 6d
período de 1ms
INCREMENTA TEMP0
TEMP0 = 16 ?
Não
Sim
ZERA TEMP0
INTENSIDADE = 15 ?
Sim
Não
INTENSIDADE > TEMP0 ?
Sim
Não
LIGA LÂMPADA
DESLIGA LÂMPADA
RECUPERA CONTEXTO
SAI DA INTERRUPÇÃO

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *                         EXPERIÊNCIA 7 - DIMMER                          *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                             DESCRIÇÃO GERAL                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; SOFTWARE QUE UTILIZA DOIS BOTÕES PARA INCREMENTAR E DECREMENTAR A
; INTENSIDADE DA LÂMPADA. A INTERRUPÇÃO DE TMR0 É UTILIZADA PARA CONTROLAR
; O PWM QUE ACIONA A LÂMPADA. A INTENSIDADE TAMBÉM É MOSTRADA NO DISPLAY
; DA UNIDADE.
; OS BOTÕES ATIVOS SÃO O DA LINHA 4

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       CONFIGURAÇÕES PARA GRAVAÇÃO                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_OFF & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               VARIÁVEIS                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DOS NOMES E ENDEREÇOS DE TODAS AS VARIÁVEIS UTILIZADAS
; PELO SISTEMA

   CBLOCK  0x20                    ; ENDEREÇO INICIAL DA MEMÓRIA DE
                                   ; USUÁRIO

           W_TEMP                  ; REGISTRADORES TEMPORÁRIOS PARA
           STATUS_TEMP             ; INTERRUPÇÕES
                                   ; ESTAS VARIÁVEIS NEM SERÃO UTI-
                                   ; LIZADAS
           INTENSIDADE             ; ARMAZENA O VALOR DA CONTAGEM
           FLAGS                   ; ARMAZENA OS FLAGS DE CONTROLE
           FILTRO11                ; FILTRAGEM  1 PARA O BOTÃO 1
           FILTRO12                ; FILTRAGEM  2 PARA O BOTÃO 1
           FILTRO21                ; FILTRAGEM  1 PARA O BOTÃO 2
           FILTRO22                ; FILTRAGEM  2 PARA O BOTÃO 2
           TEMPO                   ; INTERVALOS DE 1 MS

   ENDC                            ; FIM DO BLOCO DE MEMÓRIA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>         ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      DEFINIÇÃO DOS BANCOS DE RAM                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
; *                              FLAGS INTERNOS                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS FLAGS UTILIZADOS PELO SISTEMA

#DEFINE    ST_BT1  FLAGS,0         ; STATUS DO BOTÃO 1
#DEFINE    ST_BT2  FLAGS,1         ; STATUS DO BOTÃO 2

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               CONSTANTES                                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODAS AS CONSTANTES UTILIZADAS PELO SISTEMA

MIN               EQU     .0      ; VALOR MÍNIMO PARA O INTENSIDADE
MAX               EQU     .15     ; VALOR MÁXIMO PARA O INTENSIDADE
T_FILTRO          EQU     .20     ; FILTRO PARA BOTÃO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               ENTRADAS                                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS PINOS QUE SERÃO UTILIZADOS COMO ENTRADA
; RECOMENDAMOS TAMBÉM COMENTAR O SIGNIFICADO DE SEUS ESTADOS (0 E 1)

#DEFINE    BOTAO1  PORTB,0         ; PORTA DO BOTÃO
                                   ; 1 -> PRESSIONADO
                                   ; 0 -> LIBERADO

#DEFINE    BOTAO2  PORTB,1         ; PORTA DO BOTÃO
                                   ; 1 -> PRESSIONADO
                                   ; 0 -> LIBERADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                SAÍDAS                                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DEFINIÇÃO DE TODOS OS PINOS QUE SERÃO UTILIZADOS COMO SAÍDA
; RECOMENDAMOS TAMBÉM COMENTAR O SIGNIFICADO DE SEUS ESTADOS (0 E 1)

#DEFINE    DSP_UNIDADE     PORTB,7 ; PINO DISPLAY DA UNIDADE
                                   ; 1 -> DISPLAY ATIVADO
                                   ; 0 -> DISPLAY DESATIVADO

#DEFINE    LINHA_4         PORTB,7 ; PINO PARA ATIVAR LINHA 4 DO TECLADO
                                   ;  MATRICIAL
                                   ; 1 -> LINHA ATIVADA
                                   ; 0 -> LINHA DESATIVADA

#DEFINE    LAMPADA         PORTC,5 ; DEFINE PINO DA LAMPADA
                                   ; 0 -> LÂMPADA APAGADA
                                   ; 1 -> LÂMPADA ACESA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           VETOR DE RESET                                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   ORG     0x00            ; ENDEREÇO INICIAL DE PROCESSAMENTO
   GOTO    INICIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                         INÍCIO DA INTERRUPÇÃO                               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; AS INTERRUPÇÕES NÃO SERÃO UTILIZADAS, POR ISSO PODEMOS SUBSTITUIR
; TODO O SISTEMA EXISTENTE NO ARQUIVO MODELO PELO APRESENTADO ABAIXO
; ESTE SISTEMA NÃO É OBRIGATÓRIO, MAS PODE EVITAR PROBLEMAS FUTUROS

   ORG     0x04                    ; ENDEREÇO INICIAL DA INTERRUPÇÃO
   MOVWF   W_TEMP                  ; SALVA W EM W_TEMP
   SWAPF   STATUS,W
   MOVWF   STATUS_TEMP             ; SALVA STATUS EM STATUS_TEMP

   BTFSS   INTCON,T0IF             ; É INTERRUPÇÃO DE TMR0?
   GOTO    SAI_INT                 ; NÃO, SAI SE AÇÃO
                                   ; SIM
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                TRATAMENTO DA INTERRUPÇÃO DE TMR0                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA SERÁ EXECUTADA A CADA 1ms.
; 1ms = 4us (PRESCALER) X 250us (TMR0)
; A INTERRUPÇÃO É RESPONSÁVEL PELO PWM QUE MODULA A INTENSIDADE DA LÂMPADA

   BCF     INTCON,T0IF             ; LIMPA FLAG DA INT.

   MOVLW   .256-.250               ; SETA TIMER P/ 1ms
   MOVWF   TMR0                    ; REINICIA TMR0

   INCF    TEMPO,F                 ; INCREMENTA TEMPO

   MOVLW   .16                     ; COLOCA 16 EM WORK
   XORWF   TEMPO,W                 ; COMPARA TEMPO COM 16
   BTFSC   STATUS,Z                ; TESTA BIT Z DO REG. STATUS
   CLRF    TEMPO                   ; ZERA TEMPO

   MOVLW   .15                     ; COLOCA 15 EM W
   XORWF   INTENSIDADE,W           ; COMPARA INTENSIDADE COM 15
   BTFSC   STATUS,Z                ; TESTA BIT Z DO REG. STATUS
   GOTO    LIGA_LAMPADA

   MOVF    INTENSIDADE,W           ; MOVE INTENSIDADE PARA W
   SUBWF   TEMPO,W                 ; SUBTRAI TEMPO DE INTENSIDADE
   BTFSS   STATUS,C                ; TESTA BIC C DO REG. STATUS
                                   ; VERIFICA SE TEMPO E MENOR QUE INTENSIDADE
   GOTO    LIGA_LAMPADA

   BCF     LAMPADA                 ; DESLIGA LÂMPADA
   GOTO    SAI_INT

LIGA_LAMPADA

   BSF     LAMPADA                 ; LIGA LÂMPADA

   GOTO    SAI_INT                 ; SAI DA INTERRUPÇÃO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          FIM DA INTERRUPÇÃO                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
SAI_INT
   SWAPF   STATUS_TEMP,W
   MOVWF   STATUS                  ; RECUPERA STATUS
   SWAPF   W_TEMP,F
   SWAPF   W_TEMP,W                ; RECUPERA W
   RETFIE                          ; RETORNA DA INTERRUPÇÃO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   ROTINA DE CONVERSÃO BINÁRIO -> DISPLAY                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA IRÁ RETORNAR EM W, O SIMBOLO CORRETO QUE DEVE SER
; MOSTRADO NO DISPLAY PARA CADA VALOR DE INTENSIDADE. O RETORNO JÁ ESTÁ
; FORMATADO PARA AS CONDIÇÕES DE LIGAÇÃO DO DISPLAY AO PORTD.
;     a
;     **********
;     *        *
;   f *        * b
;     *    g   *
;     **********
;     *        *
;   e *        * c
;     *    d   *
;     **********  *.

CONVERTE
   MOVF    INTENSIDADE,W           ; COLOCA INTENSIDADE EM W
   ANDLW   B'00001111'             ; MASCARA VALOR DE INTENSIDADE
                                   ; CONSIDERAR SOMENTE ATÉ 15
   ADDWF   PCL,F
```

```
;          B'.GFEDCBA'             ; POSIÇÃO CORRETA DOS SEGMENTOS
   RETLW   B'00111111'             ; 00 - RETORNA SÍMBOLO CORRETO 0
   RETLW   B'00000110'             ; 01 - RETORNA SÍMBOLO CORRETO 1
   RETLW   B'01011011'             ; 02 - RETORNA SÍMBOLO CORRETO 2
   RETLW   B'01001111'             ; 03 - RETORNA SÍMBOLO CORRETO 3
   RETLW   B'01100110'             ; 04 - RETORNA SÍMBOLO CORRETO 4
   RETLW   B'01101101'             ; 05 - RETORNA SÍMBOLO CORRETO 5
   RETLW   B'01111101'             ; 06 - RETORNA SÍMBOLO CORRETO 6
   RETLW   B'00000111'             ; 07 - RETORNA SÍMBOLO CORRETO 7
   RETLW   B'01111111'             ; 08 - RETORNA SÍMBOLO CORRETO 8
   RETLW   B'01101111'             ; 09 - RETORNA SÍMBOLO CORRETO 9
   RETLW   B'01110111'             ; 10 - RETORNA SÍMBOLO CORRETO A
   RETLW   B'01111100'             ; 11 - RETORNA SÍMBOLO CORRETO b
   RETLW   B'00111001'             ; 12 - RETORNA SÍMBOLO CORRETO C
   RETLW   B'01011110'             ; 13 - RETORNA SÍMBOLO CORRETO d
   RETLW   B'01111001'             ; 14 - RETORNA SÍMBOLO CORRETO E
   RETLW   B'01110001'             ; 15 - RETORNA SÍMBOLO CORRETO F

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          INICIO DO PROGRAMA                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

INICIO
   CLRF    PORTA                   ; LIMPA O PORTA
   CLRF    PORTB                   ; LIMPA O PORTB
   CLRF    PORTC                   ; LIMPA O PORTC
   CLRF    PORTD                   ; LIMPA O PORTD
   CLRF    PORTE                   ; LIMPA O PORTE


   BANK1                           ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                   ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                   ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011001'
   MOVWF   TRISC                   ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                   ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                   ; CONFIGURA I/O DO PORTE

   MOVLW   B'00000111'
   MOVWF   CMCON                   ; DESLIGA COMPARADORES ANALÓGICOS

   MOVLW   B'00000111'
   MOVWF   ADCON1                  ; DESLIGA CONVERSORES A/D

   MOVLW   B'10000001'
   MOVWF   OPTION_REG              ; PRESCALER 1:4 NO TMR0
                                   ; PULL-UPS DESABILITADOS
                                   ; AS DEMAIS CONFG. SÃO IRRELEVANTES
   MOVLW   B'10100000'
   MOVWF   INTCON                  ; HABILITA INTERRUPÇÃO DE TMR0

   BANK0                           ; RETORNA PARA O BANCO 0

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       INICIALIZAÇÃO DO HARDWARE                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   BSF     DSP_UNIDADE             ; ATIVA DISPLAY DA UNIDADE

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       INICIALIZAÇÃO DAS VARIÁVEIS                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   CLRF    FLAGS                   ; LIMPA TODOS OS FLAGS
```

```
   MOVLW   MIN
   MOVWF   INTENSIDADE             ; INICIA INTENSIDADE = MIN
   MOVLW   .256-.250               ; SETA TIMER P/ 1ms
   MOVWF   TMR0                    ; REINICIA TMR0
   GOTO    ATUALIZA                ; ATUALIZA O DISPLAY INICIALMENTE

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          ROTINA PRINCIPAL                               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

MAIN
   CLRF    FILTRO11
   CLRF    FILTRO21
   MOVLW   T_FILTRO
   MOVWF   FILTRO12                ; INICIALIZA FILTRO1 = T_FILTRO
   MOVWF   FILTRO22                ; INICIALIZA FILTRO2 = T_FILTRO

CHECA_BT1

   BTFSS   BOTAO1                  ; O BOTÃO 1 ESTÁ PRESSIONADO?
   GOTO    BT1_LIB                 ; NÃO, ENTÃO TRATA COMO LIBERADO
                                   ; SIM

   DECFSZ  FILTRO11,F              ; DECREMENTA O FILTRO DO BOTÃO
                                   ; TERMINOU?
   GOTO    CHECA_BT1               ; NÃO, CONTINUA ESPERANDO
                                   ; SIM

   DECFSZ  FILTRO12,F              ; DECREMENTA O FILTRO DO BOTÃO
                                   ; TERMINOU?
   GOTO    CHECA_BT1               ; NÃO, CONTINUA ESPERANDO
                                   ; SIM

   BTFSS   ST_BT1                  ; BOTÃO JÁ ESTAVA PRESSIONADO?
   GOTO    DEC                     ; NÃO, EXECUTA AÇÃO DO BOTÃO
   GOTO    CHECA_BT2               ; SIM, CHECA BOTÃO 2

BT1_LIB
   BCF     ST_BT1                  ; MARCA BOTÃO 1 COMO LIBERADO

CHECA_BT2

   BTFSS   BOTAO2                  ; O BOTÃO 2 ESTÁ PRESSIONADO?
   GOTO    BT2_LIB                 ; NÃO, ENTÃO TRATA COMO LIBERADO
                                   ; SIM

   DECFSZ  FILTRO21,F              ; DECREMENTA O FILTRO DO BOTÃO
                                   ; TERMINOU?
   GOTO    CHECA_BT2               ; NÃO, CONTINUA ESPERANDO
                                   ; SIM

   DECFSZ  FILTRO22,F              ; DECREMENTA O FILTRO DO BOTÃO
                                   ; TERMINOU?
   GOTO    CHECA_BT2               ; NÃO, CONTINUA ESPERANDO
                                   ; SIM

   BTFSS   ST_BT2                  ; BOTÃO JÁ ESTAVA PRESSIONADO?
   GOTO    INC                     ; NÃO, EXECUTA AÇÃO DO BOTÃO
   GOTO    MAIN                    ; SIM, VOLTA AO LOOPING

BT2_LIB
   BCF     ST_BT2                  ; MARCA BOTÃO 2 COMO LIBERADO
   GOTO    MAIN                    ; RETORNA AO LOOPING

DEC                                ; AÇÃO DE DECREMENTAR
   BSF     ST_BT1                  ; MARCA BOTÃO 1 COMO JÁ PRESSIONADO
   MOVF    INTENSIDADE,W           ; COLOCA INTENSIDADE EM W
   XORLW   MIN                     ; APLICA XOR ENTRE INTENSIDADE E MIN
                                   ; PARA TESTAR IGUALDADE. SE FOREM
                                   ; IGUAIS, O RESULTADO SERÁ ZERO
   BTFSC   STATUS,Z                ; RESULTOU EM ZERO?
   GOTO    MAIN                    ; SIM, RETORNA SEM AFETAR CONT.
```

```
                                ; NÃO
   DECF    INTENSIDADE,F        ; DECREMENTA O INTENSIDADE
   GOTO    ATUALIZA             ; ATUALIZA O DISPLAY

INC                             ; AÇÃO DE INCREMENTAR
   BSF     ST_BT2               ; MARCA BOTÃO 2 COMO JÁ PRESSIONADO
   MOVF    INTENSIDADE,W        ; COLOCA INTENSIDADE EM W
   XORLW   MAX                  ; APLICA XOR ENTRE INTENSIDADE E MAX
                                ; PARA TESTAR IGUALDADE. SE FOREM
                                ; IGUAIS, O RESULTADO SERÁ ZERO
   BTFSC   STATUS,Z             ; RESULTOU EM ZERO?
   GOTO    MAIN                 ; SIM, RETORNA SEM AFETAR CONT.
                                ; NÃO
   INCF    INTENSIDADE,F        ; INCREMENTA O INTENSIDADE
   GOTO    ATUALIZA             ; ATUALIZA O DISPLAY

ATUALIZA
   CALL    CONVERTE             ; CONVERTE INTENSIDADE NO NÚMERO DO
                                ; DISPLAY
   MOVWF   PORTD                ; ATUALIZA O PORTD PARA
                                ; VISUALIZARMOS O VALOR DE INTENSIDADE
                                ; NO DISPLAY
   GOTO    MAIN                 ; NÃO, VOLTA AO LOOP PRINCIPAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        FIM DO PROGRAMA                                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                          ; OBRIGATÓRIO
```

Dicas e Comentários

A técnica utilizada nesta experiência para implementar o PWM apresenta como inconveniente resultar em PWMs de baixa freqüência. Por exemplo, no caso desta experiência a interrupção ocorre a cada 1ms e como o PWM têm 4 bits de resolução, temos um período total de 16ms para o PWM. Isto significa que a freqüência é de 62,5Hz, ou seja, o PWM implementado via software apresenta baixa freqüência. Além disso, se ele fosse de por exemplo 8 bits, o período passaria para 256ms e, portanto, a freqüência cairia para aproximadamente 4Hz, ou seja, extremamente baixa. Note que quanto maior a resolução desejada maior é o período do PWM.

Exercícios Propostos

1. Altere a freqüência do PWM alterando a base de tempo do TMR0 para 100us.
2. Altere o PWM para trabalhar com apenas 2 bits de resolução.
3. Acrescente as rotinas de leitura e escrita na EEPROM para salvar e recuperar a intensidade da lâmpada mesmo na falta de energia.

Capítulo 10 - Experiência 8 – Botões, Leds e Buzzer

Objetivo

O objetivo desta experiência é dar continuidade ao aprendizado das interrupções e em particular à interrupção de TMR2 utilizada nesta experiência para excitar o buzzer.

Descrição

O buzzer utilizado no MCMASTER não é do tipo auto-oscilante, ou seja, este tipo de buzzer precisa ser excitado externamente para emitir algum som. Assim, o microcontrolador deve ser responsável por gerar uma onda quadrada a fim de excitar o piezo-elétrico e fazer com que o buzzer emita algum som. Operar com este tipo de buzzer é muito mais trabalhoso do que com um buzzer auto-oscilante, já que não basta ligar-lo, é necessário aplicar uma onda quadrada para que o buzzer funcione. Porém, ele dá margem à criação de diferentes tons, pois, quem gera a freqüência da onda quadrada e conseqüentemente o tom com que o buzzer irá tocar é o microcontrolador e se este gerar freqüências adequadas, pode-se implementar até mesmo uma simples melodia.

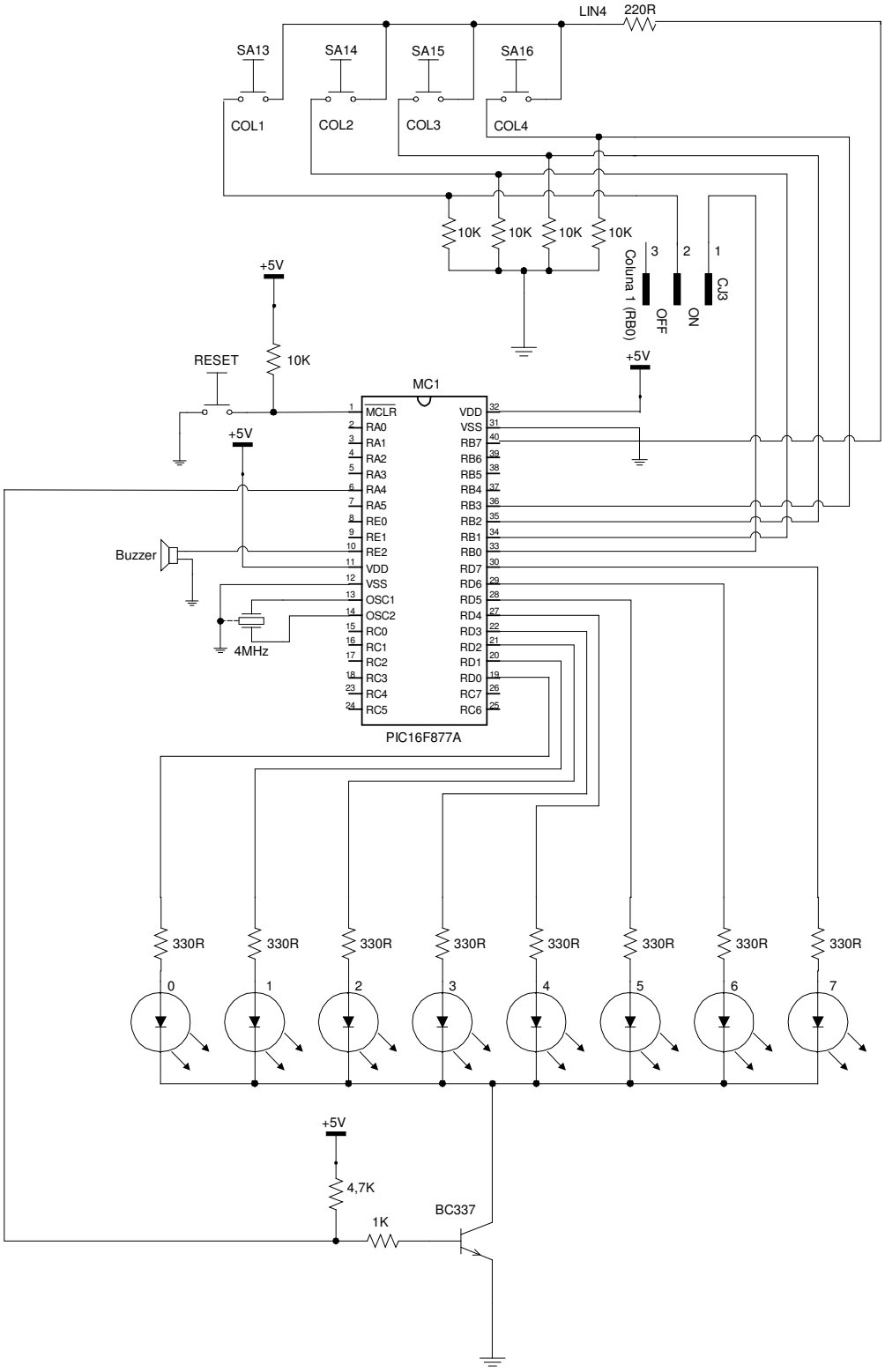
Apesar desta experiência utilizar o TMR2 para excitar o buzzer com freqüência variável, o objetivo não é a implementação de notas musicais, embora isto seja possível.

Diferente dos outros timers do microcontrolador, o TMR2 estoura e pode gerar uma interrupção sempre que o seu valor ultrapassa o valor de um registrador especial, denominado **PR2**. Ou seja, o TMR2, diferentemente dos outros timers conta desde zero até o valor programado no registrador PR2 e não até 255 (a não ser que o valor do registrador PR2 seja 255).

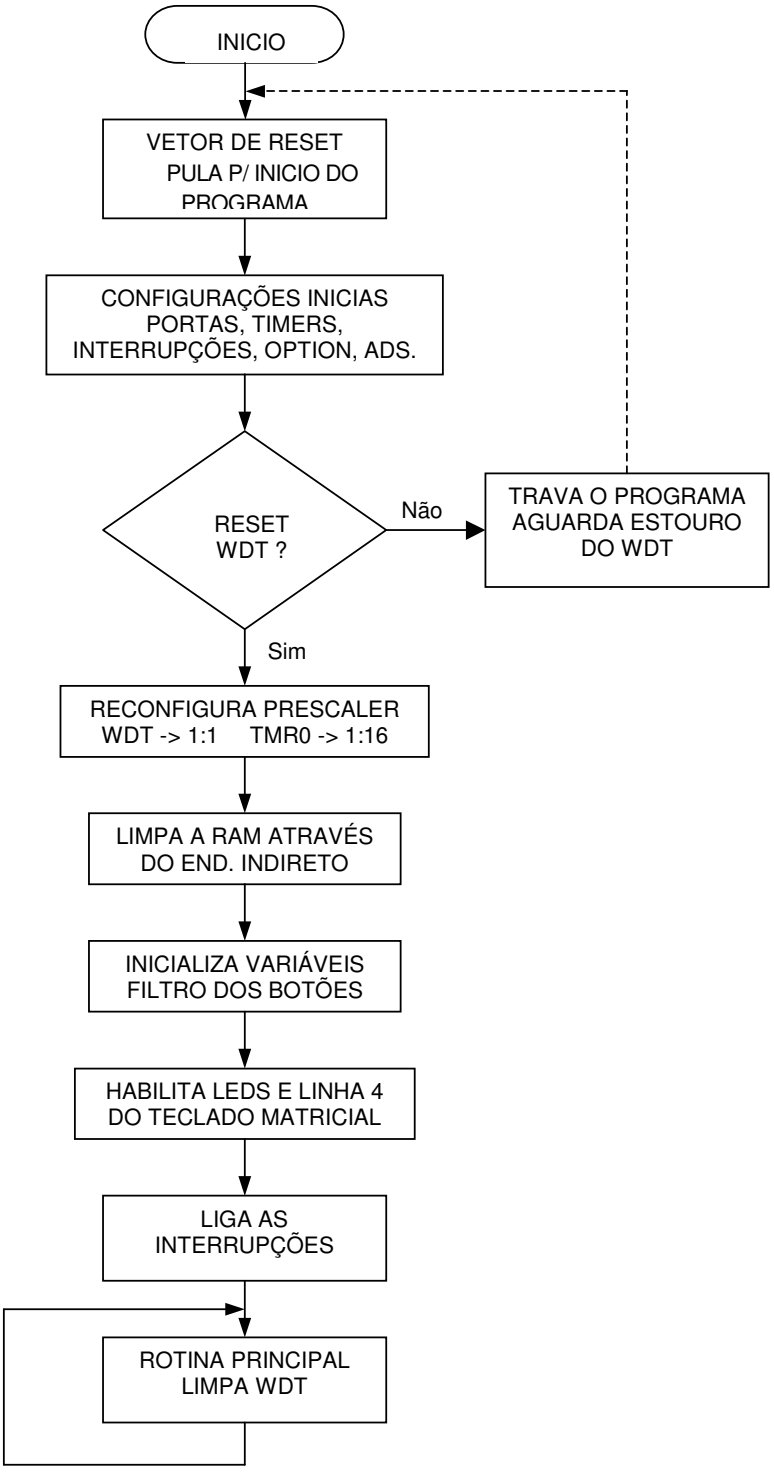
Desta forma, o que o software faz é ler o estado das teclas da linha 4 e em função deste alterar o valor do registrador **PR2**. Conseqüentemente, a interrupção de TMR2 varia de período, ou seja, a freqüência com que a interrupção ocorre depende do estado das teclas da linha 4. Como é a interrupção a responsável por gerar a onda quadrada que excita o buzzer e como a freqüência pode ser alterada pelas teclas, dependendo da configuração de teclas pressionadas o buzzer emite som em diferentes tonalidades, criando, embora com freqüência erradas uma espécie de mini-teclado.

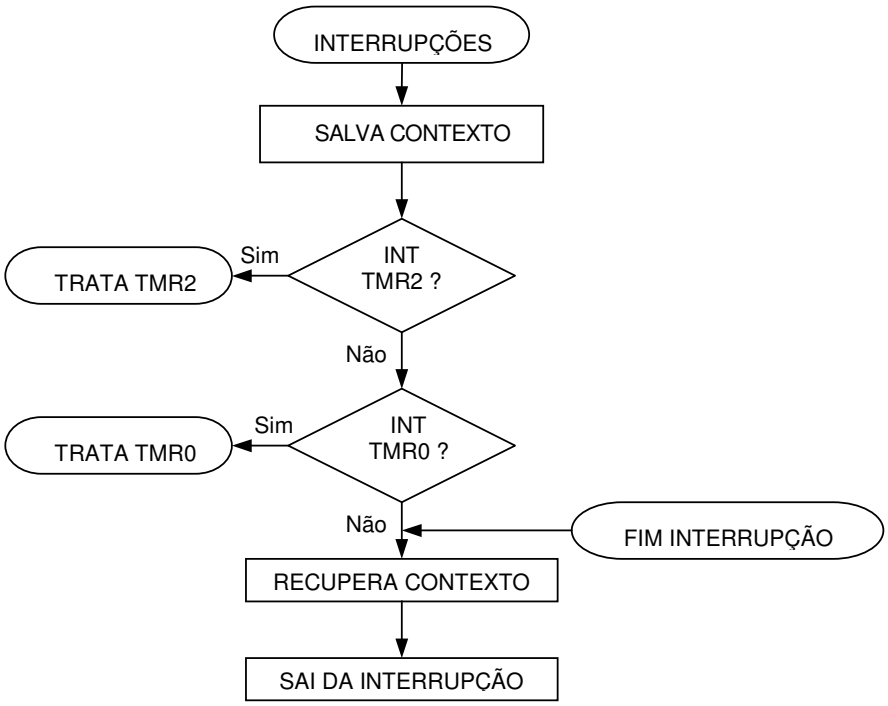
Os leds ligados ao PORTD são utilizados apenas para repetir o estado das teclas pressionadas a fim de criar um efeito visual para a experiência.

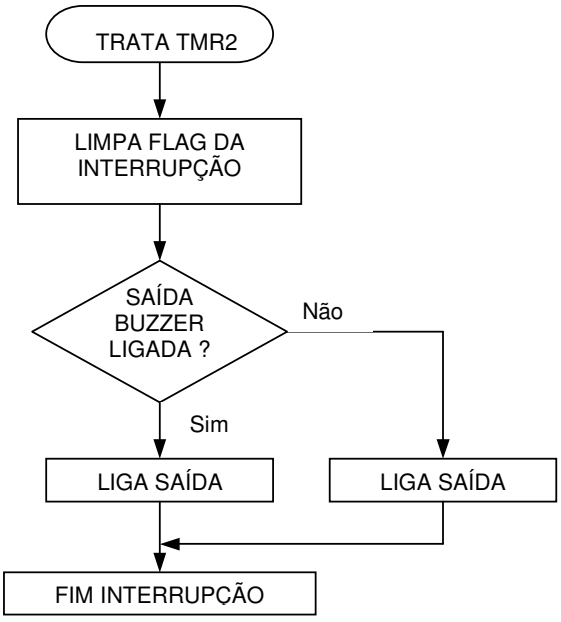
Esquema Elétrico

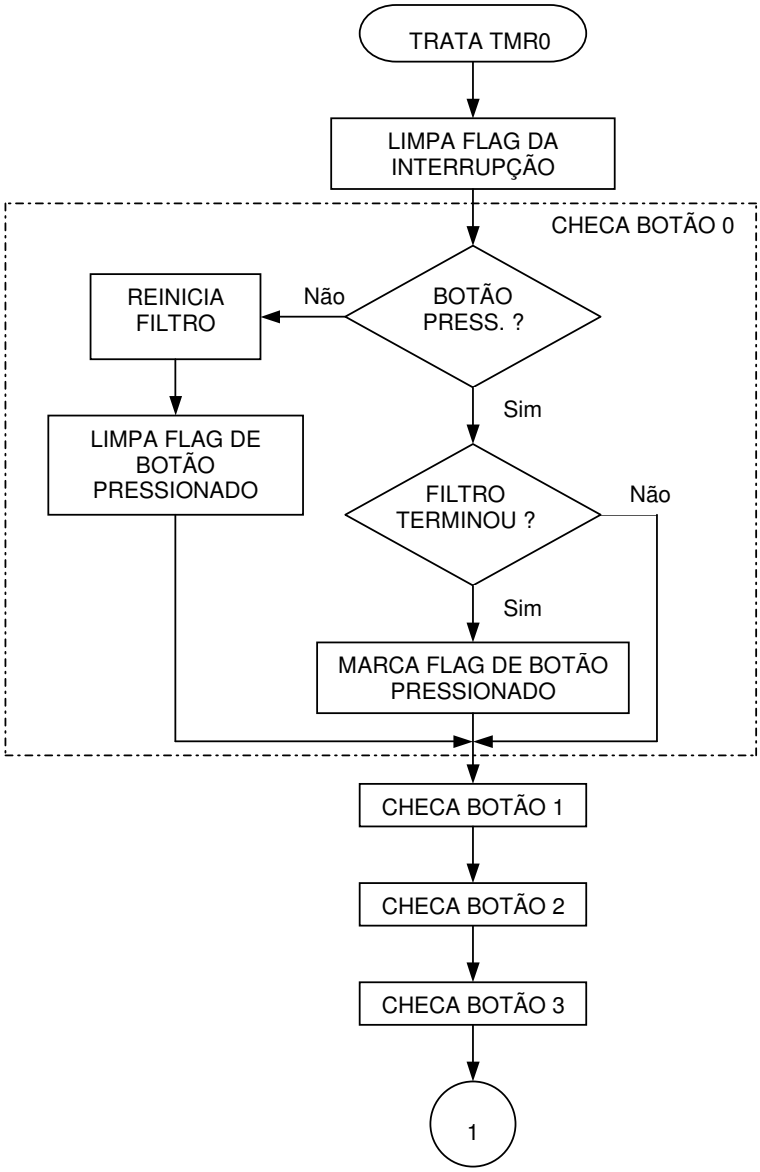


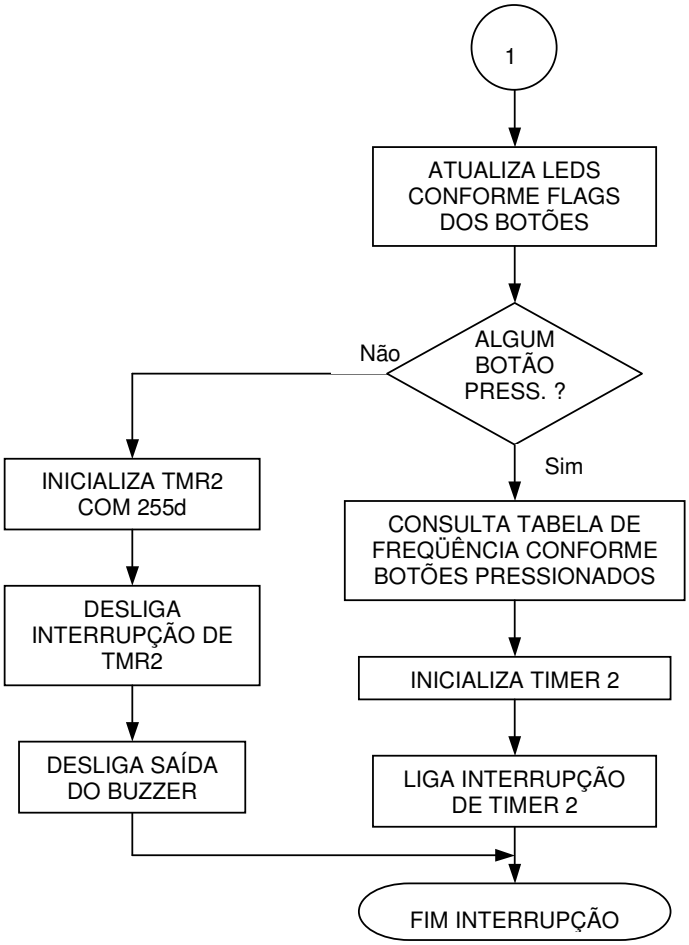
Fluxograma


INICIO
VETOR DE RESET
PULA P/ INICIO DO
PROGRAMA
CONFIGURAÇÕES INICIAS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION, ADS.
RESET
WDT ?
Não
TRAVA O PROGRAMA
AGUARDA ESTOURO
DO WDT
Sim
RECONFIGURA PRESCALER
WDT -> 1:1 TMR0 -> 1:16
LIMPA A RAM ATRAVÉS
DO END. INDIRETO
INICIALIZA VARIÁVEIS
FILTRO DOS BOTÕES
HABILITA LEDS E LINHA 4
DO TECLADO MATRICIAL
LIGA AS
INTERRUPÇÕES
ROTINA PRINCIPAL
LIMPA WDT


INTERRUPÇÕES
SALVA CONTEXTO
INT
TMR2 ?
Sim
TRATA TMR2
Não
INT
TMR0 ?
Sim
TRATA TMR0
Não
FIM INTERRUPÇÃO
RECUPERA CONTEXTO
SAI DA INTERRUPÇÃO


TRATA TMR2
LIMPA FLAG DA
INTERRUPÇÃO
SAÍDA
BUZZER
LIGADA ?
Não
Sim
LIGA SAÍDA
LIGA SAÍDA
FIM INTERRUPÇÃO


TRATA TMR0
LIMPA FLAG DA INTERRUPÇÃO
CHECA BOTÃO 0
BOTÃO PRESS. ?
Não
REINICIA FILTRO
Sim
LIMPA FLAG DE BOTÃO PRESSIONADO
FILTRO TERMINOU ?
Não
Sim
MARCA FLAG DE BOTÃO PRESSIONADO
CHECA BOTÃO 1
CHECA BOTÃO 2
CHECA BOTÃO 3
1


1
ATUALIZA LEDS CONFORME FLAGS DOS BOTÕES
ALGUM BOTÃO PRESS. ?
Não
Sim
INICIALIZA TMR2 COM 255d
DESLIGA INTERRUPÇÃO DE TMR2
DESLIGA SAÍDA DO BUZZER
CONSULTA TABELA DE FREQÜÊNCIA CONFORME BOTÕES PRESSIONADOS
INICIALIZA TIMER 2
LIGA INTERRUPÇÃO DE TIMER 2
FIM INTERRUPÇÃO

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *                   EXPERIÊNCIA 8 - BOTÕES, LEDS E BUZZER                 *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                             DESCRIÇÃO GERAL                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTE SOFTWARE ESTÁ PREPARADO PARA LER QUATRO BOTÕES E TOCAR O BUZZER COM
; DURAÇÃO VARIÁVEL CONFORME A TECLA PRESSIONADA, ALÉM DE ACENDER O LED
; INDICANDO AS TECLAS PRESSIONADAS.
; AS TECLAS DA LINHA 4 DEVEM SER UTILIZADAS PARA TOCAR O BUZZER.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       CONFIGURAÇÕES PARA GRAVAÇÃO                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_ON & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           DEFINIÇÃO DAS VARIÁVEIS                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE BLOCO DE VARIÁVEIS ESTÁ LOCALIZADO NO FINAL DO BANCO 0, A PARTIR
;  DO ENDEREÇO 0X70, POIS ESTA LOCALIZAÇÃO É ACESSADA DE QUALQUER BANCO,
;  FACILITANDO A OPERAÇÃO COM AS VARIÁVEIS AQUI LOCALIZADAS.

   CBLOCK  0X70                    ; POSIÇÃO COMUM A TODOS OS BANCOS

           W_TEMP                  ; REGISTRADOR TEMPORÁRIO PARA W
           STATUS_TEMP             ; REGISTRADOR TEMPORÁRIO PARA STATUS

           STATUS_BOTOES           ; REGISTRADOR PARA ARMAZENAR O STATUS DOS BOTÕES

           FILTRO_BT0              ; FILTRO PARA BOTAO 0
           FILTRO_BT1              ; FILTRO PARA BOTAO 1
           FILTRO_BT2              ; FILTRO PARA BOTAO 2
           FILTRO_BT3              ; FILTRO PARA BOTAO 3

   ENDC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>         ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     DEFINIÇÃO DOS BANCOS DE RAM                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0       ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0       ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           CONSTANTES INTERNAS                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE CONSTANTES FACILITA A PROGRAMAÇÃO E A MANUTENÇÃO.
```

```
FILTRO_BOTAO      EQU     .20     ; FILTRO P/ EVITAR RUIDOS DOS BOTÕES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    DECLARAÇÃO DOS FLAGs DE SOFTWARE                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE FLAGs AJUDA NA PROGRAMAÇÃO E ECONOMIZA MEMÓRIA RAM.

;  ESTE PROGRAMA NÃO UTILIZA NENHUM FLAG DE USUÁRIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                ENTRADAS                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS ENTRADAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    BOTAO_0         PORTB,0 ; ESTADO DO BOTÃO 0
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_1         PORTB,1 ; ESTADO DO BOTÃO 1
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_2         PORTB,2 ; ESTADO DO BOTÃO 2
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_3         PORTB,3 ; ESTADO DO BOTÃO 3
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                SAÍDAS                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS SAÍDAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    LINHA_4         PORTB,7 ; PINO PARA ATIVAR LINHA 4 DO TECLADO
                                   ;  MATRICIAL
                                   ; 1 -> LINHA ATIVADA
                                   ; 0 -> LINHA DESATIVADA

#DEFINE    LEDS            PORTD   ; PORT LIGADO AOS LEDS

#DEFINE    LED_BOTAO_0     PORTD,0 ; LED CORRESPONDENTE AO BOTÃO 0
                                   ; 1 -> LED LIGADO
                                   ; 0 -> LED DESLIGADO

#DEFINE    LED_BOTAO_1     PORTD,1 ; LED CORRESPONDENTE AO BOTÃO 1
                                   ; 1 -> LED LIGADO
                                   ; 0 -> LED DESLIGADO

#DEFINE    LED_BOTAO_2     PORTD,2 ; LED CORRESPONDENTE AO BOTÃO 2
                                   ; 1 -> LED LIGADO
                                   ; 0 -> LED DESLIGADO

#DEFINE    LED_BOTAO_3     PORTD,3 ; LED CORRESPONDENTE AO BOTÃO 3
                                   ; 1 -> LED LIGADO
                                   ; 0 -> LED DESLIGADO

#DEFINE    C_LEDS          PORTA,4 ; PINO PARA ATIVAR GRUPO DE 8 LEDS
                                   ; 1 -> LEDS ATIVADOS
                                   ; 0 -> LEDS DESATIVADOS

#DEFINE    BUZZER          PORTE,2 ; SAÍDA PARA BUZZER

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  VETOR DE RESET DO MICROCONTROLADOR                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO INICIAL PARA EXECUÇÃO DO PROGRAMA
```

```
   ORG     0X0000                  ; ENDEREÇO DO VETOR DE RESET
   GOTO    CONFIG                  ; PULA PARA CONFIG DEVIDO A REGIÃO
                                   ; DESTINADA ÀS INTERRUPÇÕES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               VETOR DE INTERRUPÇÃO DO MICROCONTROLADOR                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO DE DESVIO DO PROGRAMA QUANDO UMA INTERRUPÇÃO ACONTECE

   ORG     0X0004                  ; ENDEREÇO DO VETOR DE INTERRUPÇÃO

;  É MUITO IMPORTANTE QUE OS REGISTRADORES PRIORITÁRIOS AO FUNCIONAMENTO DA
;  MÁQUINA, E QUE PODEM SER ALTERADOS TANTO DENTRO QUANTO FORA DAS INTS SEJAM
;  SALVOS EM REGISTRADORES TEMPORÁRIOS PARA PODEREM SER POSTERIORMENTE
;  RECUPERADOS.

SALVA_CONTEXTO
   MOVWF   W_TEMP                  ; COPIA W PARA W_TEMP
   SWAPF   STATUS,W
   MOVWF   STATUS_TEMP             ; COPIA STATUS PARA STATUS_TEMP

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                TESTA QUAL INTERRUPÇÃO FOI SOLICITADA                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  TESTA O FLAG DAS INTERRUPÇÕES PARA SABER PARA QUAL ROTINA DESVIAR.

TESTA_INT
   BTFSC   INTCON,T0IF             ; FOI INTERRUPÇÃO DE TMR0 ?
   GOTO    INT_TMR0                ; SIM - PULA P/ INT_TMR0
                                   ; NÃO - ENTÃO FOI TMR2

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                TRATAMENTO DA INTERRUPÇÃO DE TIMER 2                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ROTINA PARA TRATAMENTO DA INTERRUPÇÃO DE TIMER 2.
;  INVERTE O ESTADO DO PINO DO BUZZER.

INT_TMR2
   BCF     PIR1,TMR2IF             ; LIMPA FLAG DA INTERRUPÇÃO

   BTFSS   BUZZER                  ; BUZZER LIGADO?
   GOTO    LIGA_BUZZER             ; NÃO -  ENTÃO LIGA
                                   ; SIM
   BCF     BUZZER                  ; DESLIGA O BUZZER
   GOTO    SAI_INT                 ; SAI DA INTERRUPÇÃO

LIGA_BUZZER
   BSF     BUZZER                  ; LIGA O BUZZER
   GOTO    SAI_INT                 ; SAI DA INTERRUPÇÃO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                 TRATAMENTO DA INTERRUPÇÃO DE TIMER 0                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  TRATAMENTO DA INTERRUPÇÃO DE TIMER 0. RESPONSÁVEL POR CONVERTER OS PINOS
;  DOS BOTÕES EM ENTRADA, SALVAR A SITUAÇÃO DOS MESMOS NUMA VARIÁVEL
;  TEMPORÁRIA, CONVERTER NOVAMENTE OS PINOS PARA SAÍDA E ATUALIZAR OS LEDS.

INT_TMR0
   BCF     INTCON,T0IF             ; LIMPA FLAG DA INTERRUPÇÃO

; ********************** TESTA O ESTADO DOS BOTÕES ************************

TESTA_BT0
   BTFSS   BOTAO_0                 ; BOTÃO 0 PRESSIONADO?
   GOTO    BT0_LIB                 ; NÃO - TRATA BOTÃO COMO LIBERADO
                                   ; SIM
   DECFSZ  FILTRO_BT0,F            ; DECREMENTA FILTRO DO BOTÃO. ACABOU?
   GOTO    TESTA_BT1               ; NÃO - TESTA PRÓXIMO BOTÃO
   BSF     STATUS_BOTOES,0         ; SIM - MARCA BOTÃO COMO PRESSIONADO
   GOTO    TESTA_BT1               ; TESTA PRÓXIMO BOTÃO

BT0_LIB
```

```
   MOVLW   FILTRO_BOTAO
   MOVWF   FILTRO_BT0              ; REINICIALIZA FILTRO
   BCF     STATUS_BOTOES,0         ; MARCA BOTÃO COMO LIBERADO

TESTA_BT1
   BTFSS   BOTAO_1                 ; BOTÃO 1 PRESSIONADO?
   GOTO    BT1_LIB                 ; NÃO - TRATA BOTÃO COMO LIBERADO
                                   ; SIM
   DECFSZ  FILTRO_BT1,F            ; DECREMENTA FILTRO DO BOTÃO. ACABOU?
   GOTO    TESTA_BT2               ; NÃO - TESTA PRÓXIMO BOTÃO
   BSF     STATUS_BOTOES,1         ; SIM - MARCA BOTÃO COMO PRESSIONADO
   GOTO    TESTA_BT2               ; TESTA PRÓXIMO BOTÃO

BT1_LIB
   MOVLW   FILTRO_BOTAO
   MOVWF   FILTRO_BT1              ; REINICIALIZA FILTRO
   BCF     STATUS_BOTOES,1         ; MARCA BOTÃO COMO LIBERADO

TESTA_BT2
   BTFSS   BOTAO_2                 ; BOTÃO 2 PRESSIONADO?
   GOTO    BT2_LIB                 ; NÃO - TRATA BOTÃO COMO LIBERADO
                                   ; SIM
   DECFSZ  FILTRO_BT2,F            ; DECREMENTA FILTRO DO BOTÃO. ACABOU?
   GOTO    TESTA_BT3               ; NÃO - TESTA PRÓXIMO BOTÃO
   BSF     STATUS_BOTOES,2         ; SIM - MARCA BOTÃO COMO PRESSIONADO
   GOTO    TESTA_BT3               ; TESTA PRÓXIMO BOTÃO

BT2_LIB
   MOVLW   FILTRO_BOTAO
   MOVWF   FILTRO_BT2              ; REINICIALIZA FILTRO
   BCF     STATUS_BOTOES,2         ; MARCA BOTÃO COMO LIBERADO

TESTA_BT3
   BTFSS   BOTAO_3                 ; BOTÃO 3 PRESSIONADO?
   GOTO    BT3_LIB                 ; NÃO - TRATA BOTÃO COMO LIBERADO
                                   ; SIM
   DECFSZ  FILTRO_BT3,F            ; DECREMENTA FILTRO DO BOTÃO. ACABOU?
   GOTO    CONTINUA                ; NÃO - CONTINUA EXECUÇÃO DO PROGRAMA
   BSF     STATUS_BOTOES,3         ; SIM - MARCA BOTÃO COMO PRESSIONADO
   GOTO    CONTINUA                ; E CONTINUA EXECUÇÃO DO PROGRAMA

BT3_LIB
   MOVLW   FILTRO_BOTAO
   MOVWF   FILTRO_BT3              ; REINICIALIZA FILTRO
   BCF     STATUS_BOTOES,3         ; MARCA BOTÃO COMO LIBERADO

CONTINUA
   MOVF    STATUS_BOTOES,W
   MOVWF   LEDS                    ; ATUALIZA LEDS CONFORME BOTÕES

   MOVF    STATUS_BOTOES,F
   BTFSS   STATUS,Z                ; TODOS OS BOTÕES SOLTOS?
   GOTO    MUDA_FREQ               ; NÃO - DEVE ALT. A FREQ.
                                   ; SIM
   BANK1                           ; MUDA PARA BANK1
   MOVLW   .255
   MOVWF   PR2                     ; PR2 = 255
   BCF     PIE1,TMR2IE             ; DESLIGA INT. TIMER2
   BANK0                           ; MUDA PARA BANK0
   BCF     BUZZER                  ; GARANTE PINO DO BUZZER EM 0
   GOTO    SAI_INT                 ; SAI DA INTERRUPÇÃO

MUDA_FREQ
   CALL    ACERTA_FREQ             ; CHAMA TABELA DE FREQ.
   BANK1                           ; MUDA PARA BANK1
   MOVWF   PR2                     ; ACERTA VALOR DE PR2 CONFORME
                                   ; RETORNO DA TABELA
   BSF     PIE1,TMR2IE             ; LIGA INT. TIMER2
   BANK0                           ; MUDA PARA BANK0

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    FIM DA ROTINA DE INTERRUPÇÃO                         *
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  RESTAURAR OS VALORES DE "W" E "STATUS" ANTES DE RETORNAR.

SAI_INT
   SWAPF   STATUS_TEMP,W
   MOVWF   STATUS                  ; COPIA STATUS_TEMP PARA STATUS
   SWAPF   W_TEMP,F
   SWAPF   W_TEMP,W                ; COPIA W_TEMP PARA W
   RETFIE                          ; RETORNA DA INTERRUPÇÃO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                 TABELA DE ACERTO DA FREQUÊNCIA DO BUZZER                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

ACERTA_FREQ
   MOVF    STATUS_BOTOES,W         ; COLOCA STATUS_BOTOES EM W
   ADDWF   PCL,F                   ; SOMA STATUS_BOTOES AO PCL
                                   ; CRIANDO UMA SELEÇÃO TIPO "CASE"
                                   ; CONFORME TABELA ABAIXO:
                                   ; - -> LIBERARO
                                   ; X -> PRESSIONADO
                                   ; BT3   BT2   BT1   BT0
   RETLW   .255                    ;  -     -     -     -
   RETLW   .16                     ;  -     -     -     X
   RETLW   .32                     ;  -     -     X     -
   RETLW   .48                     ;  -     -     X     X
   RETLW   .64                     ;  -     X     -     -
   RETLW   .80                     ;  -     X     -     X
   RETLW   .96                     ;  -     X     X     -
   RETLW   .112                    ;  -     X     X     X
   RETLW   .128                    ;  X     -     -     -
   RETLW   .144                    ;  X     -     -     X
   RETLW   .160                    ;  X     -     X     -
   RETLW   .176                    ;  X     -     X     X
   RETLW   .192                    ;  X     X     -     -
   RETLW   .208                    ;  X     X     -     X
   RETLW   .224                    ;  X     X     X     -
   RETLW   .240                    ;  X     X     X     X

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               CONFIGURAÇÕES INICIAIS DE HARDWARE E SOFTWARE               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  NESTA ROTINA SÃO INICIALIZADAS AS PORTAS DE I/O DO MICROCONTROLADOR E AS
;  CONFIGURAÇÕES DOS REGISTRADORES ESPECIAIS (SFR). A ROTINA INICIALIZA A
;  MÁQUINA E AGUARDA O ESTOURO DO WDT.

CONFIG
   CLRF    PORTA                   ; LIMPA O PORTA
   CLRF    PORTB                   ; LIMPA O PORTB
   CLRF    PORTC                   ; LIMPA O PORTC
   CLRF    PORTD                   ; LIMPA O PORTD
   CLRF    PORTE                   ; LIMPA O PORTE


   BANK1                           ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                   ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                   ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011000'
   MOVWF   TRISC                   ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                   ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                   ; CONFIGURA I/O DO PORTE

   MOVLW   B'00000111'
   MOVWF   CMCON                   ; DESLIGA COMPARADORES ANALÓGICOS
```

```
   MOVLW   B'00000111'
   MOVWF   ADCON1                  ; DESLIGA CONVERSORES A/D

   MOVLW   B'11001111'
   MOVWF   OPTION_REG              ; CONFIGURA OPTIONS
                                   ; PULL-UPs DESABILITADOS
                                   ; INTER. NA BORDA DE SUBIDA DO RB0
                                   ; TIMER0 INCREM. PELO CICLO DE MÁQUINA
                                   ; WDT   - 1:128
                                   ; TIMER - 1:1
   MOVLW   B'01100000'
   MOVWF   INTCON                  ; CONFIGURA INTERRUPÇÕES
                                   ; HABILITA AS INT. DE TMR0 E PERIF.
   MOVLW   B'00000000'
   MOVWF   PIE1                    ; CONFIGURA INTERRUPÇÕES
                                   ; DESABILITA TODAS AS INT. DE PERIF.

   BANK0                           ; RETORNA PARA O BANCO 0

   MOVLW   B'00001111'
   MOVWF   T2CON                   ; TIMER2: PRESCALE  - 1:16
                                   ;         POSTSCALE - 1:2

;  AS INSTRUÇÕES A SEGUIR FAZEM COM QUE O PROGRAMA TRAVE QUANDO HOUVER UM
;  RESET OU POWER-UP, MAS PASSE DIRETO SE O RESET FOR POR WDT. DESTA FORMA,
;  SEMPRE QUE O PIC É LIGADO, O PROGRAMA TRAVA, AGUARDA UM ESTOURO DE WDT
;  E COMEÇA NOVAMENTE. ISTO EVITA PROBLEMAS NO START-UP DO PIC.

   BTFSC   STATUS,NOT_TO           ; RESET POR ESTOURO DE WDT?
   GOTO    $                       ; NÃO - AGUARDA ESTOURO DO WDT
                                   ; SIM

;  RECONFIGURA O VALOR DO OPTION_REG PARA ACERTAR O PRESCALE.

   BANK1                           ; SELECIONA BANCO 1 DA RAM
   MOVLW   B'11000010'
   MOVWF   OPTION_REG              ; RECONFIGURA OPTIONS
                                   ; PULL-UPs DESABILITADOS
                                   ; INTER. NA BORDA DE SUBIDA DO RB0
                                   ; TIMER0 INCR. PELO CICLO DE MÁQUINA
                                   ; WDT    - 1:1
                                   ; TIMER0 - 1:8
   BANK0                           ; SELECIONA BANCO 0 DA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INICIALIZAÇÃO DA RAM                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA IRÁ LIMPAR TODA A RAM DO BANCO 0, INDO DE 0X20 A 0X7F.
;  EM SEGUIDA, AS VARIÁVEIS DE RAM DO PROGRAMA SÃO INICIALIZADAS.

   MOVLW   0X20
   MOVWF   FSR                     ; APONTA O ENDEREÇAMENTO INDIRETO PARA
                                   ; A PRIMEIRA POSIÇÃO DA RAM
LIMPA_RAM
   CLRF    INDF                    ; LIMPA A POSIÇÃO
   INCF    FSR,F                   ; INCREMENTA PONTEIRO P/ A PRÓX. POS.
   MOVF    FSR,W
   XORLW   0X80                    ; COMPARA PONTEIRO COM A ÚLT. POS. +1
   BTFSS   STATUS,Z                ; JÁ LIMPOU TODAS AS POSIÇÕES?
   GOTO    LIMPA_RAM               ; NÃO - LIMPA A PRÓXIMA POSIÇÃO
                                   ; SIM

   MOVLW   FILTRO_BOTAO            ; INICIALIZA OS FILTROS DOS BOTÕES
   MOVWF   FILTRO_BT0
   MOVWF   FILTRO_BT1
   MOVWF   FILTRO_BT2
   MOVWF   FILTRO_BT3

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           LOOP PRINCIPAL                                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
;  ESTA ROTINA PRINCIPAL SIMPLESMENTE LIMPA O WDT, POIS TODA A LÓGICA DO
;  PROGRAMA É TRATADA DENTRO DAS INTERRUPÇÕES.

   BSF     C_LEDS                  ; ATIVA LEDS LIGADOS AO PORTD

   BSF     LINHA_4                 ; ATIVA BOTÕES DA LINHA 4

   BSF     INTCON,GIE              ; LIGA AS INTERRUPÇÕES

LOOP
   CLRWDT                          ; LIMPA WATCHDOG TIMER
   GOTO    LOOP                    ; VOLTA AO LOOP

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            FIM DO PROGRAMA                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                             ; FIM DO PROGRAMA
```

Dicas e Comentários

Notar que nesta experiência foi utilizada uma tabela para retornar em função do estado das teclas pressionadas o valor que deve ser carregado no registrador PR2. Ou seja, conforme comentado na experiência 4, a utilização de tabelas não se limita apenas à conversão de BCD para display de 7 segmentos. Prova disso é que nesta experiência a tabela foi útil para exercer função completamente diferente.

O software desta experiência apresenta uma particularidade. Veja que não existe nenhuma instrução, a não ser o CLRWDT, no loop principal do programa. Assim, todas as instruções do software e toda a sua lógica está implementada apenas dentro das interrupções, sendo parte na interrupção de TMR0 e parte na interrupção de TMR2.

Exercícios Propostos

1. Alterar a configuração do TMR2 para que o buzzer seja excitado em outra faixa de freqüências.
2. Utilizar a instrução boleana XOR para inverter o estado do pino do buzzer.
3. Inverter a escala de freqüências, trocando a mais alta pela mais baixa.

Capítulo 11 - Experiência 9 – Varredura de displays e utilização do TIMER 1

Objetivo

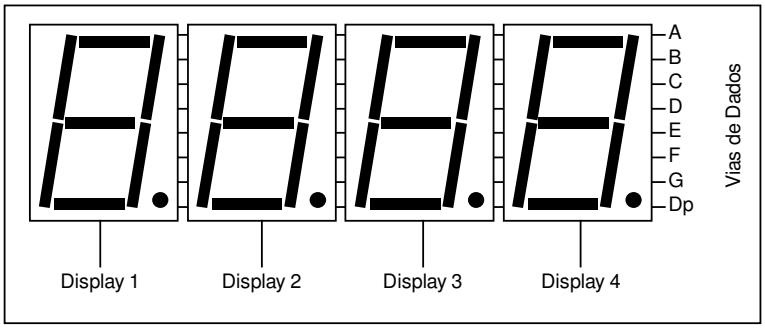
O objetivo desta experiência é o aprendizado do sistema de varredura comumente utilizado para varrer displays de 7 segmentos. Além disso, é visto o TMR1 fechando assim o estudo sobre os três times do PIC16F877A.

Descrição

Cada display é formado por um conjunto de 8 leds, sendo 7 para a caracterização do dígito e 1 para o ponto decimal. Desta forma, precisaremos de um pino do PIC para controlar cada um dos leds e, portanto, serão necessários 8 pinos para acionar os 7 segmentos e mais o ponto decimal. Porém, o MCMASTER não possui apenas um display e sim quatro. Seriam necessários então 32 pinos do microcontrolador para acionar os quatro displays? Não, existe uma saída para isso. O segredo para minimizar a quantidade de pinos utilizados é o emprego de um conceito denominado varredura.

Para isso, interligam-se todos os displays, juntando todos os pinos de um mesmo segmento numa única via, de forma a criar um barramento de dados com as vias de A até Dp. Em seguida, utiliza-se um pino para controlar o comum de cada um dos displays (total de 4 pinos). Assim, quando se deseja escrever em um dos displays, bastar informar os segmentos a serem acionados nas vias de dados e ligar o comum do display desejado.

Utilizando o hardware desta forma é fácil notar que apenas um dos displays poderá ser acionado de cada vez. Porém, se acionarmos os displays continuamente, um após o outro e de forma rápida, nossos olhos não conseguiram perceber que apenas um display está acionado por vez dando a impressão de que todos os displays estão acionados o tempo todo.



Assim, empregando-se a técnica de varredura, consegue-se controlar os 4 displays utilizando apenas 12 pinos do microcontrolador.

O exemplo desenvolvido para esta experiência faz muito mais que simplesmente implementar a varredura dos displays. Trata-se de um contador regressivo de segundos, ou seja, um temporizador capaz de contar até 9.999 segundos.

As teclas habilitadas são as da linha 4 e seguem as funções descritas na tabela.

Coluna	Descrição

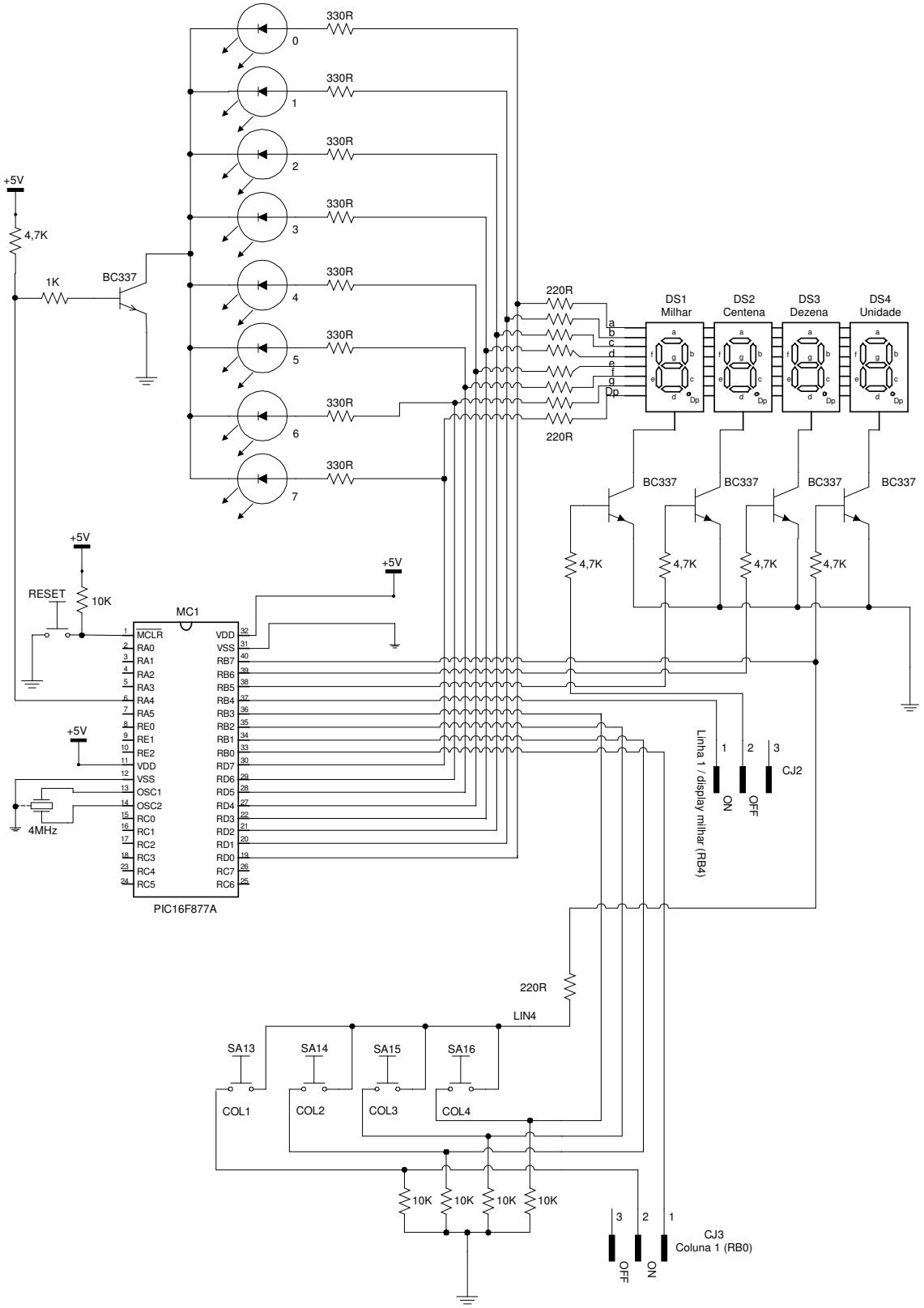
1	nenhuma função
2	Incrementa o valor inicial em 1 segundo
3	Decrementa o valor inicial em 1 segundo
4	Inicia e paralisa o temporizador

Para a contagem do tempo utilizou-se a interrupção de TMR1, configurada conforme a tabela a seguir.

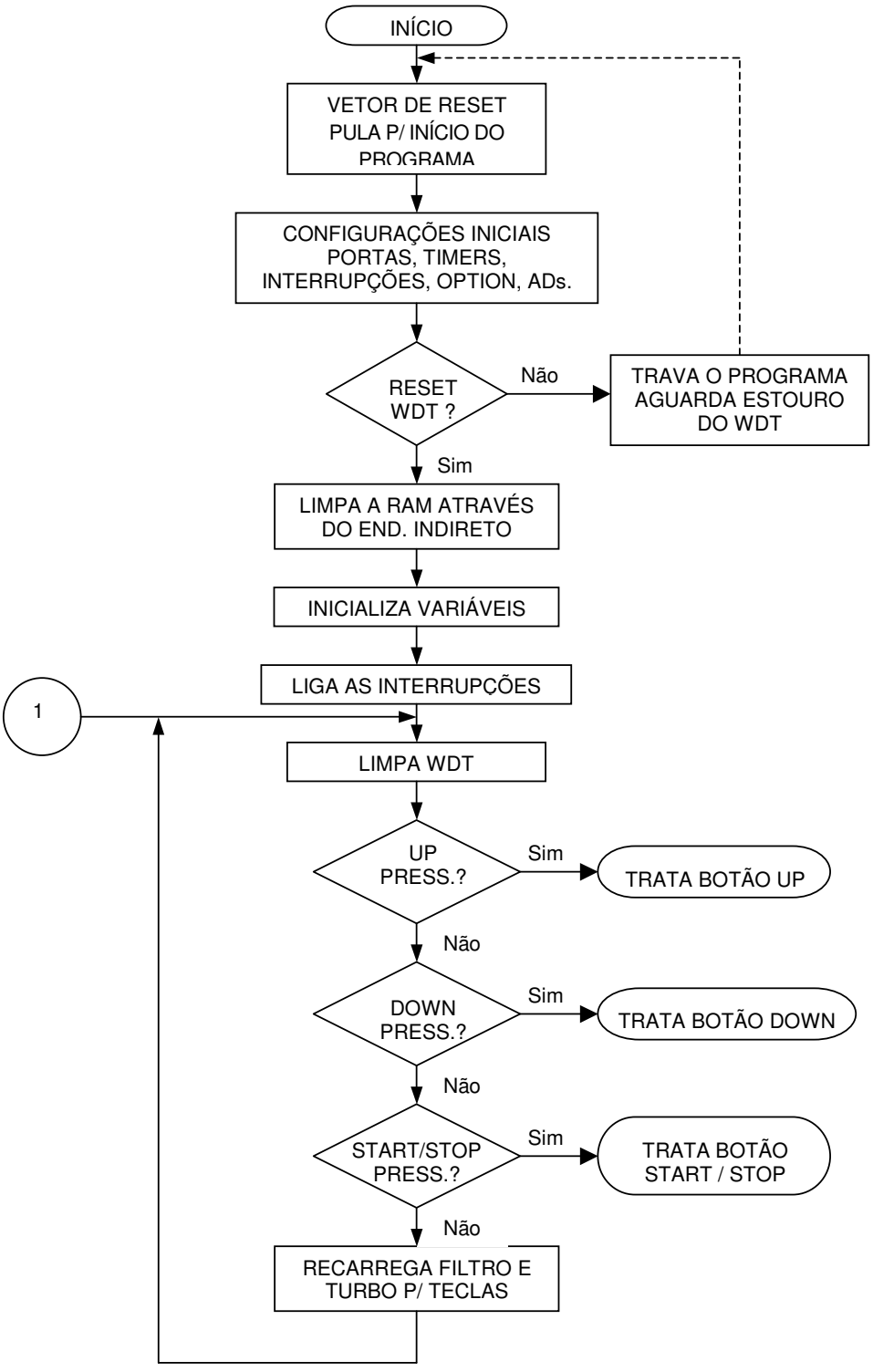
Ciclo de Maq.	***Prescale***	**Conta TMR1**	**Auxiliar**	**Período**	**Freqüência**
1 µs	8	62500	2	1.000.000 µs	1 Hz

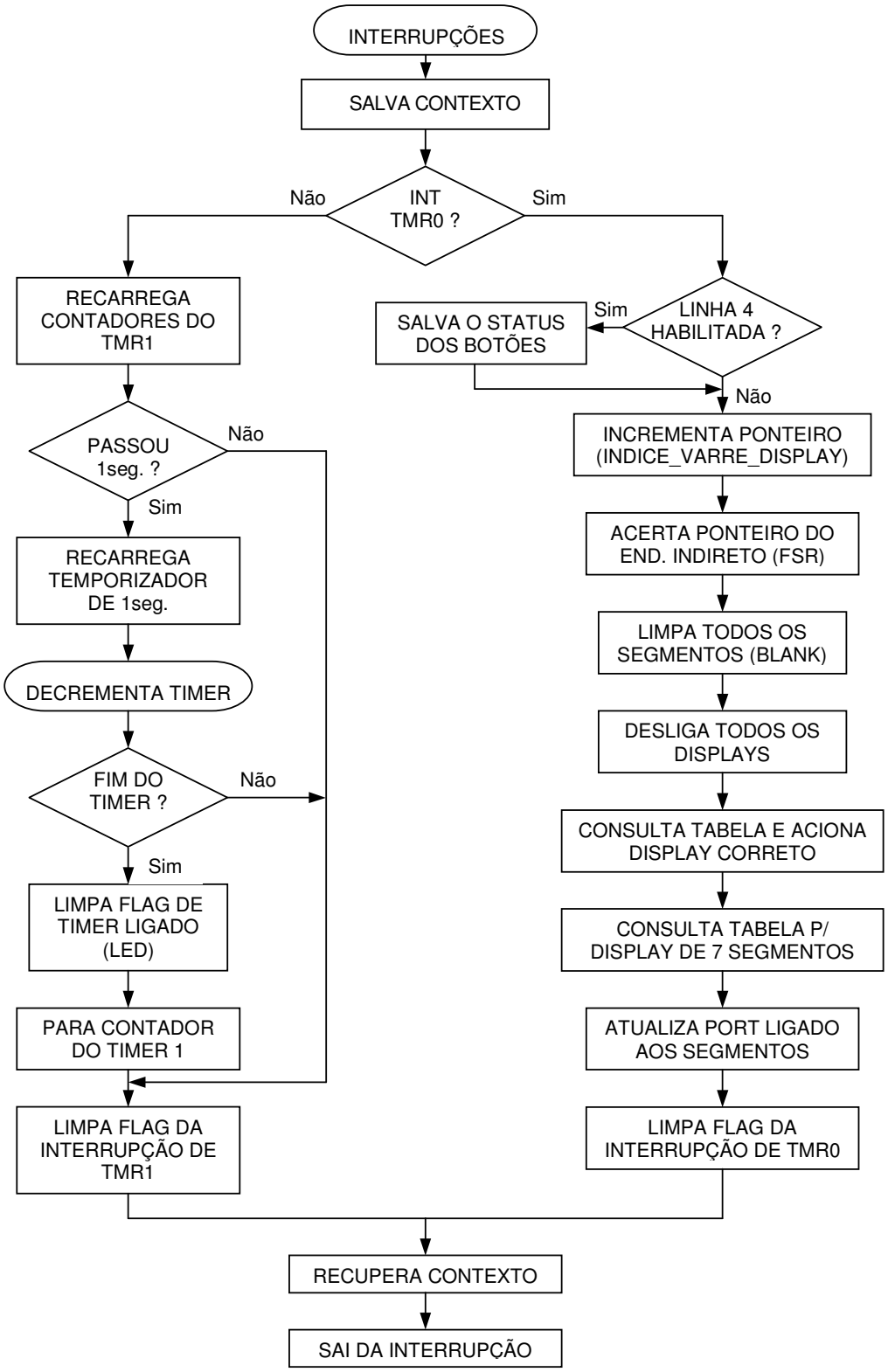
Configurou-se o *prescale* do TMR1 em 1:8 e o contador foi inicializado com o valor total menos o desejado para a contagem (65.536 – 62.500). Desta maneira a interrupção ocorre a cada 0,5 segundo. A fim de criar a base de tempo de 1 segundo foi utilizada uma variável auxiliar que é decrementada a cada ocorrência da interrupção.

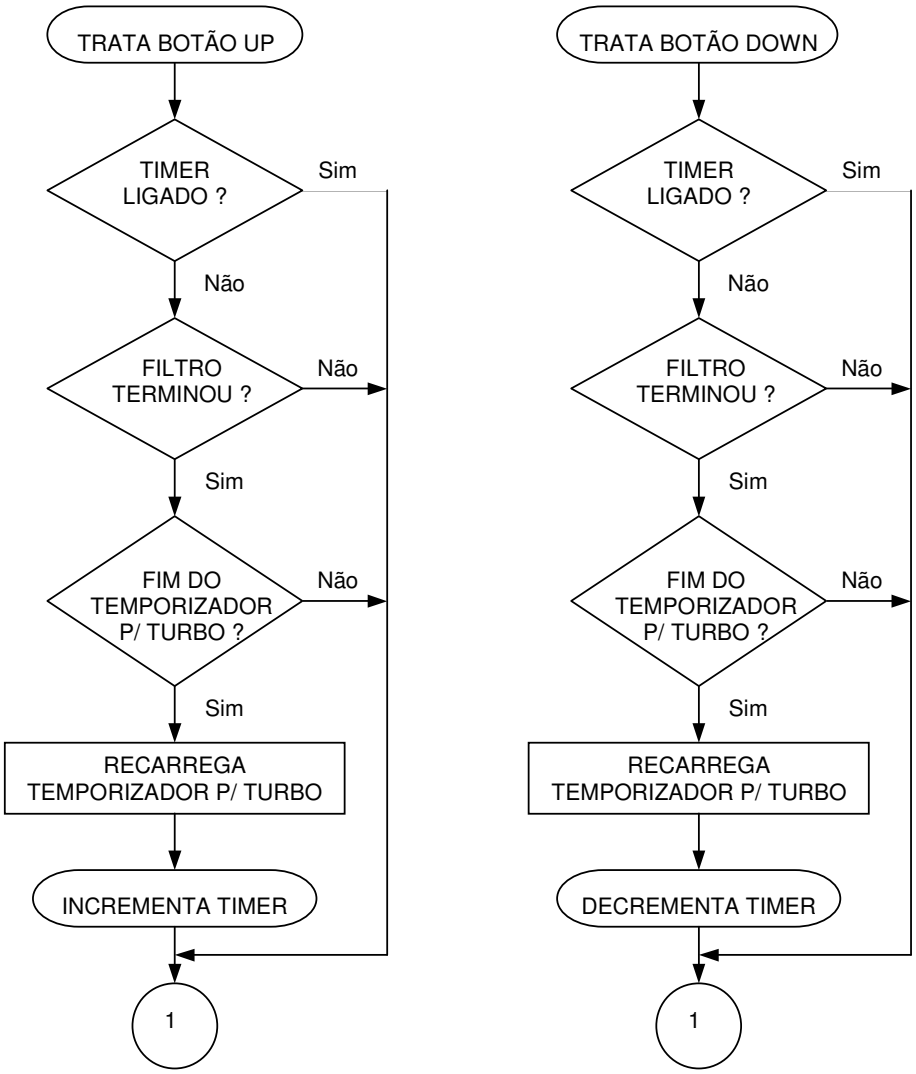
Esquema Elétrico

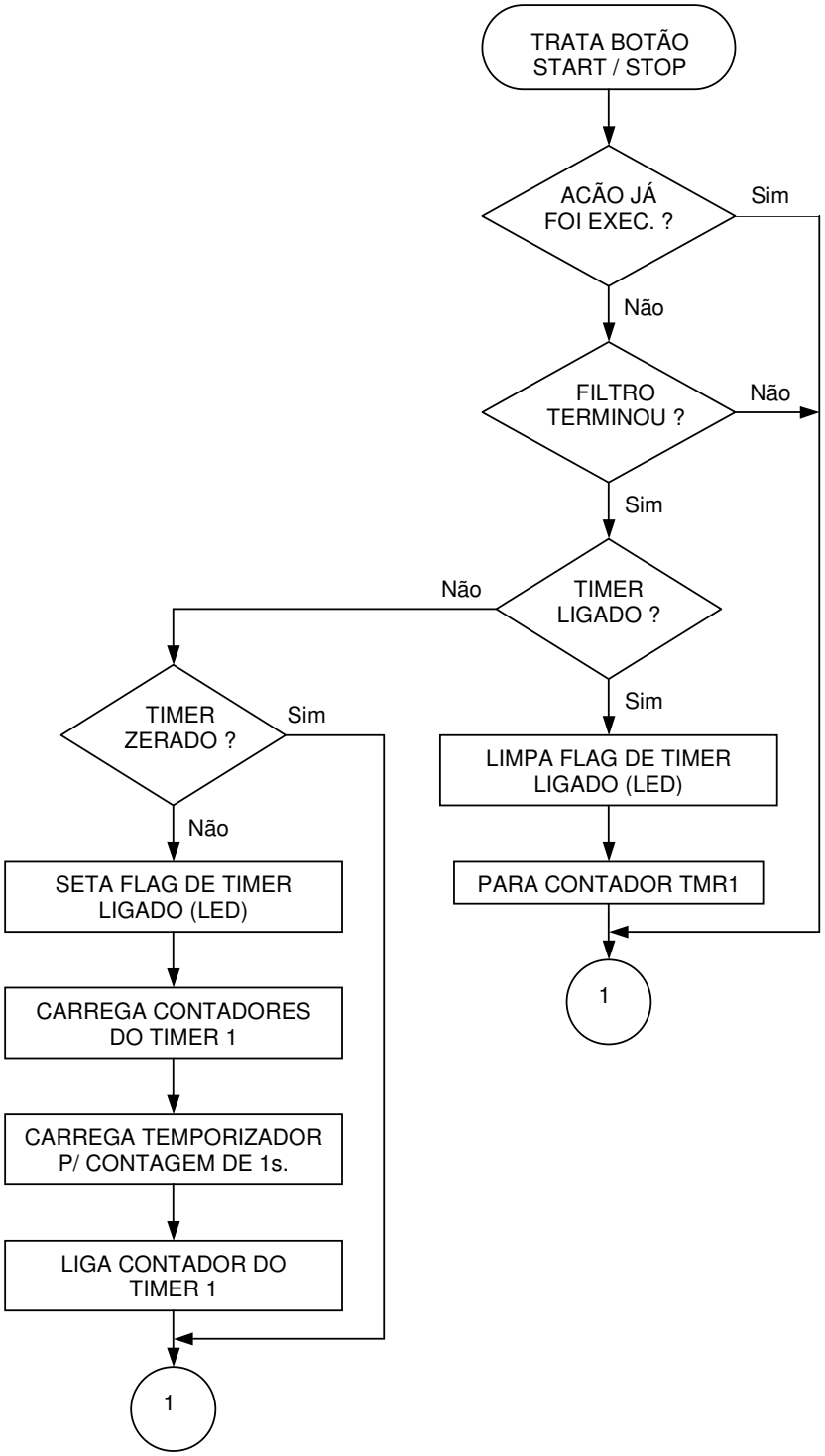


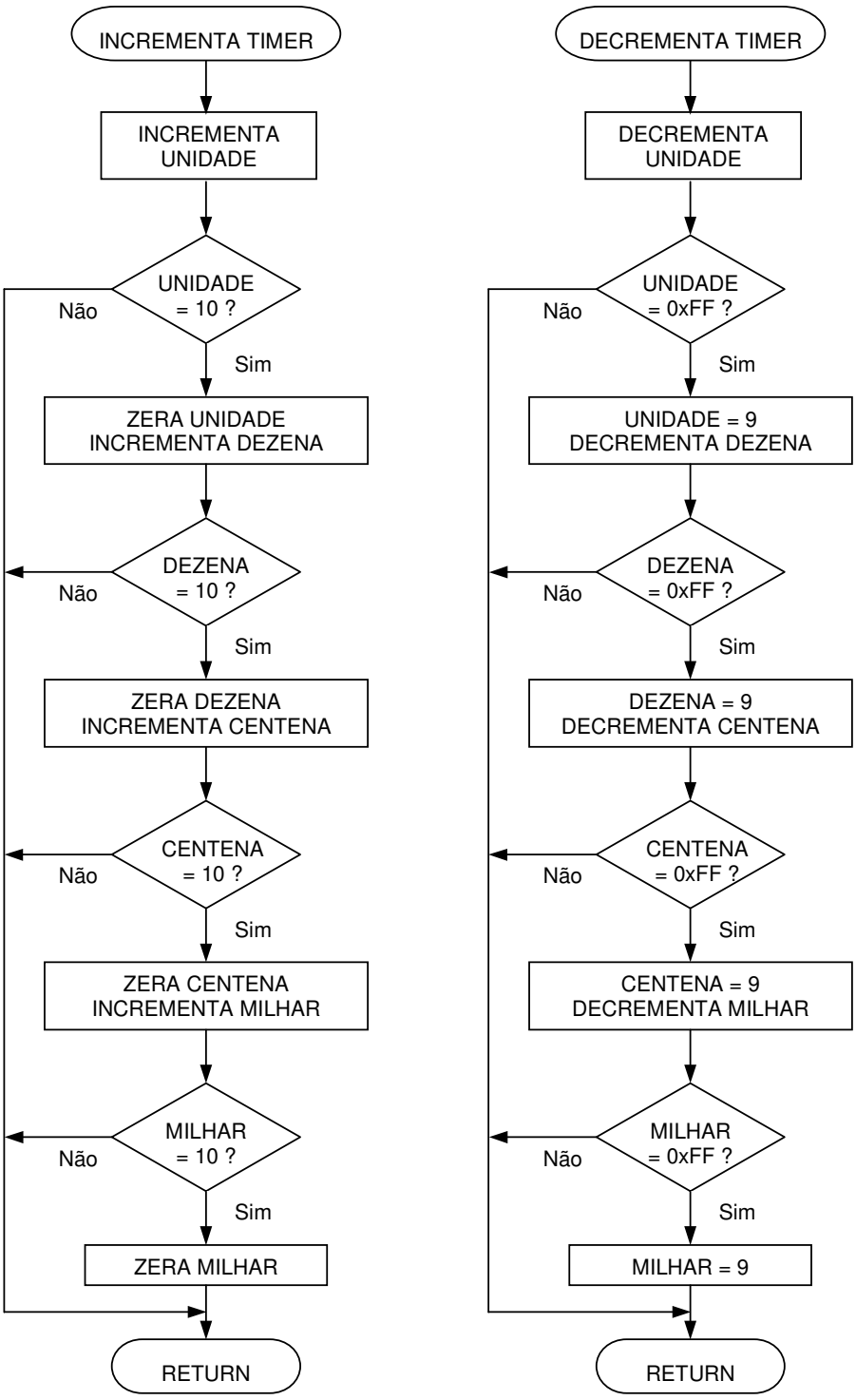
Fluxograma


INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURAÇÕES INICIAIS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION, ADs.
RESET
WDT ?
Não
TRAVA O PROGRAMA
AGUARDA ESTOURO
DO WDT
Sim
LIMPA A RAM ATRAVÉS
DO END. INDIRETO
INICIALIZA VARIÁVEIS
LIGA AS INTERRUPÇÕES
1
LIMPA WDT
UP
PRESS.?
Sim
TRATA BOTÃO UP
Não
DOWN
PRESS.?
Sim
TRATA BOTÃO DOWN
Não
START/STOP
PRESS.?
Sim
TRATA BOTÃO
START / STOP
Não
RECARREGA FILTRO E
TURBO P/ TECLAS


INTERRUPÇÕES
SALVA CONTEXTO
INT TMR0 ?
Não
Sim
RECARREGA CONTADORES DO TMR1
PASSOU 1seg. ?
Não
Sim
RECARREGA TEMPORIZADOR DE 1seg.
DECREMENTA TIMER
FIM DO TIMER ?
Não
Sim
LIMPA FLAG DE TIMER LIGADO (LED)
PARA CONTADOR DO TIMER 1
LIMPA FLAG DA INTERRUPÇÃO DE TMR1
SALVA O STATUS DOS BOTÕES
Sim
LINHA 4 HABILITADA ?
Não
INCREMENTA PONTEIRO (INDICE_VARRE_DISPLAY)
ACERTA PONTEIRO DO END. INDIRETO (FSR)
LIMPA TODOS OS SEGMENTOS (BLANK)
DESLIGA TODOS OS DISPLAYS
CONSULTA TABELA E ACIONA DISPLAY CORRETO
CONSULTA TABELA P/ DISPLAY DE 7 SEGMENTOS
ATUALIZA PORT LIGADO AOS SEGMENTOS
LIMPA FLAG DA INTERRUPÇÃO DE TMR0
RECUPERA CONTEXTO
SAI DA INTERRUPÇÃO


TRATA BOTÃO UP
TIMER LIGADO ?
Sim
Não
FILTRO TERMINOU ?
Não
Sim
FIM DO TEMPORIZADOR P/ TURBO ?
Não
Sim
RECARREGA TEMPORIZADOR P/ TURBO
INCREMENTA TIMER
1
TRATA BOTÃO DOWN
TIMER LIGADO ?
Sim
Não
FILTRO TERMINOU ?
Não
Sim
FIM DO TEMPORIZADOR P/ TURBO ?
Não
Sim
RECARREGA TEMPORIZADOR P/ TURBO
DECREMENTA TIMER
1


TRATA BOTÃO
START / STOP
ACÃO JÁ
FOI EXEC. ?
Sim
Não
FILTRO
TERMINOU ?
Não
Sim
TIMER
LIGADO ?
Não
Sim
TIMER
ZERADO ?
Sim
Não
LIMPA FLAG DE TIMER
LIGADO (LED)
PARA CONTADOR TMR1
SETA FLAG DE TIMER
LIGADO (LED)
CARREGA CONTADORES
DO TIMER 1
CARREGA TEMPORIZADOR
P/ CONTAGEM DE 1s.
LIGA CONTADOR DO
TIMER 1
1
1


INCREMENTA TIMER
INCREMENTA UNIDADE
UNIDADE = 10 ?
Não
Sim
ZERA UNIDADE INCREMENTA DEZENA
DEZENA = 10 ?
Não
Sim
ZERA DEZENA INCREMENTA CENTENA
CENTENA = 10 ?
Não
Sim
ZERA CENTENA INCREMENTA MILHAR
MILHAR = 10 ?
Não
Sim
ZERA MILHAR
RETURN
DECREMENTA TIMER
DECREMENTA UNIDADE
UNIDADE = 0xFF ?
Não
Sim
UNIDADE = 9 DECREMENTA DEZENA
DEZENA = 0xFF ?
Não
Sim
DEZENA = 9 DECREMENTA CENTENA
CENTENA = 0xFF ?
Não
Sim
CENTENA = 9 DECREMENTA MILHAR
MILHAR = 0xFF ?
Não
Sim
MILHAR = 9
RETURN

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *       EXPERIÊNCIA 9 - VARREDURA DE DISPLAYS E UTILIZAÇÃO DO TIMER 1     *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            DESCRIÇÃO GERAL                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTE EXEMPLO FOI PREPARADO PARA DEMONSTRAR O FUNCIONAMENTO DO TIMER DE
; 16 BITS DO PIC (TMR1) E DA VARREDURA DE DISPLAYS.
; CONSISTE NUM TEMPORIZADOR DE SEGUNDOS. DOIS BOTÕES FORAM UTILIZADOS PARA
; PROGRAMAR O TEMPO DA CONTAGEM. UM OUTRO BOTÃO FOI UTILIZADO PARA DISPARAR
; E PARALIZAR O CONTADOR. O TEMPORIZADOR CONSEGUE CONTAR ATÉ 9999 SEGUNDOS,
; DE FORMA QUE OS 4 DISPLAYS DE 7 SEGMENTOS FORAM NECESSÁRIOS.
; A CONTAGEM É REGRESSIVA.
; UM LED INDICA QUE O TEMPORIZADOR ESTÁ OPERANDO. QUANDO O SISTEMA CHEGA
; A 0000 (ZERO) O LED É DESLIGADO AUTOMATICAMENTE.
; NESTE EXEMPLO APENAS OS BOTÕES DA LINHA 4 PODEM SER UTILIZADOS.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        CONFIGURAÇÕES PARA GRAVAÇÃO                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_ON & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           DEFINIÇÃO DAS VARIÁVEIS                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O PRIMEIRO BLOCO DE VARIÁVEIS ESTÁ LOCALIZADO LOGO NO INÍCIO DO BANCO 0

   CBLOCK  0X20                    ; POSIÇÃO INICIAL DA RAM

           UNIDADE                 ; (LSD)
           DEZENA                  ;
           CENTENA                 ;
           MILHAR                  ; (MSD)

           LEDS

           STATUS_BOTOES           ; ARMAZENA O STATUS DOS BOTOES

           FILTRO_BOTOES           ; FILTRO PARA RUIDOS

           TEMPO_TURBO             ; TEMPORIZADOR P/ TURBO DAS TECLAS

           INDICE_VARRE_DISPLAY    ; INDEXADOR P/ VARREDURA DOS DISPLAYS

           DIVISOR_TMR1            ; CONTADOR AUXILIAR P/ SEGUNDOS

   ENDC

;  O SEGUNDO BLOCO DE VARIÁVEIS ESTÁ LOCALIZADO NO FINAL DO BANCO 0, A PARTIR
;  DO ENDEREÇO 0X70, POIS ESTA LOCALIZAÇÃO É ACESSADA DE QUALQUER BANCO,
;  FACILITANDO A OPERAÇÃO COM AS VARIÁVEIS AQUI LOCALIZADAS.

   CBLOCK  0X70                    ; REGIÃO COMUM A TODOS OS BANCOS

           STATUS_TEMP             ; REGISTRADOR DE STATUS TEMPORÁRIO
           WORK_TEMP               ; REGISTRADOR DE TRABALHO TEMPORÁRIO
           FSR_TEMP                ; REG. DE ENDERECO INDIRETO TEMPORÁRIO
           PCLATH_TEMP             ; REGISTRADOR DE PAGINAÇÃO TEMPORÁRIO
   ENDC
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>        ; ARQUIVO DE DEFINIÇÕES DO PIC ATUAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    DEFINIÇÃO DOS BANCOS DE RAM                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           CONSTANTES INTERNAS                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE CONSTANTES FACILITA A PROGRAMAÇÃO E A MANUTENÇÃO.

FILTRO_TECLA       EQU     .200            ; FILTRO P/ EVITAR RUIDOS DOS BOTÕES
TURBO_TECLA        EQU     .70             ; TEMPORIZADOR P/ TURBO DAS TECLAS

TMR1_HIGH  EQU     HIGH (.65536-.62500)
TMR1_LOW   EQU     LOW  (.65536-.62500)    ; VALOR PARA CONTAGEM DE
                                           ; 62500 CICLOS DE CONTAGEM
                                           ; DO TMR1 (PROGRAMADO P/
                                           ; PRESCALER DE 1:8)

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  DECLARAÇÃO DOS FLAGs DE SOFTWARE                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE FLAGs AJUDA NA PROGRAMAÇÃO E ECONOMIZA MEMÓRIA RAM.

;  ESTE PROGRAMA NÃO UTILIZA NENHUM FLAG DE USUÁRIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               ENTRADAS                               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS ENTRADAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    BT_UP           STATUS_BOTOES,1 ; ESTADO DO BOTÃO 1
                                           ; 0 -> LIBERADO
                                           ; 1 -> PRESSIONADO

#DEFINE    BT_DOWN         STATUS_BOTOES,2 ; ESTADO DO BOTÃO 2
                                           ; 0 -> LIBERADO
                                           ; 1 -> PRESSIONADO

#DEFINE    BT_START_STOP   STATUS_BOTOES,3 ; ESTADO DO BOTÃO 3
                                           ; 0 -> LIBERADO
                                           ; 1 -> PRESSIONADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               SAÍDAS                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS SAÍDAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    ESTADO_TIMER    LEDS,0  ; LED DE ESTADO DO TIMER
                                   ; (FUNCIONA TAMBÉM COMO FLAG)
                                   ; 1 -> TIMER CONTANDO
                                   ; 0 -> TIMER PARADO

#DEFINE    MUX             PORTB   ; MUX PARA ACIONAMENTO DOS DISPLAYS
                                   ; (DE RB4 ATÉ RB7)
```

```
#DEFINE    C_LEDS          PORTA,4 ; PINO PARA ATIVAR GRUPO DE 8 LEDS
                                   ; 1 -> LEDS ATIVADOS
                                   ; 0 -> LEDS DESATIVADOS

#DEFINE    SEGMENTOS       PORTD   ; SEGMENTOS DOS DISPLAYS

#DEFINE    LINHA_4         PORTB,7 ; PINO P/ ATIVAR LINHA 4 (TECLADO MATRICIAL)
                                   ; 0 -> LINHA 4 ATIVADA
                                   ; 1 -> LINHA 4 DESATIVADA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  VETOR DE RESET DO MICROCONTROLADOR                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO INICIAL PARA EXECUÇÃO DO PROGRAMA

   ORG     0X0000                  ; ENDEREÇO DO VETOR DE RESET
   GOTO    CONFIG                  ; PULA PARA CONFIG DEVIDO A REGIÃO
                                   ; DESTINADA ÀS INTERRUPÇÕES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           VETOR DE INTERRUPÇÃO                            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO DE DESVIO DO PROGRAMA QUANDO UMA INTERRUPÇÃO ACONTECE

   ORG     0X0004                  ; ENDEREÇO DO VETOR DE INTERRUPÇÃO

;  É MUITO IMPORTANTE QUE OS REGISTRADORES PRIORITÁRIOS AO FUNCIONAMENTO DA
;  MÁQUINA, E QUE PODEM SER ALTERADOS TANTO DENTRO QUANTO FORA DAS INTS SEJAM
;  SALVOS EM REGISTRADORES TEMPORÁRIOS PARA PODEREM SER POSTERIORMENTE
;  RECUPERADOS.

SALVA_CONTEXTO
   MOVWF   WORK_TEMP               ; SALVA REGISTRADOR DE TRABALHO E
   SWAPF   STATUS,W                ; DE STATUS DURANTE O TRATAMENTO
   MOVWF   STATUS_TEMP             ; DA INTERRUPÇÃO.
   MOVF    FSR,W
   MOVWF   FSR_TEMP                ; SALVA REGISTRADOR FSR
   MOVF    PCLATH,W
   MOVWF   PCLATH_TEMP             ; SALVA REGISTRADOR PCLATH

   CLRF    PCLATH                  ; LIMPA REGISTRADOR PCLATH
                                   ; (SELECIONA PÁGINA 0)
   CLRF    STATUS                  ; LIMPA REGISTRADOR STATUS
                                   ; (SELECIONA BANCO 0)

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                 TESTA QUAL INTERRUPÇÃO FOI SOLICITADA                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  TESTA O FLAG DAS INTERRUPÇÕES PARA SABER PARA QUAL ROTINA DESVIAR.

   BTFSS   INTCON,T0IF             ; FOI INTERRUPÇÃO DE TMR0 ?
   GOTO    INT_TMR1                ; NÃO - ENTÃO PULA P/ INT_TMR1
                                   ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    TRATAMENTO DA INTERRUPÇÃO DE TIMER 0                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ROTINA PARA EXECUTAR AS AÇÕES NECESSÁRIAS SEMPRE QUE A INTERRUPÇÃO
;  ACONTECE. NESTE CASO, A INTERRUPÇÃO ESTA SENDO UTILIZADA PARA GERAR A
;  FREQÜÊNCIA DE VARREDURA DOS DISPLAYS. POR ISSO, CADA VEZ QUE ELA ACONTECER,
;  O PRÓXIMO DISPLAY SERÁ ACIONADO.

INT_TMR0
   BTFSS   LINHA_4                 ; ESTA VARRENDO A LINHA 4 ?
   GOTO    VARRE_DISPLAY           ; NAO - PULA P/ VARRE_DISPLAY
                                   ; SIM
   MOVF    PORTB,W
   MOVWF   STATUS_BOTOES           ; SALVA O STATUS DOS BOTOES

VARRE_DISPLAY
   INCF    INDICE_VARRE_DISPLAY,F  ; INCR. O ÍNDICE DE VAR. DOS DISPLAYS
```

```
   MOVLW   .5
   XORWF   INDICE_VARRE_DISPLAY,W  ; LIMITA CONTAGEM DE 0 A 4
   BTFSC   STATUS,Z                ; INDICE_VARRE_DISPLAY = 5 ?
   CLRF    INDICE_VARRE_DISPLAY    ; SIM - LIMPA CONTADOR
                                   ; NÃO

   MOVF    INDICE_VARRE_DISPLAY,W  ; CARREGA NO WORK O VALOR DO ÍNDICE
   ADDLW   UNIDADE                 ; SOMA ENDEREÇO DO PRIMEIRO DÍGITO
   MOVWF   FSR                     ; SALVA RESULTADO NO FSR, APONTANDO
                                   ; PARA O ENDEREÇO DO DÍGITO ATUAL.
                                   ; (ENDEREÇAMENTO INDIRETO)

   CLRF    SEGMENTOS               ; LIMPA OS SEGMENTOS (BLANK)
                                   ; UTILIZADO P/ EVITAR SOMBRAS NOS
                                   ; DISPLAYS

   MOVLW   B'00001111'             ; PREPARA MÁSCARA
   ANDWF   MUX,F                   ; EXECUTA MÁSCARA (DESLIGA OS DISP.)
   BCF     C_LEDS                  ; DESLIGA O COMUM DOS LEDS

   MOVLW   .4
   XORWF   INDICE_VARRE_DISPLAY,W
   BTFSS   STATUS,Z                ; DEVE ACENDER LEDS ?
   GOTO    ACENDE_DISPLAYS         ; NAO - PULA P/ ACENDE_DISPLAYS
                                   ; SIM
ACENDE_LEDS
   BSF     C_LEDS                  ; HABILITA GRUPO DE LEDS

   GOTO    $+1                     ; DELAY DE 2US

   MOVF    LEDS,W
   MOVWF   SEGMENTOS               ; ATUALIZA OS SEGMENTOS COM
                                   ;  O VALOR DE LEDS

   GOTO    SAI_INT_TMR0            ; PULA P/ SAI_INT_TMR0

ACENDE_DISPLAYS
   MOVF    INDICE_VARRE_DISPLAY,W  ; SALVA NO WORK O VALOR DO ÍNDICE
   CALL    TABELA_MUX              ; CONSULTA TABELA MUX
   IORWF   MUX,F                   ; ATUALIZA MUX, SELECIONANDO O
                                   ; DISPLAYS CORRETO PARA O MOMENTO
   GOTO    $+1                     ; DELAY DE 2US
                                   ; (TEMPO DE RESPOSTA DO TRANSISTOR)

   MOVF    INDF,W                  ; RECUPERA NO WORK O VALOR DO DÍGITO
   CALL    TABELA_DISPLAY_7_SEG    ; CONSULTA TABELA P/ DISPLAYS
   MOVWF   SEGMENTOS               ; ATUALIZA OS SEGMENTOS, ESCREVENDO
                                   ; O VALOR DO DÍGITO CORRETO (PORTD)

SAI_INT_TMR0
   BCF     INTCON,T0IF             ; LIMPA FLAG DA INTERRUPÇÃO DE TMR0
   GOTO    SAI_INT                 ; PULA P/ SAI_INT

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   TRATAMENTO DA INTERRUPÇÃO DE TIMER 1                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ROTINA PARA EXECUTAR AS AÇÕES NECESSÁRIAS SEMPRE QUE A INTERRUPÇÃO
;  ACONTECE. NESTE CASO, A INTERRUPÇÃO ESTA SENDO UTILIZADA PARA CONTAR O
;  TEMPO DO TEMPORIZADOR. POR ISSO, CADA VEZ QUE ELA ACONTECER O VALOR DO
;  TIMER SERÁ DECREMENTADO, CASO JÁ TENHA SE PASSADO 1SEG.
;  PERIODO DA INTERRUPÇÃO: 1US (CICLO DE MAQUINA) * 8 (PRESCALER DO TMR1) *
;                    * 62500 (CONTAGEM DO TMR1) = 0,5SEG.

INT_TMR1
   MOVLW   TMR1_HIGH
   MOVWF   TMR1H
   MOVLW   TMR1_LOW
   MOVWF   TMR1L                   ; RECARREGA CONTADOR DO TMR1
                                   ; PERIODICIDADE DE 0,5SEG.

   DECFSZ  DIVISOR_TMR1,F          ; PASSOU-SE 1 SEGUNDO ?
   GOTO    SAI_INT_TMR1            ; NÃO - ENTÃO SAI DA INTERRUPÇÃO
```

```
                                ; SIM
   MOVLW   .2
   MOVWF   DIVISOR_TMR1         ; RECARREGA CONTADOR DE 1SEG.

   CALL    DECREMENTA_TIMER     ; DECREMENTA O VALOR DO TIMER

   MOVF    UNIDADE,F
   BTFSS   STATUS,Z
   GOTO    SAI_INT_TMR1
   MOVF    DEZENA,F
   BTFSS   STATUS,Z
   GOTO    SAI_INT_TMR1
   MOVF    CENTENA,F
   BTFSS   STATUS,Z
   GOTO    SAI_INT_TMR1
   MOVF    MILHAR,F
   BTFSS   STATUS,Z             ; FINAL DA CONTAGEM ? (TIMER=0?)
   GOTO    SAI_INT_TMR1         ; NÃO - SAI DA INTERRUPÇÃO
                                ; SIM
   BCF     ESTADO_TIMER         ; DESLIGA LED DE TIMER OPERANDO
   BCF     T1CON,TMR1ON         ; PARALIZA CONTADOR DO TMR1

SAI_INT_TMR1
   BCF     PIR1,TMR1IF          ; LIMPA FLAG DA INTERRUPÇÃO DE TMR1

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          SAÍDA DA INTERRUPÇÃO                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ANTES DE SAIR DA INTERRUPÇÃO, O CONTEXTO SALVO NO INÍCIO DEVE SER
;  RECUPERADO PARA QUE O PROGRAMA NÃO SOFRA ALTERAÇÕES INDESEJADAS.

SAI_INT
   MOVF    PCLATH_TEMP,W
   MOVWF   PCLATH               ; RECUPERA REG. PCLATH (PAGINAÇÃO)
   MOVF    FSR_TEMP,W
   MOVWF   FSR                  ; RECUPERA REG. FSR (END. INDIRETO)
   SWAPF   STATUS_TEMP,W
   MOVWF   STATUS               ; RECUPERA REG. STATUS
   SWAPF   WORK_TEMP,F
   SWAPF   WORK_TEMP,W          ; RECUPERA REG. WORK
   RETFIE                       ; RETORNA DA INTERRUPÇÃO (HABILITA GIE)

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                TABELA PARA OS DISPLAYS DE 7 SEGMENTOS                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ROTINA PARA CONVERSÃO DO VALOR NÚMÉRICO DO DÍGITO EM RELAÇÃO AOS SEGMENTOS
;  QUE DEVEM SER ACESOS E APAGADOS NO DISPLAY

TABELA_DISPLAY_7_SEG
   ANDLW   B'00001111'          ; EXECUTA MASCARA P/ EVITAR PULOS ERRADOS
   ADDWF   PCL,F                ; SOMA DESLOCAMENTO AO PROGRAM COUNTER,
                                ; GERANDO UMA TABELA DO TIPO "CASE".
;            PGFEDCBA           ; POSIÇÃO RELATIVA AOS SEGMENTOS
   RETLW   B'00111111'          ; 0H - 0
   RETLW   B'00000110'          ; 1H - 1
   RETLW   B'01011011'          ; 2H - 2
   RETLW   B'01001111'          ; 3H - 3
   RETLW   B'01100110'          ; 4H - 4
   RETLW   B'01101101'          ; 5H - 5
   RETLW   B'01111101'          ; 6H - 6
   RETLW   B'00000111'          ; 7H - 7
   RETLW   B'01111111'          ; 8H - 8
   RETLW   B'01101111'          ; 9H - 9
   RETLW   B'00000000'          ; AH - BLANK
   RETLW   B'00000000'          ; BH - BLANK
   RETLW   B'00000000'          ; CH - BLANK
   RETLW   B'00000000'          ; DH - BLANK
   RETLW   B'00000000'          ; EH - BLANK
   RETLW   B'00000000'          ; FH - BLANK

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                TABELA PARA ACIONAMENTO DOS DISPLAYS                     *
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ROTINA PARA CONVERTER O DÍGITO ATUAL EM RELAÇÃO AO PORT QUE DEVE SER
;  LIGADO PARA ACIONAMENTO DO DISPLAY RELACIONADO.

TABELA_MUX
   ADDWF   PCL,F                  ; SOMA DESLOCAMENTO AO PROGRAM COUNTER
                                  ; GERANDO UMA TABELA DO TIPO "CASE".
   RETLW   B'10000000'            ; 0 - ACIONA DISPLAY 0
   RETLW   B'01000000'            ; 1 - ACIONA DISPLAY 1
   RETLW   B'00100000'            ; 2 - ACIONA DISPLAY 2
   RETLW   B'00010000'            ; 2 - ACIONA DISPLAY 3

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *             ROTINA PARA INCREMENTAR O VALOR DO TIMER (BCD)             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ROTINA UTILIZADA PARA INCREMENTAR O VALOR DOS REGISTRADORES UNIDADE,
;  DEZENA, CENTENA E MILHAR, QUE SÃO OS CONTADORES DO TIMER. A CONTAGEM É
;  FEITA DIRETAMENTE EM BCD.

INCREMENTA_TIMER
   INCF    UNIDADE,F              ; INCREMENTA UNIDADE

   MOVLW   .10
   XORWF   UNIDADE,W
   BTFSS   STATUS,Z               ; UNIDADE = 10 ?
   RETURN                         ; NÃO - RETORNA
                                  ; SIM
   CLRF    UNIDADE                ; ZERA A UNIDADE
   INCF    DEZENA,F               ; INCREMENTA A DEZENA

   MOVLW   .10
   XORWF   DEZENA,W
   BTFSS   STATUS,Z               ; DEZENA = 10 ?
   RETURN                         ; NÃO - RETORNA
                                  ; SIM
   CLRF    DEZENA                 ; ZERA A DEZENA
   INCF    CENTENA,F              ; INCREMENTA A CENTENA

   MOVLW   .10
   XORWF   CENTENA,W
   BTFSS   STATUS,Z               ; CENTENA = 10 ?
   RETURN                         ; NÃO - RETORNA
                                  ; SIM
   CLRF    CENTENA                ; ZERA A CENTENA
   INCF    MILHAR,F               ; INCREMENTA O MILHAR

   MOVLW   .10
   XORWF   MILHAR,W
   BTFSC   STATUS,Z               ; MILHAR = 10 ?
   CLRF    MILHAR                 ; SIM - ZERA MILHAR
   RETURN                         ; NÃO - RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *             ROTINA PARA DECREMENTAR O VALOR DO TIMER (BCD)             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ROTINA UTILIZADA PARA DECREMENTAR O VALOR DOS REGISTRADORES UNIDADE,
;  DEZENA, CENTENA E MILHAR, QUE SÃO OS CONTADORES DO TIMER. A CONTAGEM É
;  FEITA DIRETAMENTE EM BCD.

DECREMENTA_TIMER
   DECF    UNIDADE,F              ; INCREMENTA UNIDADE

   MOVLW   0XFF
   XORWF   UNIDADE,W
   BTFSS   STATUS,Z               ; UNIDADE = 0XFF ?
   RETURN                         ; NÃO - RETORNA
                                  ; SIM
   MOVLW   .9
   MOVWF   UNIDADE                ; CARREGA UNIDADE COM 9
   DECF    DEZENA,F               ; DECREMENTA A DEZENA

   MOVLW   0XFF
```

```
   XORWF   DEZENA,W
   BTFSS   STATUS,Z               ; DEZENA = 0XFF ?
   RETURN                         ; NÃO - RETORNA
                                  ; SIM
   MOVLW   .9
   MOVWF   DEZENA                 ; CARREGA A DEZENA COM 9
   DECF    CENTENA,F              ; DECREMENTA A CENTENA

   MOVLW   0XFF
   XORWF   CENTENA,W
   BTFSS   STATUS,Z               ; CENTENA = 0XFF ?
   RETURN                         ; NÃO - RETORNA
                                  ; SIM
   MOVLW   .9
   MOVWF   CENTENA                ; CARREGA CENTENA COM 9
   DECF    MILHAR,F               ; DECREMENTA O MILHAR

   MOVLW   0XFF
   XORWF   MILHAR,W
   BTFSS   STATUS,Z               ; MILHAR = 0XFF ?
   RETURN                         ; NÃO - RETORNA
                                  ; SIM
   MOVLW   .9
   MOVWF   MILHAR                 ; CARREGA O MILHAR COM 9
   RETURN                         ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               CONFIGURAÇÕES INICIAIS DE HARDWARE E SOFTWARE            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  NESTA ROTINA SÃO INICIALIZADAS AS PORTAS DE I/O DO MICROCONTROLADOR E AS
;  CONFIGURAÇÕES DOS REGISTRADORES ESPECIAIS (SFR). A ROTINA INICIALIZA A
;  MÁQUINA E AGUARDA O ESTOURO DO WDT.

CONFIG
   CLRF    PORTA                  ; LIMPA O PORTA
   CLRF    PORTB                  ; LIMPA O PORTB
   CLRF    PORTC                  ; LIMPA O PORTC
   CLRF    PORTD                  ; LIMPA O PORTD
   CLRF    PORTE                  ; LIMPA O PORTE


   BANK1                          ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                  ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                  ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011000'
   MOVWF   TRISC                  ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                  ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                  ; CONFIGURA I/O DO PORTE

   MOVLW   B'11011111'
   MOVWF   OPTION_REG             ; CONFIGURA OPTIONS
                                  ; PULL-UPs DESABILITADOS
                                  ; INTER. NA BORDA DE SUBIDA DO RB0
                                  ; TIMER0 INCREM. PELO CICLO DE MÁQUINA
                                  ; WDT   - 1:128
                                  ; TIMER0- 1:1

   MOVLW   B'01100000'
   MOVWF   INTCON                 ; CONFIGURA INTERRUPÇÕES
                                  ; HABILITADA A INTERRUPÇÃO DE TIMER0
                                  ; HABILITA AS INTERRUPÇÕES DE PERIFÉRICO

   MOVLW   B'00000001'
   MOVWF   PIE1                   ; CONFIGURA INTERRUPÇÕES DE PERIFIÉRICOS
```

```
                                ; HABILITADA A INTERRUPÇÃO DE TMR1
   MOVLW   B'00000111'
   MOVWF   CMCON                ; DESLIGA OS COMPARADORES

   MOVLW   B'00000111'
   MOVWF   ADCON1               ; CONFIGURA CONVERSOR A/D
                                ; CONFIGURA PORTA E PORTE COMO I/O DIGITAL

   BANK0                        ; SELECIONA BANCO 0 DA RAM

   MOVLW   B'00110000'
   MOVWF   T1CON                ; CONFIGURA TMR1
                                ; PRESCALER -> 1:8
                                ; INCREMENTADO PELO CICLO DE MÁQUINA

;  AS INSTRUÇÕES A SEGUIR FAZEM COM QUE O PROGRAMA TRAVE QUANDO HOUVER UM
;  RESET OU POWER-UP, MAS PASSE DIRETO SE O RESET FOR POR WDT. DESTA FORMA,
;  SEMPRE QUE O PIC É LIGADO, O PROGRAMA TRAVA, AGUARDA UM ESTOURO DE WDT
;  E COMEÇA NOVAMENTE. ISTO EVITA PROBLEMAS NO START-UP DO PIC.

   BTFSC   STATUS,NOT_TO        ; RESET POR ESTOURO DE WATCHDOG TIMER?
   GOTO    $                    ; NÃO - AGUARDA ESTOURO DO WDT
                                ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INICIALIZAÇÃO DA RAM                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA IRÁ LIMPAR TODA A RAM DO BANCO 0, INDO DE 0X20 A 0X7F.
;  EM SEGUIDA, AS VARIÁVEIS DE RAM DO PROGRAMA SÃO INICIALIZADAS.

   MOVLW   0X20
   MOVWF   FSR                  ; APONTA O ENDEREÇAMENTO INDIRETO PARA
                                ; A PRIMEIRA POSIÇÃO DA RAM
LIMPA_RAM
   CLRF    INDF                 ; LIMPA A POSIÇÃO
   INCF    FSR,F                ; INCREMENTA O PONTEIRO P/ A PRÓX. POS.
   MOVF    FSR,W
   XORLW   0X80                 ; COMPARA O PONTEIRO COM A ÚLT. POS. +1
   BTFSS   STATUS,Z             ; JÁ LIMPOU TODAS AS POSIÇÕES?
   GOTO    LIMPA_RAM            ; NÃO - LIMPA A PRÓXIMA POSIÇÃO
                                ; SIM

   BCF     ESTADO_TIMER         ; INICIA COM ESTADO EM OFF

   MOVLW   .2
   MOVWF   DIVISOR_TMR1         ; CARREGA CONTADOR DE 1SEG.

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        VARREDURA DOS BOTÕES                           *
; *                          LOOP PRINCIPAL                               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A ROTINA PRINCIPAL FICA CHECANDO O ESTADO DOS BOTÕES. CASO ALGUM SEJA
;  PRESSIONADO, A ROTINA DE TRATAMENTO DO BOTÃO É CHAMADA.

   BSF     INTCON,GIE           ; HABILITA AS INTERRUPÇÕES
                                ; USADA INT. TMR0 PARA VARREDURA
                                ; DOS DISPLAYS

VARRE
   CLRWDT                       ; LIMPA WATCHDOG TIMER

   BTFSC   BT_UP                ; O BOTÃO DE UP ESTÁ PRESSIONADO?
   GOTO    TRATA_BT_UP          ; SIM - PULA P/ TRATA_BT_UP
                                ; NÃO

   BTFSC   BT_DOWN              ; O BOTÃO DE DOWN ESTÁ PRESSIONADO?
   GOTO    TRATA_BT_DOWN        ; SIM - PULA P/ TRATA_BT_DOWN
                                ; NÃO

   BTFSC   BT_START_STOP        ; O BOTÃO START/STOP ESTÁ PRESSIONADO?
   GOTO    TRATA_BT_START_STOP  ; SIM - PULA P/ TRATA_BT_START_STOP
                                ; NÃO
```

```
   MOVLW   FILTRO_TECLA           ; CARREGA NO WORK O VALOR DE FILTRO_TECLA
   MOVWF   FILTRO_BOTOES          ; SALVA EM FILTRO_BOTOES
                                  ; RECARREGA FILTRO P/ EVITAR RUIDOS

   MOVLW   .1
   MOVWF   TEMPO_TURBO            ; CARREGA TEMPO DO TURBO DAS TECLAS
                                  ; COM 1 - IGNORA O TURBO A PRIMEIRA
                                  ; VEZ QUE A TECLA É PRESSIONADA

   GOTO    VARRE                  ; VOLTA PARA VARRER TECLADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        TRATAMENTO DOS BOTÕES                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

; ************************ TRATAMENTO DO BOTÃO DE UP **********************

TRATA_BT_UP
   BTFSC   ESTADO_TIMER           ; TIMER ESTÁ PARADO ?
   GOTO    VARRE                  ; NÃO - VOLTA P/ VARRE
                                  ; SIM

   DECFSZ  FILTRO_BOTOES,F        ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                  ; NÃO - VOLTA P/ VARRE
                                  ; SIM - BOTÃO PRESSIONADO

   DECFSZ  TEMPO_TURBO,F          ; FIM DO TEMPO DE TURBO ?
   GOTO    VARRE                  ; NÃO - VOLTA P/ VARRE
                                  ; SIM
   MOVLW   TURBO_TECLA
   MOVWF   TEMPO_TURBO            ; RECARREGA TEMPORIZADOR DO TURBO
                                  ; DAS TECLAS

   CALL    INCREMENTA_TIMER       ; INCREMENTA O VALOR DO TIMER

   GOTO    VARRE                  ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************ TRATAMENTO DO BOTÃO DE DOWN **********************

TRATA_BT_DOWN
   BTFSC   ESTADO_TIMER           ; TIMER ESTÁ PARADO ?
   GOTO    VARRE                  ; NÃO - VOLTA P/ VARRE
                                  ; SIM

   DECFSZ  FILTRO_BOTOES,F        ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                  ; NÃO - VOLTA P/ VARRE
                                  ; SIM - BOTÃO PRESSIONADO

   DECFSZ  TEMPO_TURBO,F          ; FIM DO TEMPO DE TURBO ?
   GOTO    VARRE                  ; NÃO - VOLTA P/ VARRE
                                  ; SIM
   MOVLW   TURBO_TECLA
   MOVWF   TEMPO_TURBO            ; RECARREGA TEMPORIZADOR DO TURBO
                                  ; DAS TECLAS

   CALL    DECREMENTA_TIMER       ; DECREMENTA O VALOR DO TIMER

   GOTO    VARRE                  ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************ TRATAMENTO DO BOTÃO START / STOP *****************

TRATA_BT_START_STOP
   MOVF    FILTRO_BOTOES,F
   BTFSC   STATUS,Z               ; FILTRO JÁ IGUAL A ZERO ?
                                  ; (FUNÇÃO JÁ FOI EXECUTADA?)
   GOTO    VARRE                  ; SIM - VOLTA P/ VARREDURA DO TECLADO
                                  ; NÃO

   DECFSZ  FILTRO_BOTOES,F        ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                  ; NÃO - VOLTA P/ VARRE
                                  ; SIM - BOTÃO PRESSIONADO
```

```
   BTFSS   ESTADO_TIMER            ; TIMER ESTA LIGADO ?
   GOTO    LIGA_TIMER              ; NÃO - ENTÃO LIGA
                                   ; SIM - ENTÃO DESLIGA

DESLIGA_TIMER
   BCF     ESTADO_TIMER            ; DESLIGA LED E FLAG DO ESTADO DO TIMER
   BCF     T1CON,TMR1ON            ; PARA CONTADOR DO TMR1
   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

LIGA_TIMER
   MOVF    UNIDADE,F
   BTFSS   STATUS,Z                ; UNIDADE ESTÁ ZERADA ?
   GOTO    LIGA_TIMER_2            ; NÃO - PULA P/ LIGA_TIMER_2
                                   ; SIM - TESTA DEZENA
   MOVF    DEZENA,F
   BTFSS   STATUS,Z                ; DEZENA ESTÁ ZERADA ?
   GOTO    LIGA_TIMER_2            ; NÃO - PULA P/ LIGA_TIMER_2
                                   ; SIM - TESTA CENTENA
   MOVF    CENTENA,F
   BTFSS   STATUS,Z                ; CENTENA ESTÁ ZERADA ?
   GOTO    LIGA_TIMER_2            ; NÃO - PULA P/ LIGA_TIMER_2
                                   ; SIM - TESTA MILHAR
   MOVF    MILHAR,F
   BTFSS   STATUS,Z                ; MILHAR ESTÁ ZERADO ?
   GOTO    LIGA_TIMER_2            ; NÃO - PULA P/ LIGA_TIMER_2
   GOTO    VARRE                   ; SIM - VOLTA P/ VARRER TECLADO
                                   ;       SEM LIGAR O TIMER

LIGA_TIMER_2
   BSF     ESTADO_TIMER            ; LIGA LED E FLAG DO ESTADO DO TIMER

   MOVLW   TMR1_HIGH
   MOVWF   TMR1H
   MOVLW   TMR1_LOW
   MOVWF   TMR1L                   ; INICIALIZA CONTADORES

   MOVLW   .2
   MOVWF   DIVISOR_TMR1            ; INICIALIZA DIVISOR

   BSF     T1CON,TMR1ON            ; LIGA CONTAGEM DO TMR1

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          FIM DO PROGRAMA                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                             ; FIM DO PROGRAMA
```

Dicas e Comentários

Observar que nesta experiência, ao entrar no tratamento das interrupções, a operação de salvar contexto é maior que nas experiências anteriores. Isto por que agora salva-se também os valores de FSR e PCLATH pois os mesmos podem ser alterados dentro da interrupção.

Exercícios Propostos

1. Implemente o quarto botão, para resetar o temporizador (voltar a zero).
2. Implemente uma segunda velocidade para os botões de incremento e decremento, de forma que facilite o ajuste de valores maiores.
3. Em vez de fazer um timer somente de segundos, utilize os dois dígitos da esquerda para mostrar o tempo em minutos e os da direita para mostrar o tempo em segundos. O ponto do display da centena pode ser usado para marcar a separação. Não se esqueça que agora os displays da direita devem contar somente de 0 a 59 e não mais de 0 a 99.

Capítulo 12 - Experiência 10 – Display de cristal líquido LCD

Objetivo

O objetivo desta experiência é o aprendizado da utilização de display de cristal líquido.

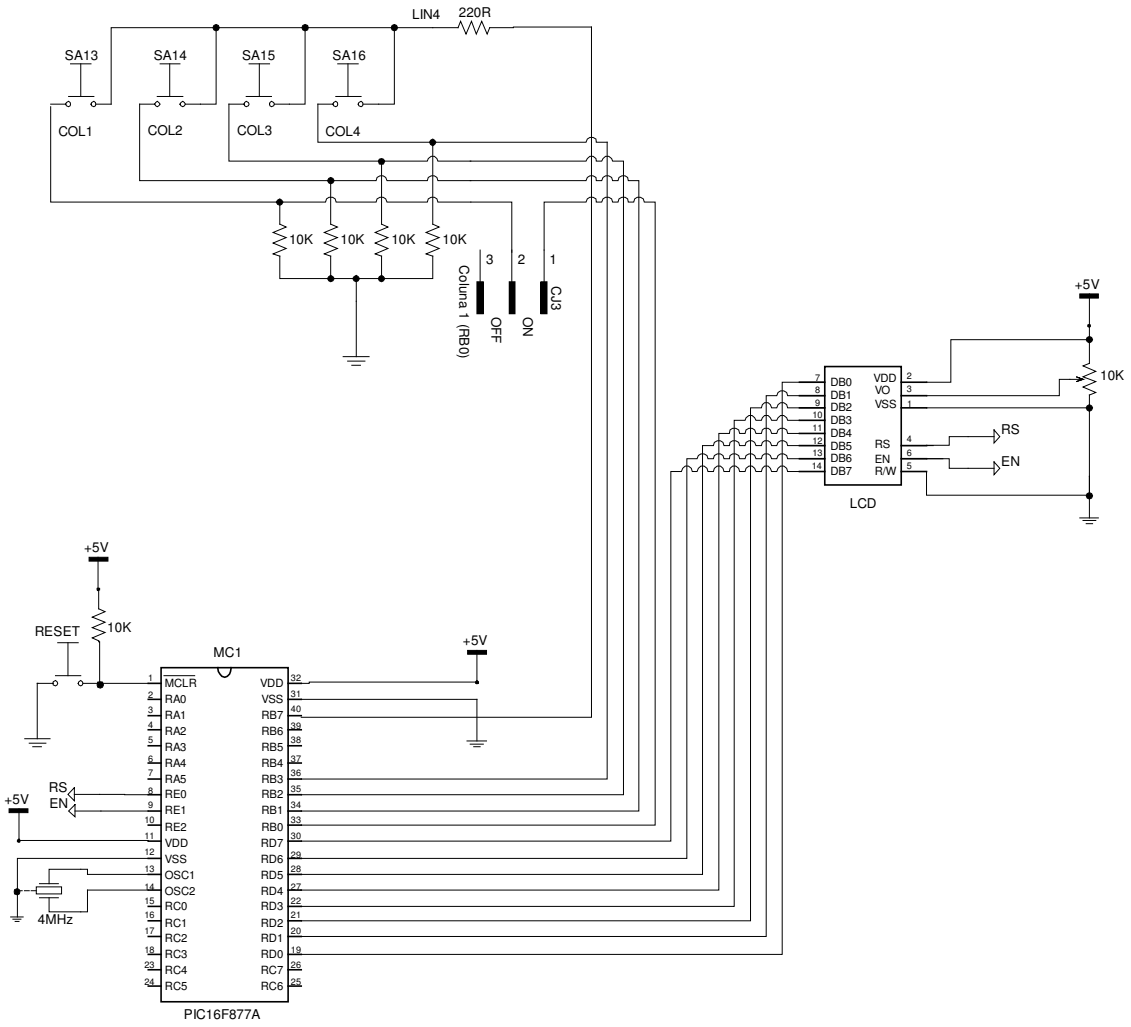
Descrição

Esta experiência foi elaborada para explicar o funcionamento do display LCD e o exemplo de software proposto é bastante reduzido. Simplesmente utilizou-se o LCD para informar ao usuário qual tecla está pressionada.

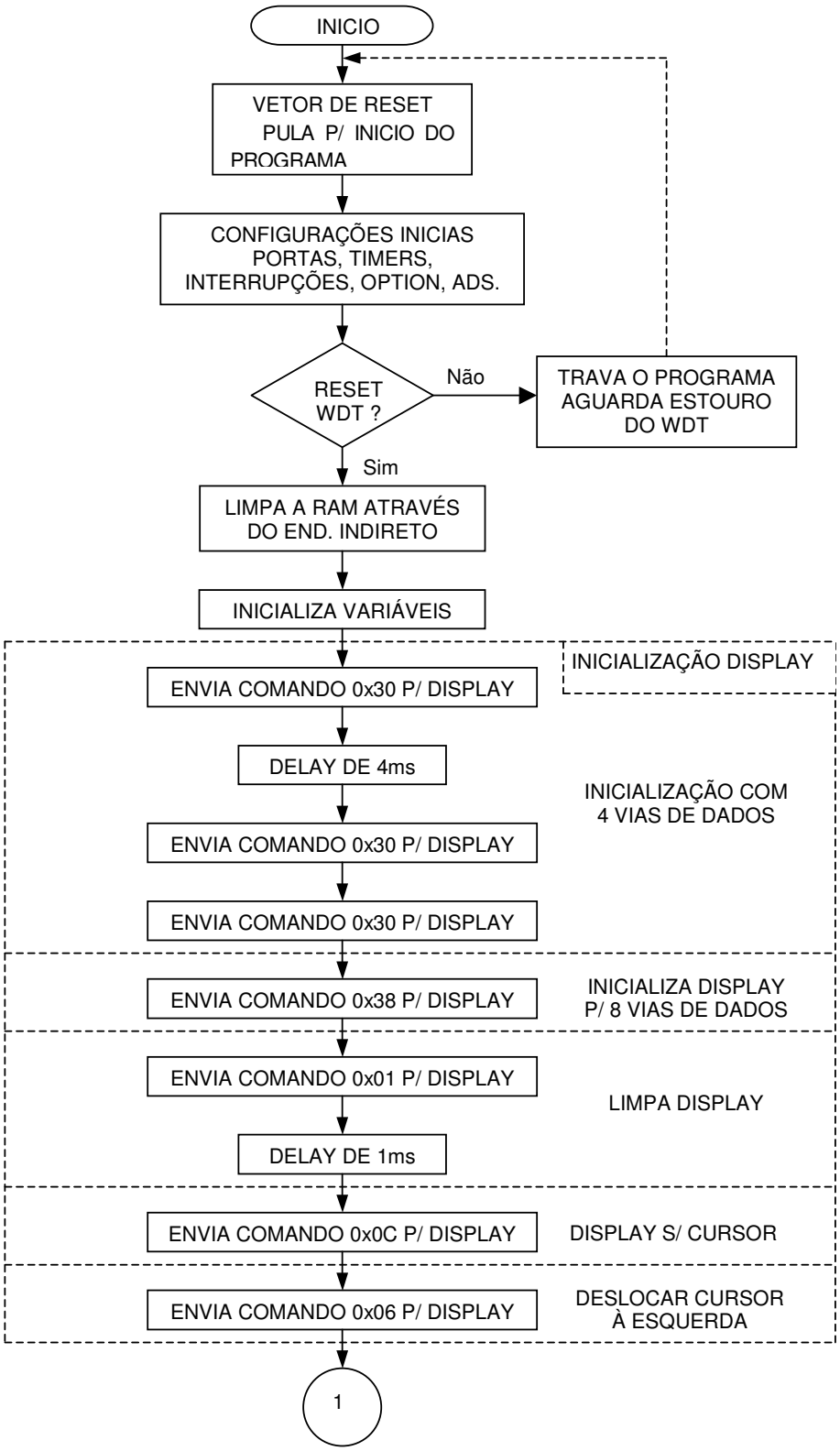
Para isso elaborou-se uma rotina chamada ESCREVE que envia a informação passada pelo Work ao display. Esta rotina pode ser utilizada tanto para enviar comandos quanto dados. Foi criada também, a rotina de inicialização do LCD. Esta rotina configura o sistema para comunicação com 8 vias, 2 linhas, sem cursor visível e com movimento automático do cursor para a direita. Além disso, ela já limpa a tela e posiciona o cursor na primeira linha, primeiro caractere da esquerda.

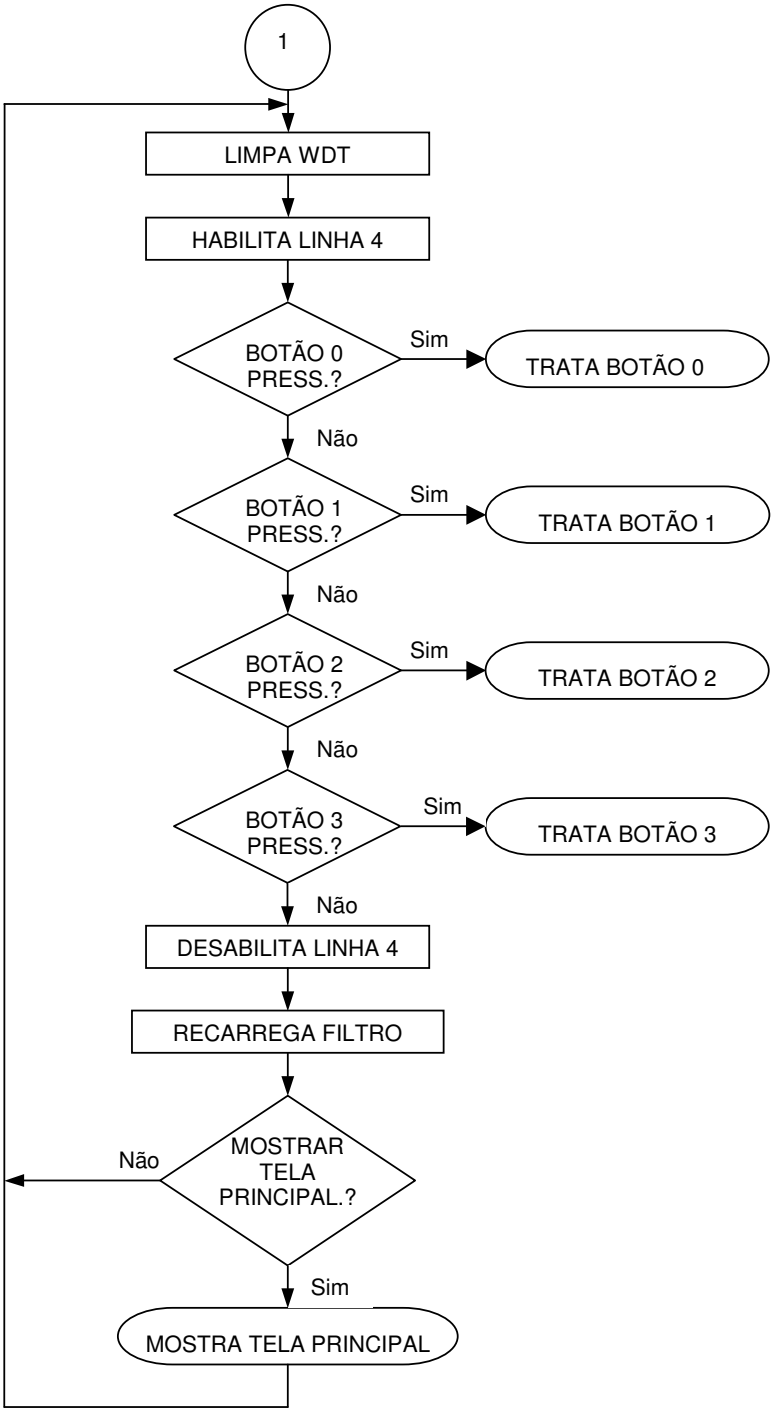
Para cada botão pressionado, posicionou-se o cursor em um local diferente da tela e escreveu-se o número do botão em questão. Após a liberação, uma tela padrão é visualizada.

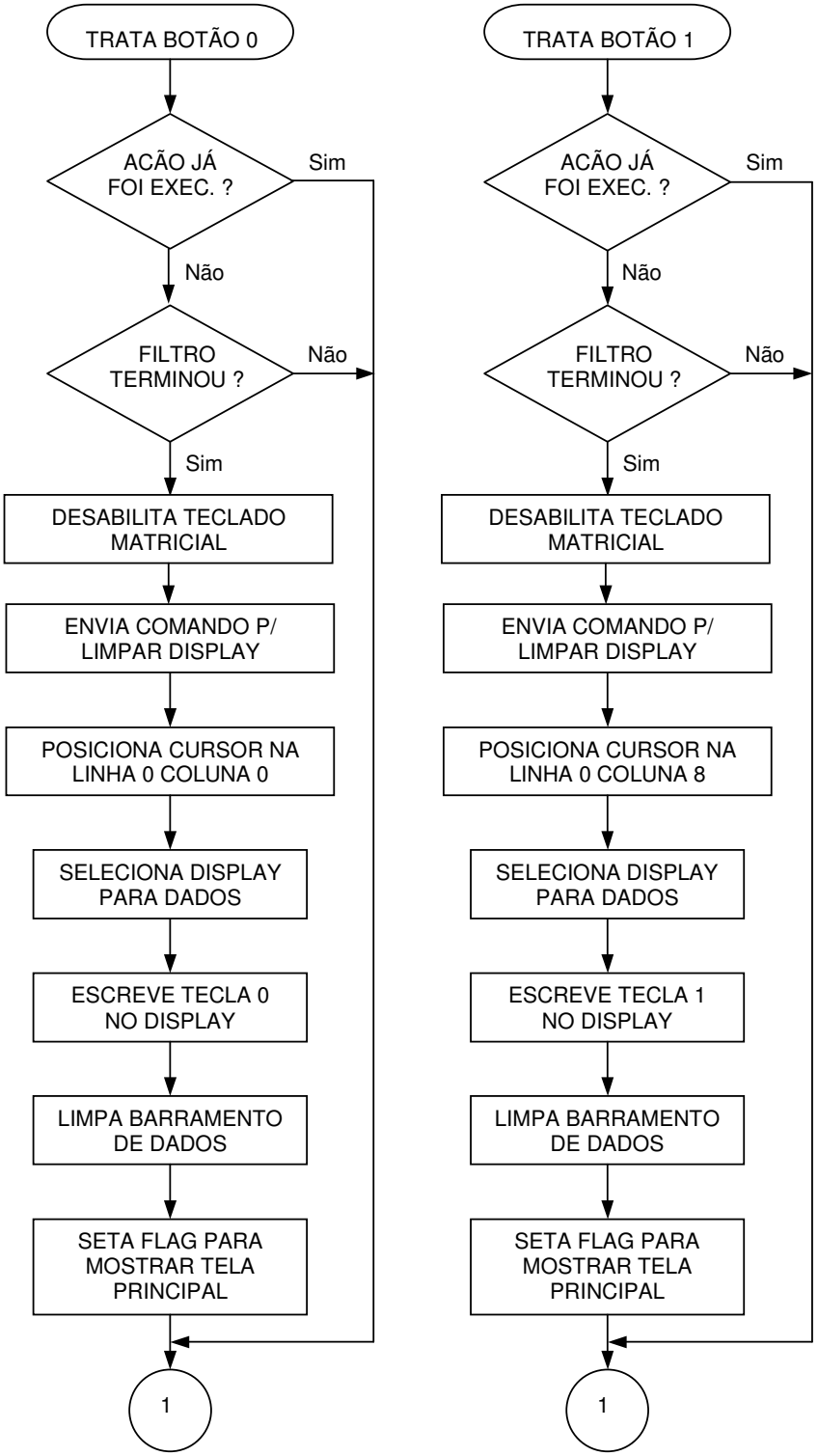
Esquema Elétrico

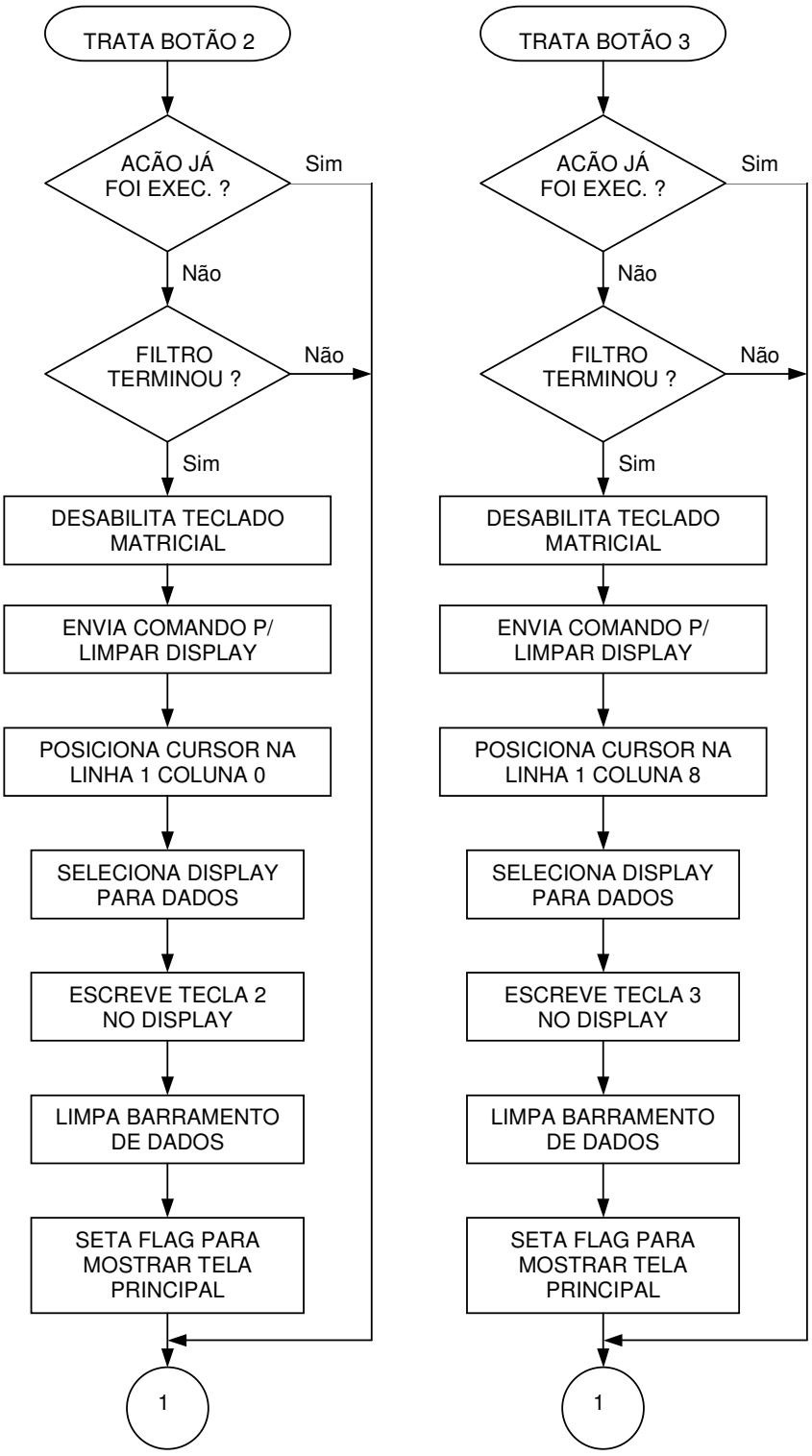


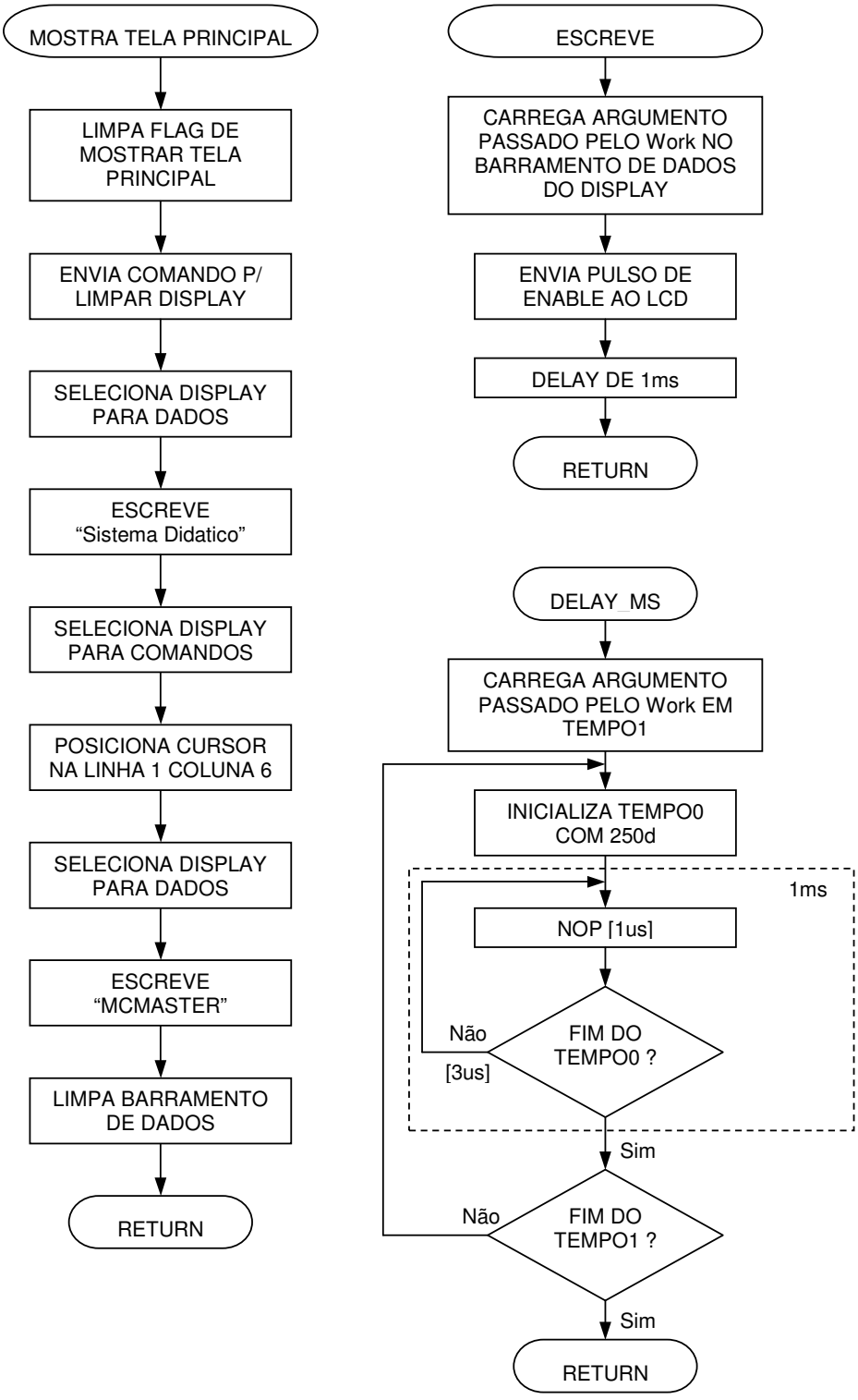
Fluxograma


INICIO
VETOR DE RESET
PULA P/ INICIO DO PROGRAMA
CONFIGURAÇÕES INICIAS PORTAS, TIMERS, INTERRUPÇÕES, OPTION, ADS.
RESET WDT ?
Não
TRAVA O PROGRAMA AGUARDA ESTOURO DO WDT
Sim
LIMPA A RAM ATRAVÉS DO END. INDIRETO
INICIALIZA VARIÁVEIS
INICIALIZAÇÃO DISPLAY
ENVIA COMANDO 0x30 P/ DISPLAY
DELAY DE 4ms
INICIALIZAÇÃO COM 4 VIAS DE DADOS
ENVIA COMANDO 0x30 P/ DISPLAY
ENVIA COMANDO 0x30 P/ DISPLAY
ENVIA COMANDO 0x38 P/ DISPLAY
INICIALIZA DISPLAY P/ 8 VIAS DE DADOS
ENVIA COMANDO 0x01 P/ DISPLAY
LIMPA DISPLAY
DELAY DE 1ms
ENVIA COMANDO 0x0C P/ DISPLAY
DISPLAY S/ CURSOR
ENVIA COMANDO 0x06 P/ DISPLAY
DESLOCAR CURSOR À ESQUERDA
1


1
LIMPA WDT
HABILITA LINHA 4
BOTÃO 0 PRESS.?
Sim
TRATA BOTÃO 0
Não
BOTÃO 1 PRESS.?
Sim
TRATA BOTÃO 1
Não
BOTÃO 2 PRESS.?
Sim
TRATA BOTÃO 2
Não
BOTÃO 3 PRESS.?
Sim
TRATA BOTÃO 3
Não
DESABILITA LINHA 4
RECARREGA FILTRO
MOSTRAR TELA PRINCIPAL.?
Não
Sim
MOSTRA TELA PRINCIPAL


TRATA BOTÃO 0
ACÃO JÁ FOI EXEC. ?
Sim
Não
FILTRO TERMINOU ?
Não
Sim
DESABILITA TECLADO MATRICIAL
ENVIA COMANDO P/ LIMPAR DISPLAY
POSICIONA CURSOR NA LINHA 0 COLUNA 0
SELECIONA DISPLAY PARA DADOS
ESCREVE TECLA 0 NO DISPLAY
LIMPA BARRAMENTO DE DADOS
SETA FLAG PARA MOSTRAR TELA PRINCIPAL
1
TRATA BOTÃO 1
ACÃO JÁ FOI EXEC. ?
Sim
Não
FILTRO TERMINOU ?
Não
Sim
DESABILITA TECLADO MATRICIAL
ENVIA COMANDO P/ LIMPAR DISPLAY
POSICIONA CURSOR NA LINHA 0 COLUNA 8
SELECIONA DISPLAY PARA DADOS
ESCREVE TECLA 1 NO DISPLAY
LIMPA BARRAMENTO DE DADOS
SETA FLAG PARA MOSTRAR TELA PRINCIPAL
1


TRATA BOTÃO 2
ACÃO JÁ FOI EXEC. ?
Sim
Não
FILTRO TERMINOU ?
Não
Sim
DESABILITA TECLADO MATRICIAL
ENVIA COMANDO P/ LIMPAR DISPLAY
POSICIONA CURSOR NA LINHA 1 COLUNA 0
SELECIONA DISPLAY PARA DADOS
ESCREVE TECLA 2 NO DISPLAY
LIMPA BARRAMENTO DE DADOS
SETA FLAG PARA MOSTRAR TELA PRINCIPAL
1
TRATA BOTÃO 3
ACÃO JÁ FOI EXEC. ?
Sim
Não
FILTRO TERMINOU ?
Não
Sim
DESABILITA TECLADO MATRICIAL
ENVIA COMANDO P/ LIMPAR DISPLAY
POSICIONA CURSOR NA LINHA 1 COLUNA 8
SELECIONA DISPLAY PARA DADOS
ESCREVE TECLA 3 NO DISPLAY
LIMPA BARRAMENTO DE DADOS
SETA FLAG PARA MOSTRAR TELA PRINCIPAL
1


MOSTRA TELA PRINCIPAL
LIMPA FLAG DE MOSTRAR TELA PRINCIPAL
ENVIA COMANDO P/ LIMPAR DISPLAY
SELECIONA DISPLAY PARA DADOS
ESCREVE "Sistema Didatico"
SELECIONA DISPLAY PARA COMANDOS
POSICIONA CURSOR NA LINHA 1 COLUNA 6
SELECIONA DISPLAY PARA DADOS
ESCREVE "MCMASTER"
LIMPA BARRAMENTO DE DADOS
RETURN
ESCREVE
CARREGA ARGUMENTO PASSADO PELO Work NO BARRAMENTO DE DADOS DO DISPLAY
ENVIA PULSO DE ENABLE AO LCD
DELAY DE 1ms
RETURN
DELAY_MS
CARREGA ARGUMENTO PASSADO PELO Work EM TEMPO1
INICIALIZA TEMPO0 COM 250d
1ms
NOP [1us]
Não
[3us]
FIM DO TEMPO0 ?
Sim
Não
FIM DO TEMPO1 ?
Sim
RETURN

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER           *
; *                                                                         *
; *              EXPERIÊNCIA 10 - DISPLAY DE CRISTAL LÍQUIDO LCD            *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            DESCRIÇÃO GERAL                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE EXEMPLO FOI ELABORADO PARA EXPLICAR O FUNCIONAMENTO DO MÓDULO DE LCD.
;  FOI CRIADA UMA ROTINA PARA ESCREVER COMANDOS OU CACACTRES NO LCD. EXISTE
;  TAMBÉM UMA ROTINA DE INICIALIZAÇÃO NECESSÁRIA PARA A CORRETA CONFIGURAÇÃO
;  DO LCD. OS BOTÕES CONTINUAM SENDO MONITORADOS. UMA MENSAGEM É ESCRITA
;  NO LCD PARA CADA UM DOS BOTÕES, QUANDO O MESMO É PRESSIONADO.
;  APENAS OS BOTÕES DA LINHA 4 ESTÃO ATIVADOS
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        CONFIGURAÇÕES PARA GRAVAÇÃO                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_ON & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          DEFINIÇÃO DAS VARIÁVEIS                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE BLOCO DE VARIÁVEIS ESTÁ LOCALIZADO LOGO NO INÍCIO DO BANCO 0

   CBLOCK  0X20                    ; POSIÇÃO INICIAL DA RAM

           TEMPO1
           TEMPO0                  ; CONTADORES P/ DELAY

           FILTRO_BOTOES           ; FILTRO PARA RUIDOS

           FLAG                    ; FLAG DE USO GERAL

   ENDC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>         ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     DEFINIÇÃO DOS BANCOS DE RAM                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          CONSTANTES INTERNAS                            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE CONSTANTES FACILITA A PROGRAMAÇÃO E A MANUTENÇÃO.

FILTRO_TECLA       EQU     .200            ; FILTRO P/ EVITAR RUIDOS DOS BOTÕES
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    DECLARAÇÃO DOS FLAGs DE SOFTWARE                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE FLAGs AJUDA NA PROGRAMAÇÃO E ECONOMIZA MEMÓRIA RAM.

#DEFINE    TELA_PRINCIPAL  FLAG,0  ; FLAG P/ INDICAR QUE DEVE MOSTRAR
                                   ; A TELA PRINCIPAL
                                   ; 1-> MOSTRA TELA PRINCIPAL
                                   ; 0-> TELA PRINCIPAL JÁ FOI MOSTRADA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                ENTRADAS                                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS ENTRADAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    BOTAO_0         PORTB,0 ; ESTADO DO BOTÃO 0
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_1         PORTB,1 ; ESTADO DO BOTÃO 1
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_2         PORTB,2 ; ESTADO DO BOTÃO 2
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_3         PORTB,3 ; ESTADO DO BOTÃO 3
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                 SAÍDAS                                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS SAÍDAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    DISPLAY         PORTD   ; BARRAMENTO DE DADOS DO DISPLAY

#DEFINE    RS              PORTE,0 ; INDICA P/ O DISPLAY UM DADO OU COMANDO
                                   ; 1 -> DADO
                                   ; 0 -> COMANDO

#DEFINE    ENABLE          PORTE,1 ; SINAL DE ENABLE P/ DISPLAY
                                   ; ATIVO NA BORDA DE DESCIDA

#DEFINE    TEC_MATRICIAL   PORTB   ; PORT DO MICROCONTROLADOR LIGADO AO
                                   ; TECLADO MATRICIAL
                                   ; <RB4:RB7> LINHAS
                                   ; 1->ATIVADAS 0->DESATIVADAS
                                   ; <RB0:RB3> COLUNAS
                                   ; 1->TECLAS PRESSIONADAS 0->TECLAS LIBERADAS

#DEFINE    LINHA_4         PORTB,7 ; PINO P/ ATIVAR LINHA 4 (TECLADO MATRICIAL)
                                   ; 0 -> LINHA 4 ATIVADA
                                   ; 1 -> LINHA 4 DESATIVADA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  VETOR DE RESET DO MICROCONTROLADOR                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO INICIAL PARA EXECUÇÃO DO PROGRAMA

   ORG     0X0000                  ; ENDEREÇO DO VETOR DE RESET
   GOTO    CONFIG                  ; PULA PARA CONFIG DEVIDO A REGIÃO
                                   ; DESTINADA AS ROTINAS SEGUINTES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    ROTINA DE DELAY (DE 1MS ATÉ 256MS)                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA É UMA ROTINA DE DELAY VARIÁVEL, COM DURAÇÃO DE 1MS X O VALOR PASSADO
;  EM WORK (W).
```

```
DELAY_MS
   MOVWF   TEMPO1                ; CARREGA TEMPO1 (UNIDADES DE MS)
   MOVLW   .250
   MOVWF   TEMPO0                ; CARREGA TEMPO0 (P/ CONTAR 1MS)

   CLRWDT                        ; LIMPA WDT (PERDE TEMPO)
   DECFSZ  TEMPO0,F              ; FIM DE TEMPO0 ?
   GOTO    $-2                   ; NÃO - VOLTA 2 INSTRUÇÕES
                                 ; SIM - PASSOU-SE 1MS
   DECFSZ  TEMPO1,F              ; FIM DE TEMPO1 ?
   GOTO    $-6                   ; NÃO - VOLTA 6 INSTRUÇÕES
                                 ; SIM
   RETURN                        ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DE UM CARACTER NO DISPLAY              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ENVIA UM CARACTER PARA O MÓDULO DE LCD. O CARACTER A SER
;  ESCRITO DEVE SER COLOCADO EM WORK (W) ANTES DE CHAMAR A ROTINA.

ESCREVE
   MOVWF   DISPLAY               ; ATUALIZA DISPLAY (PORTD)
   NOP                           ; PERDE 1US PARA ESTABILIZAÇÃO
   BSF     ENABLE                ; ENVIA UM PULSO DE ENABLE AO DISPLAY
   GOTO    $+1                   ; .
   BCF     ENABLE                ; .

   MOVLW   .1
   CALL    DELAY_MS              ; DELAY DE 1MS
   RETURN                        ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DA TELA PRINCIPAL                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A TELA PRINCIPAL DO PROGRAMA, COM AS FRASES:
;  LINHA 1 - "Sistema Didatico"
;  LINHA 2 - "     MCMASTER    "

MOSTRA_TELA_PRINCIPAL
   BCF     TELA_PRINCIPAL        ; LIMPA FLAG DE MOSTRAR TELA PRINCIPAL

   CLRF    TEC_MATRICIAL         ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                    ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0X01                  ; ESCREVE COMANDO PARA
   CALL    ESCREVE               ; LIMPAR A TELA
   MOVLW   .1
   CALL    DELAY_MS              ; DELAY DE 1MS

   BSF     RS                    ; SELECIONA O DISPLAY P/ DADOS

   MOVLW   'S'
   CALL    ESCREVE
   MOVLW   'i'
   CALL    ESCREVE
   MOVLW   's'
   CALL    ESCREVE
   MOVLW   't'
   CALL    ESCREVE
   MOVLW   'e'
   CALL    ESCREVE
   MOVLW   'm'
   CALL    ESCREVE
   MOVLW   'a'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   'D'
   CALL    ESCREVE
   MOVLW   'i'
   CALL    ESCREVE
```

```
   MOVLW   'd'
   CALL    ESCREVE
   MOVLW   'a'
   CALL    ESCREVE
   MOVLW   't'
   CALL    ESCREVE
   MOVLW   'i'
   CALL    ESCREVE
   MOVLW   'c'
   CALL    ESCREVE
   MOVLW   'o'
   CALL    ESCREVE                 ; ESCREVE Sistema Didatico

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDO
   MOVLW   0XC5
   CALL    ESCREVE                 ; POSICIONA O CURSOR - POSIÇÃO 5 LINHA 1
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

   MOVLW   'S'
   CALL    ESCREVE
   MOVLW   'D'
   CALL    ESCREVE
   MOVLW   '-'
   CALL    ESCREVE
   MOVLW   '1'
   CALL    ESCREVE
   MOVLW   '7'
   CALL    ESCREVE
   MOVLW   '0'
   CALL    ESCREVE
   MOVLW   '0'
   CALL    ESCREVE                 ; ESCREVE MCMASTER

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   RETURN

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               CONFIGURAÇÕES INICIAIS DE HARDWARE E SOFTWARE           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  NESTA ROTINA SÃO INICIALIZADAS AS PORTAS DE I/O DO MICROCONTROLADOR E AS
;  CONFIGURAÇÕES DOS REGISTRADORES ESPECIAIS (SFR). A ROTINA INICIALIZA A
;  MÁQUINA E AGUARDA O ESTOURO DO WDT.

CONFIG
   CLRF    PORTA                   ; LIMPA O PORTA
   CLRF    PORTB                   ; LIMPA O PORTB
   CLRF    PORTC                   ; LIMPA O PORTC
   CLRF    PORTD                   ; LIMPA O PORTD
   CLRF    PORTE                   ; LIMPA O PORTE


   BANK1                           ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                   ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                   ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011000'
   MOVWF   TRISC                   ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                   ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                   ; CONFIGURA I/O DO PORTE

   MOVLW   B'11011111'
   MOVWF   OPTION_REG              ; CONFIGURA OPTIONS
                                   ; PULL-UPs DESABILITADOS
                                   ; INTER. NA BORDA DE SUBIDA DO RB0
```

```
                                ; TIMER0 INCREM. PELO CICLO DE MÁQUINA
                                ; WDT   - 1:128
                                ; TIMER - 1:1

   MOVLW   B'00000000'
   MOVWF   INTCON               ; CONFIGURA INTERRUPÇÕES
                                ; DESABILITADA TODAS AS INTERRUPÇÕES
   MOVLW   B'00000111'
   MOVWF   CMCON                ; DESLIGA OS COMPARADORES

   MOVLW   B'00000111'
   MOVWF   ADCON1               ; CONFIGURA CONVERSOR A/D
                                ; CONFIGURA PORTA E PORTE COM I/O DIGITAL

   BANK0                        ; SELECIONA BANCO 0 DA RAM

;  AS INSTRUÇÕES A SEGUIR FAZEM COM QUE O PROGRAMA TRAVE QUANDO HOUVER UM
;  RESET OU POWER-UP, MAS PASSE DIRETO SE O RESET FOR POR WDT. DESTA FORMA,
;  SEMPRE QUE O PIC É LIGADO, O PROGRAMA TRAVA, AGUARDA UM ESTOURO DE WDT
;  E COMEÇA NOVAMENTE. ISTO EVITA PROBLEMAS NO START-UP DO PIC.

   BTFSC   STATUS,NOT_TO        ; RESET POR ESTOURO DE WATCHDOG TIMER?
   GOTO    $                    ; NÃO - AGUARDA ESTOURO DO WDT
                                ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INICIALIZAÇÃO DA RAM                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA IRÁ LIMPAR TODA A RAM DO BANCO 0, INDO DE 0X20 A 0X7F.
;  EM SEGUIDA, AS VARIÁVEIS DE RAM DO PROGRAMA SÃO INICIALIZADAS.

   MOVLW   0X20
   MOVWF   FSR                  ; APONTA O ENDEREÇAMENTO INDIRETO PARA
                                ; A PRIMEIRA POSIÇÃO DA RAM
LIMPA_RAM
   CLRF    INDF                 ; LIMPA A POSIÇÃO
   INCF    FSR,F                ; INCREMENTA O PONTEIRO P/ A PRÓX. POS.
   MOVF    FSR,W
   XORLW   0X80                 ; COMPARA O PONTEIRO COM A ÚLT. POS. +1
   BTFSS   STATUS,Z             ; JÁ LIMPOU TODAS AS POSIÇÕES?
   GOTO    LIMPA_RAM            ; NÃO - LIMPA A PRÓXIMA POSIÇÃO
                                ; SIM

   BSF     TELA_PRINCIPAL       ; INICIALIZA MOSTRANDO TELA PRINCIPAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  CONFIGURAÇÕES INICIAIS DO DISPLAY                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA INICIALIZA O DISPLAY P/ COMUNICAÇÃO DE 8 VIAS, DISPLAY PARA 2
; LINHAS, CURSOR APAGADO E DESLOCAMENTO DO CURSOR À DIREITA.

INICIALIZACAO_DISPLAY
   BCF     RS                   ; SELECIONA O DISPLAY P/ COMANDOS

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO

   MOVLW   .3
   CALL    DELAY_MS             ; DELAY DE 3MS (EXIGIDO PELO DISPLAY)

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO

   MOVLW   B'00111000'          ; ESCREVE COMANDO PARA
   CALL    ESCREVE              ; INTERFACE DE 8 VIAS DE DADOS

   MOVLW   B'00000001'          ; ESCREVE COMANDO PARA
   CALL    ESCREVE              ; LIMPAR TODO O DISPLAY
```

```
   MOVLW   .1
   CALL    DELAY_MS                ; DELAY DE 1MS

   MOVLW   B'00001100'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; LIGAR O DISPLAY SEM CURSOR

   MOVLW   B'00000110'             ; ESCREVE COMANDO PARA INCREM.
   CALL    ESCREVE                 ; AUTOMÁTICO À DIREITA

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          VARREDURA DOS BOTÕES                           *
; *                            LOOP PRINCIPAL                               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A ROTINA PRINCIPAL FICA CHECANDO O ESTADO DOS BOTÕES. CASO ALGUM SEJA
;  PRESSIONADO, A ROTINA DE TRATAMENTO DO BOTÃO É CHAMADA.


VARRE
   CLRWDT                          ; LIMPA WATCHDOG TIMER

   BSF     LINHA_4                 ; ATIVA BOTÕES DA LINHA 4

   GOTO    $+1                     ; DELAY PARA ESTABILIZAÇÃO
                                   ; E LEITURA DO TECLADO

   BTFSC   BOTAO_0                 ; O BOTÃO 0 ESTÁ PRESSIONADO ?
   GOTO    TRATA_BOTAO_0           ; SIM - PULA P/ TRATA_BOTAO_0
                                   ; NÃO

   BTFSC   BOTAO_1                 ; O BOTÃO 1 ESTÁ PRESSIONADO ?
   GOTO    TRATA_BOTAO_1           ; SIM - PULA P/ TRATA_BOTAO_1
                                   ; NÃO

   BTFSC   BOTAO_2                 ; O BOTÃO 2 ESTÁ PRESSIONADO ?
   GOTO    TRATA_BOTAO_2           ; SIM - PULA P/ TRATA_BOTAO_2
                                   ; NÃO

   BTFSC   BOTAO_3                 ; O BOTÃO 3 ESTÁ PRESSIONADO ?
   GOTO    TRATA_BOTAO_3           ; SIM - PULA P/ TRATA_BOTAO_3
                                   ; NÃO

   BCF     LINHA_4                 ; DESATIVA BOTÕES DA LINHA 4

   MOVLW   FILTRO_TECLA            ; CARREGA NO WORK O VALOR DE FILTRO_TECLA
   MOVWF   FILTRO_BOTOES           ; SALVA EM FILTRO_BOTOES
                                   ; RECARREGA FILTRO P/ EVITAR RUIDOS

   BTFSS   TELA_PRINCIPAL          ; DEVE MOSTRAR TELA PRINCIPAL ?
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM
   CALL    MOSTRA_TELA_PRINCIPAL
   GOTO    VARRE                   ; VOLTA PARA VARRER TECLADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        TRATAMENTO DOS BOTÕES                            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

; ************************ TRATAMENTO DO BOTÃO 0 ***************************

TRATA_BOTAO_0
   MOVF    FILTRO_BOTOES,F
   BTFSC   STATUS,Z                ; FILTRO JÁ IGUAL A ZERO ?
                                   ; (FUNÇÃO JÁ FOI EXECUTADA?)
   GOTO    VARRE                   ; SIM - VOLTA P/ VARREDURA DO TECLADO
                                   ; NÃO

   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
```

```
                                ; SIM - BOTÃO PRESSIONADO

;  OS COMANDOS A SEGUIR SÃO PARA ESCREVER A FRASE RELACIONADA AO BOTÃO 0

   CLRF    TEC_MATRICIAL        ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                   ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0X01
   CALL    ESCREVE              ; COMANDO P/ LIMPAR A TELA
   MOVLW   .1
   CALL    DELAY_MS             ; DELAY DE 1MS

   MOVLW   0X80                 ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE              ; LINHA 0 / COLUNA 0
   BSF     RS                   ; SELECIONA O DISPLAY P/ DADOS

                                ; COMANDOS PARA ESCREVER AS
                                ; LETRAS DE "TECLA 0"
   MOVLW   'T'
   CALL    ESCREVE
   MOVLW   'E'
   CALL    ESCREVE
   MOVLW   'C'
   CALL    ESCREVE
   MOVLW   'L'
   CALL    ESCREVE
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   '0'
   CALL    ESCREVE

   CLRF    DISPLAY              ; LIMPA BARRAMENTO DE DADOS

   BSF     TELA_PRINCIPAL       ; SETA FLAG P/ MOSTRAR TELA PRINCIPAL

   GOTO    VARRE                ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************* TRATAMENTO DO BOTÃO 1 ***************************

TRATA_BOTAO_1
   MOVF    FILTRO_BOTOES,F
   BTFSC   STATUS,Z             ; FILTRO JÁ IGUAL A ZERO ?
                                ; (FUNÇÃO JÁ FOI EXECUTADA?)
   GOTO    VARRE                ; SIM - VOLTA P/ VARREDURA DO TECLADO
                                ; NÃO

   DECFSZ  FILTRO_BOTOES,F      ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                ; NÃO - VOLTA P/ VARRE
                                ; SIM - BOTÃO PRESSIONADO

;  OS COMANDOS A SEGUIR SÃO PARA ESCREVER A FRASE RELACIONADA AO BOTÃO 1

   CLRF    TEC_MATRICIAL        ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                   ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0X01
   CALL    ESCREVE              ; COMANDO P/ LIMPAR A TELA
   MOVLW   .1
   CALL    DELAY_MS             ; DELAY DE 1MS

   MOVLW   0X88                 ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE              ; LINHA 0 / COLUNA 8
   BSF     RS                   ; SELECIONA O DISPLAY P/ DADOS

                                ; COMANDOS PARA ESCREVER AS
                                ; LETRAS DE "TECLA 1"
   MOVLW   'T'
   CALL    ESCREVE
   MOVLW   'E'
   CALL    ESCREVE
```

```
   MOVLW   'C'
   CALL    ESCREVE
   MOVLW   'L'
   CALL    ESCREVE
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   '1'
   CALL    ESCREVE

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   BSF     TELA_PRINCIPAL          ; SETA FLAG P/ MOSTRAR TELA PRINCIPAL

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************* TRATAMENTO DO BOTÃO 2 ***************************

TRATA_BOTAO_2
   MOVF    FILTRO_BOTOES,F
   BTFSC   STATUS,Z                ; FILTRO JÁ IGUAL A ZERO ?
                                   ; (FUNÇÃO JA FOI EXECUTADA?)
   GOTO    VARRE                   ; SIM - VOLTA P/ VARREDURA DO TECLADO
                                   ; NÃO

   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO

;  OS COMANDOS A SEGUIR SÃO PARA ESCREVER A FRASE RELACIONADA AO BOTÃO 2

   CLRF    TEC_MATRICIAL           ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0X01
   CALL    ESCREVE                 ; COMANDO P/ LIMPAR A TELA
   MOVLW   .1
   CALL    DELAY_MS                ; DELAY DE 1MS

   MOVLW   0XC0                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 0
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

                                   ; COMANDOS PARA ESCREVER AS
                                   ; LETRAS DE "TECLA 2"
   MOVLW   'T'
   CALL    ESCREVE
   MOVLW   'E'
   CALL    ESCREVE
   MOVLW   'C'
   CALL    ESCREVE
   MOVLW   'L'
   CALL    ESCREVE
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   '2'
   CALL    ESCREVE

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   BSF     TELA_PRINCIPAL          ; SETA FLAG P/ MOSTRAR TELA PRINCIPAL

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************* TRATAMENTO DO BOTÃO 3 ***************************

TRATA_BOTAO_3
   MOVF    FILTRO_BOTOES,F
   BTFSC   STATUS,Z                ; FILTRO JÁ IGUAL A ZERO ?
```

```
                                    ; (FUNÇÃO JA FOI EXECUTADA?)
   GOTO    VARRE                    ; SIM - VOLTA P/ VARREDURA DO TECLADO
                                    ; NÃO

   DECFSZ  FILTRO_BOTOES,F          ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                    ; NÃO - VOLTA P/ VARRE
                                    ; SIM - BOTÃO PRESSIONADO

;  OS COMANDOS A SEGUIR SÃO PARA ESCREVER A FRASE RELACIONADA AO BOTÃO 3

   CLRF    TEC_MATRICIAL            ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                       ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0X01
   CALL    ESCREVE                  ; COMANDO P/ LIMPAR A TELA
   MOVLW   .1
   CALL    DELAY_MS                 ; DELAY DE 1MS

   MOVLW   0XC8                     ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                  ; LINHA 1 / COLUNA 8
   BSF     RS                       ; SELECIONA O DISPLAY P/ DADOS

                                    ; COMANDOS PARA ESCREVER AS
                                    ; LETRAS DE "TECLA 3"
   MOVLW   'T'
   CALL    ESCREVE
   MOVLW   'E'
   CALL    ESCREVE
   MOVLW   'C'
   CALL    ESCREVE
   MOVLW   'L'
   CALL    ESCREVE
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   '3'
   CALL    ESCREVE

   CLRF    DISPLAY                  ; LIMPA BARRAMENTO DE DADOS

   BSF     TELA_PRINCIPAL           ; SETA FLAG P/ MOSTRAR TELA PRINCIPAL

   GOTO    VARRE                    ; VOLTA P/ VARREDURA DOS BOTÕES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            FIM DO PROGRAMA                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                              ; FIM DO PROGRAMA
```

Dicas e Comentários

Apesar da estrutura do sistema ficar muito simples com a implementação da rotina ESCREVE, nunca se deve esquecer de confirmar o estado da saída RS (define comando ou dado) antes de utilizá-la.

Notar que para enviar um caractere ao LCD deve-se utilizar o código ASCII do caractere.

Apesar do sistema MCMASTER possuir ligação com o módulo de LCD através de 8 vias de dados é possível utilizá-lo para testar e implementar a comunicação com 4 vias. Basta modificar a rotina de inicialização e a de escrita de um byte.

Exercícios Propostos

1. Altere a comunicação para 4 vias.
2. Mantenha a tela principal disponível somente quando o sistema é ligado. Após alguns segundos, mostre uma tela com o nome das quatro teclas e indique a tecla pressionada através de um caractere de seta (←) ou outro qualquer.

Capítulo 13 - Experiência 11 – Conversor A/D

Objetivo

Nesta experiência será estudado o módulo de conversão A/D interno do PIC16F877A

Descrição

Este exemplo foi elaborado para explicar o funcionamento do módulo de conversão analógico digital interno do PIC16F877A. É convertido o valor analógico presente no pino RA0 do microcontrolador, sendo que este valor pode ser alterado através do potenciômetro presente na placa do sistema MCMASTER.

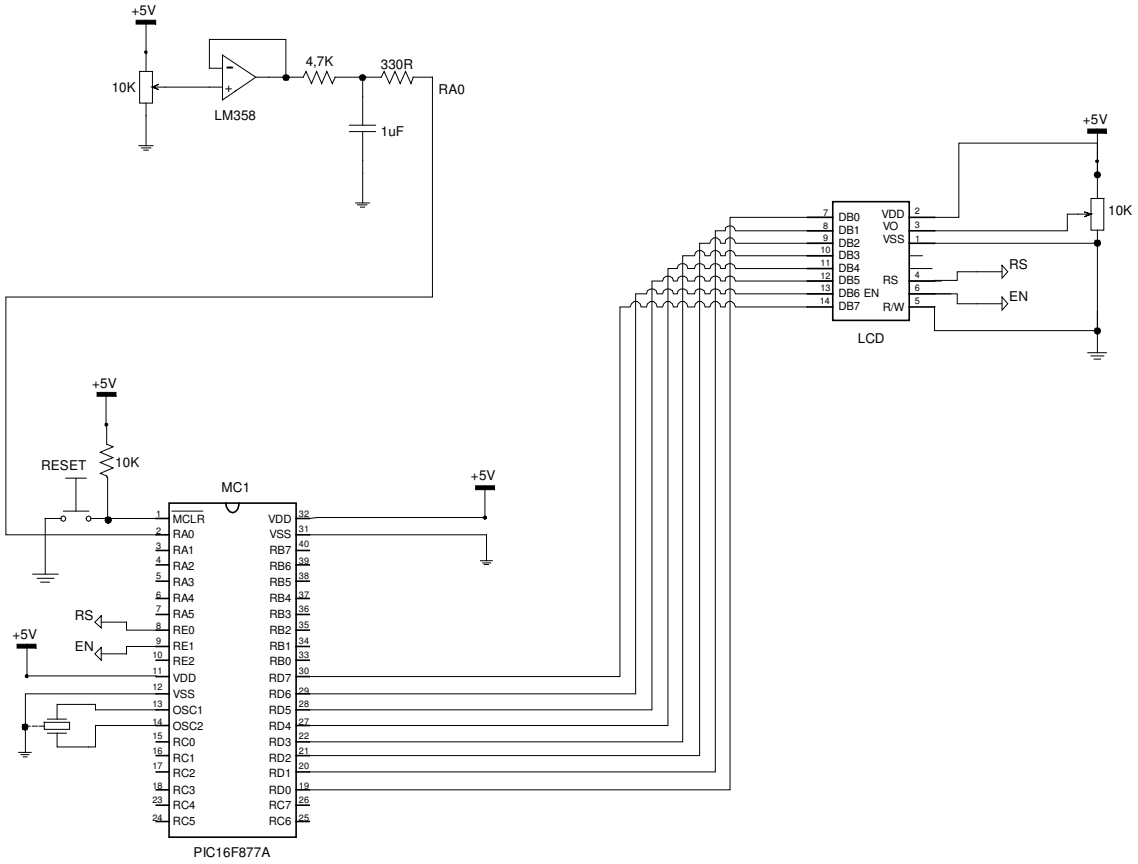
A conversão é feita diretamente no loop principal, sem a utilização de nenhuma interrupção, nem para checar o término da conversão nem para definir uma freqüência de amostragem. Desta forma, a conversão será feita uma após a outra, na freqüência definida pelo período do loop principal.

Uma vez terminada a conversão, descarta-se os 2 bits menos significativos e considera-se somente o resultado armazenado em **ADRESH**. Com isso já se está executando uma espécie de filtragem, evitando assim que o valor final fique oscilando.

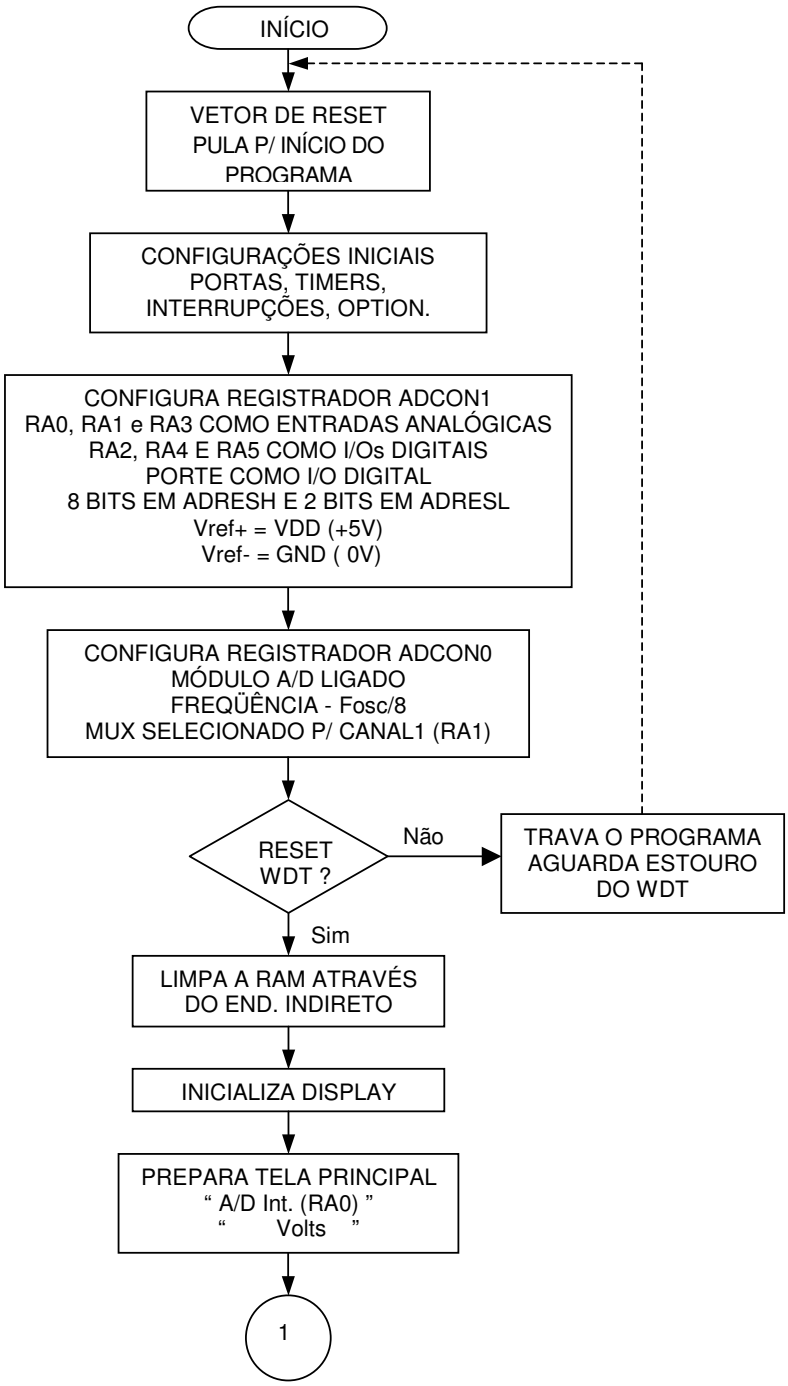
Aplica-se então uma regra de 3 para converter o valor do A/D para a unidade desejada: Volts. Considerando-se que quando o A/D resulta em 0 (zero) a entrada possui 0,0V, e quando o A/D resulta em 255 a entrada é equivalente a 5,0V, aplica-se a regra de 3 é mostra-se o valor da tensão, já em Volts, no LCD.

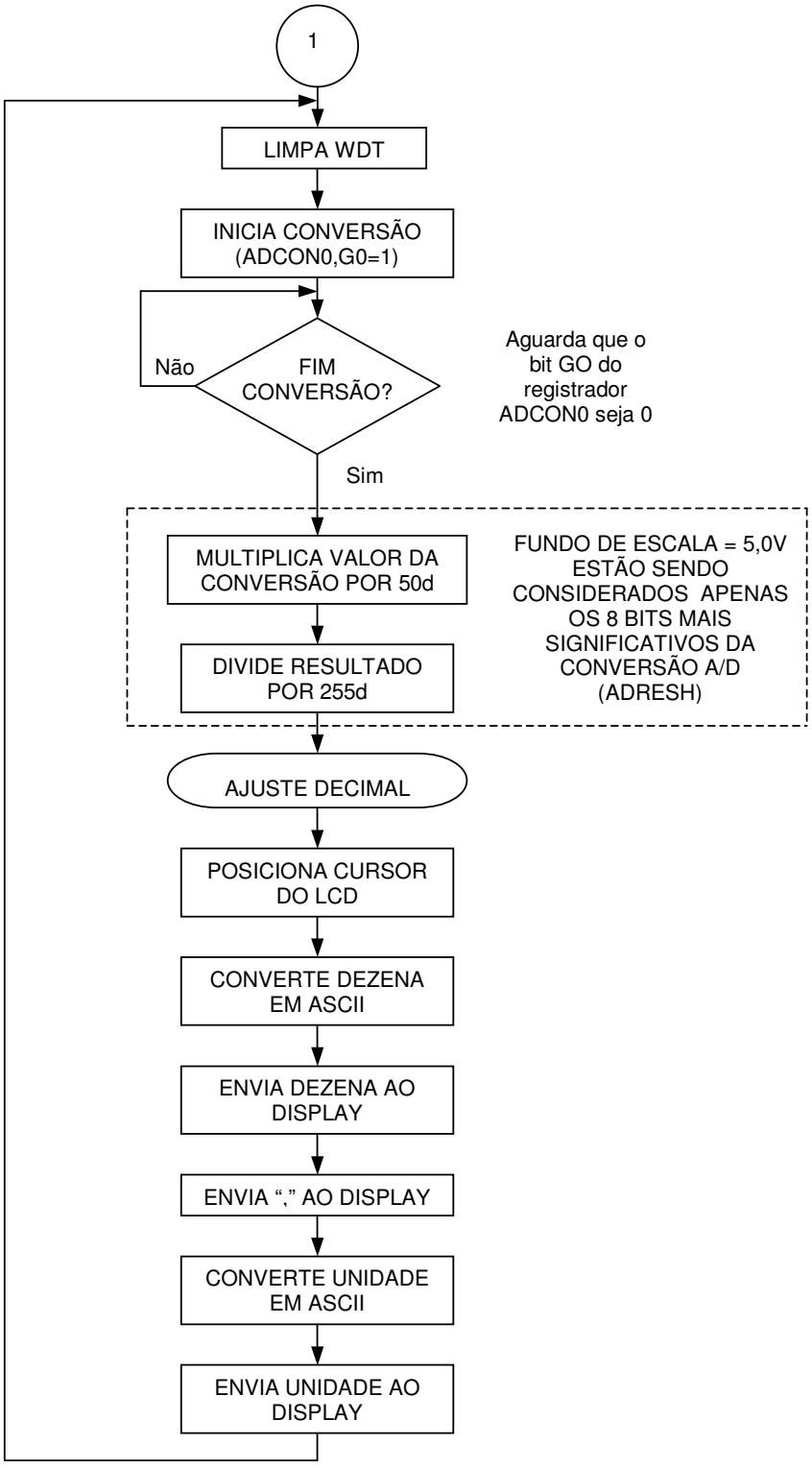
Para a execução da regra de 3 foram utilizadas rotinas de multiplicação de 8x8 e divisão de 16x16 retiradas de Aplication Notes da própria Microchip.

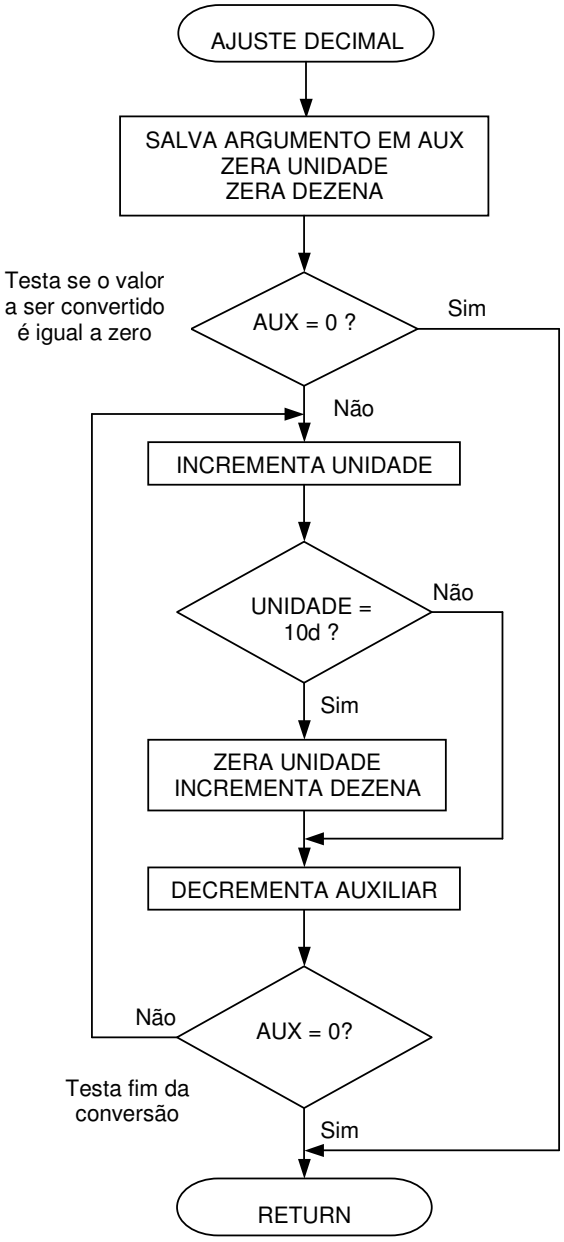
Esquema Elétrico



Fluxograma


INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURAÇÕES INICIAIS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION.
CONFIGURA REGISTRADOR ADCON1
RA0, RA1 e RA3 COMO ENTRADAS ANALÓGICAS
RA2, RA4 E RA5 COMO I/Os DIGITAIS
PORTE COMO I/O DIGITAL
8 BITS EM ADRESH E 2 BITS EM ADRESL
Vref+ = VDD (+5V)
Vref- = GND (0V)
CONFIGURA REGISTRADOR ADCON0
MÓDULO A/D LIGADO
FREQÜÊNCIA - Fosc/8
MUX SELECIONADO P/ CANAL1 (RA1)
RESET
WDT ?
Não
TRAVA O PROGRAMA
AGUARDA ESTOURO
DO WDT
Sim
LIMPA A RAM ATRAVÉS
DO END. INDIRETO
INICIALIZA DISPLAY
PREPARA TELA PRINCIPAL
“ A/D Int. (RA0) ”
“ Volts ”
1


1
LIMPA WDT
INICIA CONVERSÃO
(ADCON0,G0=1)
Não
FIM
CONVERSÃO?
Aguarda que o
bit GO do
registrador
ADCON0 seja 0
Sim
MULTIPLICA VALOR DA
CONVERSÃO POR 50d
DIVIDE RESULTADO
POR 255d
FUNDO DE ESCALA = 5,0V
ESTÃO SENDO
CONSIDERADOS APENAS
OS 8 BITS MAIS
SIGNIFICATIVOS DA
CONVERSÃO A/D
(ADRESH)
AJUSTE DECIMAL
POSICIONA CURSOR
DO LCD
CONVERTE DEZENA
EM ASCII
ENVIA DEZENA AO
DISPLAY
ENVIA "," AO DISPLAY
CONVERTE UNIDADE
EM ASCII
ENVIA UNIDADE AO
DISPLAY


AJUSTE DECIMAL
SALVA ARGUMENTO EM AUX
ZERA UNIDADE
ZERA DEZENA
Testa se o valor
a ser convertido
é igual a zero
AUX = 0 ?
Sim
Não
INCREMENTA UNIDADE
UNIDADE =
10d ?
Não
Sim
ZERA UNIDADE
INCREMENTA DEZENA
DECREMENTA AUXILIAR
Não
AUX = 0?
Testa fim da
conversão
Sim
RETURN

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *                       EXPERIÊNCIA 11 - CONVERSOR A/D                    *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                             DESCRIÇÃO GERAL                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE EXEMPLO FOI ELABORADO PARA EXPLICAR O FUNCIONAMENTO DO MÓDULO DE
;  CONVERSÃO ANALÓGICO DIGITAL INTERNO DO PIC. É CONVERTIDO O VALOR ANALÓGICO
;  PRESENTE NO PINO RA0 DO MICROCONTROLADOR, SENDO QUE ESTE VALOR PODE SER
;  ALTERADO ATRAVÉS DO POTENCIÔMETRO DA PLACA. O VALOR DA CONVERSÃO A/D É
;  AJUSTADO NUMA ESCALA DE 0 À 5V E MOSTRADO NO LCD.
;  FORAM UTILIZADAS ROTINAS DE MULTIPLICAÇÃO DE 8x8 E DIVISÃO DE 16x16. ESTAS
;  ROTINAS FORAM RETIRADAS DE APLICATION NOTES DA PRÓPRIA MICROCHIP.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        CONFIGURAÇÕES PARA GRAVAÇÃO                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_ON & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           DEFINIÇÃO DAS VARIÁVEIS                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE BLOCO DE VARIÁVEIS ESTÁ LOCALIZADO LOGO NO INÍCIO DO BANCO 0

   CBLOCK  0X20                    ; POSIÇÃO INICIAL DA RAM

           ACCaHI                  ; ACUMULADOR a DE 16 BITS UTILIZADO
           ACCaLO                  ; NA ROTINA DE DIVISÃO

           ACCbHI                  ; ACUMULADOR b DE 16 BITS UTILIZADO
           ACCbLO                  ; NA ROTINA DE DIVISÃO

           ACCcHI                  ; ACUMULADOR c DE 16 BITS UTILIZADO
           ACCcLO                  ; NA ROTINA DE DIVISÃO

           ACCdHI                  ; ACUMULADOR d DE 16 BITS UTILIZADO
           ACCdLO                  ; NA ROTINA DE DIVISÃO

           temp                    ; CONTADOR TEMPORÁRIO UTILIZADO
                                   ; NA ROTINA DE DIVISÃO

           H_byte                  ; ACUMULADOR DE 16 BITS UTILIZADO
           L_byte                  ; P/ RETORNAR O VALOR DA ROTINA
                                   ; DE MULTIPLICAÇÃO

           mulplr                  ; OPERADOR P/ ROTINA DE MUTIPLICAÇÃO
           mulcnd                  ; OPERADOR P/ ROTINA DE MUTIPLICAÇÃO

           TEMPO0
           TEMPO1                  ; TEMPORIZADORES P/ ROTINA DE DELAY

           AUX                     ; REGISTRADOR AUXILIAR DE USO GERAL

           UNIDADE                 ; ARMAZENA VALOR DA UNIDADE DA TENSÃO
           DEZENA                  ; ARMAZENA VALOR DA DEZENA DA TENSÃO

   ENDC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
; *                DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>        ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    DEFINIÇÃO DOS BANCOS DE RAM                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           CONSTANTES INTERNAS                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE CONSTANTES FACILITA A PROGRAMAÇÃO E A MANUTENÇÃO.

;  ESTE PROGRAMA NÃO UTILIZA NENHUMA CONSTANTE.

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   DECLARAÇÃO DOS FLAGs DE SOFTWARE                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE FLAGs AJUDA NA PROGRAMAÇÃO E ECONOMIZA MEMÓRIA RAM.

;  ESTE PROGRAMA NÃO UTILIZA NENHUM FLAG DE USUÁRIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               ENTRADAS                                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS ENTRADAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

;  ESTE PROGRAMA UTILIZA APENAS UMA ENTRADA P/ O CONVERSOR A/D.
;  ESTA ENTRADA NÃO PRECISA SER DECLARADA, POIS O SOFTWARE NUNCA FAZ
;  REFERÊNCIA A ELA DE FORMA DIRETA, POIS O CANAL A/D A SER CONVERTIDO É
;  SELECIONADO NO REGISTRADOS ADCON0 DE FORMA BINÁRIA E NÃO ATRAVÉS DE
;  DEFINES. PORÉM PARA FACILITAR O ENTENDIMENTO DO HARDWARE VAMOS DECLARAR
;  ESTA ENTRADA NORMALMENTE.

#DEFINE    CAD             PORTA,0 ; ENTRADA A/D DO POTENCIÔMETRO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               SAÍDAS                                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS SAÍDAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    DISPLAY         PORTD   ; BARRAMENTO DE DADOS DO DISPLAY

#DEFINE    RS              PORTE,0 ; INDICA P/ O DISPLAY UM DADO OU COMANDO
                                   ; 1 -> DADO
                                   ; 0 -> COMANDO

#DEFINE    ENABLE          PORTE,1 ; SINAL DE ENABLE P/ DISPLAY
                                   ; ATIVO NA BORDA DE DESCIDA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  VETOR DE RESET DO MICROCONTROLADOR                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO INICIAL PARA EXECUÇÃO DO PROGRAMA

   ORG     0X0000                  ; ENDEREÇO DO VETOR DE RESET
   GOTO    CONFIG                  ; PULA PARA CONFIG DEVIDO A REGIÃO
                                   ; DESTINADA AS ROTINAS SEGUINTES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  ROTINA DE DELAY (DE 1MS ATÉ 256MS)                   *
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA É UMA ROTINA DE DELAY VARIÁVEL, COM DURAÇÃO DE 1MS X O VALOR PASSADO
;  EM WORK (W).

DELAY_MS
   MOVWF   TEMPO1                ; CARREGA TEMPO1 (UNIDADES DE MS)
   MOVLW   .250
   MOVWF   TEMPO0                ; CARREGA TEMPO0 (P/ CONTAR 1MS)

   CLRWDT                        ; LIMPA WDT (PERDE TEMPO)
   DECFSZ  TEMPO0,F              ; FIM DE TEMPO0 ?
   GOTO    $-2                   ; NÃO - VOLTA 2 INSTRUÇÕES
                                 ; SIM - PASSOU-SE 1MS
   DECFSZ  TEMPO1,F              ; FIM DE TEMPO1 ?
   GOTO    $-6                   ; NÃO - VOLTA 6 INSTRUÇÕES
                                 ; SIM
   RETURN                        ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DE UM CARACTER NO DISPLAY                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ENVIA UM CARACTER PARA O MÓDULO DE LCD. O CARACTER A SER
;  ESCRITO DEVE SER COLOCADO EM WORK (W) ANTES DE CHAMAR A ROTINA.

ESCREVE
   MOVWF   DISPLAY               ; ATUALIZA DISPLAY (PORTD)
   NOP                           ; PERDE 1US PARA ESTABILIZAÇÃO
   BSF     ENABLE                ; ENVIA UM PULSO DE ENABLE AO DISPLAY
   GOTO    $+1                   ; .
   BCF     ENABLE                ; .

   MOVLW   .1
   CALL    DELAY_MS              ; DELAY DE 1MS
   RETURN                        ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                AJUSTE DECIMAL                             *
; *               W [HEX] =  DEZENA [DEC] ; UNIDADE [DEC]                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA RECEBE UM ARGUMENTO PASSADO PELO WORK E RETORNA NAS VARIÁVEIS
;  DEZENA E UNIDADE O NÚMERO BCD CORRESPONDÊNTE AO PARÂMETRO PASSADO.

AJUSTE_DECIMAL
   MOVWF   AUX                   ; SALVA VALOR A CONVERTER EM AUX
   CLRF    UNIDADE
   CLRF    DEZENA                ; RESETA REGISTRADORES

   MOVF    AUX,F
   BTFSC   STATUS,Z              ; VALOR A CONVERTER = 0 ?
   RETURN                        ; SIM - RETORNA
                                 ; NÃO
   INCF    UNIDADE,F             ; INCREMENTA UNIDADE

   MOVF    UNIDADE,W
   XORLW   0X0A
   BTFSS   STATUS,Z              ; UNIDADE = 10d ?
   GOTO    $+3                   ; NÃO
                                 ; SIM
   CLRF    UNIDADE               ; RESETA UNIDADE
   INCF    DEZENA,F              ; INCREMENTA DEZENA

   DECFSZ  AUX,F                 ; FIM DA CONVERSÃO ?
   GOTO    $-.8                  ; NÃO - VOLTA P/ CONTINUAR CONVERSÃO
   RETURN                        ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           ROTINA DE DIVISÃO                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;*****************************************************************************
;                        Double Precision Division
;*****************************************************************************
;   Division : ACCb(16 bits) / ACCa(16 bits) -> ACCb(16 bits) with
```

```
;                                                  Remainder in ACCc (16 bits)
;      (a) Load the Denominator in location ACCaHI & ACCaLO ( 16 bits )
;      (b) Load the Numerator in location ACCbHI & ACCbLO ( 16 bits )
;      (c) CALL D_divF
;      (d) The 16 bit result is in location ACCbHI & ACCbLO
;      (e) The 16 bit Remainder is in locations ACCcHI & ACCcLO
;*****************************************************************************

D_divF
   MOVLW   .16
   MOVWF   temp                    ; CARREGA CONTADOR PARA DIVISÃO

   MOVF    ACCbHI,W
   MOVWF   ACCdHI
   MOVF    ACCbLO,W
   MOVWF   ACCdLO                  ; SALVA ACCb EM ACCd

   CLRF    ACCbHI
   CLRF    ACCbLO                  ; LIMPA ACCb

   CLRF    ACCcHI
   CLRF    ACCcLO                  ; LIMPA ACCc

DIV
   BCF     STATUS,C
   RLF     ACCdLO,F
   RLF     ACCdHI,F
   RLF     ACCcLO,F
   RLF     ACCcHI,F
   MOVF    ACCaHI,W
   SUBWF   ACCcHI,W                ;check if a>c
   BTFSS   STATUS,Z
   GOTO    NOCHK
   MOVF    ACCaLO,W
   SUBWF   ACCcLO,W                ;if msb equal then check lsb
NOCHK
   BTFSS   STATUS,C                ;carry set if c>a
   GOTO    NOGO
   MOVF    ACCaLO,W                ;c-a into c
   SUBWF   ACCcLO,F
   BTFSS   STATUS,C
   DECF    ACCcHI,F
   MOVF    ACCaHI,W
   SUBWF   ACCcHI,F
   BSF     STATUS,C                ;shift a 1 into b (result)
NOGO
   RLF     ACCbLO,F
   RLF     ACCbHI,F

   DECFSZ  temp,F                  ; FIM DA DIVISÃO ?
   GOTO    DIV                     ; NÃO - VOLTA P/ DIV
                                   ; SIM
   RETURN                          ; RETORNA


; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          ROTINA DE MULTIPLICAÇÃO                          *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;*****************************************************************************
;                  8x8 Software Multiplier
;               ( Fast Version : Straight Line Code )
;*****************************************************************************
;
;   The 16 bit result is stored in 2 bytes
; Before calling the subroutine " mpy ", the multiplier should
; be loaded in location " mulplr ", and the multiplicand in
; " mulcnd " . The 16 bit result is stored in locations
; H_byte & L_byte.
;       Performance :
;                       Program Memory  :  37 locations
;                       # of cycles     :  38
;                       Scratch RAM     :   0 locations
```

```
;*****************************************************************

; ****************************************
;  Define a macro for adding & right shifting
; ****************************************

mult    MACRO   bit               ; Begin macro

   BTFSC   mulplr,bit
   ADDWF   H_byte,F
   RRF     H_byte,F
   RRF     L_byte,F

   ENDM                           ; End of macro

; ****************************
;   Begin Multiplier Routine
; ****************************

mpy_F
   CLRF    H_byte
   CLRF    L_byte
   MOVF    mulcnd,W               ; move the multiplicand to W reg.
   BCF     STATUS,C               ; Clear carry bit in the status Reg.

   mult    0
   mult    1
   mult    2
   mult    3
   mult    4
   mult    5
   mult    6
   mult    7

   RETURN                         ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               CONFIGURAÇÕES INICIAIS DE HARDWARE E SOFTWARE           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  NESTA ROTINA SÃO INICIALIZADAS AS PORTAS DE I/O DO MICROCONTROLADOR E AS
;  CONFIGURAÇÕES DOS REGISTRADORES ESPECIAIS (SFR). A ROTINA INICIALIZA A
;  MÁQUINA E AGUARDA O ESTOURO DO WDT.

CONFIG
   CLRF    PORTA                  ; LIMPA O PORTA
   CLRF    PORTB                  ; LIMPA O PORTB
   CLRF    PORTC                  ; LIMPA O PORTC
   CLRF    PORTD                  ; LIMPA O PORTD
   CLRF    PORTE                  ; LIMPA O PORTE


   BANK1                          ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                  ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                  ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011000'
   MOVWF   TRISC                  ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                  ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                  ; CONFIGURA I/O DO PORTE


   MOVLW   B'11011011'
   MOVWF   OPTION_REG             ; CONFIGURA OPTIONS
                                  ; PULL-UPs DESABILITADOS
                                  ; INTER. NA BORDA DE SUBIDA DO RB0
```

```
                                ; TIMER0 INCREM. PELO CICLO DE MÁQUINA
                                ; WDT   - 1:8
                                ; TIMER - 1:1

   MOVLW   B'00000000'
   MOVWF   INTCON               ; CONFIGURA INTERRUPÇÕES
                                ; DESABILITA TODAS AS INTERRUPÇÕES
   MOVLW   B'00000111'
   MOVWF   CMCON                ; DESLIGA OS COMPARADORES

   MOVLW   B'00000100'
   MOVWF   ADCON1               ; CONFIGURA CONVERSOR A/D
                                ; RA0, RA1 E RA3 COMO ANALÓGICO
                                ; RA2, RA4 E RA5 COMO I/O DIGITAL
                                ; PORTE COMO I/O DIGITAL
                                ; JUSTIFICADO À ESQUERDA
                                ; 8 BITS EM ADRESH E 2 BITS EM ADRESL
                                ; Vref+ = VDD (+5V)
                                ; Vref- = GND ( 0V)

   BANK0                        ; SELECIONA BANCO 0 DA RAM

   MOVLW   B'01000001'
   MOVWF   ADCON0               ; CONFIGURA CONVERSOR A/D
                                ; VELOCIDADE -> Fosc/8
                                ; CANAL 0
                                ; MÓDULO LIGADO

;  AS INSTRUÇÕES A SEGUIR FAZEM COM QUE O PROGRAMA TRAVE QUANDO HOUVER UM
;  RESET OU POWER-UP, MAS PASSE DIRETO SE O RESET FOR POR WDT. DESTA FORMA,
;  SEMPRE QUE O PIC É LIGADO, O PROGRAMA TRAVA, AGUARDA UM ESTOURO DE WDT
;  E COMEÇA NOVAMENTE. ISTO EVITA PROBLEMAS NO START-UP DO PIC.

   BTFSC   STATUS,NOT_TO        ; RESET POR ESTOURO DE WATCHDOG TIMER ?
   GOTO    $                    ; NÃO - AGUARDA ESTOURO DO WDT
                                ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INICIALIZAÇÃO DA RAM                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA IRÁ LIMPAR TODA A RAM DO BANCO 0, INDO DE 0X20 A 0X7F.
;  EM SEGUIDA, AS VARIÁVEIS DE RAM DO PROGRAMA SÃO INICIALIZADAS.

   MOVLW   0X20
   MOVWF   FSR                  ; APONTA O ENDEREÇAMENTO INDIRETO PARA
                                ; A PRIMEIRA POSIÇÃO DA RAM
LIMPA_RAM
   CLRF    INDF                 ; LIMPA A POSIÇÃO
   INCF    FSR,F                ; INCREMENTA O PONTEIRO P/ A PRÓX. POS.
   MOVF    FSR,W
   XORLW   0X80                 ; COMPARA O PONTEIRO COM A ÚLT. POS. +1
   BTFSS   STATUS,Z             ; JÁ LIMPOU TODAS AS POSIÇÕES?
   GOTO    LIMPA_RAM            ; NÃO - LIMPA A PRÓXIMA POSIÇÃO
                                ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     CONFIGURAÇÕES INICIAIS DO DISPLAY                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA INICIALIZA O DISPLAY P/ COMUNICAÇÃO DE 8 VIAS, DISPLAY PARA 2
; LINHAS, CURSOR APAGADO E DESLOCAMENTO DO CURSOR À DIREITA.

INICIALIZACAO_DISPLAY
   BCF     RS                   ; SELECIONA O DISPLAY P/ COMANDOS

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO
   MOVLW   .3
   CALL    DELAY_MS             ; DELAY DE 3MS (EXIGIDO PELO DISPLAY)

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO
```

```
   MOVLW   0X30                    ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   B'00111000'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; INTERFACE DE 8 VIAS DE DADOS

   MOVLW   B'00000001'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; LIMPAR TODO O DISPLAY

   MOVLW   .1
   CALL    DELAY_MS                ; DELAY DE 1MS

   MOVLW   B'00001100'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; LIGAR O DISPLAY SEM CURSOR

   MOVLW   B'00000110'             ; ESCREVE COMANDO PARA INCREM.
   CALL    ESCREVE                 ; AUTOMÁTICO À DIREITA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                 ROTINA DE ESCRITA DA TELA PRINCIPAL                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A TELA PRINCIPAL DO PROGRAMA, COM AS FRASES:
;  LINHA 1 - " A/D Int.  (RA0)"
;  LINHA 2 - "        Volts   "

   MOVLW   0X81                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 0 / COLUNA 1

   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

                                   ; COMANDOS PARA ESCREVER AS
                                   ; LETRAS DE "A/D Int.  (P2)"
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   '/'
   CALL    ESCREVE
   MOVLW   'D'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   'I'
   CALL    ESCREVE
   MOVLW   'n'
   CALL    ESCREVE
   MOVLW   't'
   CALL    ESCREVE
   MOVLW   '.'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   '('
   CALL    ESCREVE
   MOVLW   'R'
   CALL    ESCREVE
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   '0'
   CALL    ESCREVE
   MOVLW   ')'
   CALL    ESCREVE

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XC7                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 7
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS
                                   ; COMANDOS PARA ESCREVER AS
                                   ; LETRAS DE "Volts"
   MOVLW   'V'
   CALL    ESCREVE
   MOVLW   'o'
   CALL    ESCREVE
   MOVLW   'l'
```

```
   CALL    ESCREVE
   MOVLW   't'
   CALL    ESCREVE
   MOVLW   's'
   CALL    ESCREVE

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            LOOP PRINCIPAL                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A ROTINA PRINCIPAL FICA CONVERTENDO O CANAL A/D, CALCULANDO O VALOR EM
;  VOLTS E MOSTRANDO NO DISPLAY.

LOOP
   CLRWDT                          ; LIMPA WATCHDOG TIMER

   BSF     ADCON0,GO               ; INICIA CONVERSÃO A/D
   BTFSC   ADCON0,GO               ; FIM DA CONVERSÃO ?
   GOTO    $-1                     ; NÃO - VOLTA 1 INSTRUÇÃO
                                   ; SIM
   MOVF    ADRESH,W                ; SALVA VALOR DA CONVERSÃO NO WORK
   MOVWF   mulplr                  ; CARREGA WORK EM mulplr

   MOVLW   .50
   MOVWF   mulcnd                  ; CARREGA 50d EM mulcnd

   CALL    mpy_F                   ; CHAMA ROTINA DE MULTIPLICAÇÃO

   MOVF    H_byte,W
   MOVWF   ACCbHI                  ; SALVA VALOR DA MULTIPLICAÇÃO
   MOVF    L_byte,W                ; EM ACCb PARA SER UTILIZADO NA
   MOVWF   ACCbLO                  ; ROTINA DE DIVISÃO

   CLRF    ACCaHI                  ; CARREGA ACCa COM 255d (FUNDO DE
   MOVLW   .255                    ; ESCALA DO CONVERSOR A/D)
   MOVWF   ACCaLO                  ; (ESTÃO SENDO UTILIZADOS 8 BITS)

   CALL    D_divF                  ; CHAMA ROTINA DE DIVISÃO

   MOVF    ACCbLO,W                ; FAZ O AJUSTE DECIMAL PARA
   CALL    AJUSTE_DECIMAL          ; MOSTRAR NO DISPLAY (LCD)

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XC3                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 3
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

   MOVF    DEZENA,W
   ADDLW   0X30                    ; CONVERTE BCD DA DEZENA EM ASCII
   CALL    ESCREVE                 ; ENVIA AO LCD

   MOVLW   ','
   CALL    ESCREVE                 ; ESCREVE UMA VIRGULA NO LCD

   MOVF    UNIDADE,W
   ADDLW   0X30                    ; CONVERTE BCD DA UNIDADE EM ASCII
   CALL    ESCREVE                 ; ENVIA AO LCD

   GOTO    LOOP                    ; VOLTA PARA LOOP

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           FIM DO PROGRAMA                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                             ; FIM DO PROGRAMA
```

Dicas e Comentários

Inicialmente notar que toda a estrutura e rotinas utilizadas para a escrita no LCD são as mesmas já aplicadas na experiência anterior.

Observar também que não foi utilizada nenhuma interrupção neste programa. Por isso, o programa permanece parado em um pequeno loop enquanto a conversão não termina. Isto é checado através do bit **ADCON0<GO/DONE>**.

Outra rotina bem interessante que aparece neste sistema é a de conversão de um número qualquer (limitado entre 0 e 99) em dois dígitos separados, facilitando assim a escrita no LCD. Esta rotina devolve os dígitos nas variáveis UNIDADE e DEZENA. Não esquecer que antes de transmitir um valor decimal ao LCD, deve-se converte-lo em um caractere ASCII.

Para facilitar as contas e não utilizarmos números fracionários, a conversão para Volts é feita considerando-se 50 no lugar de 5,0 de forma que ao enviar o valor final ao LCD é simplesmente colocada uma vírgula entre os dois dígitos.

Exercícios Propostos

1. Simule que a entrada analógica é um sensor de temperatura linear que deve marcar de 10 a 80°C;
2. Altere o exemplo para indicar a tensão entre 0 e 2,50V, utilizando 10 bits de resolução. Para isso, faça uso da tensão de referência externa existente na placa de periféricos (pino RA3).

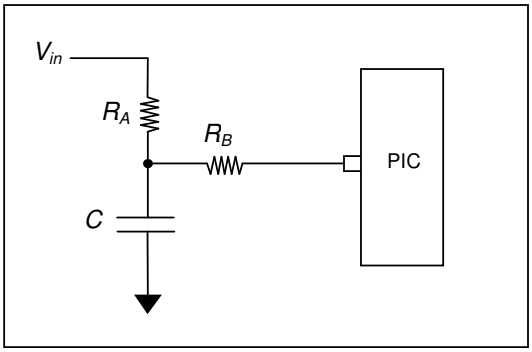
Capítulo 14 - Experiência 12 – Conversão A/D via RC

Objetivo

Nesta experiência será vista uma forma de conversão A/D fundamentada no tempo de carga de um capacitor. A vantagem desta técnica é que o microcontrolador não necessita possuir um conversor A/D interno para estimar o valor de uma variável analógica.

Descrição

A técnica descrita nesta experiência baseia-se na carga e descarga de um capacitor. A idéia consiste em medir, através do microcontrolador, o tempo de carga de um capacitor num circuito RC. Veja o exemplo de hardware colocado a seguir:



Admitindo-se que o pino do PIC está configurado como entrada, o tempo de carga do capacitor *C* está relacionado com o valor de entrada (V_{in}), do resistor R_A e do próprio capacitor *C*. O resistor R_B não interfere no tempo de carga, pois o pino do PIC está em alta impedância (entrada). Já se o pino do PIC estiver configurado como saída em nível lógico 0 o capacitor tende a se descarregar pelo resistor R_B e carregar pelo resistor R_A. Porém, vamos admitir que o valor do resistor R_B seja muito menor do que o de R_A e, portanto, nesta configuração, podemos desprezar a carga proveniente do resistor R_A e admitir que o capacitor *C* apenas se descarrega através de R_B. Em resumo, o capacitor se carrega através de R_A (pino como entrada) e se descarrega através de R_B (pino como saída em 0) sendo que o tempo de carga/descarga depende do próprio valor do capacitor, da tensão de entrada (V_{in}) e do resistor em questão.

Como funciona então a conversão A/D?

1. O software deve configurar o pino do PIC como saída em 0;
2. Esperar o tempo de descarrega do capacitor *C*. Este tempo deve ser garantido por software conforme os valores dos componentes utilizados;
3. Configurar o pino como entrada, ou seja, permitir a carga do capacitor;
4. Contar o tempo que o capacitor leva para que o PIC entenda nível lógico 1, ou seja, contar o tempo de carga do capacitor;
5. Repetir o processo para uma nova conversão.

O tempo de carga do capacitor será inversamente proporcional à tensão de entrada.

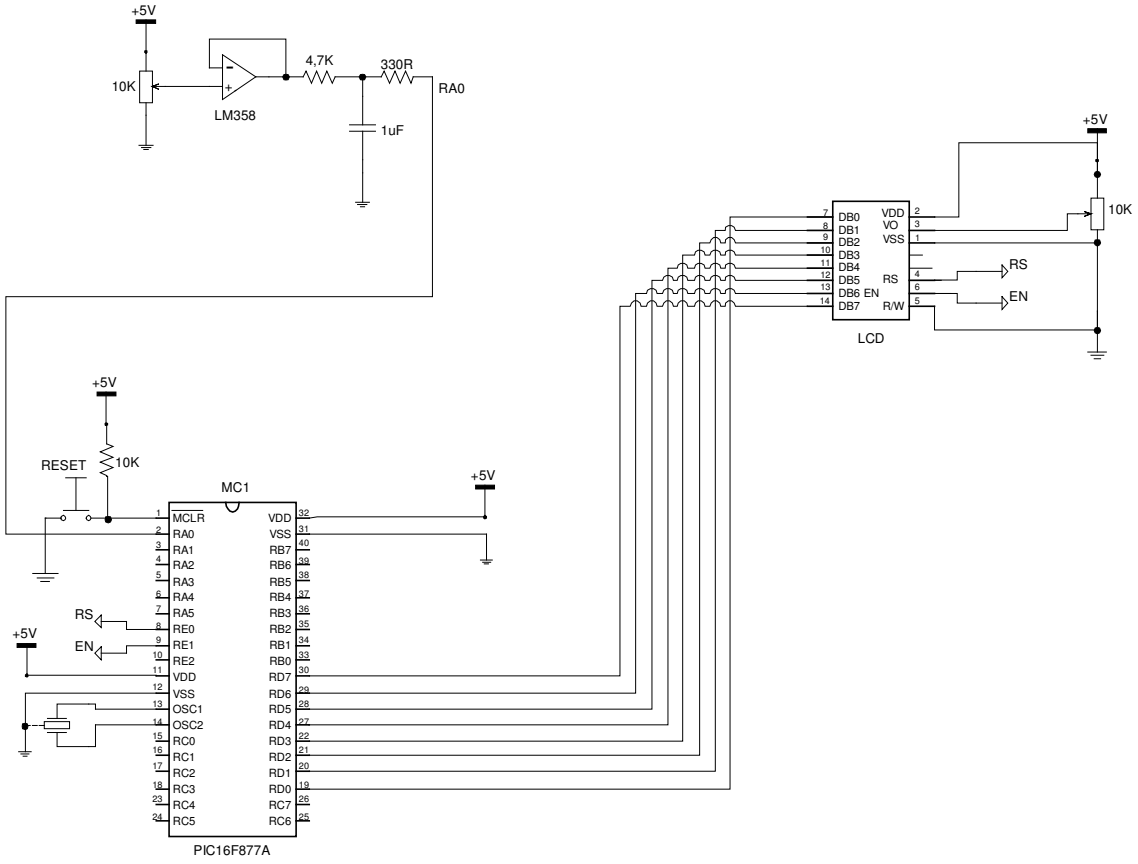
Admitindo-se que a tensão de entrada não varie durante a conversão A/D, o modelo matemático aproximado para a curva de carga do capacitor é

$$V_{cap}(t) = V_{in}\left(1 - e^{-t/R_A C}\right).$$

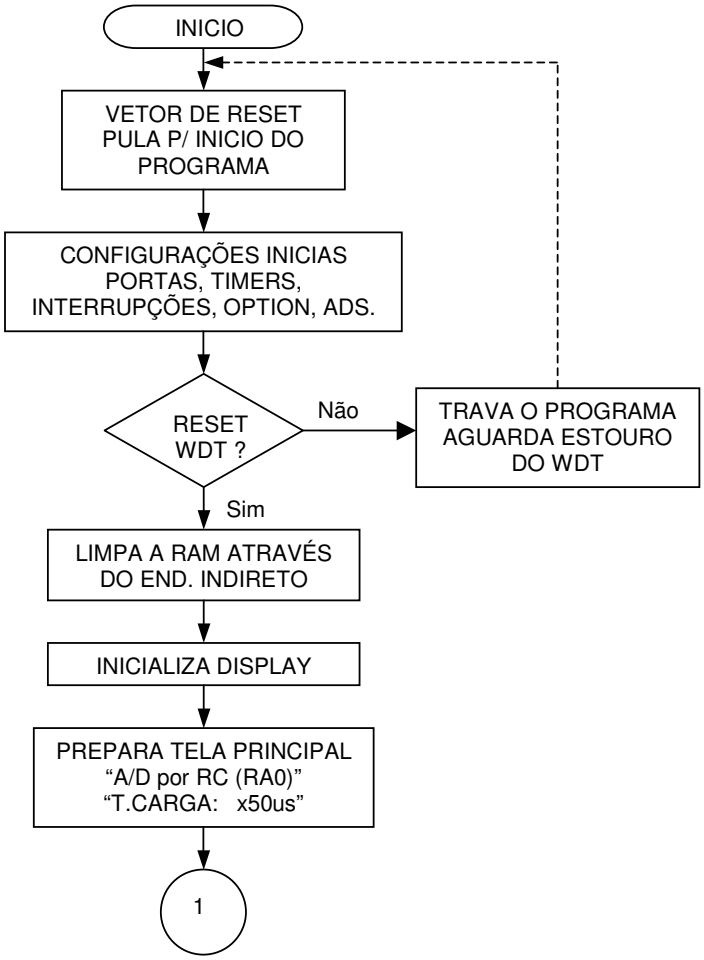
Assim, conhecidos os valores do resistor, do capacitor e do tempo de carga, pode-se estimar a tensão de entrada.

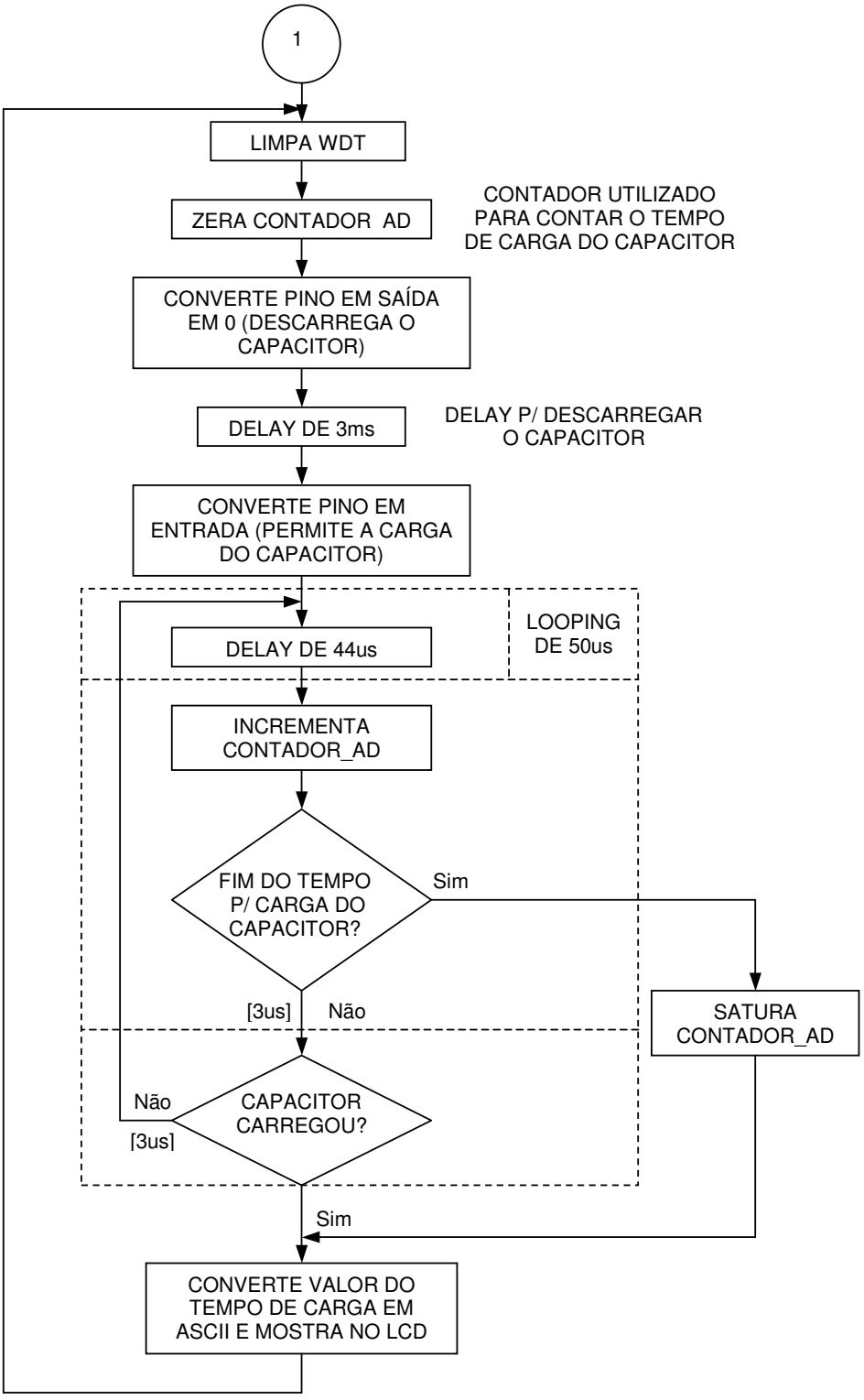
O exemplo de software da experiência calcula o tempo de carga do capacitor e mostra este tempo no display LCD. O software não calcula a tensão de entrada, apenas o tempo de carga do capacitor. O tempo de carga pode ser alterado através do potenciômetro do MCMASTER.

Esquema Elétrico



Fluxograma


INICIO
VETOR DE RESET
PULA P/ INICIO DO
PROGRAMA
CONFIGURAÇÕES INICIAS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION, ADS.
RESET
WDT ?
Não
TRAVA O PROGRAMA
AGUARDA ESTOURO
DO WDT
Sim
LIMPA A RAM ATRAVÉS
DO END. INDIRETO
INICIALIZA DISPLAY
PREPARA TELA PRINCIPAL
“A/D por RC (RA0)”
“T.CARGA: x50us”
1


1
LIMPA WDT
ZERA CONTADOR AD
CONTADOR UTILIZADO PARA CONTAR O TEMPO DE CARGA DO CAPACITOR
CONVERTE PINO EM SAÍDA EM 0 (DESCARREGA O CAPACITOR)
DELAY DE 3ms
DELAY P/ DESCARREGAR O CAPACITOR
CONVERTE PINO EM ENTRADA (PERMITE A CARGA DO CAPACITOR)
LOOPING DE 50us
DELAY DE 44us
INCREMENTA CONTADOR_AD
FIM DO TEMPO P/ CARGA DO CAPACITOR?
Sim
[3us]
Não
SATURA CONTADOR_AD
Não
CAPACITOR CARREGOU?
[3us]
Sim
CONVERTE VALOR DO TEMPO DE CARGA EM ASCII E MOSTRA NO LCD

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *                 EXPERIÊNCIA 12 - CONVERSÃO A/D VIA RC                   *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            DESCRIÇÃO GERAL                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE EXEMPLO FOI ELABORADO PARA EXPLICAR O FUNCIONAMENTO DE UM TIPO DE
;  CONVERSOR A/D FUNDAMENTADO NO TEMPO DE CARGA DE UM CAPACITOR. O TEMPO DE
;  CARGA DO CAPACITOR É MOSTRADO NO LCD E É INVERSAMENTE PROPORCIONAL À
;  TENSÃO APLICADA ATRVÉS DO POTENCIÔMETRO.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       CONFIGURAÇÕES PARA GRAVAÇÃO                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_ON & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>        ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            DEFINIÇÃO DAS VARIÁVEIS                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE BLOCO DE VARIÁVEIS ESTÁ LOCALIZADO LOGO NO INÍCIO DO BANCO 0

   CBLOCK  0X20                   ; POSIÇÃO INICIAL DA RAM

           TEMPO1
           TEMPO0                 ; CONTADORES P/ DELAY

           FILTRO_BOTOES          ; FILTRO PARA RUIDOS DOS BOTÕES

           CONTADOR_AD            ; CONTADOR PARA CONVERSOR A/D

           AUX                    ; REGISTRADOR AUXILIAR DE USO GERAL

   ENDC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     DEFINIÇÃO DOS BANCOS DE RAM                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           CONSTANTES INTERNAS                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE CONSTANTES FACILITA A PROGRAMAÇÃO E A MANUTENÇÃO.

FILTRO_TECLA       EQU     .200            ; FILTRO P/ EVITAR RUIDOS DOS BOTÕES
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    DECLARAÇÃO DOS FLAGs DE SOFTWARE                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE FLAGs AJUDA NA PROGRAMAÇÃO E ECONOMIZA MEMÓRIA RAM.

;  ESTE PROGRAMA NÃO UTILIZA NENHUM FLAG DE USUÁRIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                ENTRADAS                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS ENTRADAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    PINO_AD         TRISA,0 ; PINO P/ LEITURA DO RC
                                   ; 0 -> FORÇA A DESCARGA DO CAPACITOR
                                   ; 1 -> LIBERA A CARGA DO CAPACITOR

#DEFINE    CAD             PORTA,0 ; PINO P/ LEITURA DO CONV. A/D
                                   ; 0 -> CAPACITOR DESCARREGADO
                                   ; 1 -> CAPACITOR CARREGADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                 SAÍDAS                                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS SAÍDAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    DISPLAY         PORTD   ; BARRAMENTO DE DADOS DO DISPLAY

#DEFINE    RS              PORTE,0 ; INDICA P/ DISPLAY UM DADO OU COMANDO
                                   ; 1 -> DADO
                                   ; 0 -> COMANDO

#DEFINE    ENABLE          PORTE,1 ; SINAL DE ENABLE P/ DISPLAY
                                   ; ATIVO NA BORDA DE SUBIDA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  VETOR DE RESET DO MICROCONTROLADOR                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO INICIAL PARA EXECUÇÃO DO PROGRAMA

   ORG     0X000                   ; ENDEREÇO DO VETOR DE RESET
   GOTO    CONFIG                  ; PULA PARA CONFIG DEVIDO A REGIÃO
                                   ; DESTINADA AS ROTINAS SEGUINTES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  ROTINA DE DELAY (DE 1MS ATÉ 256MS)                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA É UMA ROTINA DE DELAY VARIÁVEL, COM DURAÇÃO DE 1MS X O VALOR PASSADO
;  EM WORK (W).

DELAY_MS
   MOVWF   TEMPO1                  ; CARREGA TEMPO1 (UNIDADES DE MS)
   MOVLW   .250
   MOVWF   TEMPO0                  ; CARREGA TEMPO0 (P/ CONTAR 1MS)

   CLRWDT                          ; LIMPA WDT (PERDE TEMPO)
   DECFSZ  TEMPO0,F                ; FIM DE TEMPO0 ?
   GOTO    $-2                     ; NÃO - VOLTA 2 INSTRUÇÕES
                                   ; SIM - PASSOU-SE 1MS
   DECFSZ  TEMPO1,F                ; FIM DE TEMPO1 ?
   GOTO    $-6                     ; NÃO - VOLTA 6 INSTRUÇÕES
                                   ; SIM
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DE UM CARACTER NO DISPLAY                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ENVIA UM CARACTER PARA O MÓDULO DE LCD. O CARACTER A SER
;  ESCRITO DEVE SER COLOCADO EM WORK (W) ANTES DE CHAMAR A ROTINA.

ESCREVE
```

```
   MOVWF   DISPLAY                ; ATUALIZA DISPLAY (PORTD)
   NOP                            ; PERDE 1US PARA ESTABILIZAÇÃO
   BSF     ENABLE                 ; ENVIA UM PULSO DE ENABLE AO DISPLAY
   GOTO    $+1                    ; .
   BCF     ENABLE                 ; .

   MOVLW   .1
   CALL    DELAY_MS               ; DELAY DE 1MS
   RETURN                         ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               CONFIGURAÇÕES INICIAIS DE HARDWARE E SOFTWARE            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  NESTA ROTINA SÃO INICIALIZADAS AS PORTAS DE I/O DO MICROCONTROLADOR E AS
;  CONFIGURAÇÕES DOS REGISTRADORES ESPECIAIS (SFR). A ROTINA INICIALIZA AS
;  VARIÁVEIS DE RAM E AGUARDA O ESTOURO DO WDT.

CONFIG
   CLRF    PORTA                  ; LIMPA O PORTA
   CLRF    PORTB                  ; LIMPA O PORTB
   CLRF    PORTC                  ; LIMPA O PORTC
   CLRF    PORTD                  ; LIMPA O PORTD
   CLRF    PORTE                  ; LIMPA O PORTE


   BANK1                          ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                  ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                  ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011000'
   MOVWF   TRISC                  ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                  ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                  ; CONFIGURA I/O DO PORTE

   MOVLW   B'11011111'
   MOVWF   OPTION_REG             ; CONFIGURA OPTIONS
                                  ; PULL-UPs DESABILITADOS
                                  ; INTER. NA BORDA DE SUBIDA DO RB0
                                  ; TIMER0 INCREM. PELO CICLO DE MÁQUINA
                                  ; WDT   - 1:128
                                  ; TIMER - 1:1
   MOVLW   B'00000111'
   MOVWF   CMCON                  ; DESLIGA OS COMPARADORES

   MOVLW   B'00000000'
   MOVWF   INTCON                 ; CONFIGURA INTERRUPÇÕES
                                  ; DESABILITADA TODAS AS INTERRUPÇÕES

   MOVLW   B'00000111'
   MOVWF   ADCON1                 ; CONFIGURA CONVERSOR A/D
                                  ; CONFIGURA PORTA COM I/O DIGITAL

   BANK0                          ; SELECIONA BANCO 0 DA RAM

;  AS INTRUÇÕES A SEGUIR FAZEM COM QUE O PROGRAMA TRAVE QUANDO HOUVER UM
;  RESET OU POWER-UP, MAS PASSE DIRETO SE O RESET FOR POR WDT. DESTA FORMA,
;  SEMPRE QUE O PIC É LIGADO, O PROGRAMA TRAVA, AGUARDA UM ESTOURO DE WDT
;  E COMEÇA NOVAMENTE. ISTO EVITA PROBLEMAS NO START-UP DO PIC.

   BTFSC   STATUS,NOT_TO          ; RESET POR ESTOURO DE WATCHDOG TIMER ?
   GOTO    $                      ; NÃO - AGUARDA ESTOURO DO WDT
                                  ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INICIALIZAÇÃO DA RAM                             *
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA IRÁ LIMPARÁ TODA A RAM DO BANCO 0, INDO DE 0X20 A 0X7F

   MOVLW   0X20
   MOVWF   FSR                     ; APONTA O ENDEREÇAMENTO INDIRETO PARA
                                   ; A PRIMEIRA POSIÇÃO DA RAM
LIMPA_RAM
   CLRF    INDF                    ; LIMPA A POSIÇÃO
   INCF    FSR,F                   ; INCREMENTA O PONTEIRO P/ A PRÓX. POS.
   MOVF    FSR,W
   XORLW   0X80                    ; COMPARA O PONTEIRO COM A ÚLT. POS. +1
   BTFSS   STATUS,Z                ; JÁ LIMPOU TODAS AS POSIÇÕES?
   GOTO    LIMPA_RAM               ; NÃO - LIMPA A PRÓXIMA POSIÇÃO
                                   ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     CONFIGURAÇÕES INICIAIS DO DISPLAY                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA INICIALIZA O DISPLAY P/ COMUNICAÇÃO DE 8 VIAS, DISPLAY PARA 2
; LINHAS, CURSOR APAGADO E DESLOCAMENTO DO CURSOR À DIREITA.

INICIALIZACAO_DISPLAY
   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS

   MOVLW   0X30                    ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   .3
   CALL    DELAY_MS                ; DELAY DE 3MS (EXIGIDO PELO DISPLAY)

   MOVLW   0X30                    ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   0X30                    ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   B'00111000'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; INTERFACE DE 8 VIAS DE DADOS

   MOVLW   B'00000001'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; LIMPAR TODO O DISPLAY

   MOVLW   .1
   CALL    DELAY_MS                ; DELAY DE 1MS

   MOVLW   B'00001100'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; LIGAR O DISPLAY SEM CURSOR

   MOVLW   B'00000110'             ; ESCREVE COMANDO PARA INCREM.
   CALL    ESCREVE                 ; AUTOMÁTICO A ESQUERDA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               ROTINA DE ESCRITA DA TELA PRINCIPAL                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A TELA PRINCIPAL DO PROGRAMA, COM AS FRASES:
;  LINHA 1 - "A/D por RC (RA0)"
;  LINHA 2 - "T.CARGA:   x50us"

   MOVLW   0X80                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 0 / COLUNA 0

   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS
                                   ; COMANDOS PARA ESCREVER AS
                                   ; LETRAS DE "A/D por RC (RA0)"
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   '/'
   CALL    ESCREVE
   MOVLW   'D'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
```

```
   MOVLW   'p'
   CALL    ESCREVE
   MOVLW   'o'
   CALL    ESCREVE
   MOVLW   'r'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   'R'
   CALL    ESCREVE
   MOVLW   'C'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   '('
   CALL    ESCREVE
   MOVLW   'R'
   CALL    ESCREVE
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   '0'
   CALL    ESCREVE
   MOVLW   ')'
   CALL    ESCREVE

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS

   MOVLW   0XC0                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 0

   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS
                                   ; COMANDOS PARA ESCREVER AS
                                   ; LETRAS DE "T.CARGA:   x50us"
   MOVLW   'T'
   CALL    ESCREVE
   MOVLW   '.'
   CALL    ESCREVE
   MOVLW   'C'
   CALL    ESCREVE
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   'R'
   CALL    ESCREVE
   MOVLW   'G'
   CALL    ESCREVE
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   ':'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   'x'
   CALL    ESCREVE
   MOVLW   '5'
   CALL    ESCREVE
   MOVLW   '0'
   CALL    ESCREVE
   MOVLW   'u'
   CALL    ESCREVE
   MOVLW   's'
   CALL    ESCREVE

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *      ROTINA PARA DESCARREGAR O CAPACITOR DE LEITURA DO CONVERSOR A/D    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA CONVERTE O PINO DO MICROCONTROLADOR EM SAÍDA COM NÍVEL LÓGICO 0
; E AGUARDA QUE O CAPACITOR SE DESCARREGUE. EM SEGUIDA O PINO É CONVERTIDO
; NOVAMENTE EM ENTRADA PARA PERMITIR QUE O CAPACITOR SE CARREGUE.
```

```
DESCARGA_CAPACITOR
   CLRWDT                          ; LIMPA WATCHDOG TIMER

   CLRF    CONTADOR_AD             ; ZERA O CONTADOR DE TEMPO DE CARGA
                                   ; DO CAPACITOR

   BANK1                           ; SELECIONA BANCO 1 DA RAM
   BCF     PINO_AD                 ; TRANSFORMA PINO EM SAIDA
   BANK0                           ; VOLTA P/ BANCO 0 DA RAM
   BCF     CAD                     ; DESCARREGA O CAPACITOR

   MOVLW   .3
   CALL    DELAY_MS                ; CHAMA ROTINA DE DELAY (3ms)
                                   ; TEMPO NECESSÁRIO P/ DESCARGA
                                   ; DO CAPACITOR

   BANK1                           ; SELECIONA BANCO 1 DA RAM
   BSF     PINO_AD                 ; TRANSFORMA PINO EM ENTRADA
   BANK0                           ; VOLTA P/ BANCO 0 DA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     LOOP P/ ESPERAR CARGA DO CAPACITOR                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; O TEMPO CONTA O TEMPO QUE O CAPACITOR LEVA PARA ATINGIR UM NÍVEL DE TENSÃO
; SUFICIENTE PARA QUE O MICROCONTROLADOR ENTENDA NÍVEL LÓGICO 1 NA ENTRADA TTL
; DO PINO RA1. CASO O CAPACITOR NUNCA SE DEMORE MAIS DO QUE 256 CICLOS DESTE
; LOOP, A ROTINA DESVIA PARA UMA ROTINA DE SATURAÇÃO.
; O LOOP DA ROTINA É DE 50us (CRISTAL DE 4MHz).

LOOP_CAD
   NOP                             ; [1us]

   MOVLW   .14                     ; [2us]
   MOVWF   AUX                     ; [3us] - CARREGA AUX COM 14d
   DECFSZ  AUX,F
   GOTO    $-1                     ; [4us] À [44us] - DELAY

   INCFSZ  CONTADOR_AD,F           ; INCREM. CONTADOR E VERIFICA ESTOURO
   GOTO    $+2                     ; NÃO HOUVE ESTOURO - PULA 1 INSTRUÇÃO
   GOTO    SATURACAO               ; HOUVE ESTOURO - PULA P/ SATURAÇÃO

   BTFSS   CAD                     ; CAPACITOR JÁ CARREGOU ?
   GOTO    LOOP_CAD                ; NÃO - VOLTA P/ LOOP_CAD
   GOTO    MOSTRA_CONTADOR         ; SIM - MOSTRA TEMPO DE CARGA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              MOSTRA O TEMPO DE CARGA DO CAPACITOR NO LCD             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA MOSTRA O TEMPO DE CARGA DO CAPACITOR EM HAXADECIMAL NO LCD.
; CASO O CAPACITOR NÃO TENHA SE CARREGADO, A ROTINA DE SATURAÇÃO GARANTE
; UM VALOR MÁXIMO PARA O TEMPO DE CARGA (0xFF).

SATURACAO
   MOVLW   0XFF
   MOVWF   CONTADOR_AD             ; SATURA O CONTADOR
                                   ; (CAPACITOR NÃO CARREGOU)

MOSTRA_CONTADOR
   MOVLW   0XC9                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 9

   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

   SWAPF   CONTADOR_AD,W           ; INVERTE NIBLE DO CONTADOR_AD
   ANDLW   B'00001111'             ; MASCARA BITS MAIS SIGNIFICATIVOS
   MOVWF   AUX                     ; SALVA EM AUXILIAR

   MOVLW   0X0A
   SUBWF   AUX,W                   ; AUX - 10d (ATUALIZA FLAG DE CARRY)
   MOVLW   0X30                    ; CARREGA WORK COM 30h
   BTFSC   STATUS,C                ; RESULTADO E POSITIVO? (É UMA LETRA?)
```

```
   MOVLW   0X37                    ; SIM - CARREGA WORK COM 37h
                                   ; NÃO - WORK FICA COM 30h (NÚMERO)
   ADDWF   AUX,W                   ; SOMA O WORK AO AUXILIAR
                                   ; (CONVERSÃO ASCII)

   CALL    ESCREVE                 ; MOSTRA NO DISPLAY

   MOVF    CONTADOR_AD,W           ; CARREGA NO WORK O CONTADOR_AD
   ANDLW   B'00001111'             ; MASCARA BITS MAIS SIGNIFICATIVOS
   MOVWF   AUX                     ; SALVA EM AUXILIAR

   MOVLW   0X0A
   SUBWF   AUX,W                   ; AUX - 10d (ATUALIZA FLAG DE CARRY)
   MOVLW   0X30                    ; CARREGA WORK COM 30h
   BTFSC   STATUS,C                ; RESULTADO E POSITIVO? (É UMA LETRA?)
   MOVLW   0X37                    ; SIM - CARREGA WORK COM 37h
                                   ; NÃO - WORK FICA COM 30h (NÚMERO)
   ADDWF   AUX,W                   ; SOMA O WORK AO AUXILIAR
                                   ; (CONVERSÃO ASCII)

   CALL    ESCREVE                 ; MOSTRA NO DISPLAY

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS

   GOTO    DESCARGA_CAPACITOR      ; VOLTA P/ DESCARREGAR O CAPACITOR

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          FIM DO PROGRAMA                                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                             ; FIM DO PROGRAMA
```

Dicas e Comentários

Este tipo de conversor A/D não apresenta uma boa precisão, além de apresentar uma série de inconvenientes:

- Note que a tensão sobre o capacitor não varia linearmente no tempo e, portanto, este tipo de conversor A/D não é linear;
- O valor da conversão, ou seja, o tempo de carga do capacitor está sujeito às variações dos componentes envolvidos.
- Normalmente o capacitor é muito suscetível a variações térmicas;
- A tensão de entrada deve ser suficientemente alta para que o PIC entenda nível lógico 1, por isso este conversor não funciona no range completo de 0 a 5V;
- O valor de tensão necessário para que o PIC entenda nível lógico 1 pode variar em função da pastilha, da tensão da fonte (alimentação do PIC) e do tipo de pino (TTL/ST).

Como dica, podemos sugerir:

- Utilizar R_B pelo menos 10 vezes menor que R_A;
- Não utilizar capacitores maiores do que $1\mu F$;
- Dar preferência ao uso de capacitores de tântalo ou cerâmico;
- Não discretizar mais do que oito níveis.

Exercícios Propostos

1. Admitindo que a tensão de entrada varia entre 0 e 5V, estimar através do tempo de carga do capacitor se a tensão de entrada encontra-se abaixo de 1,25V, entre 1,25V e 2,5V, entre 2,5V e 3,75V ou acima de 3,75V.

Capítulo 15 - Experiência 13 – Leitura de jumpers via RC

Objetivo

O objetivo desta experiência é aplicar a técnica apresentada na aula anterior para viabilizar a leitura de jumpers através de um único pino do microcontrolador.

Descrição

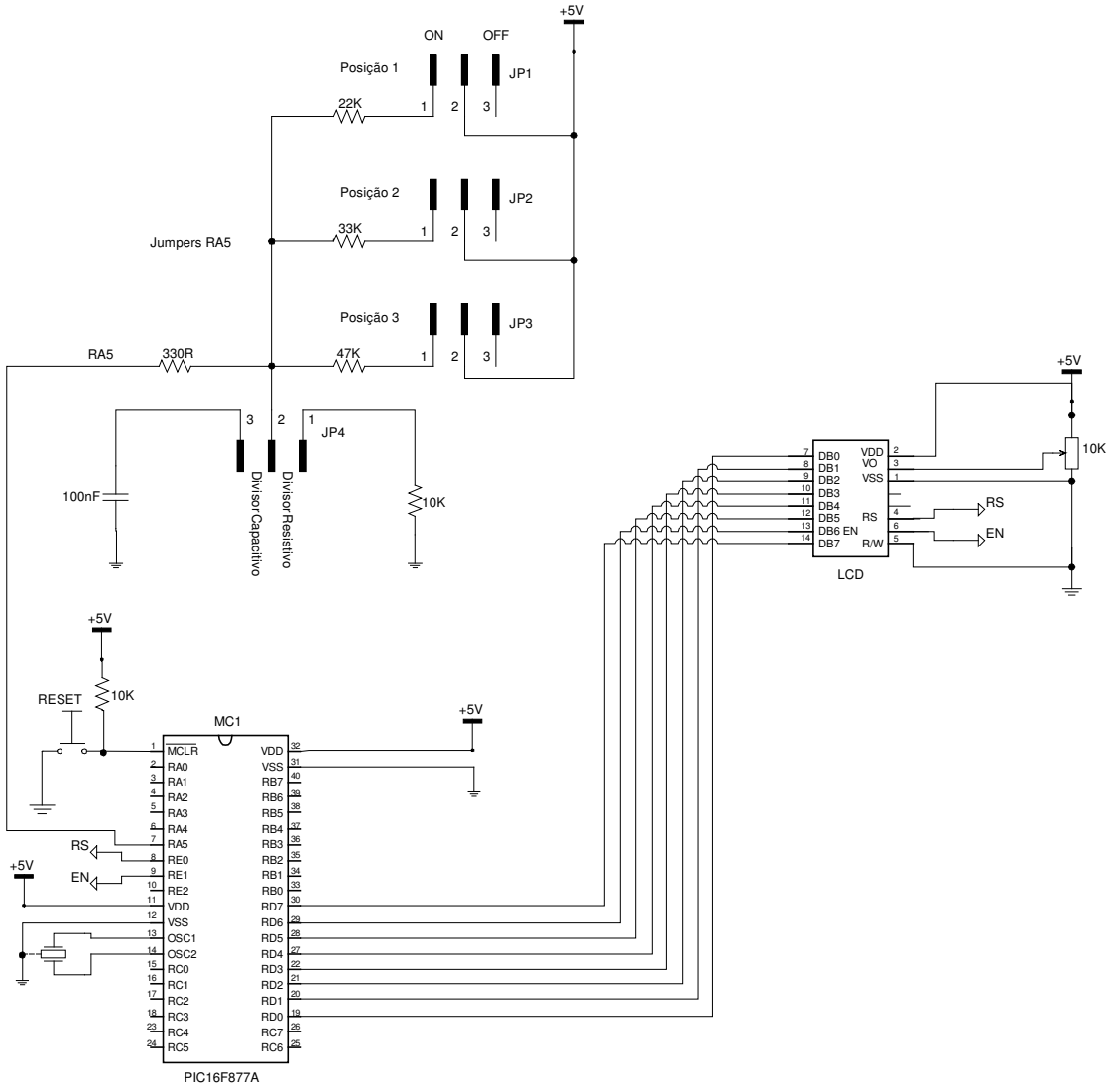
Assim como na experiência anterior, este exemplo foi elaborado utilizando a técnica de conversão A/D através de um circuito RC.

Diferente do caso anterior, a tensão de entrada nesta experiência é fixa e o que varia é o valor do resistor que fornece a carga do capacitor. Utilizando os jumpers A, B e C do MCMASTER pode-se alterar o valor do resistor de carga e conseqüentemente o tempo de carga do capacitor. Como são 3 jumpers, existem até 8 combinações possíveis que alteram o tempo de carga do capacitor.

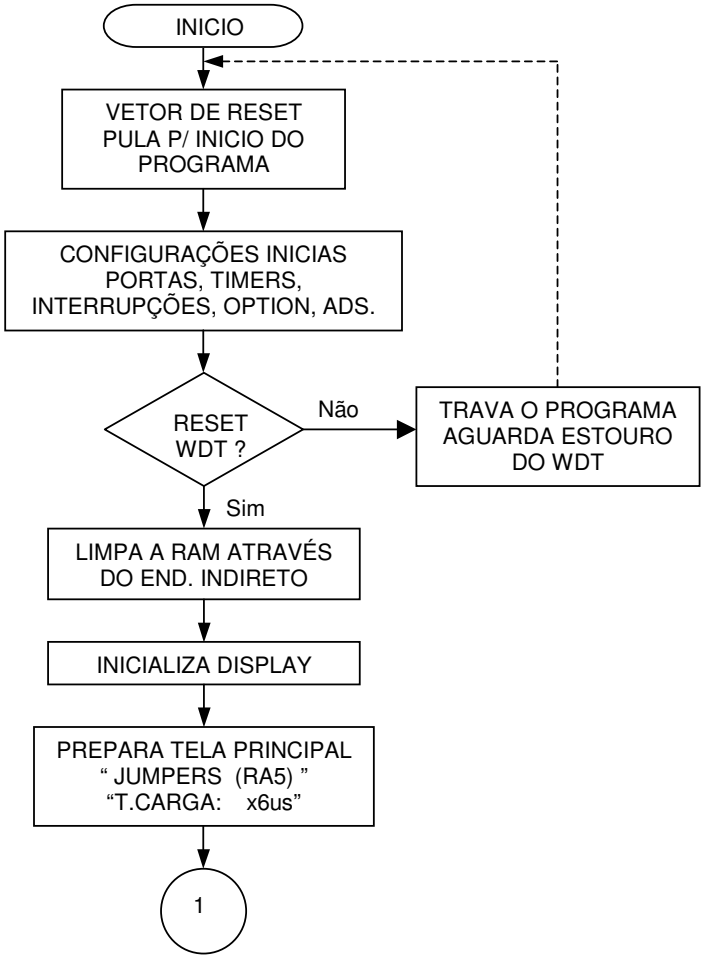
O quarto jumper deve estar selecionado na posição resistor/capacitor para que o exemplo possa ser utilizado.

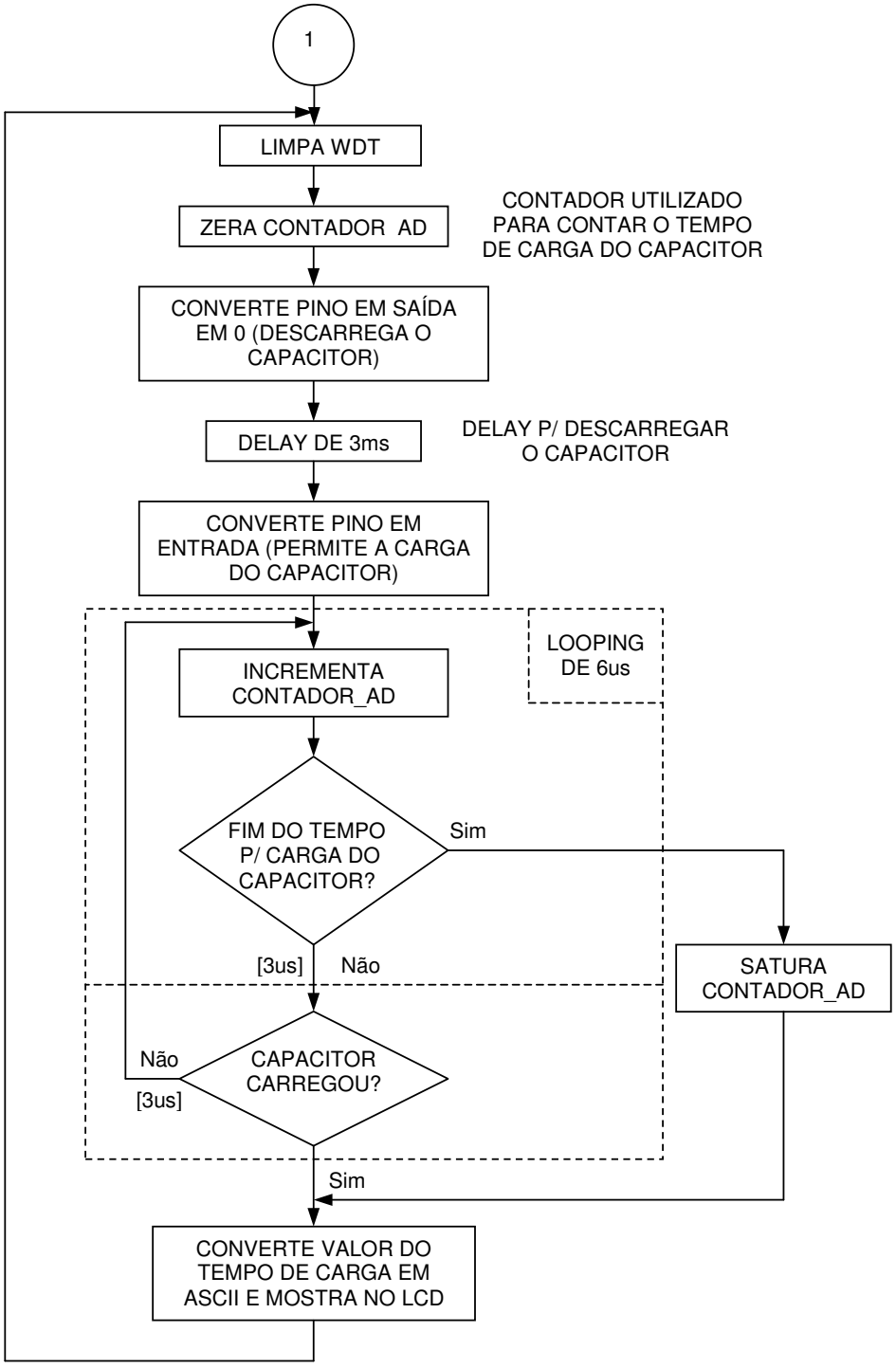
O software calcula o tempo de carga do capacitor e mostra o valor no display LCD. O interessante desta técnica é que ao invés de utilizar 3 pinos do microcontrolador, um para cada jumper, ela utiliza apenas um.

Esquema Elétrico



Fluxograma


INICIO
VETOR DE RESET
PULA P/ INICIO DO
PROGRAMA
CONFIGURAÇÕES INICIAS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION, ADS.
RESET
WDT ?
Não
TRAVA O PROGRAMA
AGUARDA ESTOURO
DO WDT
Sim
LIMPA A RAM ATRAVÉS
DO END. INDIRETO
INICIALIZA DISPLAY
PREPARA TELA PRINCIPAL
“ JUMPERS (RA5) ”
“T.CARGA: x6us”
1


1
LIMPA WDT
ZERA CONTADOR AD
CONTADOR UTILIZADO PARA CONTAR O TEMPO DE CARGA DO CAPACITOR
CONVERTE PINO EM SAÍDA EM 0 (DESCARREGA O CAPACITOR)
DELAY DE 3ms
DELAY P/ DESCARREGAR O CAPACITOR
CONVERTE PINO EM ENTRADA (PERMITE A CARGA DO CAPACITOR)
LOOPING DE 6us
INCREMENTA CONTADOR_AD
FIM DO TEMPO P/ CARGA DO CAPACITOR?
Sim
[3us]
Não
SATURA CONTADOR_AD
Não
[3us]
CAPACITOR CARREGOU?
Sim
CONVERTE VALOR DO TEMPO DE CARGA EM ASCII E MOSTRA NO LCD

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *                EXPERIÊNCIA 13 - LEITURA DE JUMPERS VIA RC               *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                             DESCRIÇÃO GERAL                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE EXEMPLO FOI ELABORADO PARA EXPLICAR O FUNCIONAMENTO DE UM TIPO DE
;  CONVERSOR A/D FUNDAMENTADO NO TEMPO DE CARGA DE UM CAPACITOR. O TEMPO DE
;  CARGA DO CAPACITOR É MOSTRADO NO LCD E É INVERSAMENTE PROPORCIONAL À
;  TENSÃO APLICADA. O TEMPO DE CARGA PODE SER ALTERADO UTILIZANDO OS JUMPERS
;  LIGADOS AO PINO RA5. COM SÃO TRÊS JUMPERS EXISTEM 8 COMBINAÇÕES POSSÍVEIS
;  QUE ALTERAM O TEMPO DE CARGA DO CAPACITOR. O QUARTO JUMPER DEVE ESTAR
;  SELECIONADO NA POSIÇÃO RESISTOR/CAPACITOR PARA QUE O EXEMPLO POSSA SER
;  UTILIZADO.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        CONFIGURAÇÕES PARA GRAVAÇÃO                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_ON & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>        ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           DEFINIÇÃO DAS VARIÁVEIS                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE BLOCO DE VARIÁVEIS ESTÁ LOCALIZADO LOGO NO INÍCIO DO BANCO 0

   CBLOCK  0X20                   ; POSIÇÃO INICIAL DA RAM

           TEMPO1
           TEMPO0                 ; CONTADORES P/ DELAY

           FILTRO_BOTOES          ; FILTRO PARA RUIDOS DOS BOTÕES

           CONTADOR_AD            ; CONTADOR PARA CONVERSOR A/D

           AUX                    ; REGISTRADOR AUXILIAR DE USO GERAL

   ENDC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     DEFINIÇÃO DOS BANCOS DE RAM                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          CONSTANTES INTERNAS                            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
;  A DEFINIÇÃO DE CONSTANTES FACILITA A PROGRAMAÇÃO E A MANUTENÇÃO.

FILTRO_TECLA      EQU     .200           ; FILTRO P/ EVITAR RUIDOS DOS BOTÕES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    DECLARAÇÃO DOS FLAGs DE SOFTWARE                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE FLAGs AJUDA NA PROGRAMAÇÃO E ECONOMIZA MEMÓRIA RAM.

;  ESTE PROGRAMA NÃO UTILIZA NENHUM FLAG DE USUÁRIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                ENTRADAS                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS ENTRADAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    PINO_AD         TRISA,5 ; PINO P/ LEITURA DOS JUMPERS
                                   ; 0 -> FORÇA A DESCARGA DO CAPACITOR
                                   ; 1 -> LIBERA A CARGA DO CAPACITOR

#DEFINE    CAD             PORTA,5 ; PINO P/ LEITURA DOS JUMPERS
                                   ; 0 -> CAPACITOR DESCARREGADO
                                   ; 1 -> CAPACITOR CARREGADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                 SAÍDAS                                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS SAÍDAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    DISPLAY         PORTD   ; BARRAMENTO DE DADOS DO DISPLAY

#DEFINE    RS              PORTE,0 ; INDICA P/ DISPLAY UM DADO OU COMANDO
                                   ; 1 -> DADO
                                   ; 0 -> COMANDO

#DEFINE    ENABLE          PORTE,1 ; SINAL DE ENABLE P/ DISPLAY
                                   ; ATIVO NA BORDA DE SUBIDA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   VETOR DE RESET DO MICROCONTROLADOR                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO INICIAL PARA EXECUÇÃO DO PROGRAMA

   ORG     0X000                   ; ENDEREÇO DO VETOR DE RESET
   GOTO    CONFIG                  ; PULA PARA CONFIG DEVIDO A REGIÃO
                                   ; DESTINADA AS ROTINAS SEGUINTES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   ROTINA DE DELAY (DE 1MS ATÉ 256MS)                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA É UMA ROTINA DE DELAY VARIÁVEL, COM DURAÇÃO DE 1MS X O VALOR PASSADO
;  EM WORK (W).

DELAY_MS
   MOVWF   TEMPO1                  ; CARREGA TEMPO1 (UNIDADES DE MS)
   MOVLW   .250
   MOVWF   TEMPO0                  ; CARREGA TEMPO0 (P/ CONTAR 1MS)

   CLRWDT                          ; LIMPA WDT (PERDE TEMPO)
   DECFSZ  TEMPO0,F                ; FIM DE TEMPO0 ?
   GOTO    $-2                     ; NÃO - VOLTA 2 INSTRUÇÕES
                                   ; SIM - PASSOU-SE 1MS
   DECFSZ  TEMPO1,F                ; FIM DE TEMPO1 ?
   GOTO    $-6                     ; NÃO - VOLTA 6 INSTRUÇÕES
                                   ; SIM
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DE UM CARACTER NO DISPLAY               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
;  ESTA ROTINA ENVIA UM CARACTER PARA O MÓDULO DE LCD. O CARACTER A SER
;  ESCRITO DEVE SER COLOCADO EM WORK (W) ANTES DE CHAMAR A ROTINA.

ESCREVE
   MOVWF   DISPLAY                ; ATUALIZA DISPLAY (PORTD)
   NOP                            ; PERDE 1US PARA ESTABILIZAÇÃO
   BSF     ENABLE                 ; ENVIA UM PULSO DE ENABLE AO DISPLAY
   GOTO    $+1                    ; .
   BCF     ENABLE                 ; .

   MOVLW   .1
   CALL    DELAY_MS               ; DELAY DE 1MS
   RETURN                         ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               CONFIGURAÇÕES INICIAIS DE HARDWARE E SOFTWARE            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  NESTA ROTINA SÃO INICIALIZADAS AS PORTAS DE I/O DO MICROCONTROLADOR E AS
;  CONFIGURAÇÕES DOS REGISTRADORES ESPECIAIS (SFR). A ROTINA INICIALIZA AS
;  VARIÁVEIS DE RAM E AGUARDA O ESTOURO DO WDT.

CONFIG
   CLRF    PORTA                  ; LIMPA O PORTA
   CLRF    PORTB                  ; LIMPA O PORTB
   CLRF    PORTC                  ; LIMPA O PORTC
   CLRF    PORTD                  ; LIMPA O PORTD
   CLRF    PORTE                  ; LIMPA O PORTE


   BANK1                          ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                  ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                  ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011000'
   MOVWF   TRISC                  ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                  ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                  ; CONFIGURA I/O DO PORTE

   MOVLW   B'11011111'
   MOVWF   OPTION_REG             ; CONFIGURA OPTIONS
                                  ; PULL-UPs DESABILITADOS
                                  ; INTER. NA BORDA DE SUBIDA DO RB0
                                  ; TIMER0 INCREM. PELO CICLO DE MÁQUINA
                                  ; WDT   - 1:128
                                  ; TIMER - 1:1
   MOVLW   B'00000111'
   MOVWF   CMCON                  ; DESLIGA OS COMPARADORES

   MOVLW   B'00000000'
   MOVWF   INTCON                 ; CONFIGURA INTERRUPÇÕES
                                  ; DESABILITADA TODAS AS INTERRUPÇÕES

   MOVLW   B'00000111'
   MOVWF   ADCON1                 ; CONFIGURA CONVERSOR A/D
                                  ; CONFIGURA PORTA COM I/O DIGITAL

   BANK0                          ; SELECIONA BANCO 0 DA RAM

;  AS INTRUÇÕES A SEGUIR FAZEM COM QUE O PROGRAMA TRAVE QUANDO HOUVER UM
;  RESET OU POWER-UP, MAS PASSE DIRETO SE O RESET FOR POR WDT. DESTA FORMA,
;  SEMPRE QUE O PIC É LIGADO, O PROGRAMA TRAVA, AGUARDA UM ESTOURO DE WDT
;  E COMEÇA NOVAMENTE. ISTO EVITA PROBLEMAS NO START-UP DO PIC.

   BTFSC   STATUS,NOT_TO          ; RESET POR ESTOURO DE WATCHDOG TIMER ?
   GOTO    $                      ; NÃO - AGUARDA ESTOURO DO WDT
```

```
                                ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INICIALIZAÇÃO DA RAM                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA IRÁ LIMPARÁ TODA A RAM DO BANCO 0, INDO DE 0X20 A 0X7F

   MOVLW   0X20
   MOVWF   FSR                  ; APONTA O ENDEREÇAMENTO INDIRETO PARA
                                ; A PRIMEIRA POSIÇÃO DA RAM
LIMPA_RAM
   CLRF    INDF                 ; LIMPA A POSIÇÃO
   INCF    FSR,F                ; INCREMENTA O PONTEIRO P/ A PRÓX. POS.
   MOVF    FSR,W
   XORLW   0X80                 ; COMPARA O PONTEIRO COM A ÚLT. POS. +1
   BTFSS   STATUS,Z             ; JÁ LIMPOU TODAS AS POSIÇÕES?
   GOTO    LIMPA_RAM            ; NÃO - LIMPA A PRÓXIMA POSIÇÃO
                                ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    CONFIGURAÇÕES INICIAIS DO DISPLAY                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA INICIALIZA O DISPLAY P/ COMUNICAÇÃO DE 8 VIAS, DISPLAY PARA 2
; LINHAS, CURSOR APAGADO E DESLOCAMENTO DO CURSOR À DIREITA.

INICIALIZACAO_DISPLAY
   BCF     RS                   ; SELECIONA O DISPLAY P/ COMANDOS

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO

   MOVLW   .3
   CALL    DELAY_MS             ; DELAY DE 3MS (EXIGIDO PELO DISPLAY)

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO

   MOVLW   B'00111000'          ; ESCREVE COMANDO PARA
   CALL    ESCREVE              ; INTERFACE DE 8 VIAS DE DADOS

   MOVLW   B'00000001'          ; ESCREVE COMANDO PARA
   CALL    ESCREVE              ; LIMPAR TODO O DISPLAY

   MOVLW   .1
   CALL    DELAY_MS             ; DELAY DE 1MS

   MOVLW   B'00001100'          ; ESCREVE COMANDO PARA
   CALL    ESCREVE              ; LIGAR O DISPLAY SEM CURSOR

   MOVLW   B'00000110'          ; ESCREVE COMANDO PARA INCREM.
   CALL    ESCREVE              ; AUTOMÁTICO A ESQUERDA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DA TELA PRINCIPAL                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A TELA PRINCIPAL DO PROGRAMA, COM AS FRASES:
;  LINHA 1 - " JUMPERS  (RA5) "
;  LINHA 2 - "T.CARGA:    x6us"

   MOVLW   0X81                 ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE              ; LINHA 0 / COLUNA 1

   BSF     RS                   ; SELECIONA O DISPLAY P/ DADOS
                                ; COMANDOS PARA ESCREVER AS
                                ; LETRAS DE "JUMPERS  (RA5)"
   MOVLW   'J'
   CALL    ESCREVE
   MOVLW   'U'
   CALL    ESCREVE
```

```
   MOVLW   'M'
   CALL    ESCREVE
   MOVLW   'P'
   CALL    ESCREVE
   MOVLW   'E'
   CALL    ESCREVE
   MOVLW   'R'
   CALL    ESCREVE
   MOVLW   'S'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   '('
   CALL    ESCREVE
   MOVLW   'R'
   CALL    ESCREVE
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   '5'
   CALL    ESCREVE
   MOVLW   ')'
   CALL    ESCREVE

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS

   MOVLW   0XC0                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 0

   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS
                                   ; COMANDOS PARA ESCREVER AS
                                   ; LETRAS DE "T.CARGA:    x6us"
   MOVLW   'T'
   CALL    ESCREVE
   MOVLW   '.'
   CALL    ESCREVE
   MOVLW   'C'
   CALL    ESCREVE
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   'R'
   CALL    ESCREVE
   MOVLW   'G'
   CALL    ESCREVE
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   ':'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   'x'
   CALL    ESCREVE
   MOVLW   '6'
   CALL    ESCREVE
   MOVLW   'u'
   CALL    ESCREVE
   MOVLW   's'
   CALL    ESCREVE

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *      ROTINA PARA DESCARREGAR O CAPACITOR DE LEITURA DO CONVERSOR A/D    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA CONVERTE O PINO DO MICROCONTROLADOR EM SAÍDA COM NÍVEL LÓGICO 0
; E AGUARDA QUE O CAPACITOR SE DESCARREGUE. EM SEGUIDA O PINO É CONVERTIDO
; NOVAMENTE EM ENTRADA PARA PERMITIR QUE O CAPACITOR SE CARREGUE.
```

```
DESCARGA_CAPACITOR
   CLRWDT                          ; LIMPA WATCHDOG TIMER

   CLRF    CONTADOR_AD             ; ZERA O CONTADOR DE TEMPO DE CARGA
                                   ; DO CAPACITOR

   BANK1                           ; SELECIONA BANCO 1 DA RAM
   BCF     PINO_AD                 ; TRANSFORMA PINO EM SAIDA
   BANK0                           ; VOLTA P/ BANCO 0 DA RAM
   BCF     CAD                     ; DESCARREGA O CAPACITOR

   MOVLW   .3
   CALL    DELAY_MS                ; CHAMA ROTINA DE DELAY (3ms)
                                   ; TEMPO NECESSÁRIO P/ DESCARGA
                                   ; DO CAPACITOR

   BANK1                           ; SELECIONA BANCO 1 DA RAM
   BSF     PINO_AD                 ; TRANSFORMA PINO EM ENTRADA
   BANK0                           ; VOLTA P/ BANCO 0 DA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     LOOP P/ ESPERAR CARGA DO CAPACITOR                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; A ROTINA CONTA O TEMPO QUE O CAPACITOR LEVA PARA ATINGIR UM NÍVEL DE TENSÃO
; SUFICIENTE PARA QUE O MICROCONTROLADOR ENTENDA NÍVEL LÓGICO 1 NA ENTRADA TTL
; DO PINO RA5. CASO O CAPACITOR DEMORE MAIS DO QUE 256 CICLOS DESTE LOOP PARA
; CARREGAR, A ROTINA DESVIA PARA UMA ROTINA DE SATURAÇÃO.
; O LOOP DA ROTINA É DE 6us (CRISTAL DE 4MHz).

LOOP_CAD
   INCFSZ  CONTADOR_AD,F           ; INCREM. CONTADOR E VERIFICA ESTOURO
   GOTO    $+2                     ; NÃO HOUVE ESTOURO - PULA 1 INSTRUÇÃO
   GOTO    SATURACAO               ; HOUVE ESTOURO - PULA P/ SATURAÇÃO

   BTFSS   CAD                     ; CAPACITOR JÁ CARREGOU ?
   GOTO    LOOP_CAD                ; NÃO - VOLTA P/ LOOP_CAD
   GOTO    MOSTRA_CONTADOR         ; SIM - MOSTRA TEMPO DE CARGA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              MOSTRA O TEMPO DE CARGA DO CAPACITOR NO LCD                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA MOSTRA O TEMPO DE CARGA DO CAPACITOR EM HAXADECIMAL NO LCD.
; CASO O CAPACITOR NÃO TENHA SE CARREGADO, A ROTINA DE SATURAÇÃO GARANTE
; UM VALOR MÁXIMO PARA O TEMPO DE CARGA (0xFF).

SATURACAO
   MOVLW   0XFF
   MOVWF   CONTADOR_AD             ; SATURA O CONTADOR
                                   ; (CAPACITOR NÃO CARREGOU)

MOSTRA_CONTADOR
   MOVLW   0XCA                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 10

   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

   SWAPF   CONTADOR_AD,W           ; INVERTE NIBLE DO CONTADOR_AD
   ANDLW   B'00001111'             ; MASCARA BITS MAIS SIGNIFICATIVOS
   MOVWF   AUX                     ; SALVA EM AUXILIAR

   MOVLW   0X0A
   SUBWF   AUX,W                   ; AUX - 10d (ATUALIZA FLAG DE CARRY)
   MOVLW   0X30                    ; CARREGA WORK COM 30h
   BTFSC   STATUS,C                ; RESULTADO E POSITIVO? (É UMA LETRA?)
   MOVLW   0X37                    ; SIM - CARREGA WORK COM 37h
                                   ; NÃO - WORK FICA COM 30h (NÚMERO)
   ADDWF   AUX,W                   ; SOMA O WORK AO AUXILIAR
                                   ; (CONVERSÃO ASCII)

   CALL    ESCREVE                 ; MOSTRA NO DISPLAY
```

```
   MOVF    CONTADOR_AD,W          ; CARREGA NO WORK O CONTADOR_AD
   ANDLW   B'00001111'            ; MASCARA BITS MAIS SIGNIFICATIVOS
   MOVWF   AUX                    ; SALVA EM AUXILIAR

   MOVLW   0X0A
   SUBWF   AUX,W                  ; AUX - 10d (ATUALIZA FLAG DE CARRY)
   MOVLW   0X30                   ; CARREGA WORK COM 30h
   BTFSC   STATUS,C               ; RESULTADO E POSITIVO? (É UMA LETRA?)
   MOVLW   0X37                   ; SIM - CARREGA WORK COM 37h
                                  ; NÃO - WORK FICA COM 30h (NÚMERO)
   ADDWF   AUX,W                  ; SOMA O WORK AO AUXILIAR
                                  ; (CONVERSÃO ASCII)

   CALL    ESCREVE                ; MOSTRA NO DISPLAY

   BCF     RS                     ; SELECIONA O DISPLAY P/ COMANDOS

   GOTO    DESCARGA_CAPACITOR     ; VOLTA P/ DESCARREGAR O CAPACITOR

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          FIM DO PROGRAMA                                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                            ; FIM DO PROGRAMA
```

Dicas e Comentários

As dicas e comentários desta experiência são os mesmos da experiência anterior.

Exercícios Propostos

1. Através do conhecimento do tempo de carga do capacitor, estime a real posição dos jumpers A, B e C do sistema.

Capítulo 16 - Experiência 14 – Modulo PWM

Objetivo

O objetivo desta experiência é ensinar ao aluno como utilizar o módulo PWM do microcontrolador PIC16F877A

Descrição

Este PIC possui 2 canais de PWMs (CCP1 e CCP2), cada um com resolução máxima de 10 bits. Isto significa que o duty cycle poderá ser regulado de 0 a 100% com uma resolução máxima de 1024 pontos. No entanto, dependendo da configuração adotada, esta resolução não será atingida.

O período do PWM é controlado diretamente pelo TMR2, através do registrador **PR2**. Como já foi visto, sempre que TMR2 = **PR2**, o timer é zerado e neste momento, um novo período do PWM é iniciado. Desta forma, pode-se definir o período e a freqüência do PWM pelas seguintes fórmulas:

$$T = [(\mathbf{PR2}) + 1] \times 4 \times Tosc \times (\text{Prescale do TMR2})$$

$$PWM_{Freq} = 1 / T$$

O duty cycle normalmente é definido em porcentagem, porém, o PIC não define o valor do duty cycle e sim o tempo do pulso em nível alto. Desta forma, o tempo do pulso pode ser calculado por:

$$tp = \mathbf{CCPRxL:CCPxCON<CCPxX:CCPxY>} \times Tosc \times (\text{Prescale do TMR2})$$

Repare que a largura do pulso é ajustada em dois registradores: **CCPRxL** que armazena os 8 bits mais significativos e **CCPxCON** que armazena os dois bits menos significativos. Assim, obtêm-se os 10 bits que controlam o duty cycle do PWM.

Apesar do PIC não definir o duty cycle, ele pode ser calculado dividindo o tempo do pulso em nível alto pelo período total do PWM.

$$\frac{t_p}{T} = \frac{\mathbf{CCPRxL:CCPxCON<DCxB1:DCxB0>} \times T_{OSC} \times (\text{Prescale do TMR2})}{[(\mathbf{PR2}) + 1] \times 4 \times T_{OSC} \times (\text{Prescale do TMR2})}$$

$$\frac{t_p}{T} = \frac{\mathbf{CCPRxL:CCPxCON<DCxB1:DCxB0>}}{[(\mathbf{PR2}) + 1] \times 4}$$

Verifica-se então que apesar do período e o do tempo de pulso dependerem do cristal (Tosc) e do ajuste do prescale do TMR2, o duty cycle depende única e exclusivamente dos valores ajustados nos registradores **PR2**, **CCPRxL** e **CCPxCON** (bits 5 e 4).

Veja que o registrador **PR2** (8 bits) que controla o período do PWM é multiplicado por 4 para poder igualar-se aos 10 bits que controlam o duty cycle. É justamente este o problema da

resolução máxima atingida. Se o registrador **PR2** for ajustado com um valor menor que 8 bits, ou seja, menor do que 255, serão necessários menos do que 10 bits para atingir um PWM com 100% de duty cycle. Portanto, o número de pontos para ajuste do duty cycle é 4 vezes maior do que o valor ajustado em (**PR2**+1). Em termos de bits, podemos dizer que a resolução do duty cycle é 2 bits maior do que o número de bits que formam o valor ajustado em **PR2**. Repare também que caso **PR2** seja ajustado com 255 nunca será atingido um duty cycle de 100%, pois o período atingirá o valor máximo de 1024 ([**PR2**+1]x4) enquanto o tempo do pulso em nível alto (**<DCxB9:DCxB0>**) será no máximo 1023.

No software da experiência ativou-se a saída do módulo CCP2 para controlar a rotação do ventilador que está ligado ao pino **RC1**.

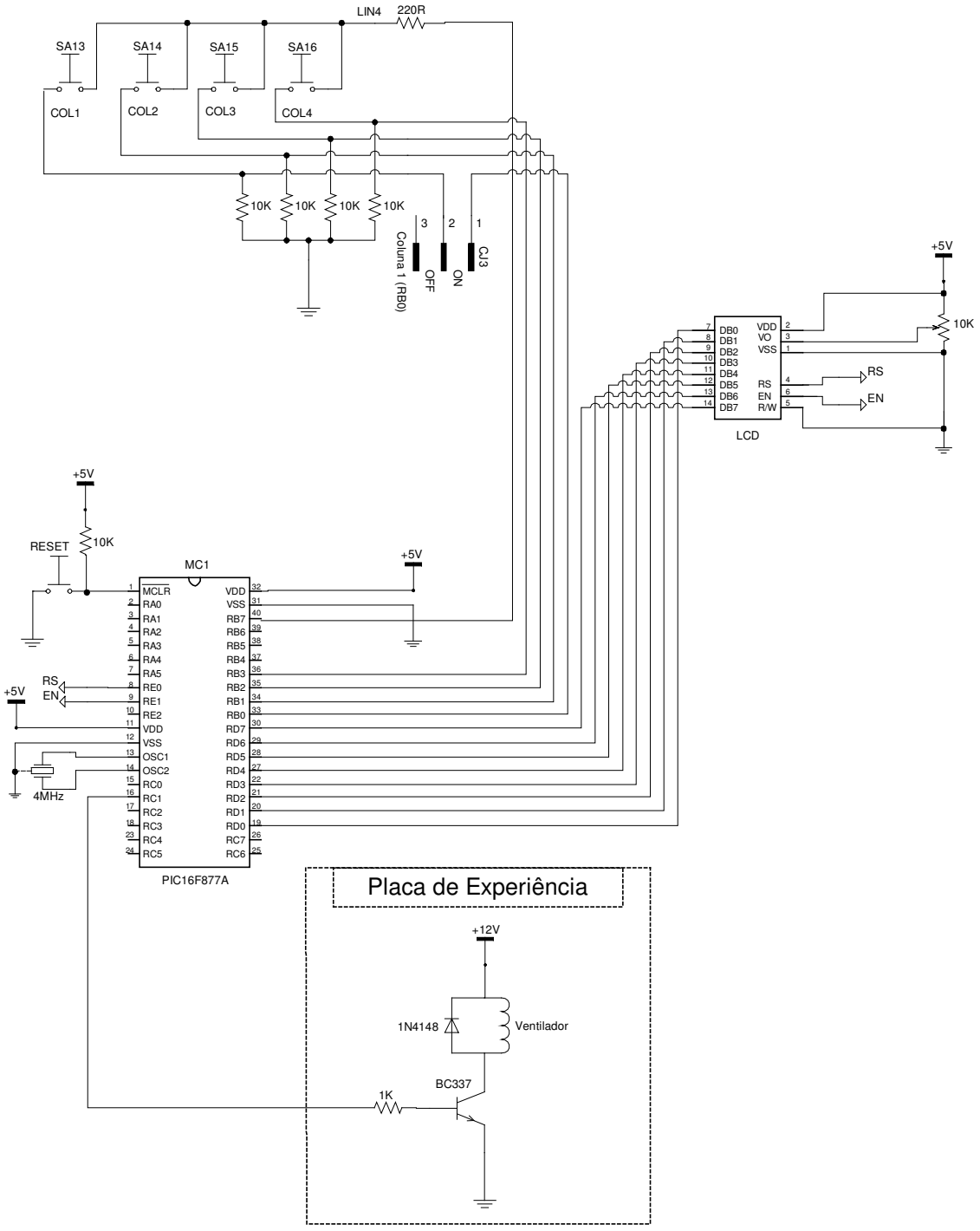
O registrador **PR2** foi ajustado com valor máximo, ou seja, 255 e o prescale do timer foi configurado para 1:16. Com isso a freqüência do PWM ficou em 244,14Hz ($PWM_{Período}$ = 4,096ms), considerando-se que o microcontrolador do sistema MCMASTER está trabalhando a 4MHz.

As teclas da linha 4 foram habilitadas e receberam as seguintes funções:

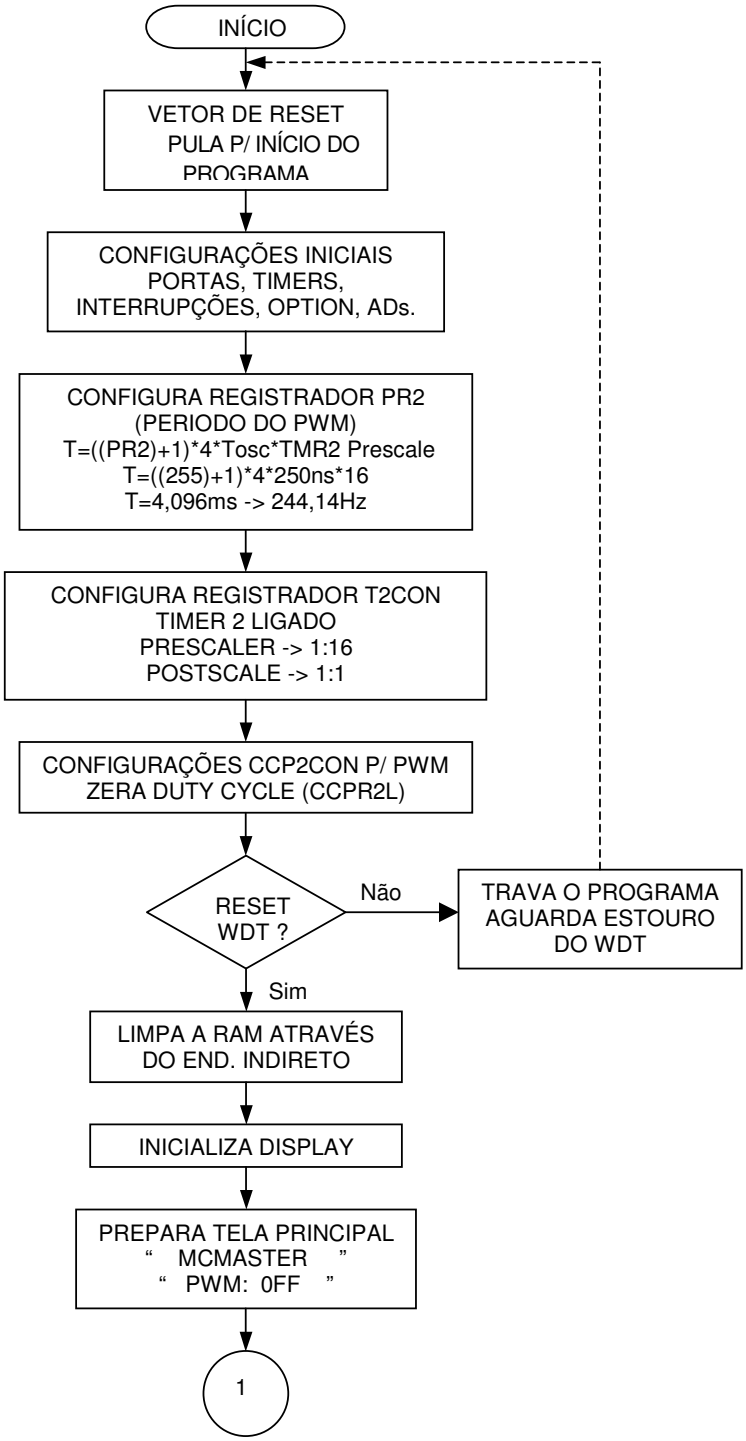
Coluna	Duty Cycle
1	0%
2	50%
3	75%
4	100%

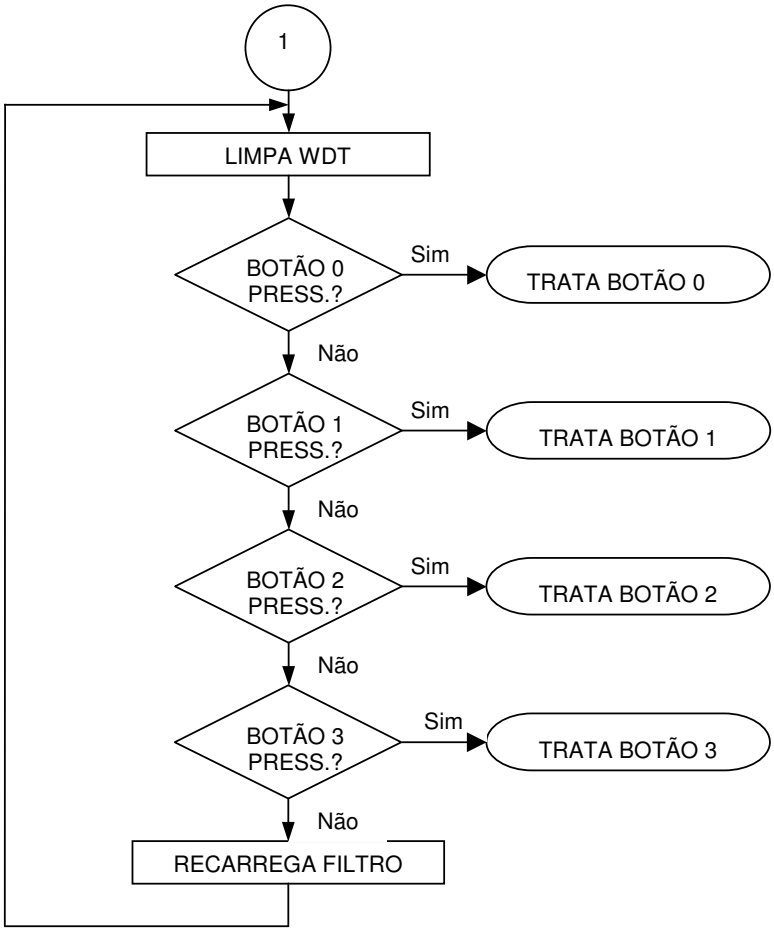
A fim de deixar o sistema mais interativo, utilizou-se o LCD para mostrar o valor atual ajustado para o PWM.

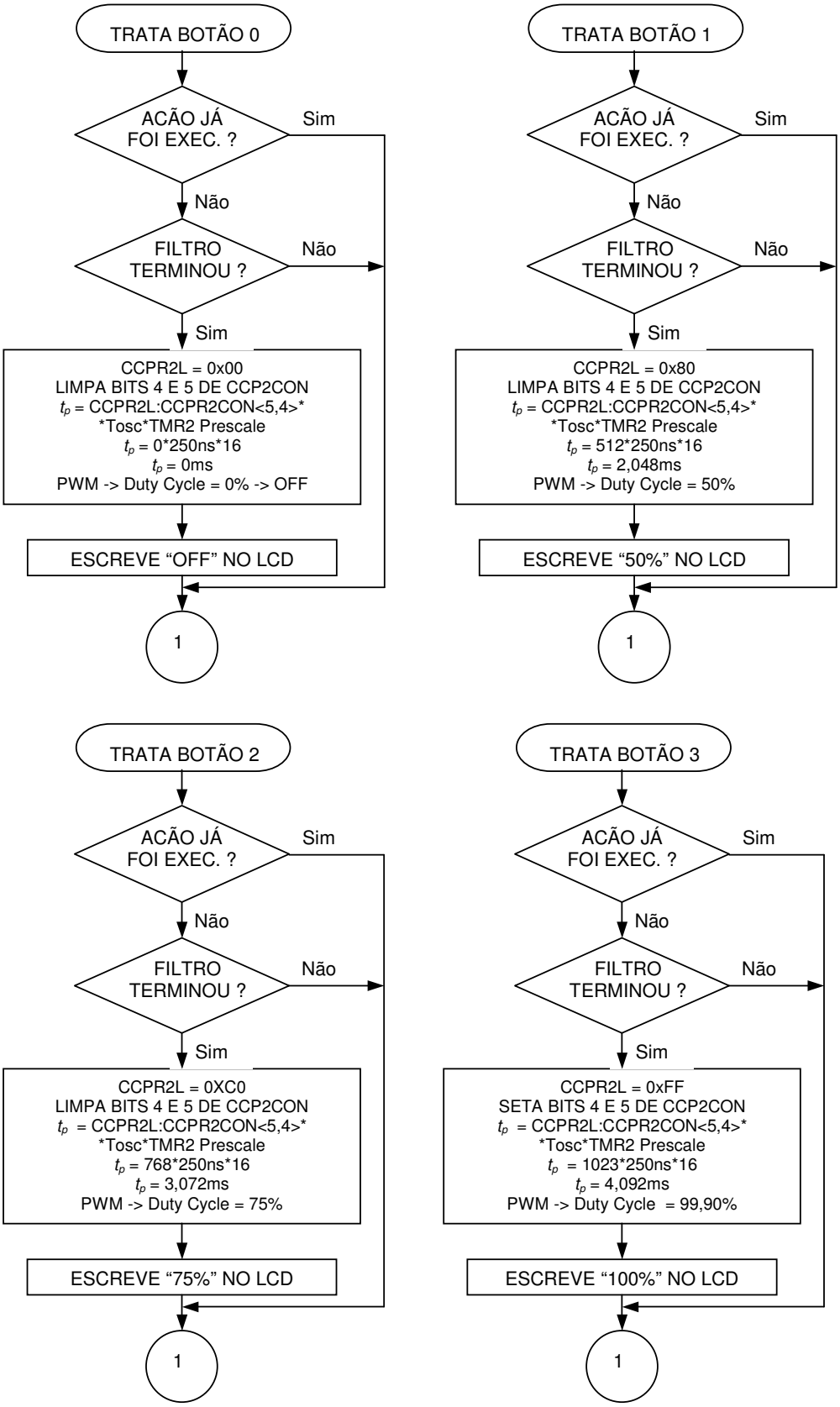
Esquema Elétrico



Fluxograma


INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURAÇÕES INICIAIS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION, ADs.
CONFIGURA REGISTRADOR PR2
(PERIODO DO PWM)
T=((PR2)+1)*4*Tosc*TMR2 Prescale
T=((255)+1)*4*250ns*16
T=4,096ms -> 244,14Hz
CONFIGURA REGISTRADOR T2CON
TIMER 2 LIGADO
PRESCALER -> 1:16
POSTSCALE -> 1:1
CONFIGURAÇÕES CCP2CON P/ PWM
ZERA DUTY CYCLE (CCPR2L)
RESET
WDT ?
Não
TRAVA O PROGRAMA
AGUARDA ESTOURO
DO WDT
Sim
LIMPA A RAM ATRAVÉS
DO END. INDIRETO
INICIALIZA DISPLAY
PREPARA TELA PRINCIPAL
“ MCMASTER ”
“ PWM: 0FF ”
1


1
LIMPA WDT
BOTÃO 0 PRESS.?
Sim
TRATA BOTÃO 0
Não
BOTÃO 1 PRESS.?
Sim
TRATA BOTÃO 1
Não
BOTÃO 2 PRESS.?
Sim
TRATA BOTÃO 2
Não
BOTÃO 3 PRESS.?
Sim
TRATA BOTÃO 3
Não
RECARREGA FILTRO


TRATA BOTÃO 0
ACÃO JÁ FOI EXEC. ?
Sim
Não
FILTRO TERMINOU ?
Não
Sim
CCPR2L = 0x00
LIMPA BITS 4 E 5 DE CCP2CON
t_p = CCPR2L:CCPR2CON<5,4>*
*Tosc*TMR2 Prescale
t_p = 0*250ns*16
t_p = 0ms
PWM -> Duty Cycle = 0% -> OFF
ESCREVE “OFF” NO LCD
1
TRATA BOTÃO 1
ACÃO JÁ FOI EXEC. ?
Sim
Não
FILTRO TERMINOU ?
Não
Sim
CCPR2L = 0x80
LIMPA BITS 4 E 5 DE CCP2CON
t_p = CCPR2L:CCPR2CON<5,4>*
*Tosc*TMR2 Prescale
t_p = 512*250ns*16
t_p = 2,048ms
PWM -> Duty Cycle = 50%
ESCREVE “50%” NO LCD
1
TRATA BOTÃO 2
ACÃO JÁ FOI EXEC. ?
Sim
Não
FILTRO TERMINOU ?
Não
Sim
CCPR2L = 0XC0
LIMPA BITS 4 E 5 DE CCP2CON
t_p = CCPR2L:CCPR2CON<5,4>*
*Tosc*TMR2 Prescale
t_p = 768*250ns*16
t_p = 3,072ms
PWM -> Duty Cycle = 75%
ESCREVE “75%” NO LCD
1
TRATA BOTÃO 3
ACÃO JÁ FOI EXEC. ?
Sim
Não
FILTRO TERMINOU ?
Não
Sim
CCPR2L = 0xFF
SETA BITS 4 E 5 DE CCP2CON
t_p = CCPR2L:CCPR2CON<5,4>*
*Tosc*TMR2 Prescale
t_p = 1023*250ns*16
t_p = 4,092ms
PWM -> Duty Cycle = 99,90%
ESCREVE “100%” NO LCD
1

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *                        EXPERIÊNCIA 14 - MÓDULO PWM                      *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            DESCRIÇÃO GERAL                              *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE EXEMPLO FOI ELABORADO PARA EXPLICAR O FUNCIONAMENTO DO MÓDULO PWM
;  DO PIC16F877. ELE MONITORA OS QUATRO BOTÕES E CONFORME O BOTÃO SELECIONADO
;  APLICA UM VALOR DIFERENTE NO PWM, FAZENDO ASSIM UM CONTROLE SOBRE A
;  VELOCIDADE DO VENTILADOR. NO LCD É MOSTRADO O VALOR ATUAL DO DUTY CYCLE.
;  OS BOTÕES ATIVOS SÃO O DA LINHA 4.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                         CONFIGURAÇÕES PARA GRAVAÇÃO                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_ON & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           DEFINIÇÃO DAS VARIÁVEIS                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE BLOCO DE VARIÁVEIS ESTÁ LOCALIZADO LOGO NO INÍCIO DO BANCO 0

   CBLOCK  0X20                    ; POSIÇÃO INICIAL DA RAM

           FILTRO_BOTOES           ; FILTRO PARA RUIDOS

           TEMPO1
           TEMPO0                  ; CONTADORES P/ DELAY

   ENDC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>         ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     DEFINIÇÃO DOS BANCOS DE RAM                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            CONSTANTES INTERNAS                          *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE CONSTANTES FACILITA A PROGRAMAÇÃO E A MANUTENÇÃO.

FILTRO_TECLA      EQU     .200          ; FILTRO P/ EVITAR RUIDOS DOS BOTÕES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     DECLARAÇÃO DOS FLAGs DE SOFTWARE                    *
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE FLAGs AJUDA NA PROGRAMAÇÃO E ECONOMIZA MEMÓRIA RAM.

;  ESTE PROGRAMA NÃO UTILIZA NENHUM FLAG DE USUÁRIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               ENTRADAS                                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS ENTRADAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    BOTAO_0         PORTB,0 ; ESTADO DO BOTÃO 0
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_1         PORTB,1 ; ESTADO DO BOTÃO 1
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_2         PORTB,2 ; ESTADO DO BOTÃO 2
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_3         PORTB,3 ; ESTADO DO BOTÃO 3
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                 SAÍDAS                                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS SAÍDAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    DISPLAY         PORTD   ; BARRAMENTO DE DADOS DO DISPLAY

#DEFINE    RS              PORTE,0 ; INDICA P/ O DISPLAY UM DADO OU COMANDO
                                   ; 1 -> DADO
                                   ; 0 -> COMANDO

#DEFINE    ENABLE          PORTE,1 ; SINAL DE ENABLE P/ DISPLAY
                                   ; ATIVO NA BORDA DE DESCIDA

#DEFINE    VENTILADOR      PORTC,1 ; SAÍDA P/ O VENTILADOR
                                   ; 1 -> VENTILADOR LIGADO
                                   ; 0 -> VENTILADOR DESLIGADO

#DEFINE    TEC_MATRICIAL   PORTB   ; PORT DO MICROCONTROLADOR LIGADO AO
                                   ; TECLADO MATRICIAL
                                   ; <RB4:RB7> LINHAS
                                   ; 1->ATIVADAS 0->DESATIVADAS
                                   ; <RB0:RB3> COLUNAS
                                   ; 1->TECLAS PRESSIONADAS 0->TECLAS LIBERADAS

#DEFINE    LINHA_4         PORTB,7 ; PINO P/ ATIVAR LINHA 4 (TECLADO MATRICIAL)
                                   ; 0 -> LINHA 4 ATIVADA
                                   ; 1 -> LINHA 4 DESATIVADA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   VETOR DE RESET DO MICROCONTROLADOR                          *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO INICIAL PARA EXECUÇÃO DO PROGRAMA

   ORG     0X0000                  ; ENDEREÇO DO VETOR DE RESET
   GOTO    CONFIG                  ; PULA PARA CONFIG DEVIDO A REGIÃO
                                   ; DESTINADA AS ROTINAS SEGUINTES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   ROTINA DE DELAY (DE 1MS ATÉ 256MS)                          *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA É UMA ROTINA DE DELAY VARIÁVEL, COM DURAÇÃO DE 1MS X O VALOR PASSADO
;  EM WORK (W).
```

```
DELAY_MS
   MOVWF   TEMPO1                  ; CARREGA TEMPO1 (UNIDADES DE MS)
   MOVLW   .250
   MOVWF   TEMPO0                  ; CARREGA TEMPO0 (P/ CONTAR 1MS)

   CLRWDT                          ; LIMPA WDT (PERDE TEMPO)
   DECFSZ  TEMPO0,F                ; FIM DE TEMPO0 ?
   GOTO    $-2                     ; NÃO - VOLTA 2 INSTRUÇÕES
                                   ; SIM - PASSOU-SE 1MS
   DECFSZ  TEMPO1,F                ; FIM DE TEMPO1 ?
   GOTO    $-6                     ; NÃO - VOLTA 6 INSTRUÇÕES
                                   ; SIM
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DE UM CARACTER NO DISPLAY              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ENVIA UM CARACTER PARA O MÓDULO DE LCD. O CARACTER A SER
;  ESCRITO DEVE SER COLOCADO EM WORK (W) ANTES DE CHAMAR A ROTINA.

ESCREVE
   MOVWF   DISPLAY                 ; ATUALIZA DISPLAY (PORTD)
   NOP                             ; PERDE 1US PARA ESTABILIZAÇÃO
   BSF     ENABLE                  ; ENVIA UM PULSO DE ENABLE AO DISPLAY
   GOTO    $+1                     ; .
   BCF     ENABLE                  ; .

   MOVLW   .1
   CALL    DELAY_MS                ; DELAY DE 1MS
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               CONFIGURAÇÕES INICIAIS DE HARDWARE E SOFTWARE           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  NESTA ROTINA SÃO INICIALIZADAS AS PORTAS DE I/O DO MICROCONTROLADOR E AS
;  CONFIGURAÇÕES DOS REGISTRADORES ESPECIAIS (SFR). A ROTINA INICIALIZA A
;  MÁQUINA E AGUARDA O ESTOURO DO WDT.

CONFIG
   CLRF    PORTA                   ; LIMPA O PORTA
   CLRF    PORTB                   ; LIMPA O PORTB
   CLRF    PORTC                   ; LIMPA O PORTC
   CLRF    PORTD                   ; LIMPA O PORTD
   CLRF    PORTE                   ; LIMPA O PORTE


   BANK1                           ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                   ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                   ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011000'
   MOVWF   TRISC                   ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                   ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                   ; CONFIGURA I/O DO PORTE

   MOVLW   B'11011111'
   MOVWF   OPTION_REG              ; CONFIGURA OPTIONS
                                   ; PULL-UPs DESABILITADOS
                                   ; INTER. NA BORDA DE SUBIDA DO RB0
                                   ; TIMER0 INCREM. PELO CICLO DE MÁQUINA
                                   ; WDT   - 1:128
                                   ; TIMER - 1:1

   MOVLW   B'00000000'
   MOVWF   INTCON                  ; CONFIGURA INTERRUPÇÕES
```

```
                                ; DESABILITADA TODAS AS INTERRUPÇÕES

   MOVLW   B'00000111'
   MOVWF   CMCON                ; DESLIGA OS COMPARADORES

   MOVLW   B'00000111'
   MOVWF   ADCON1               ; CONFIGURA CONVERSOR A/D
                                ; CONFIGURA PORTA E PORTE COMO I/O DIGITAL

   MOVLW   .255
   MOVWF   PR2                  ; CONFIGURA PERÍODO DO PWM
                                ; T=((PR2)+1)*4*Tosc*TMR2 Prescale
                                ; T=((255)+1)*4*250ns*16
                                ; T=4,096ms -> 244,14Hz

   BANK0                        ; SELECIONA BANCO 0 DA RAM

   MOVLW   B'00000111'
   MOVWF   T2CON                ; CONFIGURA TMR2
                                ; TIMER 2 LIGADO
                                ; PRESCALE  - 1:16
                                ; POSTSCALE - 1:1

   MOVLW   B'00001111'
   MOVWF   CCP2CON              ; CONFIGURA CCP2CON PARA PWM
                                ; (PINO RC1)

   CLRF    CCPR2L               ; INICIA COM DUTY CYCLE EM ZERO

;  AS INSTRUÇÕES A SEGUIR FAZEM COM QUE O PROGRAMA TRAVE QUANDO HOUVER UM
;  RESET OU POWER-UP, MAS PASSE DIRETO SE O RESET FOR POR WDT. DESTA FORMA,
;  SEMPRE QUE O PIC É LIGADO, O PROGRAMA TRAVA, AGUARDA UM ESTOURO DE WDT
;  E COMEÇA NOVAMENTE. ISTO EVITA PROBLEMAS NO START-UP DO PIC.

   BTFSC   STATUS,NOT_TO        ; RESET POR ESTOURO DE WATCHDOG TIMER ?
   GOTO    $                    ; NÃO - AGUARDA ESTOURO DO WDT
                                ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INICIALIZAÇÃO DA RAM                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA IRÁ LIMPAR TODA A RAM DO BANCO 0, INDO DE 0X20 A 0X7F.
;  EM SEGUIDA, AS VARIÁVEIS DE RAM DO PROGRAMA SÃO INICIALIZADAS.

   MOVLW   0X20
   MOVWF   FSR                  ; APONTA O ENDEREÇAMENTO INDIRETO PARA
                                ; A PRIMEIRA POSIÇÃO DA RAM
LIMPA_RAM
   CLRF    INDF                 ; LIMPA A POSIÇÃO
   INCF    FSR,F                ; INCREMENTA O PONTEIRO P/ A PRÓX. POS.
   MOVF    FSR,W
   XORLW   0X80                 ; COMPARA O PONTEIRO COM A ÚLT. POS. +1
   BTFSS   STATUS,Z             ; JÁ LIMPOU TODAS AS POSIÇÕES?
   GOTO    LIMPA_RAM            ; NÃO - LIMPA A PRÓXIMA POSIÇÃO
                                ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  CONFIGURAÇÕES INICIAIS DO DISPLAY                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA INICIALIZA O DISPLAY P/ COMUNICAÇÃO DE 8 VIAS, DISPLAY PARA 2
; LINHAS, CURSOR APAGADO E DESLOCAMENTO DO CURSOR À DIREITA.

INICIALIZACAO_DISPLAY
   BCF     RS                   ; SELECIONA O DISPLAY P/ COMANDOS

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO

   MOVLW   .3
   CALL    DELAY_MS             ; DELAY DE 3MS (EXIGIDO PELO DISPLAY)

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
```

```
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   0X30                    ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   B'00111000'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; INTERFACE DE 8 VIAS DE DADOS

   MOVLW   B'00000001'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; LIMPAR TODO O DISPLAY

   MOVLW   .1
   CALL    DELAY_MS                ; DELAY DE 1MS

   MOVLW   B'00001100'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; LIGAR O DISPLAY SEM CURSOR

   MOVLW   B'00000110'             ; ESCREVE COMANDO PARA INCREM.
   CALL    ESCREVE                 ; AUTOMÁTICO À DIREITA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DA TELA PRINCIPAL                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A TELA PRINCIPAL DO PROGRAMA, COM AS FRASES:
;  LINHA 1 - "    MCMASTER     "
;  LINHA 2 - "   PWM:  OFF    "

MOSTRA_TELA_PRINCIPAL
   MOVLW   0X84
   CALL    ESCREVE                 ; POSICIONA O CURSOR - POSIÇÃO 4 LINHA 0
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

   MOVLW   'S'
   CALL    ESCREVE
   MOVLW   'D'
   CALL    ESCREVE
   MOVLW   '-'
   CALL    ESCREVE
   MOVLW   '1'
   CALL    ESCREVE
   MOVLW   '7'
   CALL    ESCREVE
   MOVLW   '0'
   CALL    ESCREVE
   MOVLW   '0'
   CALL    ESCREVE                 ; ESCREVE MCMASTER

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XC3                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 3
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

                                   ; COMANDOS PARA ESCREVER AS
                                   ; LETRAS DE "   PWM:  OFF    "
   MOVLW   'P'
   CALL    ESCREVE
   MOVLW   'W'
   CALL    ESCREVE
   MOVLW   'M'
   CALL    ESCREVE
   MOVLW   ':'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   'O'
   CALL    ESCREVE
   MOVLW   'F'
   CALL    ESCREVE
   MOVLW   'F'
   CALL    ESCREVE
```

```
   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          VARREDURA DOS BOTÕES                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA VERIFICA SE ALGUM BOTÃO ESTÁ PRESSIONADO E CASO AFIRMATIVO
; DESVIA PARA O TRATAMENTO DO MESMO.

VARRE
   CLRWDT                          ; LIMPA WATCHDOG TIMER

; ***************** VERIFICA ALGUM BOTÃO PRESSIONADO ************************

VARRE_BOTOES
   BSF     LINHA_4                 ; ATIVA A LINHA 4 DO TECLADO MATRICIAL

   GOTO    $+1                     ; DELAY PARA ESTABILIZAÇÃO
                                   ; E LEITURA DO TECLADO

   BTFSC   BOTAO_0                 ; O BOTÃO 0 ESTA PRESSIONADO ?
   GOTO    TRATA_BOTAO_0           ; SIM - PULA P/ TRATA_BOTAO_0
                                   ; NÃO

   BTFSC   BOTAO_1                 ; O BOTÃO 1 ESTA PRESSIONADO ?
   GOTO    TRATA_BOTAO_1           ; SIM - PULA P/ TRATA_BOTAO_1
                                   ; NÃO

   BTFSC   BOTAO_2                 ; O BOTÃO 2 ESTA PRESSIONADO ?
   GOTO    TRATA_BOTAO_2           ; SIM - PULA P/ TRATA_BOTAO_2
                                   ; NÃO

   BTFSC   BOTAO_3                 ; O BOTÃO 3 ESTA PRESSIONADO ?
   GOTO    TRATA_BOTAO_3           ; SIM - PULA P/ TRATA_BOTAO_3
                                   ; NÃO

   BCF     LINHA_4                 ; DESATIVA A LINHA 4 DO TECLADO MATRICIAL

; *************************** FILTRO P/ EVITAR RUIDOS ***********************

   MOVLW   FILTRO_TECLA            ; CARREGA O VALOR DE FILTRO_TECLA
   MOVWF   FILTRO_BOTOES           ; SALVA EM FILTRO_BOTOES
                                   ; RECARREGA FILTRO P/ EVITAR RUIDOS
                                   ; NOS BOTÕES

   GOTO    VARRE                   ; VOLTA PARA VARRER TECLADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          TRATAMENTO DOS BOTÕES                          *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; NESTE TRECHO DO PROGRAMA ESTÃO TODOS OS TRATAMENTOS DOS BOTÕES

; ************************* TRATAMENTO DO BOTÃO 0 ***************************

TRATA_BOTAO_0
   MOVF    FILTRO_BOTOES,F
   BTFSC   STATUS,Z                ; FILTRO JÁ IGUAL A ZERO ?
                                   ; (FUNÇÃO JA FOI EXECUTADA?)
   GOTO    VARRE                   ; SIM - VOLTA P/ VARREDURA DO TECLADO
                                   ; NÃO
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO

   CLRF    CCPR2L                  ; ZERA CCPR2L
   BCF     CCP2CON,5               ; ZERA OS BITS 5 e 4
   BCF     CCP2CON,4               ; (LSB DO DUTY CYCLE)
                           ; Tp = CCPR2L:CCP2CON<5,4>*Tosc*TMR2 Prescale
                           ; Tp = 0 * 250ns * 16
                           ; Tp = 0
                           ; PWM -> DUTY CYCLE = 0% -> OFF
```

```
   CLRF    TEC_MATRICIAL           ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XC8                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 8
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

                                   ; COMANDOS PARA ESCREVER AS
                                   ; LETRAS DE " OFF"
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   'O'
   CALL    ESCREVE
   MOVLW   'F'
   CALL    ESCREVE
   MOVLW   'F'
   CALL    ESCREVE

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************* TRATAMENTO DO BOTÃO 1 ***************************

TRATA_BOTAO_1
   MOVF    FILTRO_BOTOES,F
   BTFSC   STATUS,Z                ; FILTRO JÁ IGUAL A ZERO ?
                                   ; (FUNÇÃO JA FOI EXECUTADA?)
   GOTO    VARRE                   ; SIM - VOLTA P/ VARREDURA DO TECLADO
                                   ; NÃO
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO
   MOVLW   0X80
   MOVWF   CCPR2L                  ; CARREGA CCPR2L COM 0X80
   BCF     CCP2CON,5               ; LIMPA OS BITS 5 e 4
   BCF     CCP2CON,4               ; LSB DO DUTY CYCLE
                          ; Tp = CCPR2L:CCP2CON<5,4>*Tosc*TMR2 Prescale
                          ; Tp = 512 * 250ns * 16
                          ; Tp = 2,048ms
                          ; PWM -> DUTY CYCLE = 50%

   CLRF    TEC_MATRICIAL           ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XC8                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 8
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

                                   ; COMANDOS PARA ESCREVER AS
                                   ; LETRAS DE " 50%"
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   '5'
   CALL    ESCREVE
   MOVLW   '0'
   CALL    ESCREVE
   MOVLW   '%'
   CALL    ESCREVE

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************* TRATAMENTO DO BOTÃO 2 ***************************

TRATA_BOTAO_2
   MOVF    FILTRO_BOTOES,F
   BTFSC   STATUS,Z                ; FILTRO JÁ IGUAL A ZERO ?
                                   ; (FUNÇÃO JA FOI EXECUTADA?)
   GOTO    VARRE                   ; SIM - VOLTA P/ VARREDURA DO TECLADO
                                   ; NÃO
```

```
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO
   MOVLW   0XC0
   MOVWF   CCPR2L                  ; CARREGA CCPR2L COM 0XC0
   BCF     CCP2CON,5               ; LIMPA OS BITS 5 e 4
   BCF     CCP2CON,4               ; LSB DO DUTY CYCLE
                           ; Tp = CCPR2L:CCP2CON<5,4>*Tosc*TMR2 Prescale
                           ; Tp = 768 * 250ns * 16
                           ; Tp = 3,072ms
                           ; PWM -> DUTY CYCLE = 75%

   CLRF    TEC_MATRICIAL           ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XC8                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 8
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

                                   ; COMANDOS PARA ESCREVER AS
                                   ; LETRAS DE "75%"
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   '7'
   CALL    ESCREVE
   MOVLW   '5'
   CALL    ESCREVE
   MOVLW   '%'
   CALL    ESCREVE

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************* TRATAMENTO DO BOTÃO 3 ***************************

TRATA_BOTAO_3
   MOVF    FILTRO_BOTOES,F
   BTFSC   STATUS,Z                ; FILTRO JÁ IGUAL A ZERO ?
                                   ; (FUNÇÃO JA FOI EXECUTADA?)
   GOTO    VARRE                   ; SIM - VOLTA P/ VARREDURA DO TECLADO
                                   ; NÃO
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO
   MOVLW   0XFF
   MOVWF   CCPR2L                  ; CARREGA CCPR2L COM 0XFF
   BSF     CCP2CON,5               ; SETA OS BITS 5 e 4
   BSF     CCP2CON,4               ; LSB DO DUTY CYCLE
                           ; Tp = CCPR2L:CCP2CON<5,4>*Tosc*TMR2 Prescale
                           ; Tp = 1023 * 250ns * 16
                           ; Tp = 4,092ms
                           ; PWM -> DUTY CYCLE = 99,90%

   CLRF    TEC_MATRICIAL           ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XC8                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 8
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

                                   ; COMANDOS PARA ESCREVER AS
                                   ; LETRAS DE "100%"
   MOVLW   '1'
   CALL    ESCREVE
   MOVLW   '0'
   CALL    ESCREVE
   MOVLW   '0'
   CALL    ESCREVE
   MOVLW   '%'
   CALL    ESCREVE
```

```
   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              FIM DO PROGRAMA                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                             ; FIM DO PROGRAMA
```

Dicas e Comentários

Para calcular o valor que deve ser carregado nos registradores que controlam o tempo do pulso em nível alto (duty cycle) a partir de um determinado valor de duty cycle expresso em porcentagem pode-se utilizar a formula a seguir:

$$\textbf{CCPRx} = [(\textbf{PR2})+1] \times 4 \times \textit{Porcentagem desejada}$$

Vejamos como exemplo o valor de 50% adotado nesta experiência. A porcentagem desejada é de 50% e o valor de **PR2** é 255, assim,

$$\textbf{CCPRx} = 256 * 4 * 0,5$$
$$\textbf{CCPRx} = 512$$

Ou seja, se carregarmos os registradores que controlam o tempo do pulso em nível alto com 512, obteremos um duty cycle de 50%

Exercícios Propostos

1. Corrija o problema encontrado no nível 100%, evitando que a saída seja colocada em zero, mesmo que por um período de tempo muito curto.
2. Em vez de trabalhar com somente 4 níveis de PWM, altere o sistema para que um botão ligue e desligue a saída e outros dois botões incremente e decremente o PWM, de 50 a 100% com passos de 5%.
3. Ative as duas saídas PWMs ao mesmo tempo, uma para o ventilador e outra para a resistência. Utilize dois botões para controlar o ajuste de cada uma delas.

Capítulo 17 - Experiência 15 – Acesso às memórias de dados e programa

Objetivo

O acesso à memória de dados EEPROM já foi visto na experiência 6, portanto, a novidade desta experiência é o acesso à memória de programa do microcontrolador.

Descrição

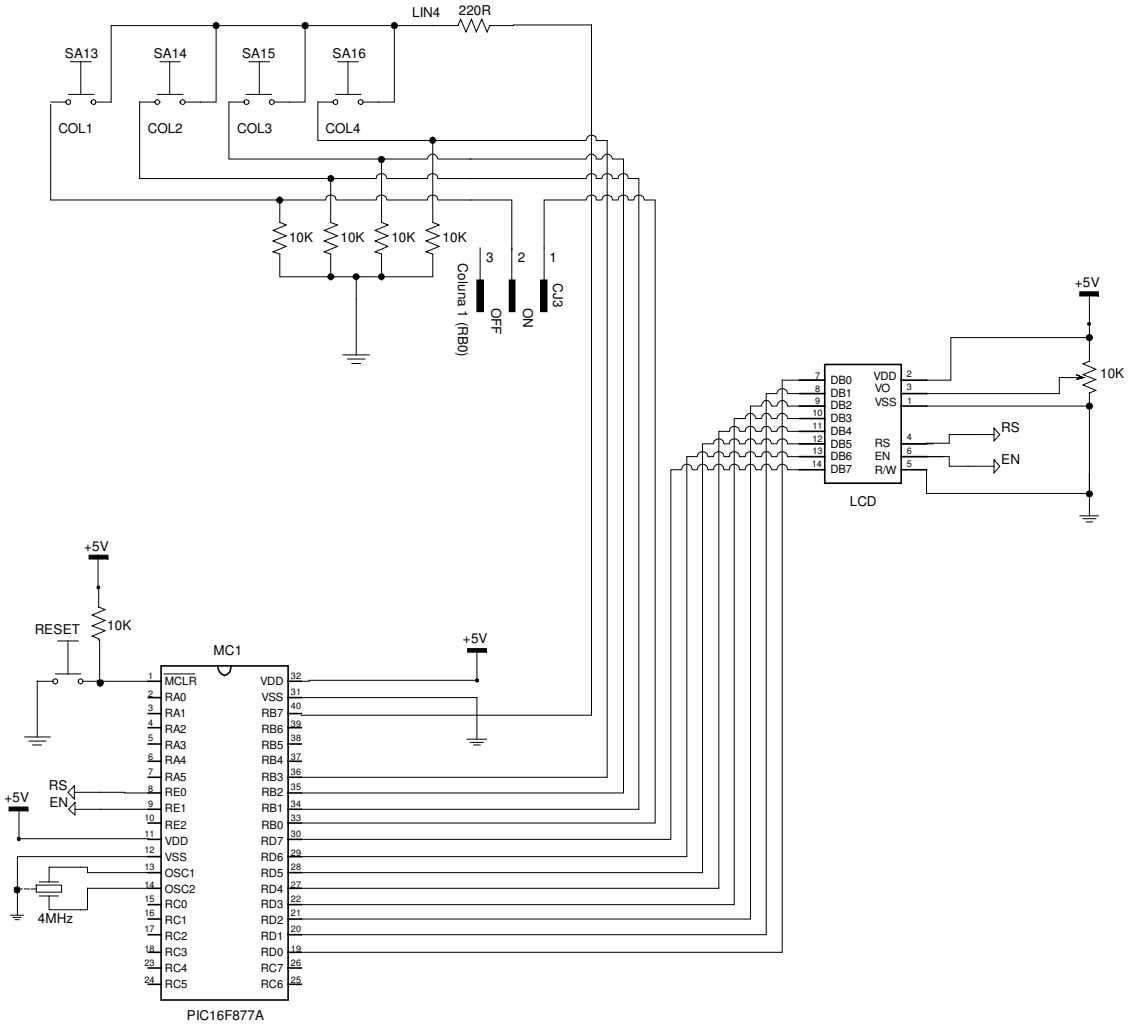
Nesta experiência o LCD está dividido em duas partes. Do lado esquerdo temos um valor relativo à memória de dados, variável de 0 a FFh (8-bits), com incremento e decremento rotativo através dos botões das colunas 2 e 3. Do lado direito o valor será para a memória de programa, também com incremento e decremento rotativo através dos botões das colunas 2 e 3, podendo ir de 0 a 3FFFh (14-bits).

Para alterar o controle dos botões entre o lado esquerdo e o lado direito deve ser usado o botão da coluna 1., sendo que o lado ativo no momento é indicado entre os sinais > e <.

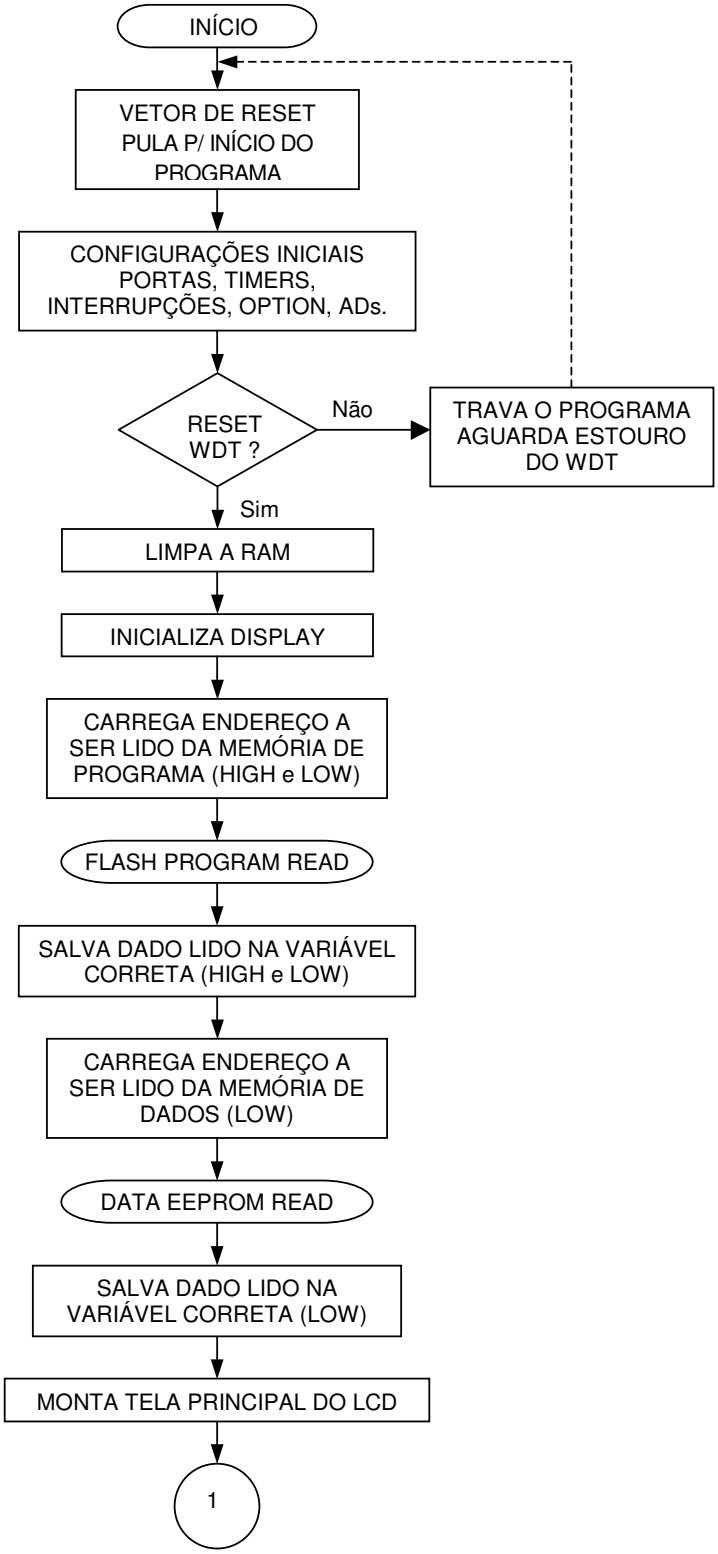
Depois de ajustados os valores desejados, basta pressionar o botão da coluna 4 para que ambos valores sejam gravados, cada um na memória correspondente.

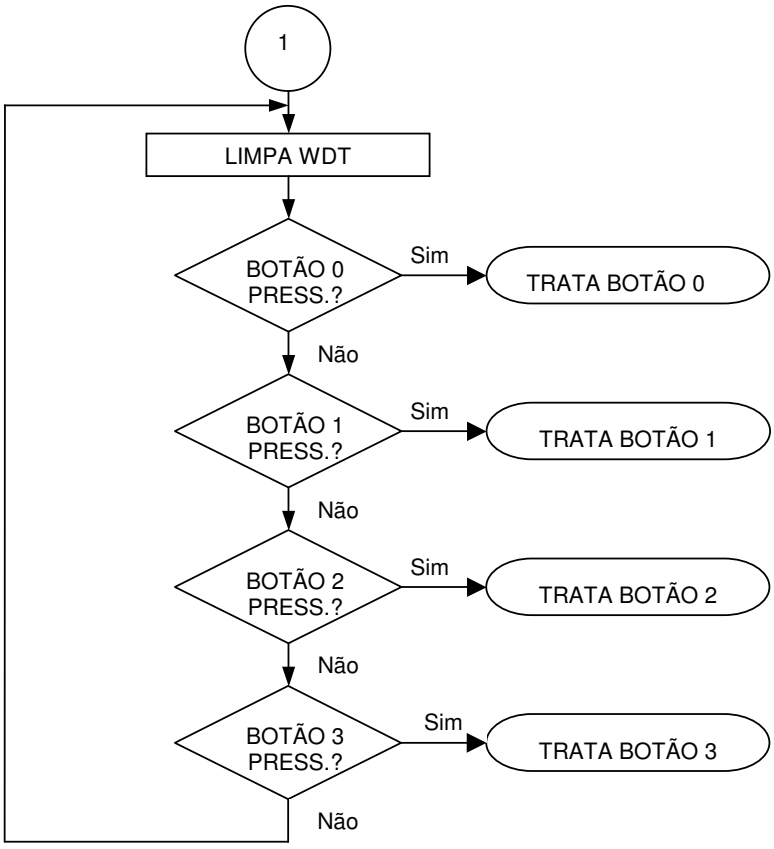
Para checar a gravação, altere os valores e reset o sistema (botão de reset). Os valores gravados serão recuperados na inicialização e mostrados no LCD.

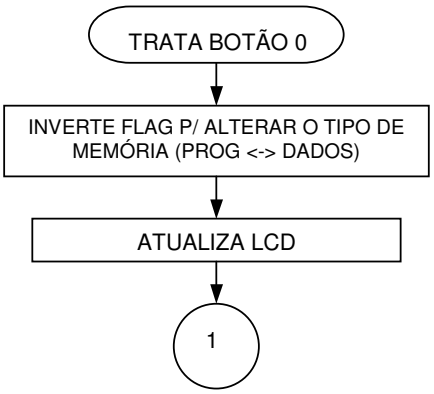
Esquema Elétrico

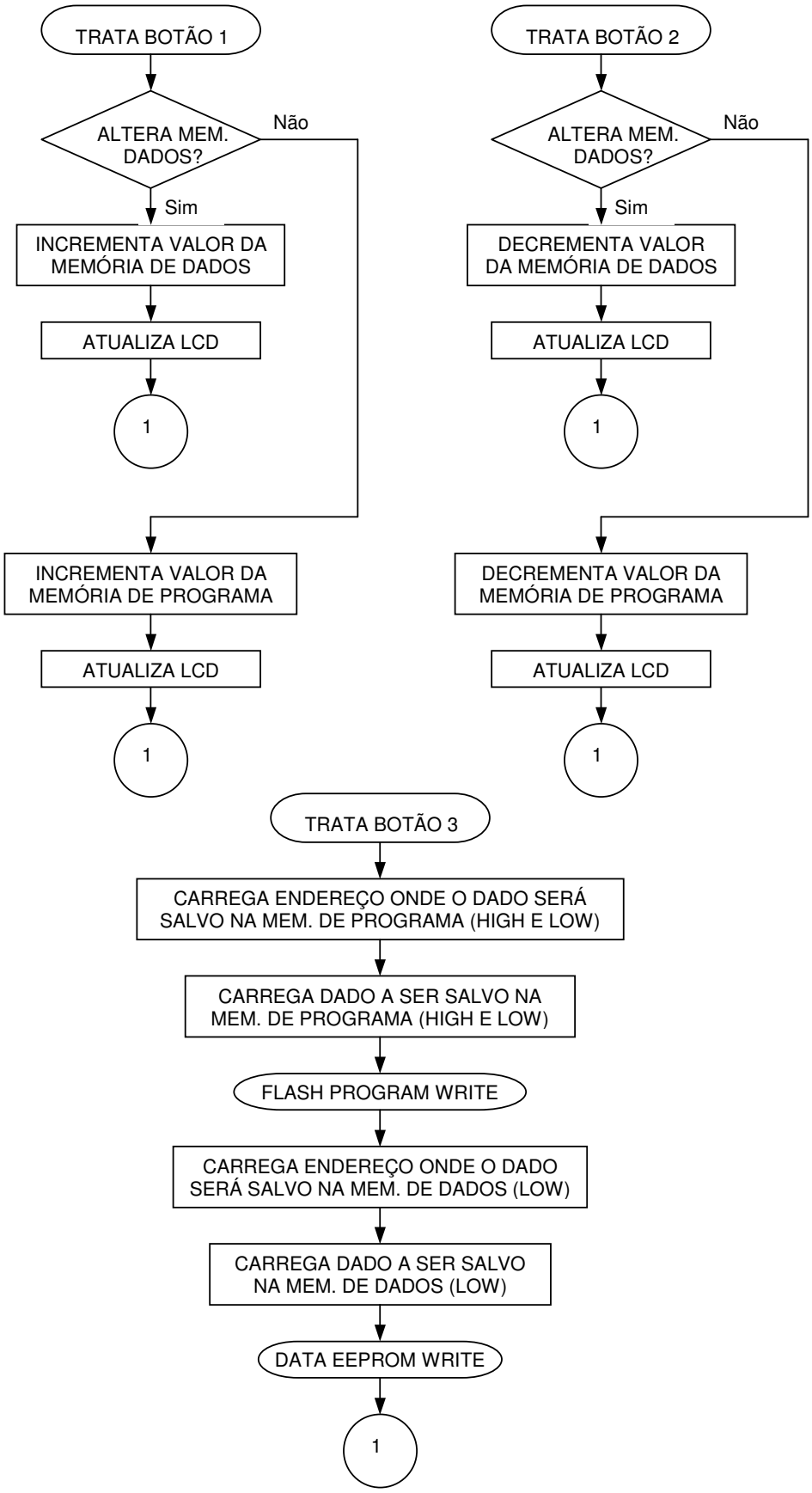


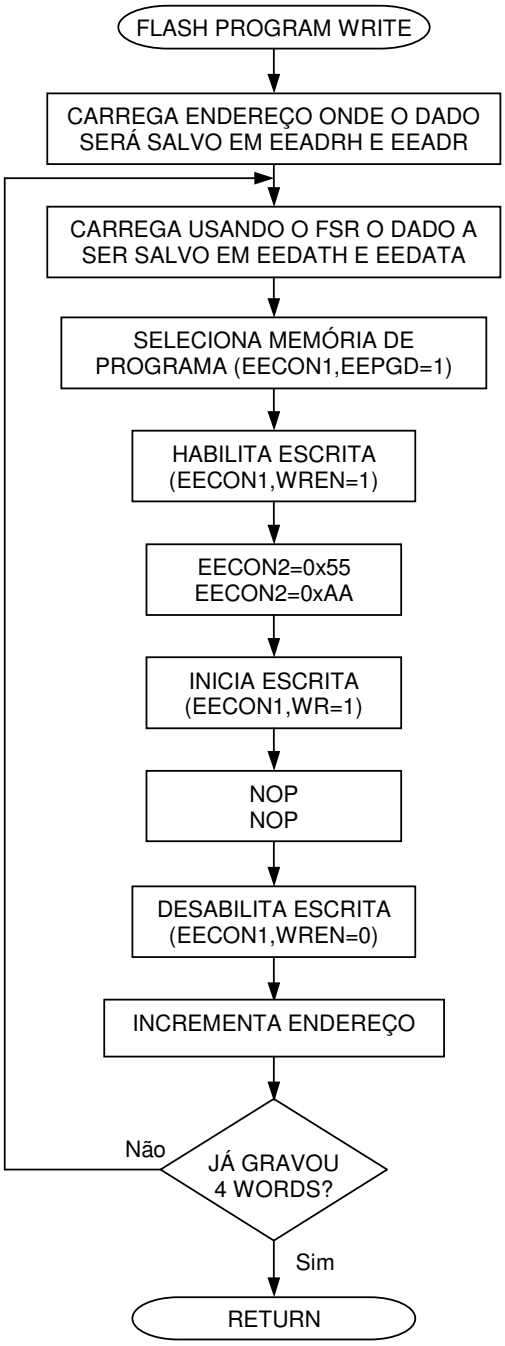
Fluxograma

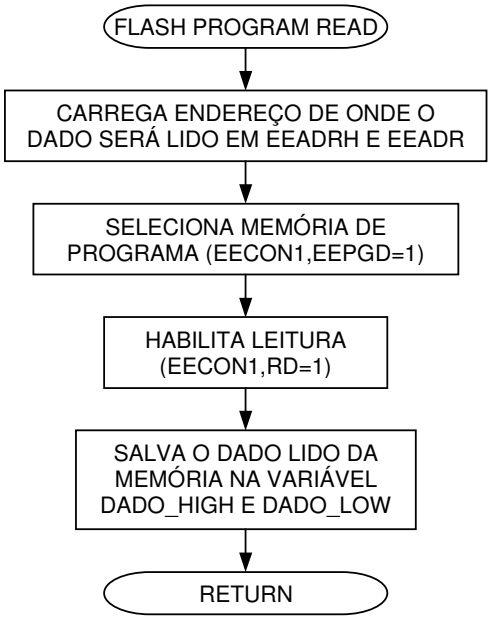

INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURAÇÕES INICIAIS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION, ADs.
RESET
WDT ?
Não
TRAVA O PROGRAMA
AGUARDA ESTOURO
DO WDT
Sim
LIMPA A RAM
INICIALIZA DISPLAY
CARREGA ENDEREÇO A
SER LIDO DA MEMÓRIA DE
PROGRAMA (HIGH e LOW)
FLASH PROGRAM READ
SALVA DADO LIDO NA VARIÁVEL
CORRETA (HIGH e LOW)
CARREGA ENDEREÇO A
SER LIDO DA MEMÓRIA DE
DADOS (LOW)
DATA EEPROM READ
SALVA DADO LIDO NA
VARIÁVEL CORRETA (LOW)
MONTA TELA PRINCIPAL DO LCD
1

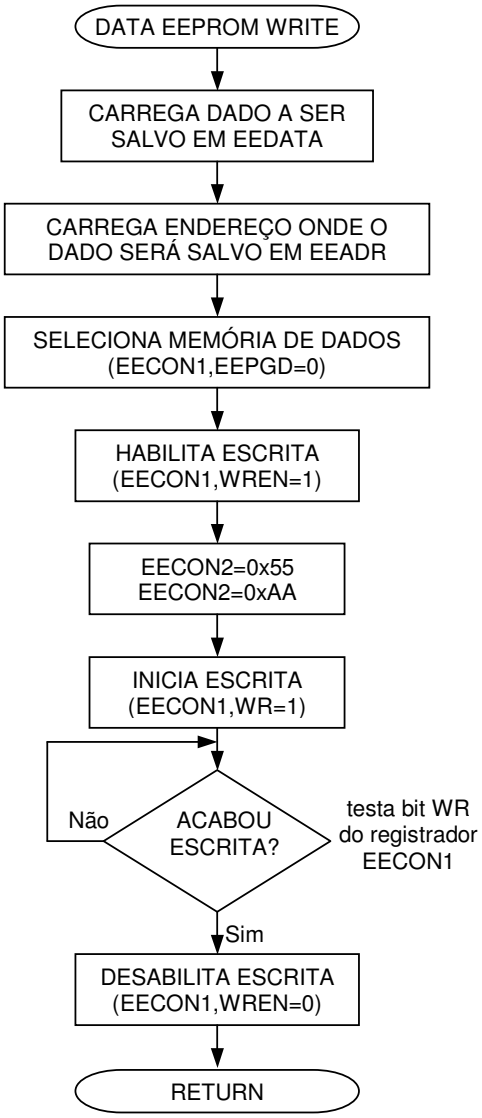

1
LIMPA WDT
BOTÃO 0 PRESS.?
Sim
TRATA BOTÃO 0
Não
BOTÃO 1 PRESS.?
Sim
TRATA BOTÃO 1
Não
BOTÃO 2 PRESS.?
Sim
TRATA BOTÃO 2
Não
BOTÃO 3 PRESS.?
Sim
TRATA BOTÃO 3
Não

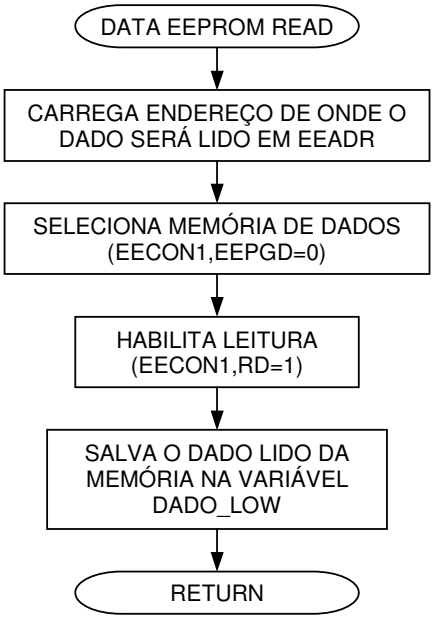

TRATA BOTÃO 0
INVERTE FLAG P/ ALTERAR O TIPO DE MEMÓRIA (PROG <-> DADOS)
ATUALIZA LCD
1


TRATA BOTÃO 1
ALTERA MEM. DADOS?
Não
Sim
INCREMENTA VALOR DA MEMÓRIA DE DADOS
ATUALIZA LCD
1
INCREMENTA VALOR DA MEMÓRIA DE PROGRAMA
ATUALIZA LCD
1
TRATA BOTÃO 2
ALTERA MEM. DADOS?
Não
Sim
DECREMENTA VALOR DA MEMÓRIA DE DADOS
ATUALIZA LCD
1
DECREMENTA VALOR DA MEMÓRIA DE PROGRAMA
ATUALIZA LCD
1
TRATA BOTÃO 3
CARREGA ENDEREÇO ONDE O DADO SERÁ SALVO NA MEM. DE PROGRAMA (HIGH E LOW)
CARREGA DADO A SER SALVO NA MEM. DE PROGRAMA (HIGH E LOW)
FLASH PROGRAM WRITE
CARREGA ENDEREÇO ONDE O DADO SERÁ SALVO NA MEM. DE DADOS (LOW)
CARREGA DADO A SER SALVO NA MEM. DE DADOS (LOW)
DATA EEPROM WRITE
1


FLASH PROGRAM WRITE
CARREGA ENDEREÇO ONDE O DADO SERÁ SALVO EM EEADRH E EEADR
CARREGA USANDO O FSR O DADO A SER SALVO EM EEDATH E EEDATA
SELECIONA MEMÓRIA DE PROGRAMA (EECON1,EEPGD=1)
HABILITA ESCRITA (EECON1,WREN=1)
EECON2=0x55
EECON2=0xAA
INICIA ESCRITA (EECON1,WR=1)
NOP
NOP
DESABILITA ESCRITA (EECON1,WREN=0)
INCREMENTA ENDEREÇO
Não
JÁ GRAVOU 4 WORDS?
Sim
RETURN


FLASH PROGRAM READ
CARREGA ENDEREÇO DE ONDE O DADO SERÁ LIDO EM EEADRH E EEADR
SELECIONA MEMÓRIA DE PROGRAMA (EECON1,EEPGD=1)
HABILITA LEITURA (EECON1,RD=1)
SALVA O DADO LIDO DA MEMÓRIA NA VARIÁVEL DADO_HIGH E DADO_LOW
RETURN


DATA EEPROM WRITE
CARREGA DADO A SER SALVO EM EEDATA
CARREGA ENDEREÇO ONDE O DADO SERÁ SALVO EM EEADR
SELECIONA MEMÓRIA DE DADOS (EECON1,EEPGD=0)
HABILITA ESCRITA (EECON1,WREN=1)
EECON2=0x55
EECON2=0xAA
INICIA ESCRITA (EECON1,WR=1)
Não
ACABOU ESCRITA?
testa bit WR do registrador EECON1
Sim
DESABILITA ESCRITA (EECON1,WREN=0)
RETURN


DATA EEPROM READ
CARREGA ENDEREÇO DE ONDE O DADO SERÁ LIDO EM EEADR
SELECIONA MEMÓRIA DE DADOS (EECON1,EEPGD=0)
HABILITA LEITURA (EECON1,RD=1)
SALVA O DADO LIDO DA MEMÓRIA NA VARIÁVEL DADO_LOW
RETURN

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *          EXPERIÊNCIA 15 - ACESSO ÀS MEMÓRIAS DE DADOS E PROGRAMA        *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            DESCRIÇÃO GERAL                              *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE EXEMPLO FOI ELABORADO PARA EXPLICAR O FUNCIONAMENTO DA LEITURA/ESCRITA
;  TANTO NA MEMÓRIA DE DADOS QUANTO NA MEMÓRIA DE PROGRAMA.
;  APENAS OS BOTÕES DA LINHA 4 ESTÃO HABILITADOS.
;  O BOTÃO DA COLUNA 1 PODE SER UTILIZADO PARA ALTERAR ENTRE OS VALORES DAS
;  MEMÓRIAS DE DADOS E PROGRAMAS. OS BOTÕES DAS COLUNAS 2 E 3 SÃO UTILIZADOS
;  PARA INCREMENTAR E DECREMENTAR OS VALORES. O BOTÃO DA COLUNA 4 É UTILIZADO
;  PARA SALVAR OS VALORES NAS MÉMORIAS DE DADOS E DE PROGRAMAS.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                         CONFIGURAÇÕES PARA GRAVAÇÃO                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_ON & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           DEFINIÇÃO DAS VARIÁVEIS                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE BLOCO DE VARIÁVEIS ESTÁ LOCALIZADO LOGO NO INÍCIO DO BANCO 0

   CBLOCK  0X20                    ; POSIÇÃO INICIAL DA RAM

           FILTRO_BOTOES           ; FILTRO PARA RUIDOS
           TEMPO_TURBO             ; TEMPORIZADOR P/ TURBO DAS TECLAS

           TEMPO1
           TEMPO0                  ; CONTADORES P/ DELAY

           FLAG                    ; FLAG DE USO GERAL

           VALOR_DADOS             ; VALOR ARMAZENADO NA MEMÓRIA
                                   ; DE DADOS (8 BITS)

           VALOR_PROG_HIGH         ; VALOR ARMAZENADO NA MEMÓRIA
           VALOR_PROG_LOW          ; DE PROGRAMAS (14 BITS)

           AUX                     ; REGISTRADOR AUXILIAR DE USO GERAL

           ENDERECO_HIGH           ; REGISTRADORES DE ENDEREÇO PARA
           ENDERECO_LOW            ; ACESSO À MEMÓRIA DE DADOS E PROGRAMA
                                   ; MAPEADOS NO BANCO 0 DA RAM

           DADO_LOW_00             ; REGISTRADORES DE DADOS PARA
           DADO_HIGH_00            ; ACESSO À MEMÓRIA DE DADOS E PROGRAMA
                                   ; MAPEADOS NO BANCO 0 DA RAM

           DADO_LOW_01             ; REGISTRADORES DE DADOS PARA
           DADO_HIGH_01            ; ACESSO À MEMÓRIA DE DADOS E PROGRAMA
                                   ; MAPEADOS NO BANCO 0 DA RAM

           DADO_LOW_10             ; REGISTRADORES DE DADOS PARA
           DADO_HIGH_10            ; ACESSO À MEMÓRIA DE DADOS E PROGRAMA
                                   ; MAPEADOS NO BANCO 0 DA RAM
```

```
          DADO_LOW_11             ; REGISTRADORES DE DADOS PARA
          DADO_HIGH_11            ; ACESSO À MEMÓRIA DE DADOS E PROGRAMA
                                  ; MAPEADOS NO BANCO 0 DA RAM

   ENDC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>        ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      DEFINIÇÃO DOS BANCOS DE RAM                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           CONSTANTES INTERNAS                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE CONSTANTES FACILITA A PROGRAMAÇÃO E A MANUTENÇÃO.

FILTRO_TECLA       EQU     .200    ; FILTRO P/ EVITAR RUIDOS DOS BOTÕES

TURBO_TECLA        EQU     .60     ; TEMPORIZADOR P/ TURBO DAS TECLAS

END_MEM_DADO       EQU     0X10    ; ENDEREÇO P/ LEITURA E GRAVAÇÃO
                                   ; NA MEMÓRIA DE DADOS

END_MEM_PROG_H     EQU     0X08    ; ENDEREÇO P/ LEITURA E GRAVAÇÃO
END_MEM_PROG_L     EQU     0X00    ; NA MEMÓRIA DE PROGRAMA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    DECLARAÇÃO DOS FLAGs DE SOFTWARE                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE FLAGs AJUDA NA PROGRAMAÇÃO E ECONOMIZA MEMÓRIA RAM.

#DEFINE    TIPO_MEMORIA    FLAG,0  ; DEFINE A MEMORIA QUE ESTA SENDO
                                   ;  UTILIZADA
                                   ; 1 -> MEMORIA DE PROGRAMA
                                   ; 0 -> MEMORIA DE DADOS

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               ENTRADAS                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS ENTRADAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    BOTAO_0         PORTB,0 ; ESTADO DO BOTÃO 0
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_1         PORTB,1 ; ESTADO DO BOTÃO 1
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_2         PORTB,2 ; ESTADO DO BOTÃO 2
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_3         PORTB,3 ; ESTADO DO BOTÃO 3
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
; *                                 SAÍDAS                                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS SAÍDAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    DISPLAY         PORTD   ; BARRAMENTO DE DADOS DO DISPLAY

#DEFINE    RS              PORTE,0 ; INDICA P/ O DISPLAY UM DADO OU COMANDO
                                   ; 1 -> DADO
                                   ; 0 -> COMANDO

#DEFINE    ENABLE          PORTE,1 ; SINAL DE ENABLE P/ DISPLAY
                                   ; ATIVO NA BORDA DE DESCIDA

#DEFINE    TEC_MATRICIAL   PORTB   ; PORT DO MICROCONTROLADOR LIGADO AO
                                   ; TECLADO MATRICIAL
                                   ; <RB4:RB7> LINHAS
                                   ; 1->ATIVADAS 0->DESATIVADAS
                                   ; <RB0:RB3> COLUNAS
                                   ; 1->TECLAS PRESSIONADAS 0->TECLAS LIBERADAS

#DEFINE    LINHA_4         PORTB,7 ; PINO P/ ATIVAR LINHA 4 (TECLADO MATRICIAL)
                                   ; 0 -> LINHA 4 ATIVADA
                                   ; 1 -> LINHA 4 DESATIVADA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  VETOR DE RESET DO MICROCONTROLADOR                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO INICIAL PARA EXECUÇÃO DO PROGRAMA

   ORG     0X0000                  ; ENDEREÇO DO VETOR DE RESET
   GOTO    CONFIG                  ; PULA PARA CONFIG DEVIDO A REGIÃO
                                   ; DESTINADA AS ROTINAS SEGUINTES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  ROTINA DE DELAY (DE 1MS ATÉ 256MS)                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA É UMA ROTINA DE DELAY VARIÁVEL, COM DURAÇÃO DE 1MS X O VALOR PASSADO
;  EM WORK (W).

DELAY_MS
   MOVWF   TEMPO1                  ; CARREGA TEMPO1 (UNIDADES DE MS)
   MOVLW   .250
   MOVWF   TEMPO0                  ; CARREGA TEMPO0 (P/ CONTAR 1MS)

   CLRWDT                          ; LIMPA WDT (PERDE TEMPO)
   DECFSZ  TEMPO0,F                ; FIM DE TEMPO0 ?
   GOTO    $-2                     ; NÃO - VOLTA 2 INSTRUÇÕES
                                   ; SIM - PASSOU-SE 1MS
   DECFSZ  TEMPO1,F                ; FIM DE TEMPO1 ?
   GOTO    $-6                     ; NÃO - VOLTA 6 INSTRUÇÕES
                                   ; SIM
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DE UM CARACTER NO DISPLAY                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ENVIA UM CARACTER PARA O MÓDULO DE LCD. O CARACTER A SER
;  ESCRITO DEVE SER COLOCADO EM WORK (W) ANTES DE CHAMAR A ROTINA.

ESCREVE
   MOVWF   DISPLAY                 ; ATUALIZA DISPLAY (PORTD)
   NOP                             ; PERDE 1US PARA ESTABILIZAÇÃO
   BSF     ENABLE                  ; ENVIA UM PULSO DE ENABLE AO DISPLAY
   GOTO    $+1                     ; .
   BCF     ENABLE                  ; .

   MOVLW   .1
   CALL    DELAY_MS                ; DELAY DE 1MS
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
; *               ROTINA DE ESCRITA LINHA 1 DO LCD                          *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A LINHA 1 DA TELA PRINCIPAL DO LCD, COM A FRASE:
;  LINHA 1 - "M.DADOS  M.PROG."

ATUALIZA_TELA_LINHA_1
   CLRF    TEC_MATRICIAL          ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                     ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0X80                   ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                ; LINHA 0 / COLUNA 0
   BSF     RS                     ; SELECIONA O DISPLAY P/ DADOS

                                  ; COMANDOS PARA ESCREVER AS
                                  ; LETRAS DE "M.DADOS  M.PROG."
   MOVLW   'M'
   CALL    ESCREVE
   MOVLW   '.'
   CALL    ESCREVE
   MOVLW   'D'
   CALL    ESCREVE
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   'D'
   CALL    ESCREVE
   MOVLW   'O'
   CALL    ESCREVE
   MOVLW   'S'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   'M'
   CALL    ESCREVE
   MOVLW   '.'
   CALL    ESCREVE
   MOVLW   'P'
   CALL    ESCREVE
   MOVLW   'R'
   CALL    ESCREVE
   MOVLW   'O'
   CALL    ESCREVE
   MOVLW   'G'
   CALL    ESCREVE
   MOVLW   '.'
   CALL    ESCREVE

   CLRF    DISPLAY                ; LIMPA BARRAMENTO DE DADOS

   RETURN                         ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               ROTINA DE ESCRITA LINHA 2 DO LCD                          *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A LINHA 2 DA TELA PRINCIPAL DO LCD.
;  A ROTINA LEVA EM CONTA TODAS AS VARIÁVEIS PERTINENTES P/ FORMAR A LINHA 2.

ATUALIZA_TELA_LINHA_2
   CLRF    TEC_MATRICIAL          ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                     ; SELECIONA O DISPLAY P/ COMANDO
   MOVLW   0XC1                   ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                ; LINHA 1 / COLUNA 1

   BSF     RS                     ; SELECIONA O DISPLAY P/ DADOS

   MOVLW   '>'
   BTFSC   TIPO_MEMORIA           ; ESTÁ UTILIZANDO A MEMÓRIA DE DADOS ?
   MOVLW   ' '                    ; NÃO - ESCREVE ESPAÇO EM BRANCO
   CALL    ESCREVE                ; SIM - ESCREVE ">" NO DISPLAY
```

```
SWAPF   VALOR_DADOS,W           ; INVERTE NIBLE DO VALOR_DADOS
ANDLW   B'00001111'             ; MASCARA BITS MAIS SIGNIFICATIVOS
MOVWF   AUX                     ; SALVA EM AUXILIAR

MOVLW   0X0A
SUBWF   AUX,W                   ; AUX - 10d (ATUALIZA FLAG DE CARRY)
MOVLW   0X30                    ; CARREGA WORK COM 30h
BTFSC   STATUS,C                ; RESULTADO É POSITIVO? (É UMA LETRA?)
MOVLW   0X37                    ; SIM - CARREGA WORK COM 37h
                                ; NÃO - WORK FICA COM 30h (NÚMERO)
ADDWF   AUX,W                   ; SOMA O WORK AO AUXILIAR
                                ; (CONVERSÃO ASCII)
CALL    ESCREVE                 ; ENVIA CARACTER AO DISPLAY LCD

MOVF    VALOR_DADOS,W           ; CARREGA WORK COM VALOR_DADOS
ANDLW   B'00001111'             ; MASCARA BITS MAIS SIGNIFICATIVOS
MOVWF   AUX                     ; SALVA EM AUXILIAR

MOVLW   0X0A
SUBWF   AUX,W                   ; AUX - 10d (ATUALIZA FLAG DE CARRY)
MOVLW   0X30                    ; CARREGA WORK COM 30h
BTFSC   STATUS,C                ; RESULTADO É POSITIVO? (É UMA LETRA?)
MOVLW   0X37                    ; SIM - CARREGA WORK COM 37h
                                ; NÃO - WORK FICA COM 30h (NÚMERO)
ADDWF   AUX,W                   ; SOMA O WORK AO AUXILIAR
                                ; (CONVERSÃO ASCII)
CALL    ESCREVE                 ; ENVIA CARACTER AO DISPLAY LCD

MOVLW   'h'
CALL    ESCREVE                 ; ESCREVE "h" NO DISPLAY

MOVLW   '<'
BTFSC   TIPO_MEMORIA            ; ESTÁ UTILIZANDO A MEMÓRIA DE DADOS ?
MOVLW   ' '                     ; NÃO - ESCREVE ESPAÇO EM BRANCO
CALL    ESCREVE                 ; SIM - ESCREVE "<" NO DISPLAY

MOVLW   ' '
CALL    ESCREVE
MOVLW   ' '
CALL    ESCREVE
MOVLW   ' '
CALL    ESCREVE                 ; ESPAÇOS EM BRANCO

MOVLW   '>'
BTFSS   TIPO_MEMORIA            ; ESTÁ UTILIZANDO A MEMÓRIA DE PROGRAMA?
MOVLW   ' '                     ; NÃO - ESCREVE ESPAÇO EM BRANCO
CALL    ESCREVE                 ; SIM - ESCREVE ">" NO DISPLAY

SWAPF   VALOR_PROG_HIGH,W       ; INVERTE NIBLE DO VALOR_PROG_HIGH
ANDLW   B'00001111'             ; MASCARA BITS MAIS SIGNIFICATIVOS
MOVWF   AUX                     ; SALVA EM AUXILIAR

MOVLW   0X0A
SUBWF   AUX,W                   ; AUX - 10d (ATUALIZA FLAG DE CARRY)
MOVLW   0X30                    ; CARREGA WORK COM 30h
BTFSC   STATUS,C                ; RESULTADO É POSITIVO? (É UMA LETRA?)
MOVLW   0X37                    ; SIM - CARREGA WORK COM 37h
                                ; NÃO - WORK FICA COM 30h (NÚMERO)
ADDWF   AUX,W                   ; SOMA O WORK AO AUXILIAR
                                ; (CONVERSÃO ASCII)
CALL    ESCREVE                 ; ENVIA CARACTER AO DISPLAY LCD

MOVF    VALOR_PROG_HIGH,W       ; CARREGA WORK COM VALOR_PROG_HIGH
ANDLW   B'00001111'             ; MASCARA BITS MAIS SIGNIFICATIVOS
MOVWF   AUX                     ; SALVA EM AUXILIAR

MOVLW   0X0A
SUBWF   AUX,W                   ; AUX - 10d (ATUALIZA FLAG DE CARRY)
MOVLW   0X30                    ; CARREGA WORK COM 30h
BTFSC   STATUS,C                ; RESULTADO É POSITIVO? (É UMA LETRA?)
MOVLW   0X37                    ; SIM - CARREGA WORK COM 37h
                                ; NÃO - WORK FICA COM 30h (NÚMERO)
```

```
   ADDWF   AUX,W                  ; SOMA O WORK AO AUXILIAR
                                  ; (CONVERSÃO ASCII)
   CALL    ESCREVE                ; ENVIA CARACTER AO DISPLAY LCD

   SWAPF   VALOR_PROG_LOW,W       ; INVERTE NIBLE DO VALOR_PROG_LOW
   ANDLW   B'00001111'            ; MASCARA BITS MAIS SIGNIFICATIVOS
   MOVWF   AUX                    ; SALVA EM AUXILIAR

   MOVLW   0X0A
   SUBWF   AUX,W                  ; AUX - 10d (ATUALIZA FLAG DE CARRY)
   MOVLW   0X30                   ; CARREGA WORK COM 30h
   BTFSC   STATUS,C               ; RESULTADO É POSITIVO? (É UMA LETRA?)
   MOVLW   0X37                   ; SIM - CARREGA WORK COM 37h
                                  ; NÃO - WORK FICA COM 30h (NÚMERO)
   ADDWF   AUX,W                  ; SOMA O WORK AO AUXILIAR
                                  ; (CONVERSÃO ASCII)
   CALL    ESCREVE                ; ENVIA CARACTER AO DISPLAY LCD

   MOVF    VALOR_PROG_LOW,W       ; CARREGA WORK COM VALOR_PROG_LOW
   ANDLW   B'00001111'            ; MASCARA BITS MAIS SIGNIFICATIVOS
   MOVWF   AUX                    ; SALVA EM AUXILIAR

   MOVLW   0X0A
   SUBWF   AUX,W                  ; AUX - 10d (ATUALIZA FLAG DE CARRY)
   MOVLW   0X30                   ; CARREGA WORK COM 30h
   BTFSC   STATUS,C               ; RESULTADO É POSITIVO? (É UMA LETRA?)
   MOVLW   0X37                   ; SIM - CARREGA WORK COM 37h
                                  ; NÃO - WORK FICA COM 30h (NÚMERO)
   ADDWF   AUX,W                  ; SOMA O WORK AO AUXILIAR
                                  ; (CONVERSÃO ASCII)
   CALL    ESCREVE                ; ENVIA CARACTER AO DISPLAY LCD

   MOVLW   'h'
   CALL    ESCREVE                ; ESCREVE "h" NO DISPLAY

   MOVLW   '<'
   BTFSS   TIPO_MEMORIA           ; ESTÁ UTILIZANDO A MEMÓRIA DE PROGRAMA?
   MOVLW   ' '                    ; NÃO - ESCREVE ESPAÇO EM BRANCO
   CALL    ESCREVE                ; SIM - ESCREVE "<" NO DISPLAY

   CLRF    DISPLAY                ; LIMPA BARRAMENTO DE DADOS

   RETURN

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA NA MEMÓRIA DE DADOS                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA ESCREVE UM DADO (8 BITS) NA MEMÓRIA DE DADOS (E2PROM).
; O DADO A SER GRAVADO DEVE SER PASSADO PELO REGISTRADOR DADO_LOW_00.
; O REGISTRADOR DADO_HIGH_00 NÃO É UTILIZADO POIS A MEMÓRIA É DE 8 BITS.
; O ENDEREÇO DEVE SER PASSADO PELO REGISTRADOR ENDERECO_LOW.
; O REGISTRADOR ENDERECO_HIGH NÃO É UTILIZADO, POIS A MEMÓRIA TEM 256 ENDER.

DATA_EEPROM_WRITE
   MOVF    DADO_LOW_00,W          ; CARREGA NO WORK DADO P/ SER GRAVADO
   BANKSEL EEDATA                 ; ALTERA P/ BANK DO REGISTRADOR EEDATA
   MOVWF   EEDATA                 ; SALVA DADO A SER GRAVADO EM EEDATA
                                  ; (CARREGA DADO NO REGISTRADOR
                                  ; CORRETO DO BANCO 2 DA RAM A PARTIR
                                  ; DO REGISTRADOR DE USUÁRIO MAPEADO
                                  ; NO BANCO 0 DA RAM)

   BANKSEL ENDERECO_LOW           ; ALTERA P/BANK DO REGIST. ENDERECO_LOW
   MOVF    ENDERECO_LOW,W         ; CARREGA NO WORK O ENDEREÇO DE DESTINO
   BANKSEL EEADR                  ; ALTERA P/ BANK DO REGISTRADOR EEADR
   MOVWF   EEADR                  ; SALVA ENDERECO EM EEADR
                                  ; (CARREGA ENDEREÇO NO REGISTRADOR
                                  ; CORRETO DO BANCO 2 DA RAM A PARTIR
                                  ; DO REGISTRADOR DE USUÁRIO MAPEADO
                                  ; NO BANCO 0 DA RAM)

   BANKSEL EECON1                 ; ALTERA P/ BANK DO REGISTRADOR EECON1
```

```
   BCF     EECON1,EEPGD            ; APONTA P/ MEMÓRIA DE DADOS
   BSF     EECON1,WREN             ; HABILITA ESCRITA

   MOVLW   0X55
   MOVWF   EECON2                  ; ESCREVE 0X55 EM EECON2 (OBRIGATÓRIO)
   MOVLW   0XAA
   MOVWF   EECON2                  ; ESCREVE 0XAA EM EECON2 (OBRIGATÓRIO)
   BSF     EECON1,WR               ; INICIA ESCRITA

   BTFSC   EECON1,WR               ; ACABOU ESCRITA ?
   GOTO    $-1                     ; NÃO - AGUARDA FIM DA ESCRITA
                                   ; SIM
   BCF     EECON1,WREN             ; DESABILITA ESCRITAS NA MEMÓRIA

   BANKSEL 0X20                    ; VOLTA P/ BANK0
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               ROTINA DE LEITURA NA MEMÓRIA DE DADOS                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA LÊ UM DADO (8 BITS) DA MEMÓRIA DE DADOS (E2PROM).
; O DADO A SER LIDO É RETORNADO NO REGISTRADOR DADO_LOW_00.
; O REGISTRADOR DADO_HIGH_00 NÃO É UTILIZADO POIS A MEMÓRIA É DE 8 BITS.
; O ENDEREÇO DEVE SER PASSADO PELO REGISTRADOR ENDERECO_LOW.
; O REGISTRADOR ENDERECO_HIGH NÃO É UTILIZADO, POIS A MEMÓRIA TEM 256 ENDER.

DATA_EEPROM_READ
   MOVF    ENDERECO_LOW,W          ; CARREGA NO WORK O ENDEREÇO DE DESTINO
   BANKSEL EEADR                   ; ALTERA P/ BANK DO REGISTRADOR EEADR
   MOVWF   EEADR                   ; SALVA ENDERECO EM EEADR
                                   ; (CARREGA ENDERECO NO REGISTRADOR
                                   ; CORRETO DO BANCO 2 DA RAM A PARTIR
                                   ; DO REGISTRADOR DE USUÁRIO MAPEADO
                                   ; NO BANCO 0 DA RAM)

   BANKSEL EECON1                  ; ALTERA P/ BANK DO REGISTRADOR EECON1
   BCF     EECON1,EEPGD            ; APONTA P/ MEMÓRIA DE DADOS
   BSF     EECON1,RD               ; HABILITA LEITURA

   BANKSEL EEDATA                  ; ALTERA P/BANK DO REGISTRADOR EEDATA
   MOVF    EEDATA,W                ; SALVA DADO LIDO NO WORK
   BANKSEL DADO_LOW_00             ; ALTERA P/BANK DO REGIST. DADO_LOW_00
   MOVWF   DADO_LOW_00             ; SALVA DADO LIDO EM DADO_LOW_00
                                   ; (SALVA DADO LIDO NO REGISTRADOR
                                   ; DE USUÁRIO MAPEADO NO BANCO 0 DA RAM
                                   ; A PARTIR DO REGISTRADOR UTILIZADO
                                   ; PELO MICROCONTROLADOR MAPEADO
                                   ; NO BANCO 2 DA RAM)

   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               ROTINA DE ESCRITA NA MEMÓRIA DE PROGRAMA                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; A ESCRITA NA MEMÓRIA DE PROGRAMA É FEITA DE 4 EM 4 WORDS OU DE 8 EM 8 BYTES
; OBRIGATORIAMENTE. O ENDEREÇO DEVE OBRIGATORIAMENTE ESTAR ALINHADO, OU SEJA,
; O ENDEREÇO INICIAL DEVERÁ SEMPRE TER OS ÚLTIMOS DOIS BITS EM 00. DESTA FORMA,
; SEMPRE A ESCRITA NA MEMÓRIA DE PROGRAMA É FEITA NOS ENDEREÇOS COM FINAIS 00,
; 01, 10 E 11, COMPLETANDO ASSIM 4 WORDS.
; ESTA ROTINA ESCREVE QUATRO WORDS (14 BITS) NA MEMÓRIA DE PROGRAMA.
; OS VAORES A SEREM SALVOS DEVEM SER PASSADOS PELOS REGISTRADORES
; DADO_HIGH_00:DADO_LOW_00, DADO_HIGH_01:DADO_LOW_01,
; DADO_HIGH_10:DADO_LOW_10 E DADO_HIGH_11:DADO_LOW_11.
; O ENDEREÇO DEVE SER PASSADO PELOS REGIST. ENDERECO_HIGH E ENDERECO_LOW.

FLASH_PROGRAM_WRITE
   MOVF    ENDERECO_HIGH,W         ; CARREGA NO WORK O ENDEREÇO DE DESTINO
   BANKSEL EEADRH                  ; ALTERA P/ BANK DO REGISTRADOR EEADH
   MOVWF   EEADRH                  ; SALVA ENDERECO EM EEADH
   BANKSEL ENDERECO_LOW            ; ALTERA P/BANK DO REGIST. ENDERECO_LOW
   MOVF    ENDERECO_LOW,W          ; CARREGA NO WORK O ENDEREÇO DE DESTINO
   ANDLW   B'11111100'             ; MASCARA PARA ZERAR OS ÚLTIMOS DOIS BIT
```

```
   BANKSEL EEADR                    ; ALTERA P/ BANK DO REGISTRADOR EEADR
   MOVWF   EEADR                    ; SALVA ENDERECO EM EEADR
                                    ; (CARREGA ENDEREÇO NOS REGISTRADOS
                                    ; CORRETOS DO BANCO 2 DA RAM A PARTIR
                                    ; DOS REGISTRADORES DE USUÁRIO MAPEADOS
                                    ; NO BANCO 0 DA RAM)

   MOVLW   DADO_LOW_00              ; CARREGA NO W ENDEREÇO DO REGISTRADOR
DADO_HIGH_00
   MOVWF   FSR                      ; SALVA O ENDEREÇO DO REGISTRADOR NO FSR

FLASH_PROGRAM_WRITE_2
   BANKSEL EEDATA                   ; ALTERA P/ BANK DO REGISTRADOR EEDATA
   MOVF    INDF,W                   ; CARREGA NO W O VALOR A SER SALVO
   MOVWF   EEDATA                   ; SALVA DADO A SER GRAVADO EM EEDATA
   INCF    FSR,F                    ; INCREMENTA PONTEIRO
   MOVF    INDF,W                   ; CARREGA NO W O VALOR A SER SALVO
   MOVWF   EEDATH                   ; SALVA DADO A SER GRAVADO EM EEDATH
   INCF    FSR,F                    ; INCREMENTA PONTEIRO

   BANKSEL EECON1                   ; ALTERA P/ BANK DO REGISTRADOR EECON1
   BSF     EECON1,EEPGD             ; APONTA P/ MEMÓRIA DE PROGRAMA
   BSF     EECON1,WREN              ; HABILITA ESCRITA

   MOVLW   0X55
   MOVWF   EECON2                   ; ESCREVE 0X55 EM EECON2 (OBRIGATÓRIO)
   MOVLW   0XAA
   MOVWF   EECON2                   ; ESCREVE 0XAA EM EECON2 (OBRIGATÓRIO)
   BSF     EECON1,WR                ; INICIA ESCRITA

   NOP
   NOP                              ; NÃO OPERA

   BCF     EECON1,WREN              ; DESABILITA ESCRITAS NA MEMÓRIA

   BANKSEL EEADR                    ; ALTERA P/ BANK DO REGISTRADOR EEADR
   INCF    EEADR,F                  ; INCREMENTA ENDEREÇO

   MOVLW   B'00000011'              ; CARREGA MASCARA NO WORK
   ANDWF   EEADR,W                  ; WORK FICA COM APENAS OS ÚLTIMOS DOIS BITS DO
ENDEREÇO
   BTFSS   STATUS,Z                 ; DEVE ESCREVER MAIS ALGUM DADO ? (WORK
DIFERENTE DE ZERO?)
   GOTO    FLASH_PROGRAM_WRITE_2    ; SIM - VOLTA PARA ESCRITA
                                    ; NÃO

   BANKSEL 0X20                     ; VOLTA P/ BANK0
   RETURN                           ; RETORNA DA SUBROTINA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               ROTINA DE LEITURA NA MEMÓRIA DE DADOS                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA LÊ UM DADO (14 BITS) DA MEMÓRIA DE PROGRAMA.
; O DADO LIDO É RETORNADO NOS REGISTRADORES DADO_HIGH_00 E DADO_LOW._00
; O ENDEREÇO DEVE SER PASSADO PELOS REGIST. ENDERECO_HIGH E ENDERECO_LOW.

FLASH_PROGRAM_READ
   MOVF    ENDERECO_HIGH,W          ; CARREGA NO WORK O ENDEREÇO DE DESTINO
   BANKSEL EEADRH                   ; ALTERA P/ BANK DO REGISTRADOR EEADH
   MOVWF   EEADRH                   ; SALVA ENDERECO EM EEADH
   BANKSEL ENDERECO_LOW             ; ALTERA P/BANK DO REGIST. ENDERECO_LOW
   MOVF    ENDERECO_LOW,W           ; CARREGA NO WORK O ENDEREÇO DE DESTINO
   BANKSEL EEADR                    ; ALTERA P/ BANK DO REGISTRADOR EEADR
   MOVWF   EEADR                    ; SALVA ENDERECO EM EEADR
                                    ; (CARREGA ENDEREÇO NOS REGISTRADOS
                                    ; CORRETOS DO BANCO 2 DA RAM A PARTIR
                                    ; DOS REGISTRADORES DE USUÁRIO MAPEADOS
                                    ; NO BANCO 0 DA RAM)

   BANKSEL EECON1                   ; ALTERA P/ BANK DO REGISTRADOR EECON1
   BSF     EECON1,EEPGD             ; APONTA P/ MEMÓRIA DE PROGRAMA
   BSF     EECON1,RD                ; HABILITA LEITURA
```

```
   NOP
   NOP

   BANKSEL EEDATH                ; ALTERA P/ BANK DO REGISTRADOR EEDATH
   MOVF    EEDATH,W              ; SALVA DADO LIDO NO WORK
   BANKSEL DADO_HIGH_00          ; ALTERA P/ BANK DO REGIST. DADO_HIGH
   MOVWF   DADO_HIGH_00          ; SALVA DADO LIDO EM DADO_HIGH_00
   BANKSEL EEDATA                ; ALTERA P/ BANK DO REGISTRADOR EEDATA_00
   MOVF    EEDATA,W              ; SALVA DADO LIDO NO WORK
   BANKSEL DADO_LOW_00           ; ALTERA P/ BANK DO REGIST. DADO_LOW_00
   MOVWF   DADO_LOW_00           ; SALVA DADO LIDO EM DADO_LOW_00
                                 ; (SALVA DADO LIDO NOS REGISTRADORES
                                 ; DE USUÁRIO MAPEADOS NO BANCO 0 DA RAM
                                 ; A PARTIR DOS REGISTRADORES UTILIZADOS
                                 ; PELO MICROCONTROLADOR MAPEADOS
                                 ; NO BANCO 2 DA RAM)

   RETURN                        ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              CONFIGURAÇÕES INICIAIS DE HARDWARE E SOFTWARE             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  NESTA ROTINA SÃO INICIALIZADAS AS PORTAS DE I/O DO MICROCONTROLADOR E AS
;  CONFIGURAÇÕES DOS REGISTRADORES ESPECIAIS (SFR). A ROTINA INICIALIZA A
;  MÁQUINA E AGUARDA O ESTOURO DO WDT.

CONFIG
   CLRF    PORTA                 ; LIMPA O PORTA
   CLRF    PORTB                 ; LIMPA O PORTB
   CLRF    PORTC                 ; LIMPA O PORTC
   CLRF    PORTD                 ; LIMPA O PORTD
   CLRF    PORTE                 ; LIMPA O PORTE


   BANK1                         ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                 ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                 ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011000'
   MOVWF   TRISC                 ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                 ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                 ; CONFIGURA I/O DO PORTE

   MOVLW   B'11011111'
   MOVWF   OPTION_REG            ; CONFIGURA OPTIONS
                                 ; PULL-UPs DESABILITADOS
                                 ; INTER. NA BORDA DE SUBIDA DO RB0
                                 ; TIMER0 INCREM. PELO CICLO DE MÁQUINA
                                 ; WDT   - 1:128
                                 ; TIMER - 1:1

   MOVLW   B'00000000'
   MOVWF   INTCON                ; CONFIGURA INTERRUPÇÕES
                                 ; DESABILITADA TODAS AS INTERRUPÇÕES

   MOVLW   B'00000111'
   MOVWF   CMCON                 ; DESLIGA OS COMPARADORES

   MOVLW   B'00000111'
   MOVWF   ADCON1                ; CONFIGURA CONVERSOR A/D
                                 ; CONFIGURA PORTA E PORTE COMO I/O DIGITAL

   BANK0                         ; SELECIONA BANCO 0 DA RAM
```

```
;  AS INSTRUÇÕES A SEGUIR FAZEM COM QUE O PROGRAMA TRAVE QUANDO HOUVER UM
;  RESET OU POWER-UP, MAS PASSE DIRETO SE O RESET FOR POR WDT. DESTA FORMA,
;  SEMPRE QUE O PIC É LIGADO, O PROGRAMA TRAVA, AGUARDA UM ESTOURO DE WDT
;  E COMEÇA NOVAMENTE. ISTO EVITA PROBLEMAS NO START-UP DO PIC.

   BTFSC   STATUS,NOT_TO          ; RESET POR ESTOURO DE WATCHDOG TIMER ?
   GOTO    $                      ; NÃO - AGUARDA ESTOURO DO WDT
                                  ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       INICIALIZAÇÃO DA RAM                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA IRÁ LIMPAR TODA A RAM DO BANCO 0, INDO DE 0X20 A 0X7F.

   MOVLW   0X20
   MOVWF   FSR                    ; APONTA O ENDEREÇAMENTO INDIRETO PARA
                                  ; A PRIMEIRA POSIÇÃO DA RAM
LIMPA_RAM
   CLRF    INDF                   ; LIMPA A POSIÇÃO
   INCF    FSR,F                  ; INCREMENTA O PONTEIRO P/ A PRÓX. POS.
   MOVF    FSR,W
   XORLW   0X80                   ; COMPARA O PONTEIRO COM A ÚLT. POS. +1
   BTFSS   STATUS,Z               ; JÁ LIMPOU TODAS AS POSIÇÕES?
   GOTO    LIMPA_RAM              ; NÃO - LIMPA A PRÓXIMA POSIÇÃO
                                  ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       CONFIGURAÇÕES INICIAIS DO DISPLAY                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA INICIALIZA O DISPLAY P/ COMUNICAÇÃO DE 8 VIAS, DISPLAY PARA 2
; LINHAS, CURSOR APAGADO E DESLOCAMENTO DO CURSOR À DIREITA.

INICIALIZACAO_DISPLAY
   BCF     RS                     ; SELECIONA O DISPLAY P/ COMANDOS

   MOVLW   0X30                   ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                ; INICIALIZAÇÃO

   MOVLW   .3
   CALL    DELAY_MS               ; DELAY DE 3MS (EXIGIDO PELO DISPLAY)

   MOVLW   0X30                   ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                ; INICIALIZAÇÃO

   MOVLW   0X30                   ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                ; INICIALIZAÇÃO

   MOVLW   B'00111000'            ; ESCREVE COMANDO PARA
   CALL    ESCREVE                ; INTERFACE DE 8 VIAS DE DADOS

   MOVLW   B'00000001'            ; ESCREVE COMANDO PARA
   CALL    ESCREVE                ; LIMPAR TODO O DISPLAY

   MOVLW   .1
   CALL    DELAY_MS               ; DELAY DE 1MS

   MOVLW   B'00001100'            ; ESCREVE COMANDO PARA
   CALL    ESCREVE                ; LIGAR O DISPLAY SEM CURSOR

   MOVLW   B'00000110'            ; ESCREVE COMANDO PARA INCREM.
   CALL    ESCREVE                ; AUTOMÁTICO À DIREITA

   BSF     RS                     ; SELECIONA O DISPLAY P/ DADOS

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       INICIALIZAÇÃO DA RAM                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE TRECHO DO PROGRAMA LÊ OS DADOS DAS MEMÓRIAS (E2PROM E FLASH) E
;  ATUALIZA A RAM.

LE_MEMORIA_PROGRAMA
   MOVLW   END_MEM_PROG_H
```

```
   MOVWF   ENDERECO_HIGH
   MOVLW   END_MEM_PROG_L
   MOVWF   ENDERECO_LOW            ; CARREGA ENDERECO P/ LEITURA

   CALL    FLASH_PROGRAM_READ      ; CHAMA ROTINA P/ LER DADO

   MOVF    DADO_HIGH_00,W
   MOVWF   VALOR_PROG_HIGH
   MOVF    DADO_LOW_00,W
   MOVWF   VALOR_PROG_LOW          ; SALVA O DADO LIDO EM
                                   ; VALOR_PROG_HIGH E VALOR_PROG_LOW
LE_MEMORIA_DADOS
   MOVLW   END_MEM_DADO
   MOVWF   ENDERECO_LOW            ; CARREGA ENDERECO P/ LEITURA

   CALL    DATA_EEPROM_READ        ; CHAMA ROTINA P/ LER DADO

   MOVF    DADO_LOW_00,W
   MOVWF   VALOR_DADOS             ; SALVA DADO LIDO EM VALOR_DADOS

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DA TELA PRINCIPAL                          *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A TELA PRINCIPAL DO PROGRAMA, COM AS FRASES:
;  LINHA 1 - "M.DADOS  M.PROG."
;  LINHA 2 - " >xxh<    xxxxh "

   CALL    ATUALIZA_TELA_LINHA_1   ; ATUALIZA TELA LINHA 1 DO LCD

   CALL    ATUALIZA_TELA_LINHA_2   ; ATUALIZA TELA LINHA 2 DO LCD

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          VARREDURA DOS BOTÕES                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA VERIFICA SE ALGUM BOTÃO ESTÁ PRESSIONADO E CASO AFIRMATIVO
; DESVIA PARA O TRATAMENTO DO MESMO.

VARRE
   CLRWDT                          ; LIMPA WATCHDOG TIMER

; ***************** VERIFICA ALGUM BOTÃO PRESSIONADO ************************

VARRE_BOTOES
   BSF     LINHA_4                 ; ATIVA LINHA 4 DO TECLADO MATRICIAL

   GOTO    $+1                     ; DELAY PARA ESTABILIZAÇÃO
                                   ; E LEITURA DO TECLADO

   BTFSC   BOTAO_0                 ; O BOTÃO 0 ESTA PRESSIONADO ?
   GOTO    TRATA_BOTAO_0           ; SIM - PULA P/ TRATA_BOTAO_0
                                   ; NÃO

   BTFSC   BOTAO_1                 ; O BOTÃO 1 ESTA PRESSIONADO ?
   GOTO    TRATA_BOTAO_1           ; SIM - PULA P/ TRATA_BOTAO_1
                                   ; NÃO

   BTFSC   BOTAO_2                 ; O BOTÃO 2 ESTA PRESSIONADO ?
   GOTO    TRATA_BOTAO_2           ; SIM - PULA P/ TRATA_BOTAO_2
                                   ; NÃO

   BTFSC   BOTAO_3                 ; O BOTÃO 3 ESTA PRESSIONADO ?
   GOTO    TRATA_BOTAO_3           ; SIM - PULA P/ TRATA_BOTAO_3
                                   ; NÃO

   BCF     LINHA_4                 ; DESATIVA A LINHA 4 DO TECLADO MATRICIAL

; ************************** FILTRO P/ EVITAR RUIDOS ************************

   MOVLW   FILTRO_TECLA            ; CARREGA O VALOR DE FILTRO_TECLA
   MOVWF   FILTRO_BOTOES           ; SALVA EM FILTRO_BOTOES
                                   ; RECARREGA FILTRO P/ EVITAR RUIDOS
                                   ; NOS BOTÕES
```

```
   MOVLW   .1
   MOVWF   TEMPO_TURBO             ; CARREGA TEMPO DO TURBO DAS TECLAS
                                   ; COM 1 - IGNORA O TURBO A PRIMEIRA
                                   ; VEZ QUE A TECLA É PRESSIONADA

   GOTO    VARRE                   ; VOLTA PARA VARRER TECLADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                         TRATAMENTO DOS BOTÕES                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; NESTE TRECHO DO PROGRAMA ESTÃO TODOS OS TRATAMENTOS DOS BOTÕES

; ************************ TRATAMENTO DO BOTÃO 0 ***************************

TRATA_BOTAO_0
   MOVF    FILTRO_BOTOES,F
   BTFSC   STATUS,Z                ; FILTRO JÁ IGUAL A ZERO ?
                                   ; (FUNÇÃO JA FOI EXECUTADA?)
   GOTO    VARRE                   ; SIM - VOLTA P/ VARREDURA DO TECLADO
                                   ; NÃO
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO

   MOVLW   B'00000001'
   XORWF   FLAG,F                  ; INVERTE FLAG
                                   ; ALTERA A MEMÓRIA UTILIZADA

   CALL    ATUALIZA_TELA_LINHA_2   ; CHAMA ROTINA P/ ATUALIZAR LCD

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************ TRATAMENTO DO BOTÃO 1 ***************************

TRATA_BOTAO_1
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO

   DECFSZ  TEMPO_TURBO,F           ; FIM DO TEMPO DE TURBO ?
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM
   MOVLW   TURBO_TECLA
   MOVWF   TEMPO_TURBO             ; RECARREGA TEMPORIZADOR DO TURBO
                                   ; DAS TECLAS

   BTFSC   TIPO_MEMORIA            ; ESTÁ UTILIZANDO MEMÓRIA DE DADOS ?
   GOTO    INC_MEM_PROG            ; NÃO - ENTÃO PULA P/ INC_MEM_PROG
                                   ; SIM
INC_MEM_DADOS
   INCF    VALOR_DADOS,F           ; INCREMENTA VALOR_DADOS

   CALL    ATUALIZA_TELA_LINHA_2   ; CHAMA ROTINA P/ ATUALIZAR LCD

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

INC_MEM_PROG
   INCF    VALOR_PROG_LOW,F        ; INCREMENTA VALOR_PROG_LOW
   BTFSC   STATUS,Z                ; HOUVE ESTOURO ?
   INCF    VALOR_PROG_HIGH,F       ; SIM - INCREMENTA VALOR_PROG_HIGH
                                   ; NÃO

   MOVLW   B'00111111'
   ANDWF   VALOR_PROG_HIGH,F       ; LIMITA CONTADOR DA MEMÓRIA DE
                                   ; PROGRAMA EM 14 BITS

   CALL    ATUALIZA_TELA_LINHA_2   ; CHAMA ROTINA P/ ATUALIZAR LCD

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************ TRATAMENTO DO BOTÃO 2 ***************************
```

```
TRATA_BOTAO_2
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO

   DECFSZ  TEMPO_TURBO,F           ; FIM DO TEMPO DE TURBO ?
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM
   MOVLW   TURBO_TECLA
   MOVWF   TEMPO_TURBO             ; RECARREGA TEMPORIZADOR DO TURBO
                                   ; DAS TECLAS

   BTFSC   TIPO_MEMORIA            ; ESTÁ UTILIZANDO MEMÓRIA DE DADOS ?
   GOTO    DEC_MEM_PROG            ; NÃO - ENTÃO PULA P/ DEC_MEM_PROG
                                   ; SIM
DEC_MEM_DADOS
   DECF    VALOR_DADOS,F           ; DECREMENTA VALOR_DADOS

   CALL    ATUALIZA_TELA_LINHA_2   ; CHAMA ROTINA P/ ATUALIZAR LCD

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

DEC_MEM_PROG
   MOVLW   .1
   SUBWF   VALOR_PROG_LOW,F        ; DECREMENTA VALOR_PROG_LOW
   BTFSS   STATUS,C                ; HOUVE ESTOURO ?
   DECF    VALOR_PROG_HIGH,F       ; SIM - DECREMENTA VALOR_PROG_HIGH
                                   ; NÃO

   MOVLW   B'00111111'
   ANDWF   VALOR_PROG_HIGH,F       ; LIMITA CONTADOR DA MEMÓRIA DE
                                   ; PROGRAMA EM 14 BITS

   CALL    ATUALIZA_TELA_LINHA_2   ; CHAMA ROTINA P/ ATUALIZAR LCD

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************* TRATAMENTO DO BOTÃO 3 ***************************

TRATA_BOTAO_3
   MOVF    FILTRO_BOTOES,F
   BTFSC   STATUS,Z                ; FILTRO JÁ IGUAL A ZERO ?
                                   ; (FUNÇÃO JA FOI EXECUTADA?)
   GOTO    VARRE                   ; SIM - VOLTA P/ VARREDURA DO TECLADO
                                   ; NÃO
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO

; ********TRECHO DO PROGRAMA PARA GRAVAR DADOS DA RAM NA MEMÓRIA *************

GRAVA_MEMORIA_PROGRAMA
   MOVLW   END_MEM_PROG_H
   MOVWF   ENDERECO_HIGH
   MOVLW   END_MEM_PROG_L
   MOVWF   ENDERECO_LOW            ; CARREGA ENDERECO ONDE O DADO SERÁ SALVO

   MOVF    VALOR_PROG_HIGH,W
   MOVWF   DADO_HIGH_00
   MOVF    VALOR_PROG_LOW,W
   MOVWF   DADO_LOW_00             ; CARREGA DADO A SER SALVO
                                   ; EM DADO_HIGH_00 E DADO_LOW_00

   CALL    FLASH_PROGRAM_WRITE     ; CHAMA ROTINA DE GRAVAÇÃO

GRAVA_MEMORIA_DADOS
   MOVLW   END_MEM_DADO
   MOVWF   ENDERECO_LOW            ; CARREGA ENDERECO ONDE O DADO SERÁ SALVO

   MOVF    VALOR_DADOS,W
   MOVWF   DADO_LOW_00             ; CARREGA DADO A SER SALVO EM DADO_LOW_00
```

```
   CALL    DATA_EEPROM_WRITE       ; CHAMA ROTINA DE GRAVAÇÃO

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              FIM DO PROGRAMA                                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                             ; FIM DO PROGRAMA
```

Dicas e Comentários

Note que a rotina que grava o dado na memória de programa, na realidade, grava o valor de 4 words. Porém, o software da experiência altera o valor de apenas uma dessas words. O problema é que a gravação de informações na memória de programa, no caso do PIC16F877A, sempre deve ser realizada de 4 em 4 words. Desta forma, não é possível gravar apenas um valor ou uma word. Sempre a gravação será de 4 words no mínimo. Portanto, sempre que se desejar gravar menos do que 4 words mantendo as outras intactas. Deve-se inicialmente realizar uma leitura de todas as 4 words, alterar as que se desejarem e regravas todas de uma única vez.

Exercícios Propostos

1. Crie um novo sistema onde do lado esquerdo você continua informando o dado para a E^2PROM, no centro você escolhe a posição, de 0 a 255 e do lado direito é informada o tipo de operação: “E” para escrita e “L” para leitura. O botão da coluna 1 continua alterando entre os parâmetros a serem ajustados e os botões das colunas 2 e 3 alteram o parâmetro atual. O botão da coluna 4 efetua a operação de escrita ou leitura dependendo da seleção ajustada no LCD.
2. Repita o exercício anterior para a memória de programa FLASH;
3. Crie um programa que copie internamente todo o código de programa para outra posição. Por exemplo, copie todas a página 0 da memória de programa para a página 1. Depois, utilize o gravador e o Mplab para ler a memória de programa e verificar se a operação foi executada com sucesso.

Capítulo 18 - Experiência 16 – Master I^2C

Objetivo

O objetivo desta experiência é mostrar ao aluno como acessar a memória de dados EEPROM externa (24LC256) utilizando os recursos de hardware do PIC para implementar o protocolo de comunicação I^2C.

Descrição

Conforme comentado na descrição do hardware, está memória está mapeada no endereço 7h (111b) da rede de comunicação I^2C a fim de evitar conflitos com o relógio de tempo real. Além disso, como a memória utilizada no MCMASTER é de 256Kbits, ou seja, 32Kbytes, são necessários 15 bits, ou seja, 2 bytes, para o correto endereçamento.

Levando-se isso em consideração, para a escrita de dados na memória o microcontrolador deverá enviar a seguinte seqüência de informações:

- Envia o byte de controle que deve incorporar o endereço de hardware da memória na rede I2C além do bit de R/W, que neste caso deve ser enviado em 0 a fim de sinalizar uma operação de escrita. Assim, o byte de controle completo considerando o mapeamento adotado no MCMASTER é 10101110b;
- Envia um start bit;
- Recebe o bit de acknowledge (ACK);
- Envia a parte mais significativa do endereço onde o dado será gravado;
- Recebe o bit de acknowledge (ACK);
- Envia a parte menos significativa do endereço onde o dado será gravado;
- Recebe o bit de acknowledge (ACK);
- Envia o dado a ser gravado;
- Recebe o bit de acknowledge (ACK);
- Envia um stop bit.

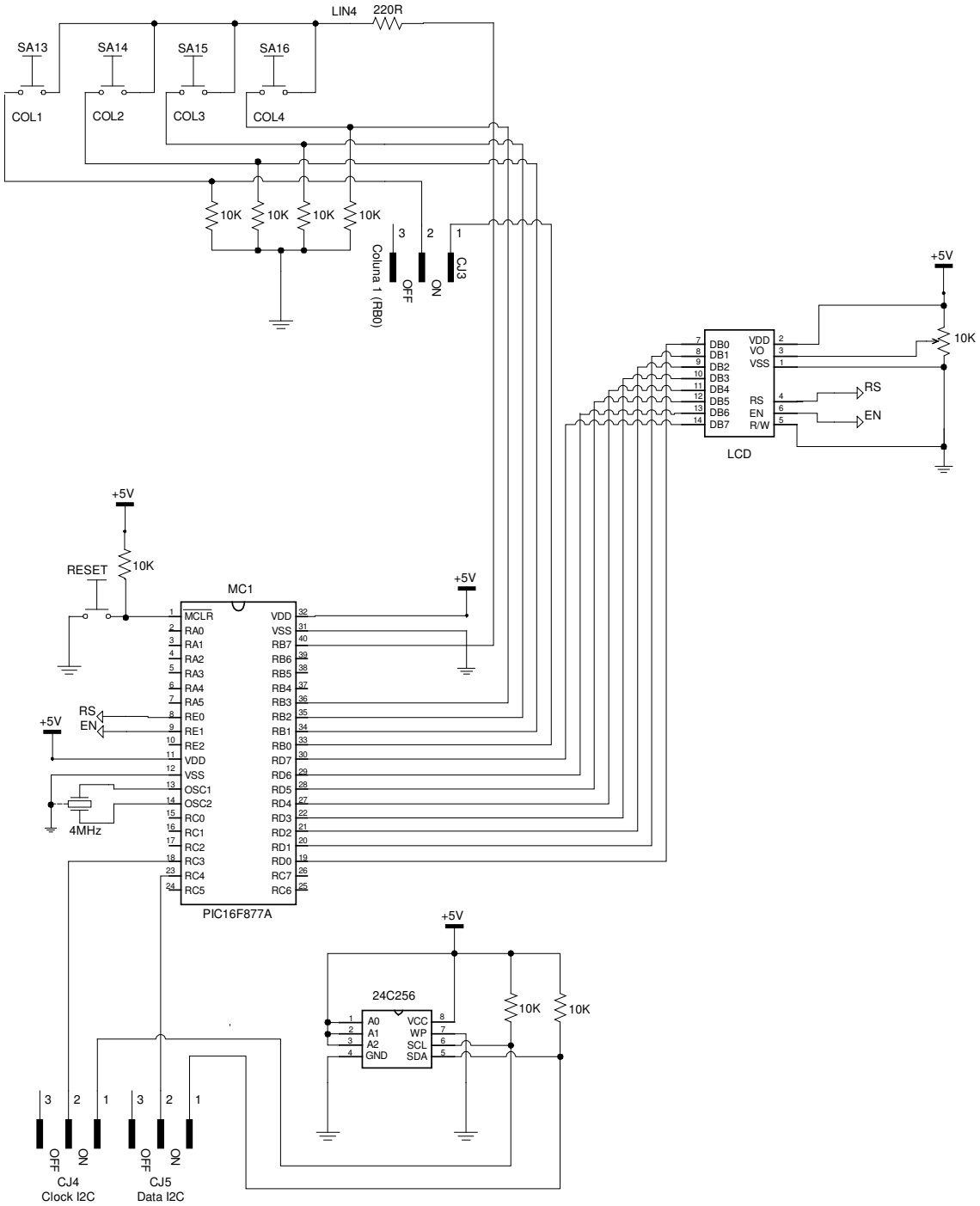
Já para uma operação de leitura a seqüência deverá ser:

- Envia um start bit;
- Envia o byte de controle que deve incorporar o endereço de hardware da memória na rede I2C além do bit de R/W, que neste caso deve ser enviado em 0 a fim de sinalizar uma operação de escrita. Note que inicialmente o endereço que se deseja ler deve ser escrito na memória. Assim, o byte de controle completo considerando o mapeamento adotado no MCMASTER é 10101110b;
- Recebe o bit de acknowledge (ACK);
- Envia a parte mais significativa do endereço de o dado será lido;
- Recebe o bit de acknowledge (ACK);
- Envia a parte menos significativa do endereço de onde o dado será lido;

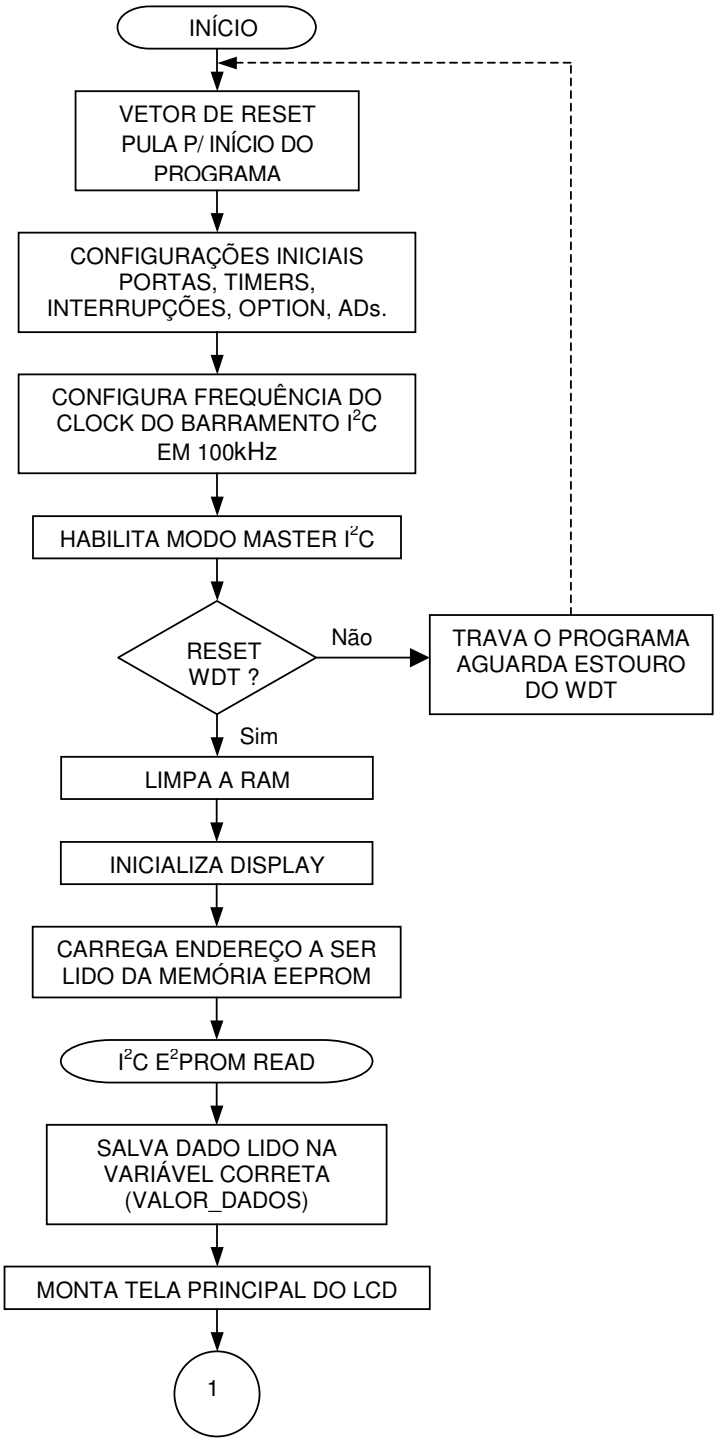
- Recebe o bit de acknowledge (ACK);
- Envia outro start bit;
- Envia novamente o byte de controle, porém agora alterando o bit R/W para sinalizar a operação de leitura. Assim, o byte de controle completo fica 10101111b;
- Recebe o bit de acknowledge (ACK);
- Recebe o byte lido da memória;
- Envia um bit em 1 sinalizando que deseja encerrar leitura (sinal de NACK);
- Envia um stop bit.

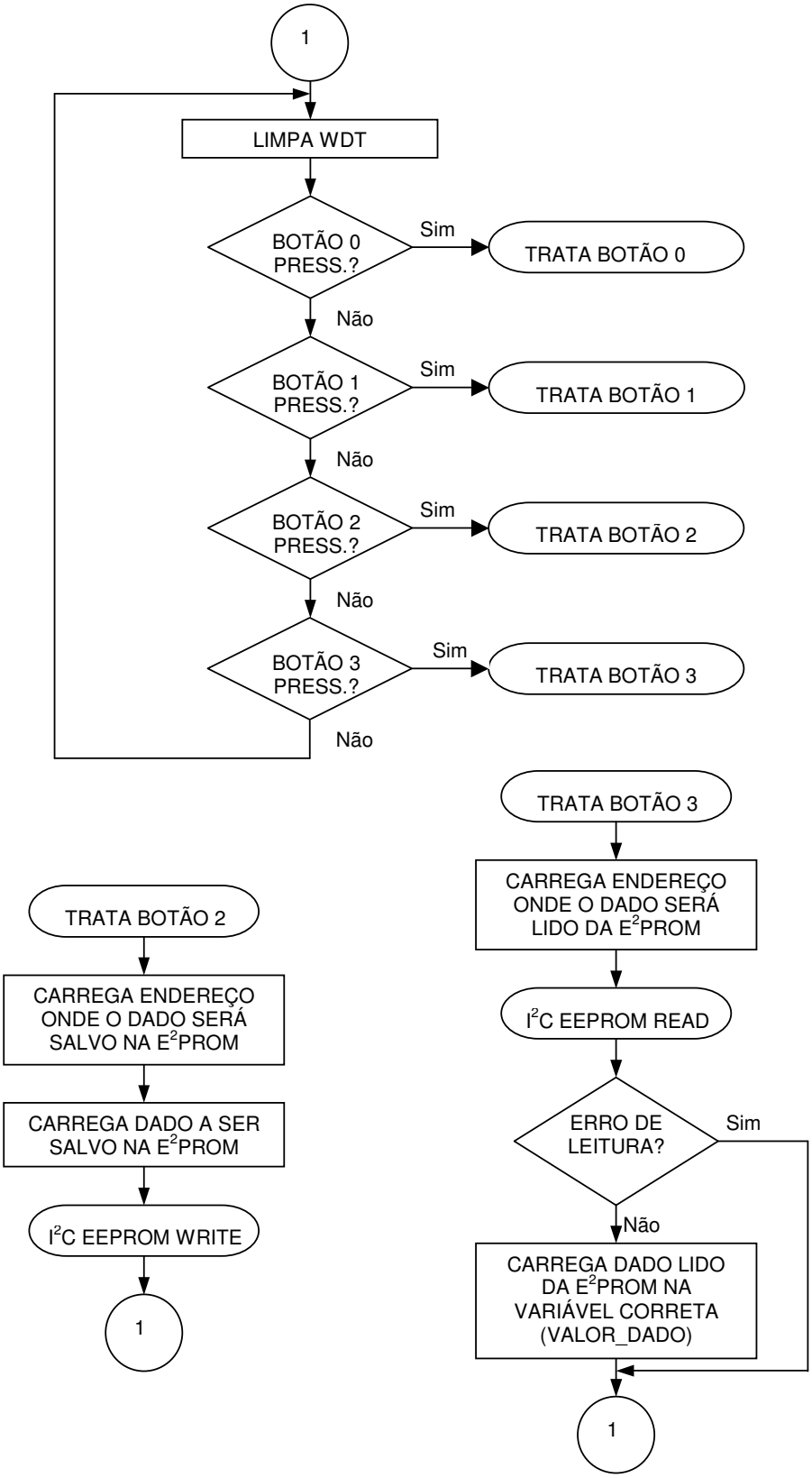
O exemplo da experiência foi elaborado utilizando os procedimentos descritos acima. São utilizados os botões da linha 4 para manipular um valor de 8 bits mostrado no display LCD. Este valor pode ser salvo e lido na memória EEPROM. Os botões das colunas 1 e 2 são utilizados para incrementar e decrementar o valor mostrado no display. O botão da coluna 3 salva o valor do display na memória serial enquanto o botão da coluna 4 é utilizado para ler o valor salvo na memória serial.

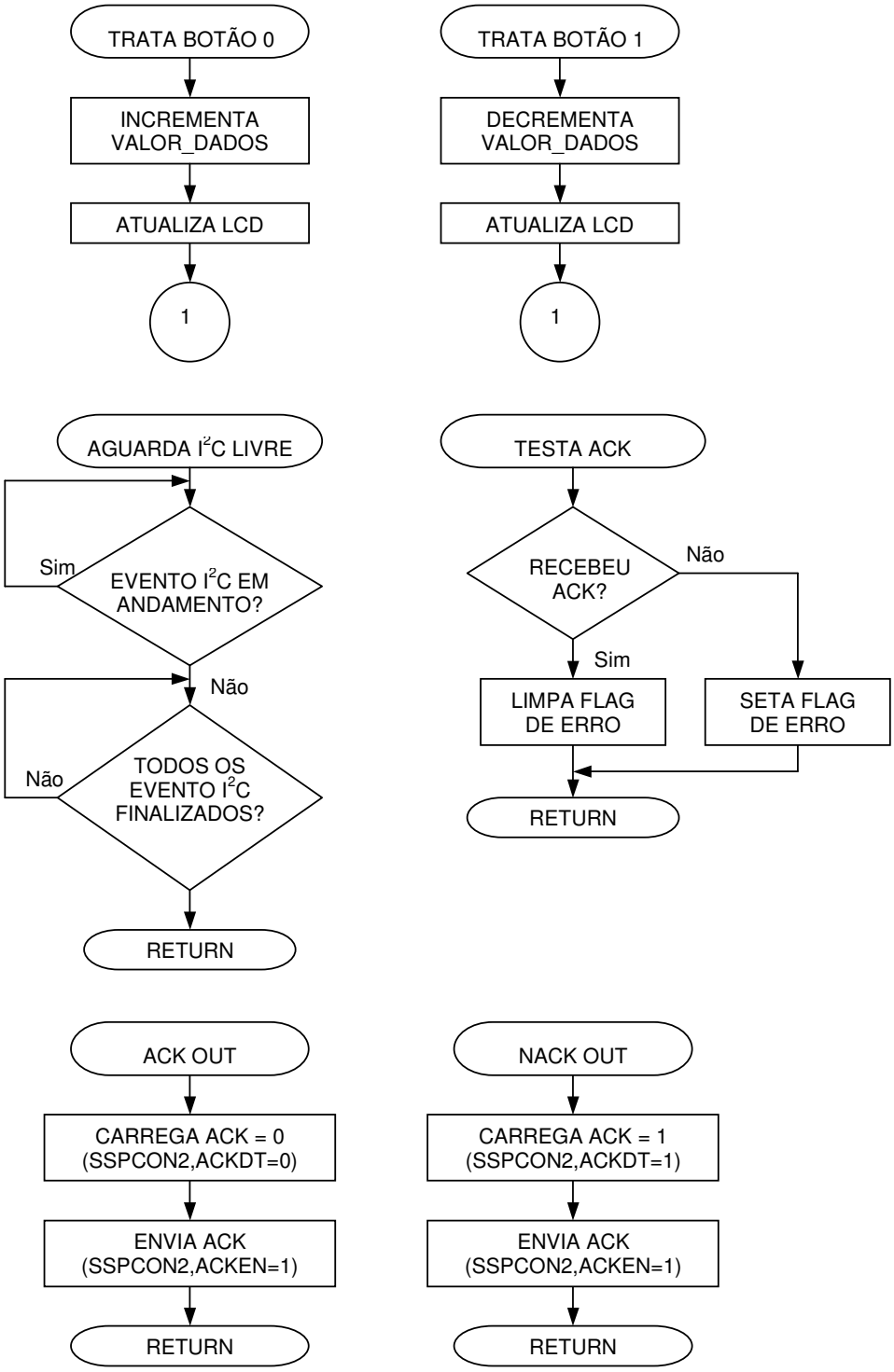
Esquema Elétrico

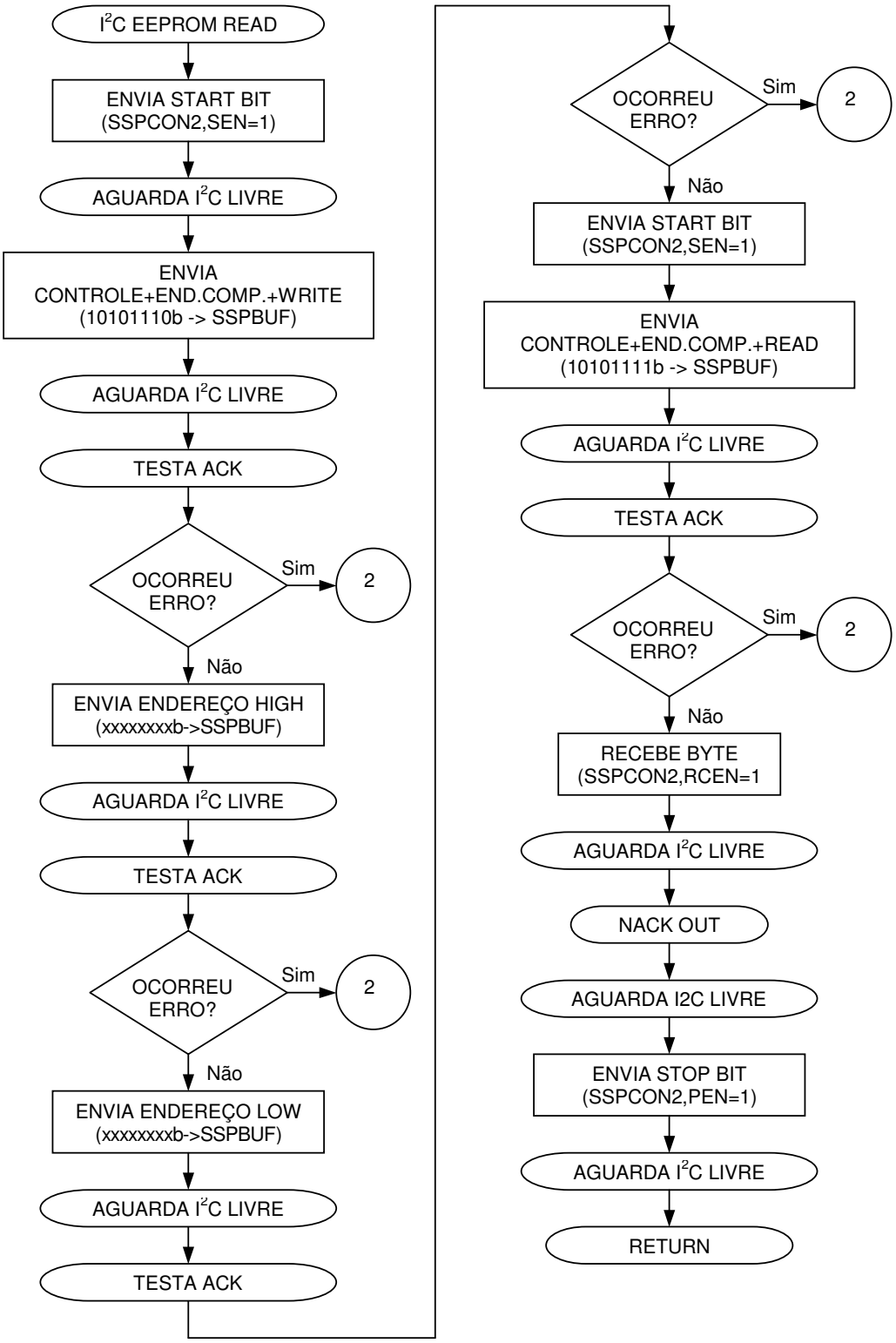


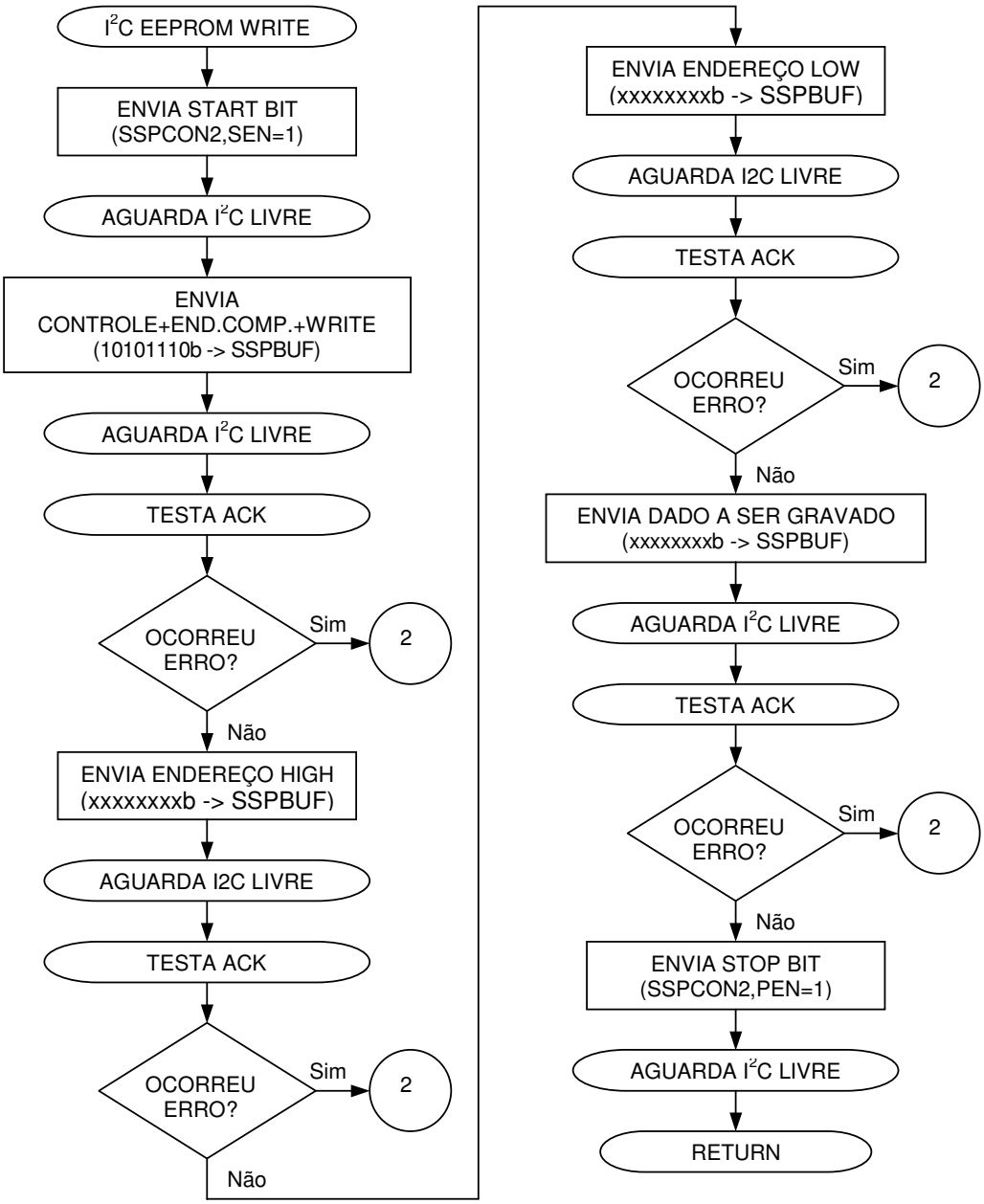
Fluxograma

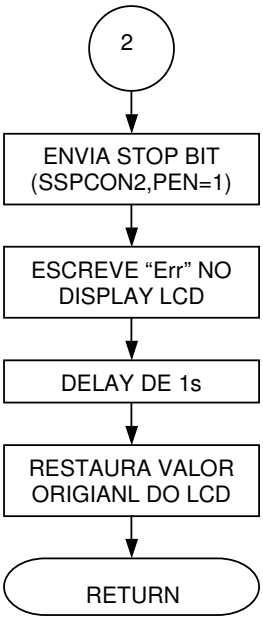

INÍCIO
VETOR DE RESET PULA P/ INÍCIO DO PROGRAMA
CONFIGURAÇÕES INICIAIS PORTAS, TIMERS, INTERRUPÇÕES, OPTION, ADs.
CONFIGURA FREQUÊNCIA DO CLOCK DO BARRAMENTO I2C EM 100kHz
HABILITA MODO MASTER I2C
RESET WDT ?
Não
TRAVA O PROGRAMA AGUARDA ESTOURO DO WDT
Sim
LIMPA A RAM
INICIALIZA DISPLAY
CARREGA ENDEREÇO A SER LIDO DA MEMÓRIA EEPROM
I2C E2PROM READ
SALVA DADO LIDO NA VARIÁVEL CORRETA (VALOR_DADOS)
MONTA TELA PRINCIPAL DO LCD
1


1
LIMPA WDT
BOTÃO 0 PRESS.?
Sim
TRATA BOTÃO 0
Não
BOTÃO 1 PRESS.?
Sim
TRATA BOTÃO 1
Não
BOTÃO 2 PRESS.?
Sim
TRATA BOTÃO 2
Não
BOTÃO 3 PRESS.?
Sim
TRATA BOTÃO 3
Não
TRATA BOTÃO 3
CARREGA ENDEREÇO ONDE O DADO SERÁ LIDO DA E2PROM
I2C EEPROM READ
ERRO DE LEITURA?
Sim
Não
CARREGA DADO LIDO DA E2PROM NA VARIÁVEL CORRETA (VALOR_DADO)
1
TRATA BOTÃO 2
CARREGA ENDEREÇO ONDE O DADO SERÁ SALVO NA E2PROM
CARREGA DADO A SER SALVO NA E2PROM
I2C EEPROM WRITE
1


TRATA BOTÃO 0
INCREMENTA VALOR_DADOS
ATUALIZA LCD
1
TRATA BOTÃO 1
DECREMENTA VALOR_DADOS
ATUALIZA LCD
1
AGUARDA I²C LIVRE
EVENTO I²C EM ANDAMENTO?
Sim
Não
TODOS OS EVENTO I²C FINALIZADOS?
Não
RETURN
TESTA ACK
RECEBEU ACK?
Não
Sim
LIMPA FLAG DE ERRO
SETA FLAG DE ERRO
RETURN
ACK OUT
CARREGA ACK = 0 (SSPCON2,ACKDT=0)
ENVIA ACK (SSPCON2,ACKEN=1)
RETURN
NACK OUT
CARREGA ACK = 1 (SSPCON2,ACKDT=1)
ENVIA ACK (SSPCON2,ACKEN=1)
RETURN


I²C EEPROM READ
ENVIA START BIT (SSPCON2,SEN=1)
AGUARDA I²C LIVRE
ENVIA CONTROLE+END.COMP.+WRITE (10101110b -> SSPBUF)
AGUARDA I²C LIVRE
TESTA ACK
OCORREU ERRO?
Sim
2
Não
ENVIA ENDEREÇO HIGH (xxxxxxxxb->SSPBUF)
AGUARDA I²C LIVRE
TESTA ACK
OCORREU ERRO?
Sim
2
Não
ENVIA ENDEREÇO LOW (xxxxxxxxb->SSPBUF)
AGUARDA I²C LIVRE
TESTA ACK
OCORREU ERRO?
Sim
2
Não
ENVIA START BIT (SSPCON2,SEN=1)
ENVIA CONTROLE+END.COMP.+READ (10101111b -> SSPBUF)
AGUARDA I²C LIVRE
TESTA ACK
OCORREU ERRO?
Sim
2
Não
RECEBE BYTE (SSPCON2,RCEN=1
AGUARDA I²C LIVRE
NACK OUT
AGUARDA I2C LIVRE
ENVIA STOP BIT (SSPCON2,PEN=1)
AGUARDA I²C LIVRE
RETURN


I²C EEPROM WRITE
ENVIA START BIT (SSPCON2,SEN=1)
AGUARDA I²C LIVRE
ENVIA CONTROLE+END.COMP.+WRITE (10101110b -> SSPBUF)
AGUARDA I²C LIVRE
TESTA ACK
OCORREU ERRO?
Sim
2
Não
ENVIA ENDEREÇO HIGH (xxxxxxxxb -> SSPBUF)
AGUARDA I2C LIVRE
TESTA ACK
OCORREU ERRO?
Sim
2
Não
ENVIA ENDEREÇO LOW (xxxxxxxxb -> SSPBUF)
AGUARDA I2C LIVRE
TESTA ACK
OCORREU ERRO?
Sim
2
Não
ENVIA DADO A SER GRAVADO (xxxxxxxxb -> SSPBUF)
AGUARDA I²C LIVRE
TESTA ACK
OCORREU ERRO?
Sim
2
Não
ENVIA STOP BIT (SSPCON2,PEN=1)
AGUARDA I²C LIVRE
RETURN


2
ENVIA STOP BIT
(SSPCON2,PEN=1)
ESCREVE “Err” NO
DISPLAY LCD
DELAY DE 1s
RESTAURA VALOR
ORIGIANL DO LCD
RETURN

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *                      EXPERIÊNCIA 16 - MASTER I2C                        *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            DESCRIÇÃO GERAL                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTE EXEMPLO FOI ELABORADO PARA EXPLICAR O FUNCIONAMENTO DA LEITURA/ESCRITA
; NA MEMÓRIA E2PROM SERIAL EXTERNA, UTILIZANDO O MASTER I2C.
; OS BOTÕES DAS COLUNAS 1 E 2 SÃO PARA INCREMENTAR E DECREMENTAR O VALOR
; MOSTRADO NO DISPLAY. O BOTÃO DA COLUNA 3 SALVA O VALOR DO DISPLAY NA
; MEMÓRIA SERIAL ENQUANTO O BOTÃO DA COLUNA 4 É UTILIZADOS PARA LER O VALOR
; SALVO NA MEMÓRIA SERIAL.
; APENAS OS BOTÕES DA LINHA 4 ESTÃO ATIVOS
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        CONFIGURAÇÕES PARA GRAVAÇÃO                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_ON & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          DEFINIÇÃO DAS VARIÁVEIS                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE BLOCO DE VARIÁVEIS ESTÁ LOCALIZADO LOGO NO INÍCIO DO BANCO 0

   CBLOCK  0X20                    ; POSIÇÃO INICIAL DA RAM

           FILTRO_BOTOES           ; FILTRO PARA RUIDOS
           TEMPO_TURBO             ; TEMPORIZADOR P/ TURBO DAS TECLAS

           TEMPO1
           TEMPO0                  ; CONTADORES P/ DELAY

           FLAG                    ; FLAG DE USO GERAL

           AUX                     ; REGISTRADOR AUXILIAR DE USO GERAL

           ENDERECO_H              ; REGISTRADORES DE ENDEREÇO PARA
           ENDERECO_L              ; ACESSO À MEMÓRIA EEPROM SERIAL EXTERNA
                                   ; MAPEADO NO BANCO 0 DA RAM

           BUFFER                  ; REGISTRADOR PARA LEITURA/GRAVAÇÃO NA EEPROM
SERIAL
                                   ; EXTERNA

           VALOR_DADOS             ; REGISTRADOR DE DADO PARA EEPROM SERIAL EXTERNA
                                   ; MAPEADO NO BANCO 0 DA RAM
   ENDC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>         ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       DEFINIÇÃO DOS BANCOS DE RAM                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          CONSTANTES INTERNAS                            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE CONSTANTES FACILITA A PROGRAMAÇÃO E A MANUTENÇÃO.

FILTRO_TECLA       EQU     .200            ; FILTRO P/ EVITAR RUIDOS DOS BOTÕES

TURBO_TECLA        EQU     .60             ; TEMPORIZADOR P/ TURBO DAS TECLAS

END_EEPROM_H       EQU     0X00            ; ENDEREÇO P/ LEITURA E GRAVAÇÃO
END_EEPROM_L       EQU     0X00            ; NA MEMÓRIA EEPROM SERIAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   DECLARAÇÃO DOS FLAGs DE SOFTWARE                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE FLAGs AJUDA NA PROGRAMAÇÃO E ECONOMIZA MEMÓRIA RAM.

#DEFINE    F_ERRO          FLAG,0          ; 1 --> ERRO NA LEITURA DA EEPROM SERIAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              ENTRADAS                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS ENTRADAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    BOTAO_0         PORTB,0         ; ESTADO DO BOTÃO 0
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_1         PORTB,1         ; ESTADO DO BOTÃO 1
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_2         PORTB,2         ; ESTADO DO BOTÃO 2
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_3         PORTB,3         ; ESTADO DO BOTÃO 3
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               SAÍDAS                                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS SAÍDAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    DISPLAY         PORTD   ; BARRAMENTO DE DADOS DO DISPLAY

#DEFINE    RS              PORTE,0 ; INDICA P/ O DISPLAY UM DADO OU COMANDO
                                   ; 1 -> DADO
                                   ; 0 -> COMANDO

#DEFINE    ENABLE          PORTE,1 ; SINAL DE ENABLE P/ DISPLAY
                                   ; ATIVO NA BORDA DE DESCIDA

#DEFINE    TEC_MATRICIAL   PORTB   ; PORT DO MICROCONTROLADOR LIGADO AO
                                   ; TECLADO MATRICIAL
                                   ; <RB4:RB7> LINHAS
                                   ; 1->ATIVADAS 0->DESATIVADAS
                                   ; <RB0:RB3> COLUNAS
                                   ; 1->TECLAS PRESSIONADAS 0->TECLAS LIBERADAS

#DEFINE    LINHA_4         PORTB,7 ; PINO P/ ATIVAR LINHA 4 (TECLADO MATRICIAL)
                                   ; 0 -> LINHA 4 ATIVADA
                                   ; 1 -> LINHA 4 DESATIVADA
```

```
#DEFINE    SCL             PORTC,3 ; VIA DE CLOCK DA EEPROM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            ENTRADAS/SAÍDAS                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

#DEFINE    SDA             PORTC,4 ; VIA DE DADOS BIDIRECIONAL DA EEPROM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  VETOR DE RESET DO MICROCONTROLADOR                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO INICIAL PARA EXECUÇÃO DO PROGRAMA

   ORG     0X0000                  ; ENDEREÇO DO VETOR DE RESET
   GOTO    CONFIG                  ; PULA PARA CONFIG DEVIDO A REGIÃO
                                   ; DESTINADA AS ROTINAS SEGUINTES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  ROTINA DE DELAY (DE 1MS ATÉ 256MS)                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA É UMA ROTINA DE DELAY VARIÁVEL, COM DURAÇÃO DE 1MS X O VALOR PASSADO
;  EM WORK (W).

DELAY_MS
   MOVWF   TEMPO1                  ; CARREGA TEMPO1 (UNIDADES DE MS)
   MOVLW   .250
   MOVWF   TEMPO0                  ; CARREGA TEMPO0 (P/ CONTAR 1MS)

   CLRWDT                          ; LIMPA WDT (PERDE TEMPO)
   DECFSZ  TEMPO0,F                ; FIM DE TEMPO0 ?
   GOTO    $-2                     ; NÃO - VOLTA 2 INSTRUÇÕES
                                   ; SIM - PASSOU-SE 1MS
   DECFSZ  TEMPO1,F                ; FIM DE TEMPO1 ?
   GOTO    $-6                     ; NÃO - VOLTA 6 INSTRUÇÕES
                                   ; SIM
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DE UM CARACTER NO DISPLAY                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ENVIA UM CARACTER PARA O MÓDULO DE LCD. O CARACTER A SER
;  ESCRITO DEVE SER COLOCADO EM WORK (W) ANTES DE CHAMAR A ROTINA.

ESCREVE
   MOVWF   DISPLAY                 ; ATUALIZA DISPLAY (PORTD)
   NOP                             ; PERDE 1US PARA ESTABILIZAÇÃO
   BSF     ENABLE                  ; ENVIA UM PULSO DE ENABLE AO DISPLAY
   GOTO    $+1                     ; .
   BCF     ENABLE                  ; .

   MOVLW   .1
   CALL    DELAY_MS                ; DELAY DE 1MS
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA LINHA 1 DO LCD                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A LINHA 1 DA TELA PRINCIPAL DO LCD, COM A FRASE:
;  LINHA 1 - "   MASTER I2C   "

ATUALIZA_TELA_LINHA_1
   CLRF    TEC_MATRICIAL           ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0X83                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 0 / COLUNA 3
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

                                   ; COMANDOS PARA ESCREVER AS
                                   ; LETRAS DE "MASTER I2C"
   MOVLW   'M'
```

```
   CALL    ESCREVE
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   'S'
   CALL    ESCREVE
   MOVLW   'T'
   CALL    ESCREVE
   MOVLW   'E'
   CALL    ESCREVE
   MOVLW   'R'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   'I'
   CALL    ESCREVE
   MOVLW   '2'
   CALL    ESCREVE
   MOVLW   'C'
   CALL    ESCREVE

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA LINHA 2 DO LCD                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A LINHA 2 DA TELA PRINCIPAL DO LCD.
;  A ROTINA LEVA EM CONTA A VARIÁVEL VALOR_DADOS PARA FORMAR A LINHA 2.

ATUALIZA_TELA_LINHA_2
   CLRF    TEC_MATRICIAL           ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDO
   MOVLW   0XC6                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 6

   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

   SWAPF   VALOR_DADOS,W           ; INVERTE NIBLE DO VALOR_DADOS
   ANDLW   B'00001111'             ; MASCARA BITS MAIS SIGNIFICATIVOS
   MOVWF   AUX                     ; SALVA EM AUXILIAR

   MOVLW   0X0A
   SUBWF   AUX,W                   ; AUX - 10d (ATUALIZA FLAG DE CARRY)
   MOVLW   0X30                    ; CARREGA WORK COM 30h
   BTFSC   STATUS,C                ; RESULTADO É POSITIVO? (É UMA LETRA?)
   MOVLW   0X37                    ; SIM - CARREGA WORK COM 37h
                                   ; NÃO - WORK FICA COM 30h (NÚMERO)
   ADDWF   AUX,W                   ; SOMA O WORK AO AUXILIAR
                                   ; (CONVERSÃO ASCII)
   CALL    ESCREVE                 ; ENVIA CARACTER AO DISPLAY LCD

   MOVF    VALOR_DADOS,W           ; CARREGA WORK COM VALOR_DADOS
   ANDLW   B'00001111'             ; MASCARA BITS MAIS SIGNIFICATIVOS
   MOVWF   AUX                     ; SALVA EM AUXILIAR

   MOVLW   0X0A
   SUBWF   AUX,W                   ; AUX - 10d (ATUALIZA FLAG DE CARRY)
   MOVLW   0X30                    ; CARREGA WORK COM 30h
   BTFSC   STATUS,C                ; RESULTADO É POSITIVO? (É UMA LETRA?)
   MOVLW   0X37                    ; SIM - CARREGA WORK COM 37h
                                   ; NÃO - WORK FICA COM 30h (NÚMERO)
   ADDWF   AUX,W                   ; SOMA O WORK AO AUXILIAR
                                   ; (CONVERSÃO ASCII)
   CALL    ESCREVE                 ; ENVIA CARACTER AO DISPLAY LCD

   MOVLW   'h'
   CALL    ESCREVE                 ; ESCREVE "h" NO DISPLAY

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS
```

```
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE CHECAGEM DE EVENTOS I2C LIBERADOS                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA AGUARDA ATÉ QUE TODOS OS EVENTOS DA I2C ESTEJAM LIBERADOS.

AGUARDA_I2C_LIVRE
   BANK1                           ; ALTERA P/ BANK1
   BTFSC   SSPSTAT,R_W             ; ESTÁ OCORRENDO ALGUM EVENTO I2C?
   GOTO    $-1                     ; SIM, ESPERA TERMINAR
   MOVF    SSPCON2,W
   ANDLW   B'00011111'             ; MASCARA SSPCON2 (ATUALIZA FLAG ZERO)
   BTFSS   STATUS,Z                ; BITS DE EVENTOS LIBERADOS?
   GOTO    $-3                     ; NÃO - AGUARDA
   BANK0                           ; SIM - VOLTA P/ BANK0
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               ACK OUT                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA ENVIA UM ACK OUT PARA O BARRAMENTO I2C.

ACK_OUT
   BANK1                           ; ALTERA P/ BANK1
   BCF     SSPCON2,ACKDT           ; CARREGA ACK
   BSF     SSPCON2,ACKEN           ; TRANSMITE
   BANK0                           ; VOLTA P/ BANK0
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               NACK OUT                                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA ENVIA UM NACK OUT PARA O BARRAMENTO I2C.

NACK_OUT
   BANK1                           ; ALTERA P/ BANK1
   BSF     SSPCON2,ACKDT           ; CARREGA NACK
   BSF     SSPCON2,ACKEN           ; TRANSMITE
   BANK0                           ; VOLTA P/ BANK0
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                 ROTINA PARA TESTAR SE O ACK FOI RECEBIDO                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA TESTA O BIT DE ACK RECEBIDO NO REGISTRADOR SSPCON2. PARA
; FACILITAR O RESTANTE DO SOFTWARE, A ROTINA COPIA ESTE FLAG NO FLAG F_ERRO
; PRESENTE NO BANCO 0 DA RAM, POIS O REGISTRADOR SSPCON2 ENCONTRA-SE NO BANK1.

TESTA_ACK
   BANK1                           ; ALTERA P/ BANK1
   BTFSC   SSPCON2,ACKSTAT         ; RECEBEU ACK ?
   GOTO    RECEBEU_NACK            ; NÃO - SINALIZA ERRO
                                   ; SIM
   BANK0                           ; VOLTA P/ BANK0
   BCF     F_ERRO                  ; LIMPA FLAG DE ERRO
   RETURN                          ; RETORNA

RECEBEU_NACK
   BANK0                           ; VOLTA P/ BANK0
   BSF     F_ERRO                  ; SETA FLAG P/ INDICAR ERRO
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  LEITURA DA EEPROM SERIAL EXTERNA                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA LÊ A MEMÓRIA SERIAL EXTERNA. O ENDEREÇO DEVE SER PASSADO PELA
; VARIÁVEL ENDERECO. O VALOR LIDO É RETORNADO EM BUFFER.
; CASO ALGUM ERRO DE LEITURA OCORRA, A ROTINA DESVIA P/ I2C_ERRO.

I2C_EEPROM_READ
   BANK1                           ; ALTERA P/ BANK1
```

```
   BSF     SSPCON2,SEN             ; INICIA START BIT
   BANK0                           ; VOLTA P/ BANK0
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO

   MOVLW   B'10101110'             ; CARREGA BYTE DE CONTROLE
   MOVWF   SSPBUF                  ; TRANSMITE CONTROLE
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO
   CALL    TESTA_ACK               ; CHAMA ROTINA P/ TESTAR ACK
   BTFSC   F_ERRO                  ; OCORREU ERRO DE ACK ?
   GOTO    I2C_ERRO                ; SIM - PULA P/ I2C_ERRO
                                   ; NÃO
   MOVF    ENDERECO_H,W
   MOVWF   SSPBUF                  ; TRANSMITE ENDEREÇO (PARTE HIGH)
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO
   CALL    TESTA_ACK               ; CHAMA ROTINA P/ TESTAR ACK
   BTFSC   F_ERRO                  ; OCORREU ERRO DE ACK ?
   GOTO    I2C_ERRO                ; SIM - PULA P/ I2C_ERRO
                                   ; NÃO

   MOVF    ENDERECO_L,W
   MOVWF   SSPBUF                  ; TRANSMITE ENDEREÇO (PARTE LOW)
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO
   CALL    TESTA_ACK               ; CHAMA ROTINA P/ TESTAR ACK
   BTFSC   F_ERRO                  ; OCORREU ERRO DE ACK ?
   GOTO    I2C_ERRO                ; SIM - PULA P/ I2C_ERRO
                                   ; NÃO

   BANK1                           ; ALTERA P/ BANK1
   BSF     SSPCON2,RSEN            ; REINICIA START BIT
   BANK0                           ; VOLTA P/ BANK0
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO

   MOVLW   B'10101111'             ; CARREGA BYTE DE CONTROLE
   MOVWF   SSPBUF                  ; TRANSMITE CONTROLE
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO
   CALL    TESTA_ACK               ; CHAMA ROTINA P/ TESTAR ACK
   BTFSC   F_ERRO                  ; OCORREU ERRO DE ACK ?
   GOTO    I2C_ERRO                ; SIM - PULA P/ I2C_ERRO
                                   ; NÃO

   BANK1                           ; ALTERA P/ BANK1
   BSF     SSPCON2,RCEN            ; INICIA LEITURA DO BYTE
   BANK0                           ; VOLTA P/ BANK0
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO
   MOVF    SSPBUF,W
   MOVWF   BUFFER                  ; SALVA DADO EM BUFFER
   CALL    NACK_OUT                ; ENVIA NACK --> FIM
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO

   BANK1                           ; ALTERA P/ BANK1
   BSF     SSPCON2,PEN             ; INICIA STOP BIT
   BANK0                           ; VOLTA P/ BANK0
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO

   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   ESCRITA NA EEPROM SERIAL EXTERNA                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA GRAVA UM DADO NA MEMÓRIA SERIAL EXTERNA. O ENDEREÇO DEVE SER
; PASSADO PELA VARIÁVEL ENDERECO. O VALOR A SER GRAVADO DEVE SER PASSADO
; EM BUFFER.
; CASO ALGUM ERRO DE GRAVAÇÃO OCORRA, A ROTINA DESVIA P/ I2C_ERRO.

I2C_EEPROM_WRITE
   BANK1                           ; ALTERA P/ BANK1
   BSF     SSPCON2,SEN             ; INICIA START BIT
   BANK0                           ; VOLTA P/ BANK0
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO

   MOVLW   B'10101110'             ; CARREGA BYTE DE CONTROLE
   MOVWF   SSPBUF                  ; TRANSMITE CONTROLE + END_HIGH
```

```
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO
   CALL    TESTA_ACK               ; CHAMA ROTINA P/ TESTAR ACK
   BTFSC   F_ERRO                  ; OCORREU ERRO DE ACK ?
   GOTO    I2C_ERRO                ; SIM - PULA P/ I2C_ERRO
                                   ; NÃO
   MOVF    ENDERECO_H,W
   MOVWF   SSPBUF                  ; TRANSMITE ENDEREÇO (PARTE HIGH)
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO
   CALL    TESTA_ACK               ; CHAMA ROTINA P/ TESTAR ACK
   BTFSC   F_ERRO                  ; OCORREU ERRO DE ACK ?
   GOTO    I2C_ERRO                ; SIM - PULA P/ I2C_ERRO
                                   ; NÃO
   MOVF    ENDERECO_L,W
   MOVWF   SSPBUF                  ; TRANSMITE ENDEREÇO (PARTE LOW)
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO
   CALL    TESTA_ACK               ; CHAMA ROTINA P/ TESTAR ACK
   BTFSC   F_ERRO                  ; OCORREU ERRO DE ACK ?
   GOTO    I2C_ERRO                ; SIM - PULA P/ I2C_ERRO
                                   ; NÃO
   MOVF    BUFFER,W
   MOVWF   SSPBUF                  ; GRAVA DADO
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO
   CALL    TESTA_ACK               ; CHAMA ROTINA P/ TESTAR ACK
   BTFSC   F_ERRO                  ; OCORREU ERRO DE ACK ?
   GOTO    I2C_ERRO                ; SIM - PULA P/ I2C_ERRO
                                   ; NÃO
   BANK1                           ; ALTERA P/ BANK1
   BSF     SSPCON2,PEN             ; INICIA STOP BIT
   BANK0                           ; VOLTA P/ BANK0
   CALL    AGUARDA_I2C_LIVRE       ; AGUARDA FIM DO EVENTO
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     ROTINA P/ SINALIZAR ERRO NA I2C                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA SOMENTE É EXECUTA CASO ALGUM ERRO DE LEITURA/GRAVAÇÃO OCORRA
; COM A MEMÓRIA SERIAL.
; A ROTINA ENVIA UM STOP BIT PARA FINALIZAR A COMUNICAÇÃO COM A MEMÓRIA
; SERIAL, ENVIA UMA MENSAGEM DE ERRO AO DISPLAY E APÓS 1s RETORNA À TELA
; PRINCIPAL.

I2C_ERRO
   BANK1                           ; ALTERA P/ BANK1
   BSF     SSPCON2,PEN             ; INICIA STOP BIT
   BANK0                           ; VOLTA P/ BANK0

   CLRF    TEC_MATRICIAL           ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDO
   MOVLW   0XC6                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 6
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

   MOVLW   'E'
   CALL    ESCREVE
   MOVLW   'r'
   CALL    ESCREVE
   MOVLW   'r'
   CALL    ESCREVE                 ; ESCREVE "Err" NO LCD

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   MOVLW   .250
   CALL    DELAY_MS
   MOVLW   .250
   CALL    DELAY_MS
   MOVLW   .250
   CALL    DELAY_MS
   MOVLW   .250
   CALL    DELAY_MS                ; DELAY DE 1seg.

   CALL    ATUALIZA_TELA_LINHA_2   ; ATUALIZA TELA PRINCIPAL
```

```
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              CONFIGURAÇÕES INICIAIS DE HARDWARE E SOFTWARE             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  NESTA ROTINA SÃO INICIALIZADAS AS PORTAS DE I/O DO MICROCONTROLADOR E AS
;  CONFIGURAÇÕES DOS REGISTRADORES ESPECIAIS (SFR). A ROTINA INICIALIZA AS
;  VARIÁVEIS DE RAM E AGUARDA O ESTOURO DO WDT.

CONFIG
   CLRF    PORTA                   ; LIMPA O PORTA
   CLRF    PORTB                   ; LIMPA O PORTB
   CLRF    PORTC                   ; LIMPA O PORTC
   CLRF    PORTD                   ; LIMPA O PORTD
   CLRF    PORTE                   ; LIMPA O PORTE


   BANK1                           ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                   ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                   ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011000'
   MOVWF   TRISC                   ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                   ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                   ; CONFIGURA I/O DO PORTE

   MOVLW   B'11011111'
   MOVWF   OPTION_REG              ; CONFIGURA OPTIONS
                                   ; PULL-UPs DESABILITADOS
                                   ; INTER. NA BORDA DE SUBIDA DO RB0
                                   ; TIMER0 INCREM. PELO CICLO DE MÁQUINA
                                   ; WDT   - 1:128
                                   ; TIMER - 1:1

   MOVLW   B'00000000'
   MOVWF   INTCON                  ; CONFIGURA INTERRUPÇÕES
                                   ; DESABILITADA TODAS AS INTERRUPÇÕES

   MOVLW   B'00000111'
   MOVWF   CMCON                   ; DESLIGA OS COMPARADORES

   MOVLW   B'00000111'
   MOVWF   ADCON1                  ; CONFIGURA CONVERSOR A/D
                                   ; CONFIGURA PORTA E PORTE COMO I/O DIGITAL

   MOVLW   B'00001001'
   MOVWF   SSPADD                  ; VELOCIDADE: 100KHz @ 4MHz

   MOVLW   B'10000000'
   MOVWF   SSPSTAT                 ; DESABILITA SLEW-RATE CONTROL (100 KHz)

   BANK0                           ; SELECIONA BANCO 0 DA RAM

   MOVLW   B'00101000'
   MOVWF   SSPCON                  ; HABILITA I2C - MASTER MODE
                                   ; CONFIGURA PINOS COMO DA I2C

;  AS INSTRUÇÕES A SEGUIR FAZEM COM QUE O PROGRAMA TRAVE QUANDO HOUVER UM
;  RESET OU POWER-UP, MAS PASSE DIRETO SE O RESET FOR POR WDT. DESTA FORMA,
;  SEMPRE QUE O PIC É LIGADO, O PROGRAMA TRAVA, AGUARDA UM ESTOURO DE WDT
;  E COMEÇA NOVAMENTE. ISTO EVITA PROBLEMAS NO START-UP DO PIC.

   BTFSC   STATUS,NOT_TO           ; RESET POR ESTOURO DE WATCHDOG TIMER ?
   GOTO    $                       ; NÃO - AGUARDA ESTOURO DO WDT
                                   ; SIM
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          INICIALIZAÇÃO DA RAM                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA IRÁ LIMPAR TODA A RAM DO BANCO 0, INDO DE 0X20 A 0X7F

   MOVLW   0X20
   MOVWF   FSR                     ; APONTA O ENDEREÇAMENTO INDIRETO PARA
                                   ; A PRIMEIRA POSIÇÃO DA RAM
LIMPA_RAM
   CLRF    INDF                    ; LIMPA A POSIÇÃO
   INCF    FSR,F                   ; INCREMENTA O PONTEIRO P/ A PRÓX. POS.
   MOVF    FSR,W
   XORLW   0X80                    ; COMPARA O PONTEIRO COM A ÚLT. POS. +1
   BTFSS   STATUS,Z                ; JÁ LIMPOU TODAS AS POSIÇÕES?
   GOTO    LIMPA_RAM               ; NÃO - LIMPA A PRÓXIMA POSIÇÃO
                                   ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   CONFIGURAÇÕES INICIAIS DO DISPLAY                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA INICIALIZA O DISPLAY P/ COMUNICAÇÃO DE 8 VIAS, DISPLAY PARA 2
; LINHAS, CURSOR APAGADO E DESLOCAMENTO DO CURSOR À DIREITA.

INICIALIZACAO_DISPLAY
   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0X30                    ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   .3
   CALL    DELAY_MS                ; DELAY DE 3MS (EXIGIDO PELO DISPLAY)

   MOVLW   0X30                    ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   0X30                    ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   B'00111000'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; INTERFACE DE 8 VIAS DE DADOS

   MOVLW   B'00000001'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; LIMPAR TODO O DISPLAY

   MOVLW   .1
   CALL    DELAY_MS                ; DELAY DE 1MS

   MOVLW   B'00001100'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; LIGAR O DISPLAY SEM CURSOR

   MOVLW   B'00000110'             ; ESCREVE COMANDO PARA INCREM.
   CALL    ESCREVE                 ; AUTOMÁTICO À DIREITA

   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INICIALIZAÇÃO DA RAM                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE TRECHO DO PROGRAMA LÊ O DADOS DA MEMÓRIAS E2PROM EXTERNA E
;  ATUALIZA A RAM.

LE_MEMORIA_EEPROM
   MOVLW   END_EEPROM_H
   MOVWF   ENDERECO_H
   MOVLW   END_EEPROM_L
   MOVWF   ENDERECO_L              ; CARREGA ENDEREÇO P/ LEITURA

   CALL    I2C_EEPROM_READ         ; CHAMA ROTINA P/ LER DADO

   MOVF    BUFFER,W
   MOVWF   VALOR_DADOS             ; SALVA DADO LIDO EM VALOR_DADOS
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DA TELA PRINCIPAL                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A TELA PRINCIPAL DO PROGRAMA, COM AS FRASES:
;  LINHA 1 - "   MASTER I2C   "
;  LINHA 2 - "      xxh       "

   CALL    ATUALIZA_TELA_LINHA_1   ; ATUALIZA TELA LINHA 1 DO LCD

   CALL    ATUALIZA_TELA_LINHA_2   ; ATUALIZA TELA LINHA 2 DO LCD

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        VARREDURA DOS BOTÕES                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA VERIFICA SE ALGUM BOTÃO ESTÁ PRESSIONADO E CASO AFIRMATIVO
; DESVIA PARA O TRATAMENTO DO MESMO.

VARRE
   CLRWDT                          ; LIMPA WATCHDOG TIMER

; ***************** VERIFICA ALGUM BOTÃO PRESSIONADO ************************

VARRE_BOTOES
   BSF     LINHA_4                 ; ATIVA LINHA 4 DO TECLADO MATRICIAL

   GOTO    $+1                     ; DELAY PARA ESTABILIZAÇÃO
                                   ; E LEITURA DO TECLADO

   BTFSC   BOTAO_0                 ; O BOTÃO 0 ESTA PRESSIONADO ?
   GOTO    TRATA_BOTAO_0           ; SIM - PULA P/ TRATA_BOTAO_0
                                   ; NÃO

   BTFSC   BOTAO_1                 ; O BOTÃO 1 ESTA PRESSIONADO ?
   GOTO    TRATA_BOTAO_1           ; SIM - PULA P/ TRATA_BOTAO_1
                                   ; NÃO

   BTFSC   BOTAO_2                 ; O BOTÃO 2 ESTA PRESSIONADO ?
   GOTO    TRATA_BOTAO_2           ; SIM - PULA P/ TRATA_BOTAO_2
                                   ; NÃO

   BTFSC   BOTAO_3                 ; O BOTÃO 3 ESTA PRESSIONADO ?
   GOTO    TRATA_BOTAO_3           ; SIM - PULA P/ TRATA_BOTAO_3
                                   ; NÃO

   BCF     LINHA_4                 ; DESATIVA A LINHA 4 DO TECLADO MATRICIAL

; ************************** FILTRO P/ EVITAR RUIDOS ************************

   MOVLW   FILTRO_TECLA            ; CARREGA O VALOR DE FILTRO_TECLA
   MOVWF   FILTRO_BOTOES           ; SALVA EM FILTRO_BOTOES
                                   ; RECARREGA FILTRO P/ EVITAR RUIDOS
                                   ; NOS BOTÕES
   MOVLW   .1
   MOVWF   TEMPO_TURBO             ; CARREGA TEMPO DO TURBO DAS TECLAS
                                   ; COM 1 - IGNORA O TURBO A PRIMEIRA
                                   ; VEZ QUE A TECLA É PRESSIONADA

   GOTO    VARRE                   ; VOLTA PARA VARRER TECLADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       TRATAMENTO DOS BOTÕES                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; NESTE TRECHO DO PROGRAMA ESTÃO TODOS OS TRATAMENTOS DOS BOTÕES

; ************************* TRATAMENTO DO BOTÃO 0 ***************************

TRATA_BOTAO_0
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO

   DECFSZ  TEMPO_TURBO,F           ; FIM DO TEMPO DE TURBO ?
```

```
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM
   MOVLW   TURBO_TECLA
   MOVWF   TEMPO_TURBO             ; RECARREGA TEMPORIZADOR DO TURBO
                                   ; DAS TECLAS

   INCF    VALOR_DADOS,F           ; INCREMENTA VALOR_DADOS

   CALL    ATUALIZA_TELA_LINHA_2   ; CHAMA ROTINA P/ ATUALIZAR LCD

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************ TRATAMENTO DO BOTÃO 1 ***************************

TRATA_BOTAO_1
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO

   DECFSZ  TEMPO_TURBO,F           ; FIM DO TEMPO DE TURBO ?
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM
   MOVLW   TURBO_TECLA
   MOVWF   TEMPO_TURBO             ; RECARREGA TEMPORIZADOR DO TURBO
                                   ; DAS TECLAS

   DECF    VALOR_DADOS,F           ; DECREMENTA VALOR_DADOS

   CALL    ATUALIZA_TELA_LINHA_2   ; CHAMA ROTINA P/ ATUALIZAR LCD

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************ TRATAMENTO DO BOTÃO 2 ***************************

TRATA_BOTAO_2
   MOVF    FILTRO_BOTOES,F
   BTFSC   STATUS,Z                ; FILTRO JÁ IGUAL A ZERO ?
                                   ; (FUNÇÃO JA FOI EXECUTADA?)
   GOTO    VARRE                   ; SIM - VOLTA P/ VARREDURA DO TECLADO
                                   ; NÃO
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO

; ********TRECHO DO PROGRAMA PARA GRAVAR DADO DA RAM NA MEMÓRIA *************

GRAVA_MEMORIA_EEPROM
   MOVLW   END_EEPROM_H
   MOVWF   ENDERECO_H
   MOVLW   END_EEPROM_L
   MOVWF   ENDERECO_L              ; CARREGA ENDERECO ONDE O DADO SERÁ SALVO
                                   ; END. -> 0x0000
                                   ; PRIMEIRA POSIÇÃO DA EEPROM
   MOVF    VALOR_DADOS,W
   MOVWF   BUFFER                  ; CARREGA DADO A SER SALVO EM BUFFER

   CALL    I2C_EEPROM_WRITE        ; CHAMA ROTINA DE GRAVAÇÃO

   MOVLW   .10
   CALL    DELAY_MS                ; GARANTE TEMPO DE ESCRITA (10ms)

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************ TRATAMENTO DO BOTÃO 3 ***************************

TRATA_BOTAO_3
   MOVF    FILTRO_BOTOES,F
   BTFSC   STATUS,Z                ; FILTRO JÁ IGUAL A ZERO ?
                                   ; (FUNÇÃO JA FOI EXECUTADA?)
   GOTO    VARRE                   ; SIM - VOLTA P/ VARREDURA DO TECLADO
                                   ; NÃO
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
```

```
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO

; ******** TRECHO DO PROGRAMA PARA LER DADO DA MEMÓRIA E ATUALIZAR RAM
*************

LER_MEMORIA_EEPROM
   MOVLW   END_EEPROM_H
   MOVWF   ENDERECO_H
   MOVLW   END_EEPROM_L
   MOVWF   ENDERECO_L              ; CARREGA ENDEREÇO DE LEITURA
                                   ; END. -> 0x0000
                                   ; PRIMEIRA POSIÇÃO DA EEPROM

   CALL    I2C_EEPROM_READ         ; CHAMA ROTINA DE LEITURA

   BTFSC   F_ERRO
   GOTO    $+3
   MOVF    BUFFER,W
   MOVWF   VALOR_DADOS             ; ATUALIZA RAM COM O VALOR LIDO

   CALL    ATUALIZA_TELA_LINHA_2   ; CHAMA ROTINA P/ ATUALIZAR LCD

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            FIM DO PROGRAMA                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                             ; FIM DO PROGRAMA
```

Dicas e Comentários

As constantes END_EEPROM_H e END_EEPROM_L representam a posição a ser gravada/lida da memória externa.

Note que este programa não utiliza as interrupções de leitura e escrita relacionadas ao protocolo I^2C. Desta forma, ele possui uma rotina (AGUARDA_I2C_LIVRE) para saber se o sistema está liberado para a próxima ação. Ele também testa o ACK e gera uma mensagem de erro caso alguma coisa saia fora do padrão.

Exercícios Propostos

1. Faça três modificações no primeiro exercício proposto da experiência 15.
   - Utilize a memória externa;
   - Limite os dados mostrados no display entre 0x41 e 0x5A;
   - Mostre os dados em ASCII, ou seja, entre A (0x41) e Z (0x5A);
2. Utilizando o exercício anterior grave na memória uma mensagem de até 16 caracteres. Depois, crie um programa que ao ser inicializado leia os 16 caracteres da memória e mostre a mensagem lida no LCD;

Capítulo 19 - Experiência 17 – Comunicação serial RS232 via USART

Objetivo

O objetivo desta experiência é o aprendizado do módulo USART do microcontrolador PIC16F877A utilizado para implementar a comunicação padrão RS232, geralmente utilizada para estabelecer um canal de comunicação entre um microcontrolador e um computador.

Descrição

Para tornar o sistema versátil e simples, criou-se um programa capaz de testar a transmissão e recepção de dados de modo isolado, ou seja, apenas com o MCMASTER sem necessariamente conecta-lo ao computador. Embora nada impeça que a comunicação com o PC seja efetivamente realizada.

Para atender esta necessidade, o software da experiência implementa uma comunicação assíncrona *Full duplex*, isto é, com a transmissão e a recepção ativadas simultaneamente.

O valor transmitido é obtido a partir da leitura da tensão do potenciômetro através do conversor A/D limitando este valor entre 0 e 255 (8-bits). O valor do A/D é então enviado para a porta serial e para o LCD. Desta forma é possível visualizar o dado transmitido. Para facilitar ainda mais o usuário, mostra-se o valor em decimal (d) e em hexadecimal (h). A comunicação é realizada no padrão 8N1 com uma velocidade de 9.600bps.

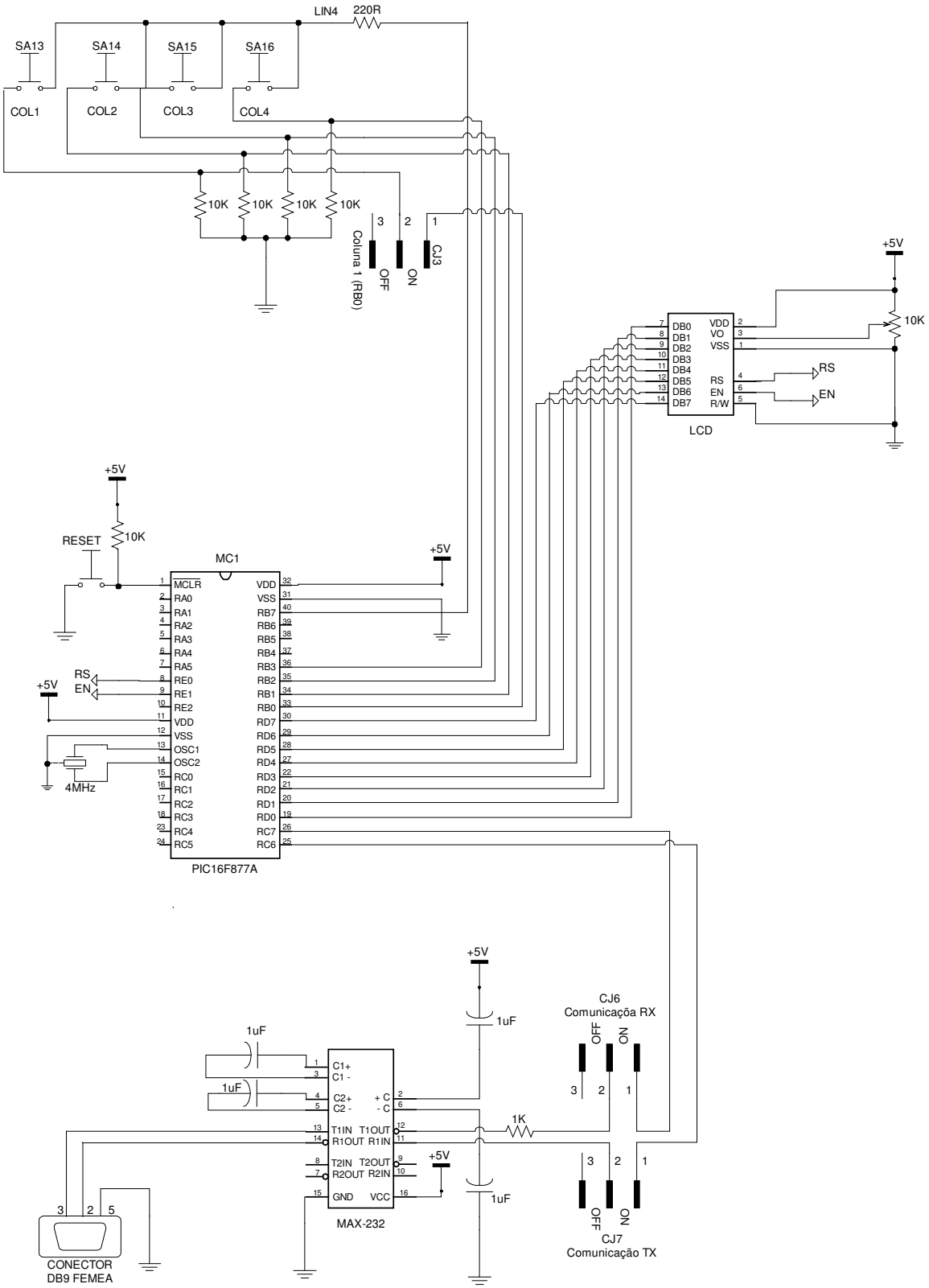
Quanto à recepção, o valor obtido pela porta serial é diretamente impresso no display de LCD, através do código ASCII.

Para que o sistema funcione sem o PC, basta interligar os pinos 2 e 3 do conector DB9. Isto fará com que tudo que seja transmitido por TX seja imediatamente recebido em RX. Tanto a transmissão quanto a recepção são contínuas.

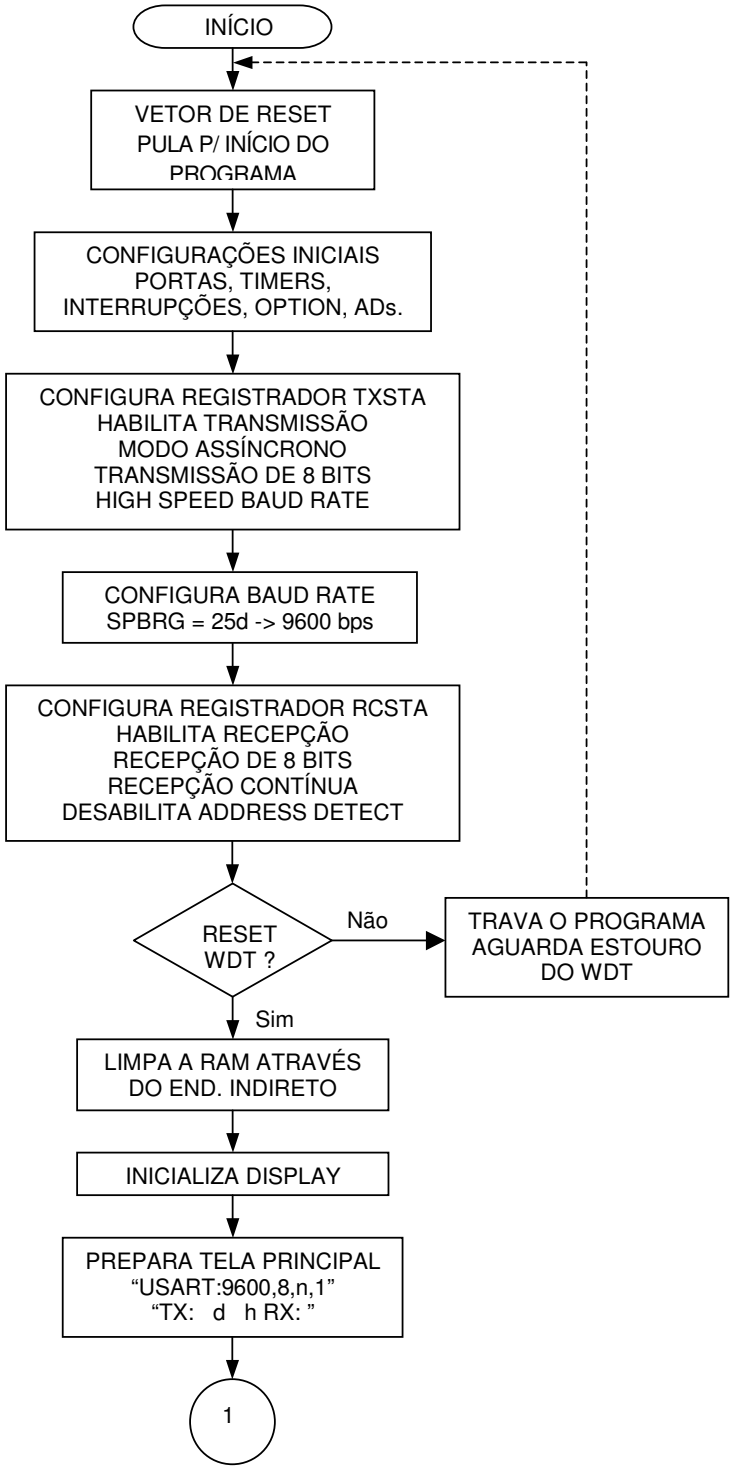
Não se deve esquecer de habilitar a porta RS232 para o microcontrolador através do botão de modo de operação após.

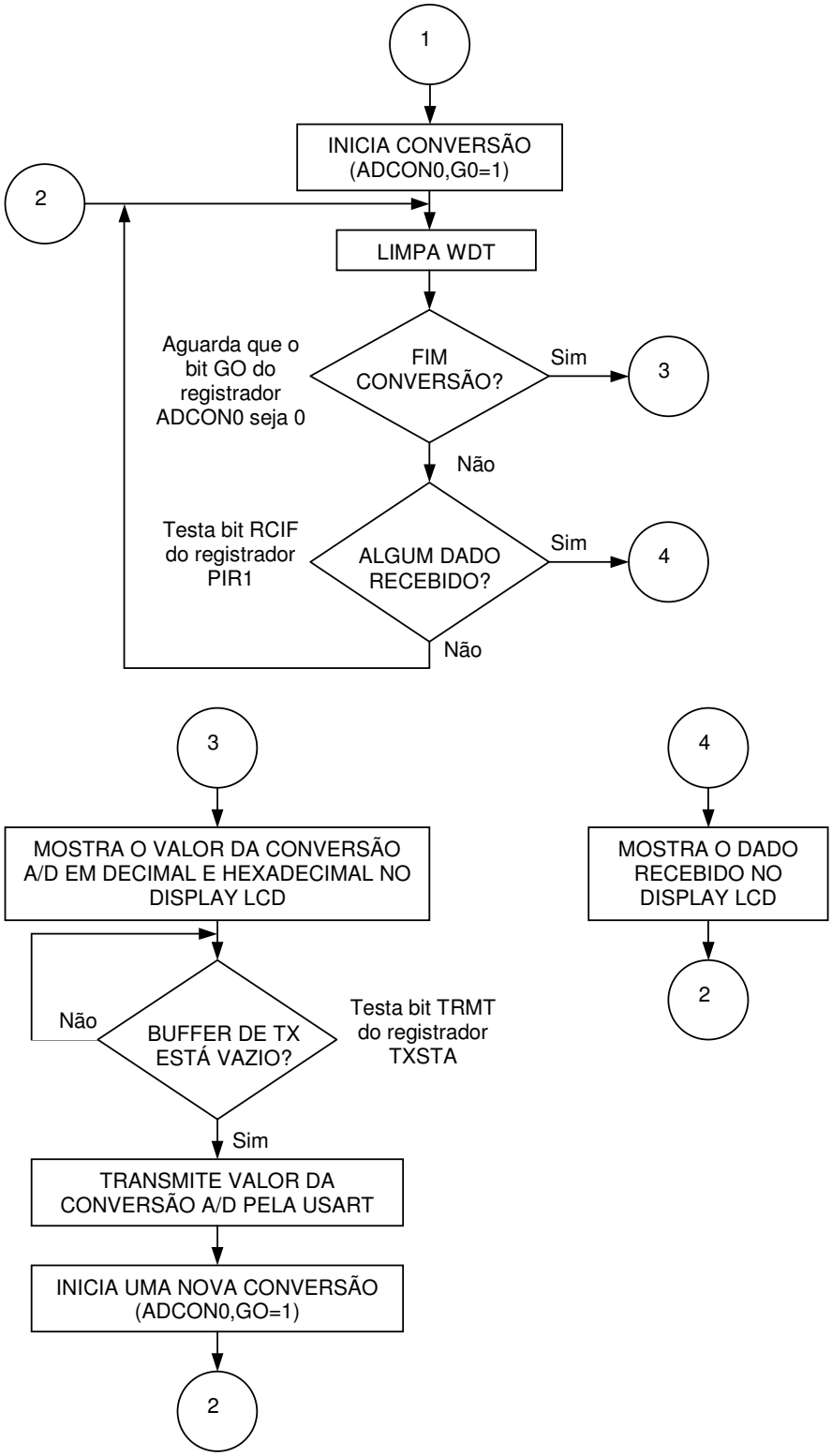
Para testar a comunicação com o computador (PC) pode-se utilizar o software SDCom disponível no CD.

Esquema Elétrico



Fluxograma


INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURAÇÕES INICIAIS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION, ADs.
CONFIGURA REGISTRADOR TXSTA
HABILITA TRANSMISSÃO
MODO ASSÍNCRONO
TRANSMISSÃO DE 8 BITS
HIGH SPEED BAUD RATE
CONFIGURA BAUD RATE
SPBRG = 25d -> 9600 bps
CONFIGURA REGISTRADOR RCSTA
HABILITA RECEPÇÃO
RECEPÇÃO DE 8 BITS
RECEPÇÃO CONTÍNUA
DESABILITA ADDRESS DETECT
RESET
WDT ?
Não
TRAVA O PROGRAMA
AGUARDA ESTOURO
DO WDT
Sim
LIMPA A RAM ATRAVÉS
DO END. INDIRETO
INICIALIZA DISPLAY
PREPARA TELA PRINCIPAL
“USART:9600,8,n,1”
“TX: d h RX: ”
1


1
INICIA CONVERSÃO
(ADCON0,G0=1)
2
LIMPA WDT
Aguarda que o
bit GO do
registrador
ADCON0 seja 0
FIM
CONVERSÃO?
Sim
3
Não
Testa bit RCIF
do registrador
PIR1
ALGUM DADO
RECEBIDO?
Sim
4
Não
3
MOSTRA O VALOR DA CONVERSÃO
A/D EM DECIMAL E HEXADECIMAL NO
DISPLAY LCD
Não
BUFFER DE TX
ESTÁ VAZIO?
Testa bit TRMT
do registrador
TXSTA
Sim
TRANSMITE VALOR DA
CONVERSÃO A/D PELA USART
INICIA UMA NOVA CONVERSÃO
(ADCON0,GO=1)
2
4
MOSTRA O DADO
RECEBIDO NO
DISPLAY LCD
2

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *            EXPERIÊNCIA 17 - COMUNICAÇÃO SERIAL RS232 VIA USART          *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            DESCRIÇÃO GERAL                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE EXEMPLO FOI ELABORADO PARA EXPLICAR O FUNCIONAMENTO DA USART DO PIC.
;  O SOFTWARE CONVERTE O CANAL 0 DO CONVERSOR A/D (POTENCIÔMETRO) E MOSTRA
;  NO DISPLAY O VALOR CONVERTIDO EM DECIMAL E HAXADECIMAL.
;  ALÉM DE MOSTRAR O VALOR NO DISPLAY, O SOFTWARE TRANSMITE PELA USART O VALOR
;  DA CONVERSÃO. OS VALORES RECEBIDOS PELA USART TAMBÉM SÃO MOSTRADOS NO LCD
;  COMO CARACTERES ASCII.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        CONFIGURAÇÕES PARA GRAVAÇÃO                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_ON & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          DEFINIÇÃO DAS VARIÁVEIS                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE BLOCO DE VARIÁVEIS ESTÁ LOCALIZADO LOGO NO INÍCIO DO BANCO 0

   CBLOCK  0X20                    ; POSIÇÃO INICIAL DA RAM

           TEMPO0
           TEMPO1                  ; TEMPORIZADORES P/ ROTINA DE DELAY

           AUX                     ; REGISTRADOR AUXILIAR DE USO GERAL

           UNIDADE                 ; ARMAZENA VALOR DA UNIDADE
           DEZENA                  ; ARMAZENA VALOR DA DEZENA
           CENTENA                 ; ARMAZENA VALOR DA CENTENA

   ENDC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>         ; MICROCONTROLADOR UTILIZADO


; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      DEFINIÇÃO DOS BANCOS DE RAM                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           CONSTANTES INTERNAS                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE CONSTANTES FACILITA A PROGRAMAÇÃO E A MANUTENÇÃO.
```

```
;  ESTE PROGRAMA NÃO UTILIZA NENHUMA CONSTANTE.

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    DECLARAÇÃO DOS FLAGs DE SOFTWARE                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE FLAGs AJUDA NA PROGRAMAÇÃO E ECONOMIZA MEMÓRIA RAM.

;  ESTE PROGRAMA NÃO UTILIZA NENHUM FLAG DE USUÁRIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                ENTRADAS                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS ENTRADAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

;  ESTE PROGRAMA UTILIZA UMA ENTRADA P/ O CONVERSOR A/D.
;  ESTA ENTRADA NÃO PRECISA SER DECLARADA, POIS O SOFTWARE NUNCA FAZ
;  REFERÊNCIA A ELA DE FORMA DIRETA, POIS O CANAL A/D A SER CONVERTIDO É
;  SELECIONADO NO REGISTRADOS ADCON0 DE FORMA BINÁRIA E NÃO ATRAVÉS DE
;  DEFINES. PORÉM PARA FACILITAR O ENTENDIMENTO DO HARDWARE VAMOS DECLARAR
;  ESTA ENTRADA NORMALMENTE.

#DEFINE    CAD     PORTA,0         ; ENTRADA A/D DO POTENCIÔMETRO

;  ALÉM DA ENTRADA DO CONVERSOR A/D, TEMOS A ENTRADA DA USART (RECEPÇÃO).
;  NOVAMENTE ESTA ENTRADA NÃO NECESSITA SER DECLARADA, PORÉM, PARA
;  FACILITAR O ENTENDIMENTO DO HARDWARE VAMOS DECLARAR ESTA ENTRADA
;  NORMALMENTE.

#DEFINE    RXUSART PORTC,7         ; ENTRADA DE RX DA USART

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                SAÍDAS                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS SAÍDAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    DISPLAY         PORTD   ; BARRAMENTO DE DADOS DO DISPLAY

#DEFINE    RS              PORTE,0 ; INDICA P/ O DISPLAY UM DADO OU COMANDO
                                   ; 1 -> DADO
                                   ; 0 -> COMANDO

#DEFINE    ENABLE          PORTE,1 ; SINAL DE ENABLE P/ DISPLAY
                                   ; ATIVO NA BORDA DE DESCIDA

;  TEMOS TAMBÉM A SAÍDA DE TX DA USART.
;  NOVAMENTE ESTA SAÍDA NÃO NECESSITA SER DECLARADA, PORÉM, PARA FACILITAR O
;  ENTENDIMENTO DO HARDWARE VAMOS DECLARAR ESTA SAÍDA NORMALMENTE.

#DEFINE    TXUSART PORTC,6         ; SAÍDA DE TX DA USART

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  VETOR DE RESET DO MICROCONTROLADOR                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO INICIAL PARA EXECUÇÃO DO PROGRAMA

   ORG     0X0000                  ; ENDEREÇO DO VETOR DE RESET
   GOTO    CONFIG                  ; PULA PARA CONFIG DEVIDO A REGIÃO
                                   ; DESTINADA AS ROTINAS SEGUINTES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  ROTINA DE DELAY (DE 1MS ATÉ 256MS)                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA É UMA ROTINA DE DELAY VARIÁVEL, COM DURAÇÃO DE 1MS X O VALOR PASSADO
;  EM WORK (W).

DELAY_MS
   MOVWF   TEMPO1                  ; CARREGA TEMPO1 (UNIDADES DE MS)
   MOVLW   .250
   MOVWF   TEMPO0                  ; CARREGA TEMPO0 (P/ CONTAR 1MS)
```

```
   CLRWDT                        ; LIMPA WDT (PERDE TEMPO)
   DECFSZ  TEMPO0,F              ; FIM DE TEMPO0 ?
   GOTO    $-2                   ; NÃO - VOLTA 2 INSTRUÇÕES
                                 ; SIM - PASSOU-SE 1MS
   DECFSZ  TEMPO1,F              ; FIM DE TEMPO1 ?
   GOTO    $-6                   ; NÃO - VOLTA 6 INSTRUÇÕES
                                 ; SIM
   RETURN                        ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DE UM CARACTER NO DISPLAY                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ENVIA UM CARACTER PARA O MÓDULO DE LCD. O CARACTER A SER
;  ESCRITO DEVE SER COLOCADO EM WORK (W) ANTES DE CHAMAR A ROTINA.

ESCREVE
   MOVWF   DISPLAY               ; ATUALIZA DISPLAY (PORTD)
   NOP                           ; PERDE 1US PARA ESTABILIZAÇÃO
   BSF     ENABLE                ; ENVIA UM PULSO DE ENABLE AO DISPLAY
   GOTO    $+1                   ; .
   BCF     ENABLE                ; .

   MOVLW   .1
   CALL    DELAY_MS              ; DELAY DE 1MS
   RETURN                        ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              AJUSTE DECIMAL                               *
; *           W [HEX] =  CENTENA [DEC] : DEZENA [DEC] ; UNIDADE [DEC]         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA RECEBE UM ARGUMENTO PASSADO PELO WORK E RETORNA NAS VARIÁVEIS
;  CENTENA, DEZENA E UNIDADE O NÚMERO BCD CORRESPONDÊNTE AO PARÂMETRO PASSADO.

AJUSTE_DECIMAL
   MOVWF   AUX                   ; SALVA VALOR A CONVERTER EM AUX

   CLRF    UNIDADE
   CLRF    DEZENA
   CLRF    CENTENA               ; RESETA REGISTRADORES

   MOVF    AUX,F
   BTFSC   STATUS,Z              ; VALOR A CONVERTER = 0 ?
   RETURN                        ; SIM - RETORNA
                                 ; NÃO
   INCF    UNIDADE,F             ; INCREMENTA UNIDADE

   MOVF    UNIDADE,W
   XORLW   0X0A
   BTFSS   STATUS,Z              ; UNIDADE = 10d ?
   GOTO    $+3                   ; NÃO
                                 ; SIM
   CLRF    UNIDADE               ; RESETA UNIDADE
   INCF    DEZENA,F              ; INCREMENTA DEZENA

   MOVF    DEZENA,W
   XORLW   0X0A
   BTFSS   STATUS,Z              ; DEZENA = 10d ?
   GOTO    $+3                   ; NÃO
                                 ; SIM
   CLRF    DEZENA                ; RESETA DEZENA
   INCF    CENTENA,F             ; INCREMENTA CENTENA

   DECFSZ  AUX,F                 ; FIM DA CONVERSÃO ?
   GOTO    $-.14                 ; NÃO - VOLTA P/ CONTINUAR CONVERSÃO
   RETURN                        ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               CONFIGURAÇÕES INICIAIS DE HARDWARE E SOFTWARE               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  NESTA ROTINA SÃO INICIALIZADAS AS PORTAS DE I/O DO MICROCONTROLADOR E AS
;  CONFIGURAÇÕES DOS REGISTRADORES ESPECIAIS (SFR). A ROTINA INICIALIZA A
```

```
;  MÁQUINA E AGUARDA O ESTOURO DO WDT.

CONFIG
   CLRF    PORTA                   ; LIMPA O PORTA
   CLRF    PORTB                   ; LIMPA O PORTB
   CLRF    PORTC                   ; LIMPA O PORTC
   CLRF    PORTD                   ; LIMPA O PORTD
   CLRF    PORTE                   ; LIMPA O PORTE


   BANK1                           ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                   ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                   ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011000'
   MOVWF   TRISC                   ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                   ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                   ; CONFIGURA I/O DO PORTE

   MOVLW   B'11011011'
   MOVWF   OPTION_REG              ; CONFIGURA OPTIONS
                                   ; PULL-UPs DESABILITADOS
                                   ; INTER. NA BORDA DE SUBIDA DO RB0
                                   ; TIMER0 INCREM. PELO CICLO DE MÁQUINA
                                   ; WDT   - 1:8
                                   ; TIMER - 1:1

   MOVLW   B'00000000'
   MOVWF   INTCON                  ; CONFIGURA INTERRUPÇÕES
                                   ; DESABILITA TODAS AS INTERRUPÇÕES

   MOVLW   B'00000111'
   MOVWF   CMCON                   ; DESLIGA OS COMPARADORES

   MOVLW   B'00000100'
   MOVWF   ADCON1                  ; CONFIGURA CONVERSOR A/D
                                   ; RA0, RA1 E RA3 COMO ANALÓGICO
                                   ; RA2, RA4 E RA5 COMO I/O DIGITAL
                                   ; PORTE COMO I/O DIGITAL
                                   ; JUSTIFICADO À ESQUERDA
                                   ; 8 BITS EM ADRESH E 2 BITS EM ADRESL
                                   ; Vref+ = VDD (+5V)
                                   ; Vref- = GND ( 0V)

   MOVLW   B'00100100'
   MOVWF   TXSTA                   ; CONFIGURA USART
                                   ; HABILITA TX
                                   ; MODO ASSINCRONO
                                   ; TRANSMISSÃO DE 8 BITS
                                   ; HIGH SPEED BAUD RATE
   MOVLW   .25
   MOVWF   SPBRG                   ; ACERTA BAUD RATE -> 9600bps

   BANK0                           ; SELECIONA BANCO 0 DA RAM

   MOVLW   B'10010000'
   MOVWF   RCSTA                   ; CONFIGURA USART
                                   ; HABILITA RX
                                   ; RECEPÇÃO DE 8 BITS
                                   ; RECEPÇÃO CONTÍNUA
                                   ; DESABILITA ADDRESS DETECT

   MOVLW   B'01000001'
   MOVWF   ADCON0                  ; CONFIGURA CONVERSOR A/D
                                   ; VELOCIDADE -> Fosc/8
```

```
                                ; CANAL 0
                                ; MÓDULO LIGADO

;  AS INSTRUÇÕES A SEGUIR FAZEM COM QUE O PROGRAMA TRAVE QUANDO HOUVER UM
;  RESET OU POWER-UP, MAS PASSE DIRETO SE O RESET FOR POR WDT. DESTA FORMA,
;  SEMPRE QUE O PIC É LIGADO, O PROGRAMA TRAVA, AGUARDA UM ESTOURO DE WDT
;  E COMEÇA NOVAMENTE. ISTO EVITA PROBLEMAS NO START-UP DO PIC.

   BTFSC   STATUS,NOT_TO        ; RESET POR ESTOURO DE WATCHDOG TIMER ?
   GOTO    $                    ; NÃO - AGUARDA ESTOURO DO WDT
                                ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INICIALIZAÇÃO DA RAM                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA IRÁ LIMPAR TODA A RAM DO BANCO 0, INDO DE 0X20 A 0X7F

   MOVLW   0X20
   MOVWF   FSR                  ; APONTA O ENDEREÇAMENTO INDIRETO PARA
                                ; A PRIMEIRA POSIÇÃO DA RAM
LIMPA_RAM
   CLRF    INDF                 ; LIMPA A POSIÇÃO
   INCF    FSR,F                ; INCREMENTA O PONTEIRO P/ A PRÓX. POS.
   MOVF    FSR,W
   XORLW   0X80                 ; COMPARA O PONTEIRO COM A ÚLT. POS. +1
   BTFSS   STATUS,Z             ; JÁ LIMPOU TODAS AS POSIÇÕES?
   GOTO    LIMPA_RAM            ; NÃO - LIMPA A PRÓXIMA POSIÇÃO
                                ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    CONFIGURAÇÕES INICIAIS DO DISPLAY                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA INICIALIZA O DISPLAY P/ COMUNICAÇÃO DE 8 VIAS, DISPLAY PARA 2
; LINHAS, CURSOR APAGADO E DESLOCAMENTO DO CURSOR À DIREITA.

INICIALIZACAO_DISPLAY
   BCF     RS                   ; SELECIONA O DISPLAY P/ COMANDOS

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO

   MOVLW   .3
   CALL    DELAY_MS             ; DELAY DE 3MS (EXIGIDO PELO DISPLAY)

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO

   MOVLW   B'00111000'          ; ESCREVE COMANDO PARA
   CALL    ESCREVE              ; INTERFACE DE 8 VIAS DE DADOS

   MOVLW   B'00000001'          ; ESCREVE COMANDO PARA
   CALL    ESCREVE              ; LIMPAR TODO O DISPLAY

   MOVLW   .1
   CALL    DELAY_MS             ; DELAY DE 1MS

   MOVLW   B'00001100'          ; ESCREVE COMANDO PARA
   CALL    ESCREVE              ; LIGAR O DISPLAY SEM CURSOR

   MOVLW   B'00000110'          ; ESCREVE COMANDO PARA INCREM.
   CALL    ESCREVE              ; AUTOMÁTICO À DIREITA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                ROTINA DE ESCRITA DA TELA PRINCIPAL                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A TELA PRINCIPAL DO PROGRAMA, COM AS FRASES:
;  LINHA 1 - "USART:9600,8,n,1"
;  LINHA 2 - "TX:   d   h RX: "
```

```
MOVLW   0X80                    ; COMANDO PARA POSICIONAR O CURSOR
CALL    ESCREVE                 ; LINHA 0 / COLUNA 0
BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

                                ; COMANDOS PARA ESCREVER AS
                                ; LETRAS DE "USART:9600,8,n,1"
MOVLW   'U'
CALL    ESCREVE
MOVLW   'S'
CALL    ESCREVE
MOVLW   'A'
CALL    ESCREVE
MOVLW   'R'
CALL    ESCREVE
MOVLW   'T'
CALL    ESCREVE
MOVLW   ':'
CALL    ESCREVE
MOVLW   '9'
CALL    ESCREVE
MOVLW   '6'
CALL    ESCREVE
MOVLW   '0'
CALL    ESCREVE
MOVLW   '0'
CALL    ESCREVE
MOVLW   ','
CALL    ESCREVE
MOVLW   '8'
CALL    ESCREVE
MOVLW   ','
CALL    ESCREVE
MOVLW   'n'
CALL    ESCREVE
MOVLW   ','
CALL    ESCREVE
MOVLW   '1'
CALL    ESCREVE

BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDO
MOVLW   0XC0                    ; COMANDO PARA POSICIONAR O CURSOR
CALL    ESCREVE                 ; LINHA 1 / COLUNA 0
BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

                                ; COMANDOS PARA ESCREVER AS
                                ; LETRAS DE "TX:   d   h RX: "
MOVLW   'T'
CALL    ESCREVE
MOVLW   'X'
CALL    ESCREVE
MOVLW   ':'
CALL    ESCREVE
MOVLW   ' '
CALL    ESCREVE
MOVLW   ' '
CALL    ESCREVE
MOVLW   ' '
CALL    ESCREVE
MOVLW   'd'
CALL    ESCREVE
MOVLW   ' '
CALL    ESCREVE
MOVLW   ' '
CALL    ESCREVE
MOVLW   ' '
CALL    ESCREVE
MOVLW   'h'
CALL    ESCREVE
MOVLW   ' '
CALL    ESCREVE
MOVLW   'R'
CALL    ESCREVE
```

```
   MOVLW   'X'
   CALL    ESCREVE
   MOVLW   ':'
   CALL    ESCREVE

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              LOOP PRINCIPAL                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A ROTINA PRINCIPAL FICA AGUARDANDO O FINAL DA CONVERSÃO A/D E VERIFICANDO
;  SE ALGUM DADO FOI RECEBIDO PELA USART

   BSF     ADCON0,GO              ; INICIA CONVERSÃO A/D
                                  ; EXECUTADA APENAS UMA VEZ

LOOP
   CLRWDT                         ; LIMPA WATCHDOG TIMER

   BTFSS   ADCON0,GO              ; FIM DA CONVERSÃO ?
   GOTO    FIM_CONVERSAO_AD       ; SIM
                                  ; NÃO
   BTFSC   PIR1,RCIF              ; RECEBEU ALGUM DADO NA SERIAL ?
   GOTO    DADO_RECEBIDO          ; SIM
                                  ; NÃO
   GOTO    LOOP                   ; VOLTA P/ LOOP

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     MOSTRA A/D NO DISPLAY E TRANSMITE                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA MOSTRA O VALOR DA CONVERSÃO A/D NO DISPLAY LCD TANTO EM DECIMAL
;  COMO EM HEXADECIMAL. O VALOR DA CONVERSÃO TAMBÉM É TRANSMITIDO PELA USART.
;  AO FINAL, A ROTINA REQUISITA UMA NOVA CONVERSÃO A/D.

FIM_CONVERSAO_AD

; ************* MOSTRA VALOR DA CONVERSÃO A/D EM DECIMAL ********************

   MOVF    ADRESH,W               ; CARREGA WORK COM VALOR DO A/D
   CALL    AJUSTE_DECIMAL         ; CHAMA ROTINA DE AJUSTE DECIMAL

   BCF     RS                     ; SELECIONA O DISPLAY P/ COMANDO
   MOVLW   0XC3                   ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                ; LINHA 1 / COLUNA 3
   BSF     RS                     ; SELECIONA O DISPLAY P/ DADOS

   MOVF    CENTENA,W
   ADDLW   0X30                   ; CONVERTE BCD DA CENTENA EM ASCII
   CALL    ESCREVE                ; ENVIA AO LCD

   MOVF    DEZENA,W
   ADDLW   0X30                   ; CONVERTE BCD DA DEZENA EM ASCII
   CALL    ESCREVE                ; ENVIA AO LCD

   MOVF    UNIDADE,W
   ADDLW   0X30                   ; CONVERTE BCD DA UNIDADE EM ASCII
   CALL    ESCREVE                ; ENVIA AO LCD

; ************* MOSTRA VALOR DA CONVERSÃO A/D EM HEXADECIMAL ****************

   BCF     RS                     ; SELECIONA O DISPLAY P/ COMANDO
   MOVLW   0XC8                   ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                ; LINHA 1 / COLUNA 8
   BSF     RS                     ; SELECIONA O DISPLAY P/ DADOS

   SWAPF   ADRESH,W               ; INVERTE NIBLE DO ADRESH
   ANDLW   B'00001111'            ; MASCARA BITS MAIS SIGNIFICATIVOS
   MOVWF   AUX                    ; SALVA EM AUXILIAR

   MOVLW   0X0A
   SUBWF   AUX,W                  ; AUX - 10d (ATUALIZA FLAG DE CARRY)
   MOVLW   0X30                   ; CARREGA WORK COM 30h
   BTFSC   STATUS,C               ; RESULTADO É POSITIVO? (É UMA LETRA?)
   MOVLW   0X37                   ; SIM - CARREGA WORK COM 37h
```

```
                                ; NÃO - WORK FICA COM 30h (NÚMERO)
   ADDWF   AUX,W                ; SOMA O WORK AO AUXILIAR
                                ; (CONVERSÃO ASCII)
   CALL    ESCREVE              ; ENVIA CARACTER AO DISPLAY LCD

   MOVF    ADRESH,W             ; CARREGA WORK COM ADRESH
   ANDLW   B'00001111'          ; MASCARA BITS MAIS SIGNIFICATIVOS
   MOVWF   AUX                  ; SALVA EM AUXILIAR

   MOVLW   0X0A
   SUBWF   AUX,W                ; AUX - 10d (ATUALIZA FLAG DE CARRY)
   MOVLW   0X30                 ; CARREGA WORK COM 30h
   BTFSC   STATUS,C             ; RESULTADO É POSITIVO? (É UMA LETRA?)
   MOVLW   0X37                 ; SIM - CARREGA WORK COM 37h
                                ; NÃO - WORK FICA COM 30h (NÚMERO)
   ADDWF   AUX,W                ; SOMA O WORK AO AUXILIAR
                                ; (CONVERSÃO ASCII)
   CALL    ESCREVE              ; ENVIA CARACTER AO DISPLAY LCD

; ************* TRANSMITE VALOR DA CONVERSÃO A/D PELA USART *****************

   MOVF    ADRESH,W             ; CARREGA WORK COM O VALOR DO A/D
   BANK1                        ; ALTERA P/ BANCO 1 DA RAM
   BTFSS   TXSTA,TRMT           ; O BUFFER DE TX ESTÁ VAZIO ?
   GOTO    $-1                  ; NÃO - AGUARDA ESVAZIAR
   BANK0                        ; SIM - VOLTA P/ BANCO 0 DA RAM
   MOVWF   TXREG                ; SALVA WORK EM TXREG (INICIA TX)

; ********************** INICIA UMA NOVA CONVERSÃO ************************

   BSF     ADCON0,GO            ; PEDE UMA NOVA CONVERSÃO A/D

   GOTO    LOOP                 ; VOLTA PARA LOOP

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    ROTINA DE RECEPÇÃO DE DADOS NA USART               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA É EXECUTADA TODA VEZ QUE UM NOVO DADO É RECEBIDO PELA USART.
;  O DADO RECEBIDO É MOSTRADO NO LCD (EM ASCII).

DADO_RECEBIDO
   BCF     RS                   ; SELECIONA O DISPLAY P/ COMANDO
   MOVLW   0XCF                 ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE              ; LINHA 1 / COLUNA 15
   BSF     RS                   ; SELECIONA O DISPLAY P/ DADOS

   MOVF    RCREG,W              ; CARREGA DADO RECEBIDO NO WORK
   CALL    ESCREVE              ; ENVIA AO LCD
                                ; AO LER O REGISTRADOR RCREG O BIT
                                ; RCIF DA INTERRUPÇÃO É LIMPO
                                ; AUTOMATICAMENTE.

   GOTO    LOOP                 ; VOLTA P/ LOOP PRINCIPAL

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           FIM DO PROGRAMA                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                          ; FIM DO PROGRAMA
```

Dicas e Comentários

A rotina de conversão Hex >>> Decimal deste exemplo é mais completa do que nos casos anteriores, pois trabalha com 3 dígitos (CENTENA, DEZENA e UNIDADE). Desta forma, ela pode converter todo o *range* do argumento de entrada (W) que vai de 0 a 255.

O sistema de conversão A/D é o mesmo apresentado na experiência 11, onde utilizou-se o conversor interno considerando apenas os 8 bits mais significativos. Com isso o valor a ser transmitido já fica limitado a um byte.

Devido a simplicidade do sistema não foi necessário o uso das interrupções, deixando-as desabilitadas. Para o caso da recepção o bit **RCIF** é testando toda vez dentro do *loop* principal. Quanto a transmissão, sempre que um novo valor foi convertido, checa-se se o *buffer* de saída está vazio para poder escrever o novo valor.

Exercícios Propostos

1. Ative o uso da interrupção de recebimento. Quanto à transmissão, em vez de deixá-la contínua, crie uma interrupção de timer como base de tempo. Por exemplo, transmita o valor atual convertido a cada 1 segundo;
2. Crie um programa no PC que receba o valor convertido, efetue alguma operação e devolva outro valor. Por exemplo, divida o valor por 25, pegue a parte inteira e some 30h para imprimir no LCD um valor de 0 a 9;
3. Mude a rotina de recepção e escrita no LCD para poder receber um número de 0 a 50 e mostrá-lo como 0.0 a 5.0. Altere o programa do PC para efetuar a regra de 3 necessária para converter um valor de 0 a 255 para 0 a 50. Com isso você voltou ao multímetro da experiência 11, só que com as contas de multiplicação e divisão não mais sendo feitas no PIC.

Capítulo 20 - Experiência 18 – Teclado matricial 4x4

Objetivo

O objetivo desta experiência é mostrar ao aluno um método simples para implementação de um teclado matricial.

Descrição

O teclado matricial do MCMASTER é composto por 4 linhas e 4 colunas formando assim uma matriz de 16 teclas. Desta forma, utilizam-se 8 pinos do microcontrolador para realizar a leitura do teclado.

Os conceitos adotados na experiência de varredura de displays de leds não são muito diferentes dos adotados para varrer os estados de um teclado matricial.

Analisando o esquema elétrico nota-se que todas as teclas de uma mesma coluna estão interligadas. Além disso, nota-se um resistor de pull-down em cada uma das vias. Veja também que todas as teclas de uma mesma linha também encontram-se interligadas. A idéia de varredura aplicada aqui é habilitar uma linha de cada vez e analisar se alguma tecla da linha habilitada está pressionada.

Para isso, deve-se configurar o microcontrolador com os pinos das linhas como saída e os pinos das colunas como entrada. Note que se todas as linhas estiverem em nível lógico 0, ou seja, se nenhuma linha estiver habilitada, ao ler o estado das colunas sempre será lido o valor 0, estando as teclas pressionadas ou não. Na verdade o microcontrolador estará lendo o estado dos resistores que no caso são de pull-down, ou seja, leitura em 0.

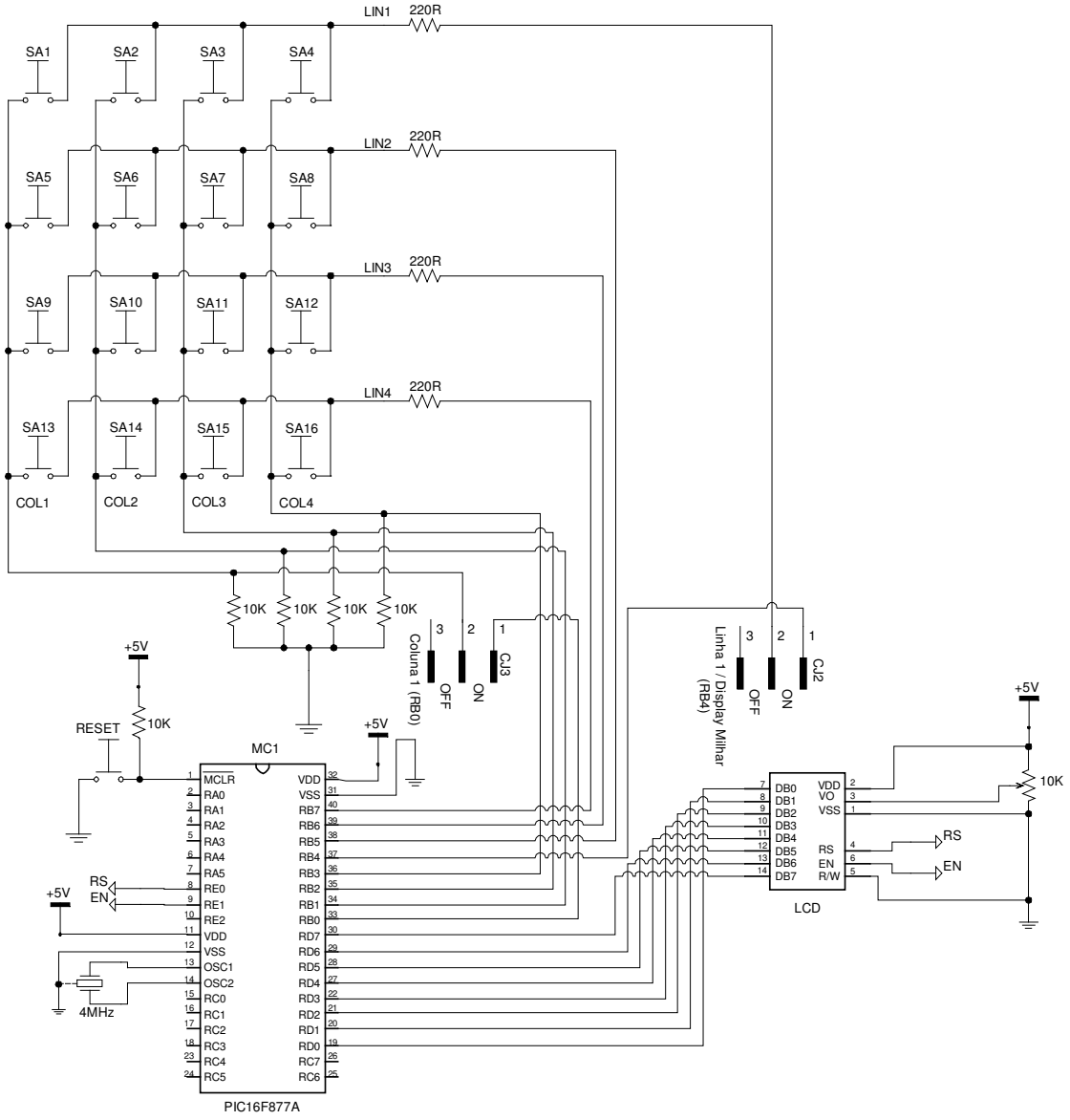
Porém, se habilitarmos uma das linhas (e apenas uma) colocando-a em nível lógico 1 e pressionarmos uma tecla dessa linha, ao lermos o estado das colunas encontraremos um bit em 1, sendo que a posição do bit em 1 sinalizará a coluna na qual a tecla foi pressionada. Como foi o próprio microcontrolador que habilitou a linha, o número da linha é conhecido e como a posição do bit em 1 define a coluna da tecla é fácil determinar a linha e coluna da tecla pressionada.

O conceito de varredura continua válido, pois apenas uma linha deve ser habilitada de cada vez e o microcontrolador deve ficar o tempo todo alterando (varrendo) a linha habilitada e lendo o estado das colunas. Enquanto nenhum bit das colunas valer 1, a varredura das linhas continua sendo executada. Ao encontrar uma coluna com tecla pressionada o software deve executar o filtro de debounce mantendo a linha atual habilitada até que o filtro de debounce seja finalizado.

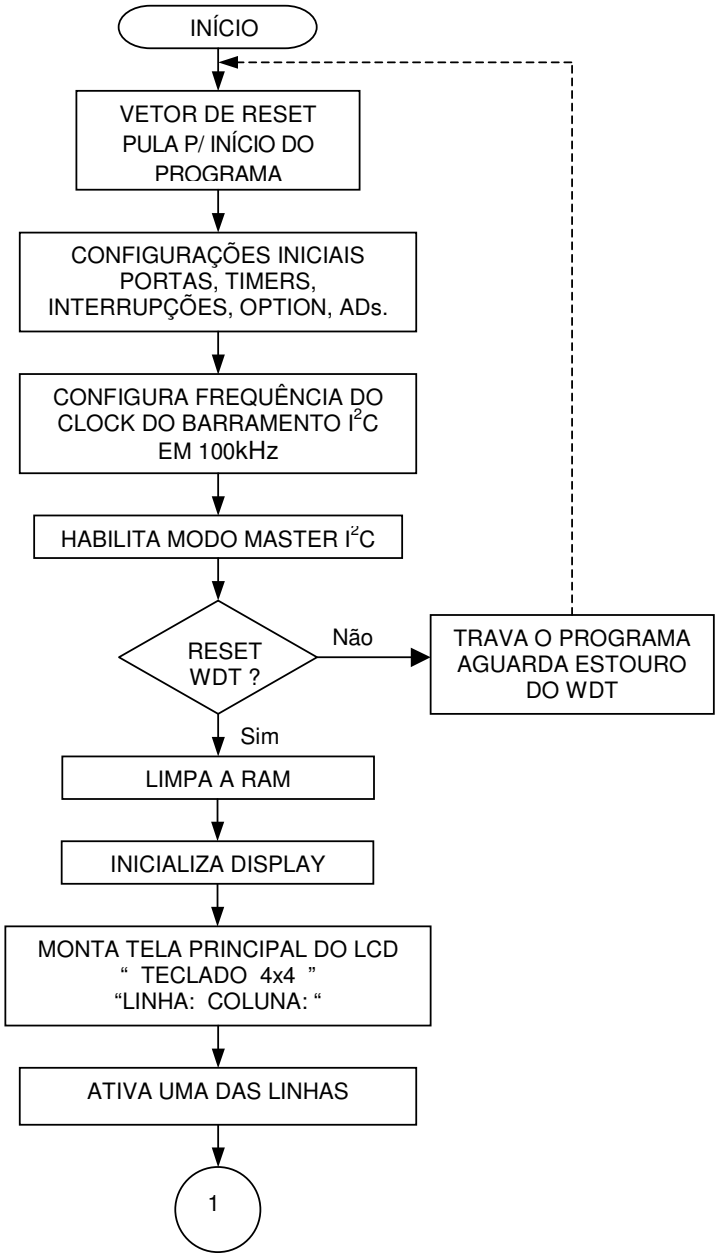
No fluxograma apresentado fica fácil de entender o conceito da varredura do teclado matricial.

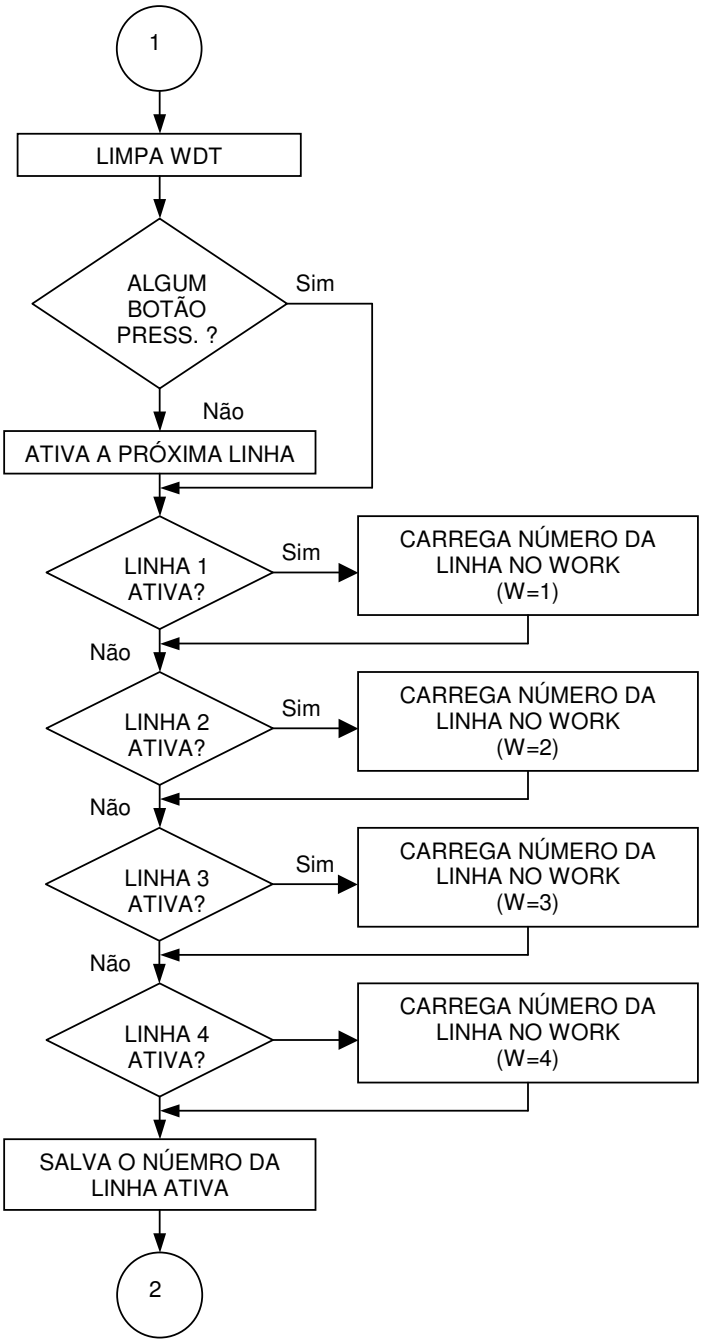
O exemplo da experiência analisa o teclado e caso alguma tecla seja pressionada, mostra a linha e coluna da mesma é mostrado no display LCD.

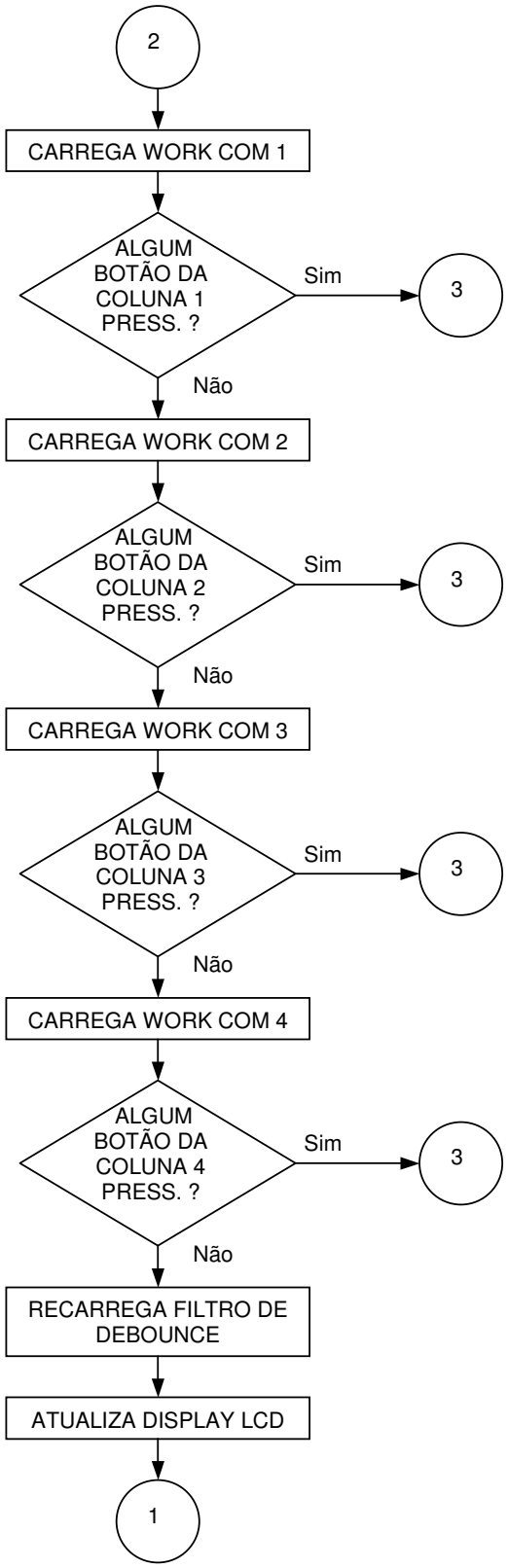
Esquema Elétrico

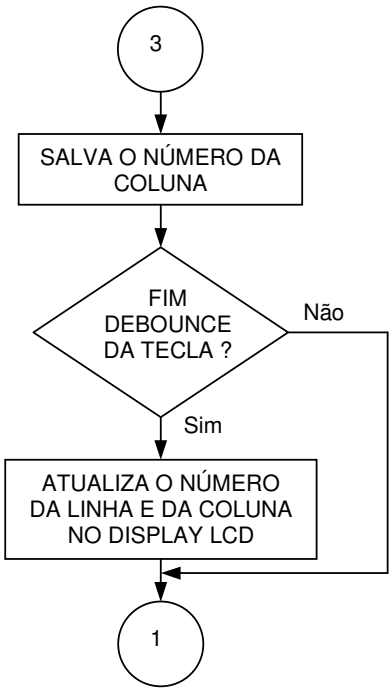


Fluxograma


INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURAÇÕES INICIAIS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION, ADs.
CONFIGURA FREQUÊNCIA DO
CLOCK DO BARRAMENTO I2C
EM 100kHz
HABILITA MODO MASTER I2C
RESET
WDT ?
Não
TRAVA O PROGRAMA
AGUARDA ESTOURO
DO WDT
Sim
LIMPA A RAM
INICIALIZA DISPLAY
MONTA TELA PRINCIPAL DO LCD
“ TECLADO 4x4 ”
“LINHA: COLUNA: “
ATIVA UMA DAS LINHAS
1


1
LIMPA WDT
ALGUM BOTÃO PRESS. ?
Sim
Não
ATIVA A PRÓXIMA LINHA
LINHA 1 ATIVA?
Sim
CARREGA NÚMERO DA LINHA NO WORK (W=1)
Não
LINHA 2 ATIVA?
Sim
CARREGA NÚMERO DA LINHA NO WORK (W=2)
Não
LINHA 3 ATIVA?
Sim
CARREGA NÚMERO DA LINHA NO WORK (W=3)
Não
LINHA 4 ATIVA?
CARREGA NÚMERO DA LINHA NO WORK (W=4)
SALVA O NÚEMRO DA LINHA ATIVA
2


2
CARREGA WORK COM 1
ALGUM BOTÃO DA COLUNA 1 PRESS. ?
Sim
3
Não
CARREGA WORK COM 2
ALGUM BOTÃO DA COLUNA 2 PRESS. ?
Sim
3
Não
CARREGA WORK COM 3
ALGUM BOTÃO DA COLUNA 3 PRESS. ?
Sim
3
Não
CARREGA WORK COM 4
ALGUM BOTÃO DA COLUNA 4 PRESS. ?
Sim
3
Não
RECARREGA FILTRO DE DEBOUNCE
ATUALIZA DISPLAY LCD
1


3
SALVA O NÚMERO DA COLUNA
FIM DEBOUNCE DA TECLA ?
Não
Sim
ATUALIZA O NÚMERO DA LINHA E DA COLUNA NO DISPLAY LCD
1

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *                 EXPERIÊNCIA 18 - TECLADO MATRICIAL 4x4                  *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            DESCRIÇÃO GERAL                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE EXEMPLO FOI ELABORADO PARA EXPLICAR O FUNCIONAMENTO DO TECLADO
;  MATRICIAL 4x4.
;  O NÚMERO DA LINHA E COLUNA DA TECLA PRESSIONADA E MOSTRADA NO LCD.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       CONFIGURAÇÕES PARA GRAVAÇÃO                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_ON & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           DEFINIÇÃO DAS VARIÁVEIS                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE BLOCO DE VARIÁVEIS ESTÁ LOCALIZADO LOGO NO INÍCIO DO BANCO 0

   CBLOCK  0X20                    ; POSIÇÃO INICIAL DA RAM

           TEMPO0
           TEMPO1                  ; TEMPORIZADORES P/ ROTINA DE DELAY

           FILTRO                  ; FILTRO PARA TECLAS

           NUM_LINHA               ; ARMAZENA O NÚMERO DA LINHA ATIVADA

           NUM_COLUNA              ; ARMAZENA O NÚMERO DA COLUNA

           LINHA_ATIVA             ; REGISTRADOR AUXILAR PARA ATIVAR
                                   ; AS LINHAS DO TECLADO 4x4
   ENDC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>         ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     DEFINIÇÃO DOS BANCOS DE RAM                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          CONSTANTES INTERNAS                            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE CONSTANTES FACILITA A PROGRAMAÇÃO E A MANUTENÇÃO.

;  ESTE PROGRAMA NÃO UTILIZA NENHUMA CONSTANTE.
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   DECLARAÇÃO DOS FLAGs DE SOFTWARE                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE FLAGs AJUDA NA PROGRAMAÇÃO E ECONOMIZA MEMÓRIA RAM.

#DEFINE    LINHA1  LINHA_ATIVA,4   ; BIT 4 DO REGISTRADOR LINHA_ATIVA
                                   ; REPRESENTA A LINHA 1 DO TECLADO 4x4
                                   ; 1 -> LINHA ESTÁ ATIVADA
                                   ; 0 -> LINHA ESTÁ DESATIVADA

#DEFINE    LINHA2  LINHA_ATIVA,5   ; BIT 5 DO REGISTRADOR LINHA_ATIVA
                                   ; REPRESENTA A LINHA 2 DO TECLADO 4x4
                                   ; 1 -> LINHA ESTÁ ATIVADA
                                   ; 0 -> LINHA ESTÁ DESATIVADA

#DEFINE    LINHA3  LINHA_ATIVA,6   ; BIT 6 DO REGISTRADOR LINHA_ATIVA
                                   ; REPRESENTA A LINHA 3 DO TECLADO 4x4
                                   ; 1 -> LINHA ESTÁ ATIVADA
                                   ; 0 -> LINHA ESTÁ DESATIVADA

#DEFINE    LINHA4  LINHA_ATIVA,7   ; BIT 7 DO REGISTRADOR LINHA_ATIVA
                                   ; REPRESENTA A LINHA 4 DO TECLADO 4x4
                                   ; 1 -> LINHA ESTÁ ATIVADA
                                   ; 0 -> LINHA ESTÁ DESATIVADA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                ENTRADAS                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS ENTRADAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    COLUNA1 PORTB,0 ; PINO DE ENTRADA DA COLUNA 1
                           ; 1 -> ALGUMA TECLA DESTA COLUNA ESTÁ PRESSIONADA
                           ; 0 -> NENHUMA TECLA DESTA COLUNA ESTÁ PRESSIONADA

#DEFINE    COLUNA2 PORTB,1 ; PINO DE ENTRADA DA COLUNA 2
                           ; 1 -> ALGUMA TECLA DESTA COLUNA ESTÁ PRESSIONADA
                           ; 0 -> NENHUMA TECLA DESTA COLUNA ESTÁ PRESSIONADA

#DEFINE    COLUNA3 PORTB,2 ; PINO DE ENTRADA DA COLUNA 3
                           ; 1 -> ALGUMA TECLA DESTA COLUNA ESTÁ PRESSIONADA
                           ; 0 -> NENHUMA TECLA DESTA COLUNA ESTÁ PRESSIONADA

#DEFINE    COLUNA4 PORTB,3 ; PINO DE ENTRADA DA COLUNA 4
                           ; 1 -> ALGUMA TECLA DESTA COLUNA ESTÁ PRESSIONADA
                           ; 0 -> NENHUMA TECLA DESTA COLUNA ESTÁ PRESSIONADA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                 SAÍDAS                                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS SAÍDAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    DISPLAY         PORTD   ; BARRAMENTO DE DADOS DO DISPLAY

#DEFINE    RS              PORTE,0 ; INDICA P/ O DISPLAY UM DADO OU COMANDO
                                   ; 1 -> DADO
                                   ; 0 -> COMANDO

#DEFINE    ENABLE          PORTE,1 ; SINAL DE ENABLE P/ DISPLAY
                                   ; ATIVO NA BORDA DE DESCIDA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   VETOR DE RESET DO MICROCONTROLADOR                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO INICIAL PARA EXECUÇÃO DO PROGRAMA

   ORG     0X0000                  ; ENDEREÇO DO VETOR DE RESET
   GOTO    CONFIG                  ; PULA PARA CONFIG DEVIDO A REGIÃO
                                   ; DESTINADA AS ROTINAS SEGUINTES
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  ROTINA DE DELAY (DE 1MS ATÉ 256MS)                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA É UMA ROTINA DE DELAY VARIÁVEL, COM DURAÇÃO DE 1MS X O VALOR PASSADO
;  EM WORK (W).

DELAY_MS
   MOVWF   TEMPO1                  ; CARREGA TEMPO1 (UNIDADES DE MS)
   MOVLW   .250
   MOVWF   TEMPO0                  ; CARREGA TEMPO0 (P/ CONTAR 1MS)

   CLRWDT                          ; LIMPA WDT (PERDE TEMPO)
   DECFSZ  TEMPO0,F                ; FIM DE TEMPO0 ?
   GOTO    $-2                     ; NÃO - VOLTA 2 INSTRUÇÕES
                                   ; SIM - PASSOU-SE 1MS
   DECFSZ  TEMPO1,F                ; FIM DE TEMPO1 ?
   GOTO    $-6                     ; NÃO - VOLTA 6 INSTRUÇÕES
                                   ; SIM
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DE UM CARACTER NO DISPLAY              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ENVIA UM CARACTER PARA O MÓDULO DE LCD. O CARACTER A SER
;  ESCRITO DEVE SER COLOCADO EM WORK (W) ANTES DE CHAMAR A ROTINA.

ESCREVE
   MOVWF   DISPLAY                 ; ATUALIZA DISPLAY (PORTD)
   NOP                             ; PERDE 1US PARA ESTABILIZAÇÃO
   BSF     ENABLE                  ; ENVIA UM PULSO DE ENABLE AO DISPLAY
   GOTO    $+1                     ; .
   BCF     ENABLE                  ; .

   MOVLW   .1
   CALL    DELAY_MS                ; DELAY DE 1MS
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               CONFIGURAÇÕES INICIAIS DE HARDWARE E SOFTWARE           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  NESTA ROTINA SÃO INICIALIZADAS AS PORTAS DE I/O DO MICROCONTROLADOR E AS
;  CONFIGURAÇÕES DOS REGISTRADORES ESPECIAIS (SFR). A ROTINA INICIALIZA A
;  MÁQUINA E AGUARDA O ESTOURO DO WDT.

CONFIG
   CLRF    PORTA                   ; LIMPA O PORTA
   CLRF    PORTB                   ; LIMPA O PORTB
   CLRF    PORTC                   ; LIMPA O PORTC
   CLRF    PORTD                   ; LIMPA O PORTD
   CLRF    PORTE                   ; LIMPA O PORTE


   BANK1                           ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                   ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                   ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011000'
   MOVWF   TRISC                   ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                   ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                   ; CONFIGURA I/O DO PORTE

   MOVLW   B'11011111'
   MOVWF   OPTION_REG              ; CONFIGURA OPTIONS
                                   ; PULL-UPs DESABILITADOS
                                   ; INTER. NA BORDA DE SUBIDA DO RB0
```

```
                                ; TIMER0 INCREM. PELO CICLO DE MÁQUINA
                                ; WDT   - 1:128
                                ; TIMER - 1:1

   MOVLW   B'00000000'
   MOVWF   INTCON               ; CONFIGURA INTERRUPÇÕES
                                ; DESABILITADA TODAS AS INTERRUPÇÕES
   MOVLW   B'00000111'
   MOVWF   CMCON                ; DESLIGA OS COMPARADORES

   MOVLW   B'00000111'
   MOVWF   ADCON1               ; CONFIGURA CONVERSOR A/D
                                ; CONFIGURA PORTA E PORTE COM I/O DIGITAL

   BANK0                        ; SELECIONA BANCO 0 DA RAM

;  AS INSTRUÇÕES A SEGUIR FAZEM COM QUE O PROGRAMA TRAVE QUANDO HOUVER UM
;  RESET OU POWER-UP, MAS PASSE DIRETO SE O RESET FOR POR WDT. DESTA FORMA,
;  SEMPRE QUE O PIC É LIGADO, O PROGRAMA TRAVA, AGUARDA UM ESTOURO DE WDT
;  E COMEÇA NOVAMENTE. ISTO EVITA PROBLEMAS NO START-UP DO PIC.

   BTFSC   STATUS,NOT_TO        ; RESET POR ESTOURO DE WATCHDOG TIMER ?
   GOTO    $                    ; NÃO - AGUARDA ESTOURO DO WDT
                                ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        INICIALIZAÇÃO DA RAM                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA IRÁ LIMPAR TODA A RAM DO BANCO 0, INDO DE 0X20 A 0X7F

   MOVLW   0X20
   MOVWF   FSR                  ; APONTA O ENDEREÇAMENTO INDIRETO PARA
                                ; A PRIMEIRA POSIÇÃO DA RAM
LIMPA_RAM
   CLRF    INDF                 ; LIMPA A POSIÇÃO
   INCF    FSR,F                ; INCREMENTA O PONTEIRO P/ A PRÓX. POS.
   MOVF    FSR,W
   XORLW   0X80                 ; COMPARA O PONTEIRO COM A ÚLT. POS. +1
   BTFSS   STATUS,Z             ; JÁ LIMPOU TODAS AS POSIÇÕES?
   GOTO    LIMPA_RAM            ; NÃO - LIMPA A PRÓXIMA POSIÇÃO
                                ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   CONFIGURAÇÕES INICIAIS DO DISPLAY                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA INICIALIZA O DISPLAY P/ COMUNICAÇÃO DE 8 VIAS, DISPLAY PARA 2
; LINHAS, CURSOR APAGADO E DESLOCAMENTO DO CURSOR À DIREITA.

INICIALIZACAO_DISPLAY
   BCF     RS                   ; SELECIONA O DISPLAY P/ COMANDOS

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO

   MOVLW   .3
   CALL    DELAY_MS             ; DELAY DE 3MS (EXIGIDO PELO DISPLAY)

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO

   MOVLW   0X30                 ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE              ; INICIALIZAÇÃO

   MOVLW   B'00111000'          ; ESCREVE COMANDO PARA
   CALL    ESCREVE              ; INTERFACE DE 8 VIAS DE DADOS

   MOVLW   B'00000001'          ; ESCREVE COMANDO PARA
   CALL    ESCREVE              ; LIMPAR TODO O DISPLAY

   MOVLW   .1
   CALL    DELAY_MS             ; DELAY DE 1MS
```

```
   MOVLW   B'00001100'            ; ESCREVE COMANDO PARA
   CALL    ESCREVE                ; LIGAR O DISPLAY SEM CURSOR

   MOVLW   B'00000110'            ; ESCREVE COMANDO PARA INCREM.
   CALL    ESCREVE                ; AUTOMÁTICO À DIREITA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                ROTINA DE ESCRITA DA TELA PRINCIPAL                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A TELA PRINCIPAL DO PROGRAMA, COM AS FRASES:
;  LINHA 1 - "  Teclado  4x4  "
;  LINHA 2 - "Linha:  Coluna: "

   MOVLW   0X82                   ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                ; LINHA 0 / COLUNA 2
   BSF     RS                     ; SELECIONA O DISPLAY P/ DADOS

                                  ; COMANDOS PARA ESCREVER AS
                                  ; LETRAS DE "Teclado  4x4"
   MOVLW   'T'
   CALL    ESCREVE
   MOVLW   'e'
   CALL    ESCREVE
   MOVLW   'c'
   CALL    ESCREVE
   MOVLW   'l'
   CALL    ESCREVE
   MOVLW   'a'
   CALL    ESCREVE
   MOVLW   'd'
   CALL    ESCREVE
   MOVLW   'o'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   '4'
   CALL    ESCREVE
   MOVLW   'x'
   CALL    ESCREVE
   MOVLW   '4'
   CALL    ESCREVE

   BCF     RS                     ; SELECIONA O DISPLAY P/ COMANDO
   MOVLW   0XC0                   ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                ; LINHA 1 / COLUNA 0
   BSF     RS                     ; SELECIONA O DISPLAY P/ DADOS

                                  ; COMANDOS PARA ESCREVER AS
                                  ; LETRAS DE "Linha:  Coluna: "
   MOVLW   'L'
   CALL    ESCREVE
   MOVLW   'i'
   CALL    ESCREVE
   MOVLW   'n'
   CALL    ESCREVE
   MOVLW   'h'
   CALL    ESCREVE
   MOVLW   'a'
   CALL    ESCREVE
   MOVLW   ':'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   'C'
   CALL    ESCREVE
   MOVLW   'o'
   CALL    ESCREVE
   MOVLW   'l'
```

```
   CALL    ESCREVE
   MOVLW   'u'
   CALL    ESCREVE
   MOVLW   'n'
   CALL    ESCREVE
   MOVLW   'a'
   CALL    ESCREVE
   MOVLW   ':'
   CALL    ESCREVE

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              LOOP PRINCIPAL                               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *


   MOVLW   B'00010000'
   MOVWF   LINHA_ATIVA            ; INICIALIZA REGISTRADOR AUXILIAR
                                  ; PARA ATIVAR LINHAS

LOOP
   CLRWDT                         ; LIMPA WATCHDOG TIMER

   MOVLW   .100
   XORWF   FILTRO,W
   BTFSS   STATUS,Z               ; FILTRO = 100 ? (NENHUMA TECLA PRESSIONADA?)
   GOTO    MANTEM_LINHA_ATUAL     ; NÃO - PULA E MANTEM A LINHA ATUAL ATIVA
                                  ; SIM - ATIVA PRÓXIMA LINHA
ATIVA_PROXIMA_LINHA
   BCF     STATUS,C
   RRF     LINHA_ATIVA,F          ; RODA REGISTRADOR LINHA_ATIVA

   MOVLW   B'10000000'            ; CARREGA W COM VALOR INICIAL DE LINHA_ATIVA
   BTFSC   LINHA_ATIVA,3          ; VARREU TODAS AS LINHAS ?
   MOVWF   LINHA_ATIVA            ; SIM - RECARREGA REGISTRADOR COM VALOR DO W
                                  ; NÃO
   MOVF    LINHA_ATIVA,W
   MOVWF   PORTB                  ; CARREGA PORTB COM VALOR DE LINHA_ATIVA
                                  ; ATIVA A LINHA CORRESPONDENTE

MANTEM_LINHA_ATUAL
   BTFSC   LINHA4                 ; A LINHA 4 ESTÁ ATIVADA ?
   MOVLW   .4                     ; SIM - CARREGA W COM 4
                                  ; NÃO
   BTFSC   LINHA3                 ; A LINHA 3 ESTÁ ATIVADA ?
   MOVLW   .3                     ; SIM - CARREGA W COM 3
                                  ; NÃO
   BTFSC   LINHA2                 ; A LINHA 2 ESTÁ ATIVADA ?
   MOVLW   .2                     ; SIM - CARREGA W COM 2
                                  ; NÃO
   BTFSC   LINHA1                 ; A LINHA 1 ESTÁ ATIVADA ?
   MOVLW   .1                     ; SIM - CARREGA W COM 1
                                  ; NÃO
   MOVWF   NUM_LINHA              ; SALVA O NÚMERO DA LINHA EM QUE EXISTE
                                  ; ALGUM TECLA PRESSIONADA

   MOVLW   .1                     ; CARREGA W COM 1
   BTFSC   COLUNA1                ; ALGUM BOTÃO DA COLUNA 1 PRESSIONADO ?
   GOTO    TRATA_COLUNAS          ; SIM - PULA P/ TRATA_COLUNAS
                                  ; NÃO
   MOVLW   .2                     ; CARREGA W COM 2
   BTFSC   COLUNA2                ; ALGUM BOTÃO DA COLUNA 2 PRESSIONADO ?
   GOTO    TRATA_COLUNAS          ; SIM - PULA P/ TRATA_COLUNAS
                                  ; NÃO
   MOVLW   .3                     ; CARREGA W COM 3
   BTFSC   COLUNA3                ; ALGUM BOTÃO DA COLUNA 3 PRESSIONADO ?
   GOTO    TRATA_COLUNAS          ; SIM - PULA P/ TRATA_COLUNAS
                                  ; NÃO
   MOVLW   .4                     ; CARREGA W COM 4
   BTFSC   COLUNA4                ; ALGUM BOTÃO DA COLUNA 4 PRESSIONADO ?
   GOTO    TRATA_COLUNAS          ; SIM - PULA P/ TRATA_COLUNAS
                                  ; NÃO
                                  ; NENHUMA TECLA ESTÁ PRESSIONADA
```

```
   MOVLW   .100
   MOVWF   FILTRO                  ; RECARREGA FILTRO

   CLRF    PORTB                   ; DESATIVA TODOS OS DISPLAYS DE 7 SEGMENTOS

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XC6
   CALL    ESCREVE                 ; POSICIONA O CURSOR
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS
   MOVLW   ' '
   CALL    ESCREVE                 ; APAGA O CARACTER DO NÚMERO DA LINHA
   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XCF
   CALL    ESCREVE                 ; POSICIONA O CURSOR
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS
   MOVLW   ' '
   CALL    ESCREVE                 ; APAGA O CARACTER DO NÚMERO DA LINHA

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   MOVF    LINHA_ATIVA,W
   MOVWF   PORTB                   ; CARREGA PORTB COM VALOR DE LINHA_ATIVA
                                   ; ATIVA A LINHA CORRESPONDENTE

   GOTO    LOOP                    ; VOLTA AO LOOP

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *             ROTINA PARA TRATAMENTO DAS COLUNAS DO TECLADO 4x4            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

TRATA_COLUNAS
   MOVWF   NUM_COLUNA              ; SALVA O NÚMERO DA COLUNA EM QUE ALGUMA
                                   ; TECLA ESTÁ PRESSIONADA

   MOVF    FILTRO,F
   BTFSC   STATUS,Z                ; TECLA JÁ FOI TRATADA ?
   GOTO    $+3                     ; SIM - PULA FILTRO DE DEBOUNCE
                                   ; NÃO
   DECFSZ  FILTRO,F                ; FIM DO FILTRO ?
   GOTO    LOOP                    ; NÃO - VOLTA AO LOOP
                                   ; SIM

   CLRF    PORTB                   ; DESATIVA TODOS OS DISPLAYS DE 7 SEGMENTOS

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XC6
   CALL    ESCREVE                 ; POSICIONA O CURSOR
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS
   MOVLW   0X30
   ADDWF   NUM_LINHA,W             ; CARREGA W COM NÚMERO DA LINHA (EM ASCII)
   CALL    ESCREVE                 ; ENVIA NÚMERO DA COLUNA AO LCD
   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XCF
   CALL    ESCREVE                 ; POSICIONA O CURSOR
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS
   MOVLW   0X30
   ADDWF   NUM_COLUNA,W            ; CARREGA W COM NÚMERO DA COLUNA (EM ASCII)
   CALL    ESCREVE                 ; ENVIA NÚMERO DA COLUNA AO LCD

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   MOVF    LINHA_ATIVA,W
   MOVWF   PORTB                   ; CARREGA PORTB COM VALOR DE LINHA_ATIVA
                                   ; ATIVA A LINHA CORRESPONDENTE

   GOTO    LOOP                    ; VOLTA AO LOOP

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          FIM DO PROGRAMA                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                             ; FIM DO PROGRAMA
```

Dicas e Comentários

Notar que ao habilitar uma linha do teclado matricial também se habilita um dos displays de 7 segmentos, ou seja, foram utilizados os mesmos pinos do microcontrolador para habilitar as linhas e os displays. Esta é uma forma econômica de varrer os displays e ao mesmo tempo varrer o teclado. Veja que com apenas 16 I/Os do microcontrolador foi possível ligar 4 displays de 7 segmentos e 16 teclas. Foram utilizados 8 I/Os para segmentos mais o ponto decimal, 4 I/Os de seleção (linhas e displays) e 4 I/Os de colunas.

Exercícios Propostos

1. Alterar o mapeamento do software invertendo a numeração das linhas e colunas.
2. Mostrar no display LCD o número da tecla pressionada e não o número da linha e da coluna.
3. Sem utilizar o display LCD, fazer um software que utilize a varredura do teclado e dos displays ao mesmo tempo, mostrando o número da tecla pressionada nos displays.

Capítulo 21 - Experiência 19 – Relógio de tempo real (RTC)

Objetivo

O objetivo desta experiência é mostrar como utilizar o relógio de tempo real (RTC).

Descrição

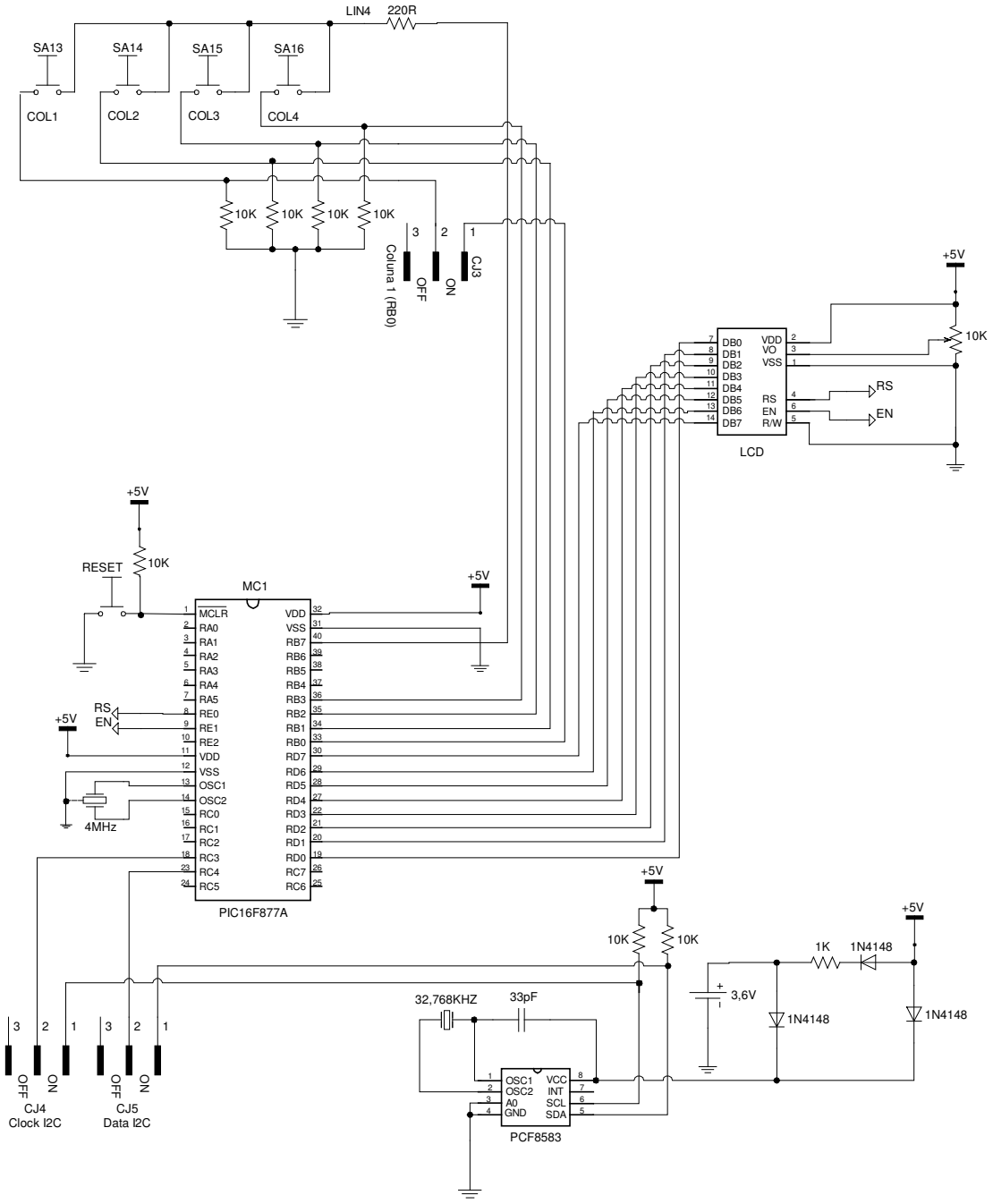
No MCMASTER optou-se pela utilização do relógio de tempo real (RTC) modelo PCF8583P da Philips. O protocolo de comunicação é do tipo I^2C e para maiores informações a respeito do componente deve-se consultar o data sheet disponível no CD.

As rotinas de acesso ao relógio seguem o padrão adotado na experiência 16 (Master I^2C) com apenas algumas modificações. Na realidade as alterações são que o relógio está mapeado no endereço 0h (000b) para evitar conflitos com a memória e ao invés de serem utilizados dois bytes para compor o endereço foi utilizado apenas um o que já é suficiente.

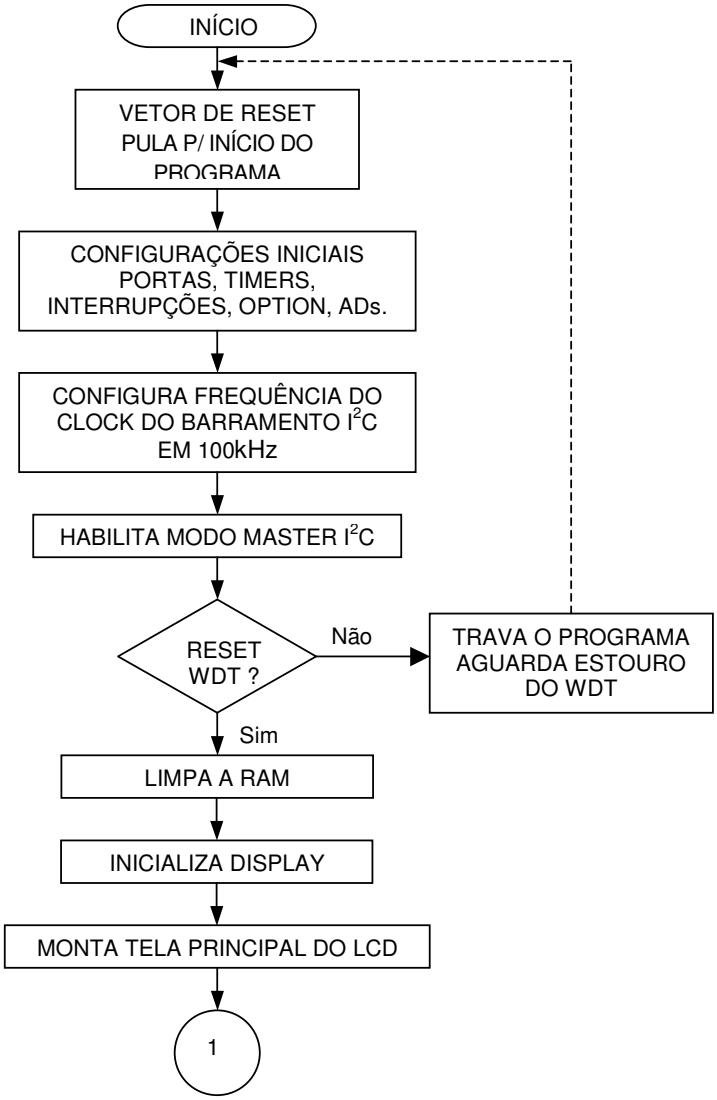
O software da experiência apenas lê a hora atual do relógio e mostra o resultado no LCD.

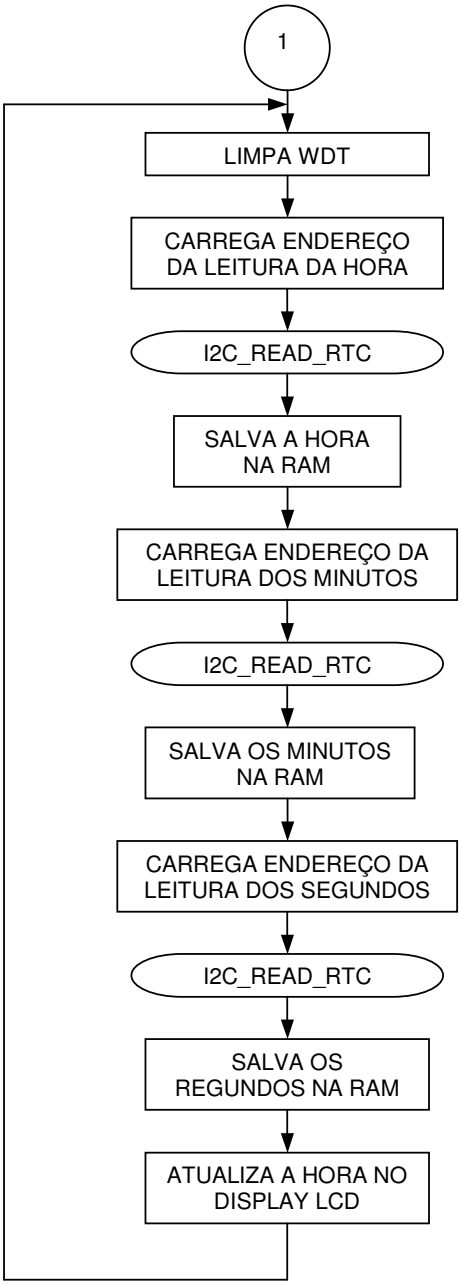
Vale lembrar que o relógio é completo, ou seja, dispõe de hora, minuto, segundo, dia, mês e ano, inclusive bissexto.

Esquema Elétrico

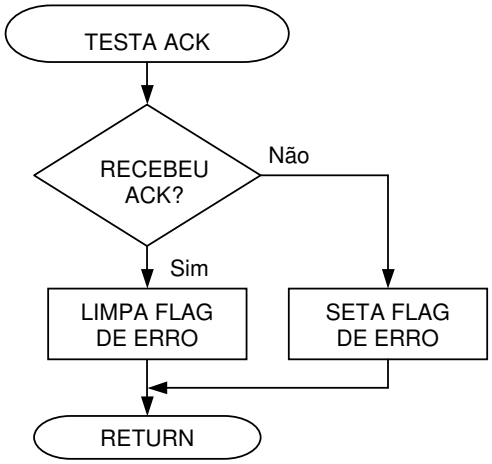


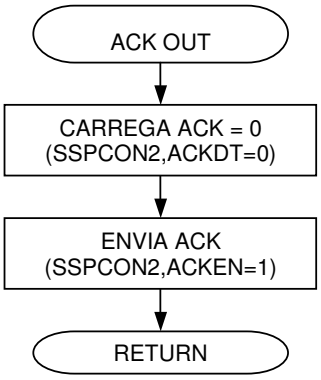
Fluxograma

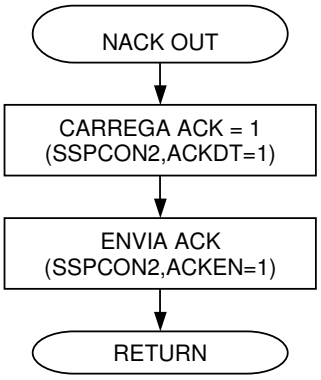

INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURAÇÕES INICIAIS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION, ADs.
CONFIGURA FREQUÊNCIA DO
CLOCK DO BARRAMENTO I2C
EM 100kHz
HABILITA MODO MASTER I2C
RESET
WDT ?
Não
TRAVA O PROGRAMA
AGUARDA ESTOURO
DO WDT
Sim
LIMPA A RAM
INICIALIZA DISPLAY
MONTA TELA PRINCIPAL DO LCD
1

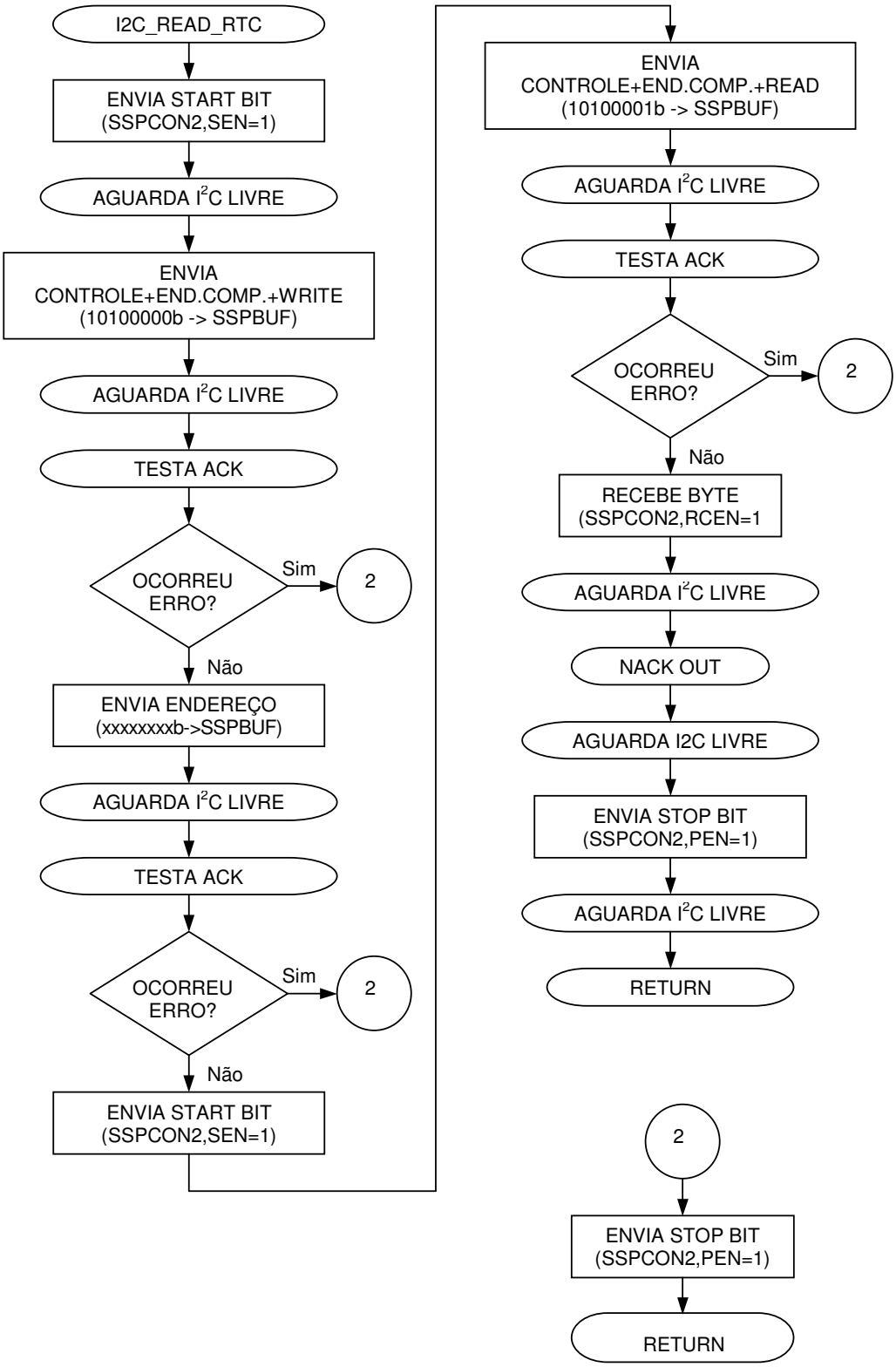

1
LIMPA WDT
CARREGA ENDEREÇO DA LEITURA DA HORA
I2C_READ_RTC
SALVA A HORA NA RAM
CARREGA ENDEREÇO DA LEITURA DOS MINUTOS
I2C_READ_RTC
SALVA OS MINUTOS NA RAM
CARREGA ENDEREÇO DA LEITURA DOS SEGUNDOS
I2C_READ_RTC
SALVA OS REGUNDOS NA RAM
ATUALIZA A HORA NO DISPLAY LCD


AGUARDA I2C LIVRE
EVENTO I2C EM ANDAMENTO?
Sim
Não
TODOS OS EVENTO I2C FINALIZADOS?
Não
RETURN


TESTA ACK
RECEBEU ACK?
Não
Sim
LIMPA FLAG DE ERRO
SETA FLAG DE ERRO
RETURN


ACK OUT
CARREGA ACK = 0 (SSPCON2,ACKDT=0)
ENVIA ACK (SSPCON2,ACKEN=1)
RETURN


NACK OUT
CARREGA ACK = 1 (SSPCON2,ACKDT=1)
ENVIA ACK (SSPCON2,ACKEN=1)
RETURN


I2C_READ_RTC
ENVIA START BIT (SSPCON2,SEN=1)
AGUARDA I2C LIVRE
ENVIA CONTROLE+END.COMP.+WRITE (10100000b -> SSPBUF)
AGUARDA I2C LIVRE
TESTA ACK
OCORREU ERRO?
Sim
2
Não
ENVIA ENDEREÇO (xxxxxxxxb->SSPBUF)
AGUARDA I2C LIVRE
TESTA ACK
OCORREU ERRO?
Sim
2
Não
ENVIA START BIT (SSPCON2,SEN=1)
ENVIA CONTROLE+END.COMP.+READ (10100001b -> SSPBUF)
AGUARDA I2C LIVRE
TESTA ACK
OCORREU ERRO?
Sim
2
Não
RECEBE BYTE (SSPCON2,RCEN=1
AGUARDA I2C LIVRE
NACK OUT
AGUARDA I2C LIVRE
ENVIA STOP BIT (SSPCON2,PEN=1)
AGUARDA I2C LIVRE
RETURN
2
ENVIA STOP BIT (SSPCON2,PEN=1)
RETURN

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *               EXPERIÊNCIA 19 - RELÓGIO DE TEMPO REAL (RTC)              *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            DESCRIÇÃO GERAL                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTE EXEMPLO FOI ELABORADO PARA EXPLICAR O FUNCIONAMENTO DO RELÓGIO RTC
; COM COMUNICAÇÃO SERIAL, UTILIZANDO A MASTER I2C DO MICROCONTROLADOR.
; A HORA ATUAL DO RELÓGIO É MOSTRADA NO DISPLAY LCD.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       CONFIGURAÇÕES PARA GRAVAÇÃO                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_ON & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           DEFINIÇÃO DAS VARIÁVEIS                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE BLOCO DE VARIÁVEIS ESTÁ LOCALIZADO LOGO NO INÍCIO DO BANCO 0

   CBLOCK  0X20                  ; POSIÇÃO INICIAL DA RAM

           TEMPO1
           TEMPO0                ; CONTADORES P/ DELAY

           FLAG                  ; REGISTRADOR DE FLAG DE USO GERAL

           ENDERECO              ; REGISTRADORES DE ENDEREÇO PARA
                                 ; ACESSO AO RELOGIO RTC

           BUFFER                ; REGISTRADOR PARA LEITURA/GRAVAÇÃO
                                 ; UTILIZADO PELA COMUNICAÇÃO I2C

           HORA                  ; ARMAZENA A HORA ATUAL
           MINUTO                ; ARMAZENA O MINUTO ATUAL
           SEGUNDO               ; ARMAZENA O SEGUNDO ATUAL

   ENDC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>       ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     DEFINIÇÃO DOS BANCOS DE RAM                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          CONSTANTES INTERNAS                            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
;  A DEFINIÇÃO DE CONSTANTES FACILITA A PROGRAMAÇÃO E A MANUTENÇÃO.

#DEFINE    END_HORA        .4      ; ENDEREÇO DA HORA NO RELOGIO RTC

#DEFINE    END_MINUTO      .3      ; ENDEREÇO DO MINUTO NO RELOGIO RTC

#DEFINE    END_SEGUNDO     .2      ; ENDEREÇO DO SEGUNDO NO RELOGIO RTC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   DECLARAÇÃO DOS FLAGs DE SOFTWARE                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE FLAGs AJUDA NA PROGRAMAÇÃO E ECONOMIZA MEMÓRIA RAM.

#DEFINE    F_ERRO          FLAG,0  ; 1 --> ERRO NA LEITURA DO RELÓGIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               ENTRADAS                                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS ENTRADAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.


; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                SAÍDAS                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS SAÍDAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    DISPLAY         PORTD   ; BARRAMENTO DE DADOS DO DISPLAY

#DEFINE    RS              PORTE,0 ; INDICA P/ O DISPLAY UM DADO OU COMANDO
                                   ; 1 -> DADO
                                   ; 0 -> COMANDO

#DEFINE    ENABLE          PORTE,1 ; SINAL DE ENABLE P/ DISPLAY
                                   ; ATIVO NA BORDA DE DESCIDA

#DEFINE    SCL             PORTC,3 ; VIA DE CLOCK DO RELÓGIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           ENTRADAS/SAÍDAS                               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

#DEFINE    SDA             PORTC,4         ; VIA DE DADOS BIDIRECIONAL DO RELÓGIO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  VETOR DE RESET DO MICROCONTROLADOR                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO INICIAL PARA EXECUÇÃO DO PROGRAMA

   ORG     0X0000                  ; ENDEREÇO DO VETOR DE RESET
   GOTO    CONFIG                  ; PULA PARA CONFIG DEVIDO A REGIÃO
                                   ; DESTINADA AS ROTINAS SEGUINTES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  ROTINA DE DELAY (DE 1MS ATÉ 256MS)                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA É UMA ROTINA DE DELAY VARIÁVEL, COM DURAÇÃO DE 1MS X O VALOR PASSADO
;  EM WORK (W).

DELAY_MS
   MOVWF   TEMPO1                  ; CARREGA TEMPO1 (UNIDADES DE MS)
   MOVLW   .250
   MOVWF   TEMPO0                  ; CARREGA TEMPO0 (P/ CONTAR 1MS)

   CLRWDT                          ; LIMPA WDT (PERDE TEMPO)
   DECFSZ  TEMPO0,F                ; FIM DE TEMPO0 ?
   GOTO    $-2                     ; NÃO - VOLTA 2 INSTRUÇÕES
                                   ; SIM - PASSOU-SE 1MS
   DECFSZ  TEMPO1,F                ; FIM DE TEMPO1 ?
   GOTO    $-6                     ; NÃO - VOLTA 6 INSTRUÇÕES
                                   ; SIM
```

```
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DE UM CARACTER NO DISPLAY                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ENVIA UM CARACTER PARA O MÓDULO DE LCD. O CARACTER A SER
;  ESCRITO DEVE SER COLOCADO EM WORK (W) ANTES DE CHAMAR A ROTINA.

ESCREVE
   MOVWF   DISPLAY                 ; ATUALIZA DISPLAY (PORTD)
   NOP                             ; PERDE 1US PARA ESTABILIZAÇÃO
   BSF     ENABLE                  ; ENVIA UM PULSO DE ENABLE AO DISPLAY
   GOTO    $+1                     ; .
   BCF     ENABLE                  ; .

   MOVLW   .1
   CALL    DELAY_MS                ; DELAY DE 1MS
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               ROTINA DE CHECAGEM DE EVENTOS I2C LIBERADOS               *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA AGUARDA ATÉ QUE TODOS OS EVENTOS DA I2C ESTEJAM LIBERADOS.

AGUARDA_I2C_LIVRE
   BANK1                           ; ALTERA P/ BANK1
   BTFSC   SSPSTAT,R_W             ; ESTÁ OCORRENDO ALGUM EVENTO I2C?
   GOTO    $-1                     ; SIM, ESPERA TERMINAR
   MOVF    SSPCON2,W
   ANDLW   B'00011111'             ; MASCARA SSPCON2 (ATUALIZA FLAG ZERO)
   BTFSS   STATUS,Z                ; BITS DE EVENTOS LIBERADOS?
   GOTO    $-3                     ; NÃO - AGUARDA
   BANK0                           ; SIM - VOLTA P/ BANK0
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                NACK OUT                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA ENVIA UM NACK OUT PARA O BARRAMENTO I2C.

NACK_OUT
   BANK1                           ; ALTERA P/ BANK1
   BSF     SSPCON2,ACKDT           ; CARREGA NACK
   BSF     SSPCON2,ACKEN           ; TRANSMITE
   BANK0                           ; VOLTA P/ BANK0
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                 ROTINA PARA TESTAR SE O ACK FOI RECEBIDO                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA TESTA O BIT DE ACK RECEBIDO NO REGISTRADOR SSPCON2. PARA
; FACILITAR O RESTANTE DO SOFTWARE, A ROTINA COPIA ESTE FLAG NO FLAG F_ERRO
; PRESENTE NO BANCO 0 DA RAM, POIS O REGISTRADOR SSPCON2 ENCONTRA-SE NO BANK1.

TESTA_ACK
   BANK1                           ; ALTERA P/ BANK1
   BTFSC   SSPCON2,ACKSTAT         ; RECEBEU ACK ?
   GOTO    RECEBEU_NACK            ; NÃO - SINALIZA ERRO
                                   ; SIM
   BANK0                           ; VOLTA P/ BANK0
   BCF     F_ERRO                  ; LIMPA FLAG DE ERRO
   RETURN                          ; RETORNA

RECEBEU_NACK
   BANK0                           ; VOLTA P/ BANK0
   BSF     F_ERRO                  ; SETA FLAG P/ INDICAR ERRO
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  LEITURA DO RELOGIO RTC VIA I2C                          *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA LÊ O RELOGIO RTC EXTERNO VIA COMUNICAÇÃO I2C.
```

```
; O ENDERECO DO RELOGIO A SER LIDO DEVE SER PREVIAMENTE ACERTADO NO
; REGISTRADOR ENDERECO. O VALOR LIDO É RETORNADO EM BUFFER.
; CASO ALGUM ERRO DE LEITURA OCORRA, A ROTINA DESVIA P/ I2C_ERRO.

I2C_READ_RTC
   BANK1                          ; ALTERA P/ BANK1
   BSF     SSPCON2,SEN            ; INICIA START BIT
   BANK0                          ; VOLTA P/ BANK0
   CALL    AGUARDA_I2C_LIVRE      ; AGUARDA FIM DO EVENTO

   MOVLW   B'10100000'            ; CARREGA BYTE DE CONTROLE
   MOVWF   SSPBUF                 ; TRANSMITE CONTROLE
   CALL    AGUARDA_I2C_LIVRE      ; AGUARDA FIM DO EVENTO
   CALL    TESTA_ACK              ; CHAMA ROTINA P/ TESTAR ACK
   BTFSC   F_ERRO                 ; OCORREU ERRO DE ACK ?
   GOTO    I2C_ERRO               ; SIM - PULA P/ I2C_ERRO
                                  ; NÃO
   MOVF    ENDERECO,W
   MOVWF   SSPBUF                 ; TRANSMITE ENDEREÇO
   CALL    AGUARDA_I2C_LIVRE      ; AGUARDA FIM DO EVENTO
   CALL    TESTA_ACK              ; CHAMA ROTINA P/ TESTAR ACK
   BTFSC   F_ERRO                 ; OCORREU ERRO DE ACK ?
   GOTO    I2C_ERRO               ; SIM - PULA P/ I2C_ERRO
                                  ; NÃO

   BANK1                          ; ALTERA P/ BANK1
   BSF     SSPCON2,RSEN           ; REINICIA START BIT
   BANK0                          ; VOLTA P/ BANK0
   CALL    AGUARDA_I2C_LIVRE      ; AGUARDA FIM DO EVENTO

   MOVLW   B'10100001'            ; CARREGA BYTE DE CONTROLE
   MOVWF   SSPBUF                 ; TRANSMITE CONTROLE
   CALL    AGUARDA_I2C_LIVRE      ; AGUARDA FIM DO EVENTO
   CALL    TESTA_ACK              ; CHAMA ROTINA P/ TESTAR ACK
   BTFSC   F_ERRO                 ; OCORREU ERRO DE ACK ?
   GOTO    I2C_ERRO               ; SIM - PULA P/ I2C_ERRO
                                  ; NÃO

   BANK1                          ; ALTERA P/ BANK1
   BSF     SSPCON2,RCEN           ; INICIA LEITURA DO BYTE
   BANK0                          ; VOLTA P/ BANK0
   CALL    AGUARDA_I2C_LIVRE      ; AGUARDA FIM DO EVENTO
   MOVF    SSPBUF,W
   MOVWF   BUFFER                 ; SALVA DADO EM BUFFER
   CALL    NACK_OUT               ; ENVIA NACK --> FIM
   CALL    AGUARDA_I2C_LIVRE      ; AGUARDA FIM DO EVENTO

   BANK1                          ; ALTERA P/ BANK1
   BSF     SSPCON2,PEN            ; INICIA STOP BIT
   BANK0                          ; VOLTA P/ BANK0
   CALL    AGUARDA_I2C_LIVRE      ; AGUARDA FIM DO EVENTO

   RETURN                         ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    ROTINA P/ SINALIZAR ERRO NA I2C                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA SOMENTE É EXECUTA CASO ALGUM ERRO DE LEITURA/GRAVAÇÃO OCORRA
; COM A MEMÓRIA SERIAL.
; A ROTINA ENVIA UM STOP BIT PARA FINALIZAR A COMUNICAÇÃO COM A MEMÓRIA
; SERIAL, ENVIA UMA MENSAGEM DE ERRO AO DISPLAY E APÓS 1s RETORNA À TELA
; PRINCIPAL.

I2C_ERRO
   BANK1                          ; ALTERA P/ BANK1
   BSF     SSPCON2,PEN            ; INICIA STOP BIT
   BANK0                          ; VOLTA P/ BANK0

   RETURN                         ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                CONFIGURAÇÕES INICIAIS DE HARDWARE E SOFTWARE           *
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  NESTA ROTINA SÃO INICIALIZADAS AS PORTAS DE I/O DO MICROCONTROLADOR E AS
;  CONFIGURAÇÕES DOS REGISTRADORES ESPECIAIS (SFR). A ROTINA INICIALIZA AS
;  VARIÁVEIS DE RAM E AGUARDA O ESTOURO DO WDT.

CONFIG
   CLRF    PORTA                  ; LIMPA O PORTA
   CLRF    PORTB                  ; LIMPA O PORTB
   CLRF    PORTC                  ; LIMPA O PORTC
   CLRF    PORTD                  ; LIMPA O PORTD
   CLRF    PORTE                  ; LIMPA O PORTE


   BANK1                          ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                  ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                  ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011000'
   MOVWF   TRISC                  ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                  ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                  ; CONFIGURA I/O DO PORTE

   MOVLW   B'11011111'
   MOVWF   OPTION_REG             ; CONFIGURA OPTIONS
                                  ; PULL-UPs DESABILITADOS
                                  ; INTER. NA BORDA DE SUBIDA DO RB0
                                  ; TIMER0 INCREM. PELO CICLO DE MÁQUINA
                                  ; WDT   - 1:128
                                  ; TIMER - 1:1

   MOVLW   B'00000000'
   MOVWF   INTCON                 ; CONFIGURA INTERRUPÇÕES
                                  ; DESABILITADA TODAS AS INTERRUPÇÕES

   MOVLW   B'00000111'
   MOVWF   ADCON1                 ; CONFIGURA CONVERSOR A/D
                                  ; CONFIGURA PORTA E PORTE COMO I/O DIGITAL

   MOVLW   B'00100111'
   MOVWF   SSPADD                 ; VELOCIDADE: 100KHz @ 4MHz

   MOVLW   B'10000000'
   MOVWF   SSPSTAT                ; DESABILITA SLEW-RATE CONTROL (100KHz)

   BANK0                          ; SELECIONA BANCO 0 DA RAM

   MOVLW   B'00101000'
   MOVWF   SSPCON                 ; HABILITA I2C - MASTER MODE
                                  ; CONFIGURA PINOS COMO DA I2C

;  AS INSTRUÇÕES A SEGUIR FAZEM COM QUE O PROGRAMA TRAVE QUANDO HOUVER UM
;  RESET OU POWER-UP, MAS PASSE DIRETO SE O RESET FOR POR WDT. DESTA FORMA,
;  SEMPRE QUE O PIC É LIGADO, O PROGRAMA TRAVA, AGUARDA UM ESTOURO DE WDT
;  E COMEÇA NOVAMENTE. ISTO EVITA PROBLEMAS NO START-UP DO PIC.

   BTFSC   STATUS,NOT_TO          ; RESET POR ESTOURO DE WATCHDOG TIMER ?
   GOTO    $                      ; NÃO - AGUARDA ESTOURO DO WDT
                                  ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      INICIALIZAÇÃO DA RAM                                    *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA IRÁ LIMPAR TODA A RAM DO BANCO 0, INDO DE 0X20 A 0X7F

   MOVLW   0X20
```

```
   MOVWF   FSR                     ; APONTA O ENDEREÇAMENTO INDIRETO PARA
                                   ; A PRIMEIRA POSIÇÃO DA RAM
LIMPA_RAM
   CLRF    INDF                    ; LIMPA A POSIÇÃO
   INCF    FSR,F                   ; INCREMENTA O PONTEIRO P/ A PRÓX. POS.
   MOVF    FSR,W
   XORLW   0X80                    ; COMPARA O PONTEIRO COM A ÚLT. POS. +1
   BTFSS   STATUS,Z                ; JÁ LIMPOU TODAS AS POSIÇÕES?
   GOTO    LIMPA_RAM               ; NÃO - LIMPA A PRÓXIMA POSIÇÃO
                                   ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                  CONFIGURAÇÕES INICIAIS DO DISPLAY                      *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA INICIALIZA O DISPLAY P/ COMUNICAÇÃO DE 8 VIAS, DISPLAY PARA 2
; LINHAS, CURSOR APAGADO E DESLOCAMENTO DO CURSOR À DIREITA.

INICIALIZACAO_DISPLAY
   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0X30                    ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   .3
   CALL    DELAY_MS                ; DELAY DE 3MS (EXIGIDO PELO DISPLAY)

   MOVLW   0X30                    ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   0X30                    ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   B'00111000'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; INTERFACE DE 8 VIAS DE DADOS

   MOVLW   B'00000001'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; LIMPAR TODO O DISPLAY

   MOVLW   .1
   CALL    DELAY_MS                ; DELAY DE 1MS

   MOVLW   B'00001100'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; LIGAR O DISPLAY SEM CURSOR

   MOVLW   B'00000110'             ; ESCREVE COMANDO PARA INCREM.
   CALL    ESCREVE                 ; AUTOMÁTICO À DIREITA


; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA LINHA 1 DO LCD                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A LINHA 1 DA TELA PRINCIPAL DO LCD, COM A FRASE:
;  LINHA 1 - "RELOGIO RTC"

ATUALIZA_TELA_LINHA_1
   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0X82                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 0 / COLUNA 2
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

                                   ; COMANDOS PARA ESCREVER AS
                                   ; LETRAS DE "RELOGIO RTC"
   MOVLW   'R'
   CALL    ESCREVE
   MOVLW   'E'
   CALL    ESCREVE
   MOVLW   'L'
   CALL    ESCREVE
   MOVLW   'O'
   CALL    ESCREVE
   MOVLW   'G'
   CALL    ESCREVE
   MOVLW   'I'
```

```
   CALL    ESCREVE
   MOVLW   'O'
   CALL    ESCREVE
   MOVLW   ' '
   CALL    ESCREVE
   MOVLW   'R'
   CALL    ESCREVE
   MOVLW   'T'
   CALL    ESCREVE
   MOVLW   'C'
   CALL    ESCREVE

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            PROGRAMA PRINCIPAL                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA LE O RELOGIO RTC EXTERNO E MOSTRA A HORA ATUAL NO LCD.

LOOP
   CLRWDT                          ; LIMPA WATCHDOG TIMER

LE_RELOGIO_RTC
   MOVLW   END_HORA
   MOVWF   ENDERECO                ; CARREGA ENDEREÇO DA HORA
   CALL    I2C_READ_RTC            ; CHAMA ROTINA DE LEITURA
   BTFSC   F_ERRO
   GOTO    $+3
   MOVF    BUFFER,W
   MOVWF   HORA                    ; ATUALIZA RAM COM O VALOR LIDO

   MOVLW   END_MINUTO
   MOVWF   ENDERECO                ; CARREGA ENDEREÇO DO MINUTO
   CALL    I2C_READ_RTC            ; CHAMA ROTINA DE LEITURA
   BTFSC   F_ERRO
   GOTO    $+3
   MOVF    BUFFER,W
   MOVWF   MINUTO                  ; ATUALIZA RAM COM O VALOR LIDO

   MOVLW   END_SEGUNDO
   MOVWF   ENDERECO                ; CARREGA ENDEREÇO DO SEGUNDO
   CALL    I2C_READ_RTC            ; CHAMA ROTINA DE LEITURA
   BTFSC   F_ERRO
   GOTO    $+3
   MOVF    BUFFER,W
   MOVWF   SEGUNDO                 ; ATUALIZA RAM COM O VALOR LIDO

MOSTRA_HORA_NO_LCD
   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDO
   MOVLW   0XC4
   CALL    ESCREVE                 ; POSICIONA O CURSOR
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

   SWAPF   HORA,W
   ANDLW   B'00000011'             ; MASCARA A HORA ATUAL (DEZENA)
   ADDLW   0X30                    ; CONVERTE EM ASCII
   CALL    ESCREVE                 ; ESCREVE A DEZENA DA HORA NO DISPLAY
   MOVF    HORA,W
   ANDLW   B'00001111'             ; MASCARA A HORA ATUAL (UNIDADE)
   ADDLW   0X30                    ; CONVERTE EM ASCII
   CALL    ESCREVE                 ; ESCREVE A UNIDADE DA HORA NO DISPLAY
   MOVLW   ':'
   CALL    ESCREVE                 ; ENVIA : AO LCD
   SWAPF   MINUTO,W
   ANDLW   B'00001111'             ; MASCARA O MINUTO ATUAL (DEZENA)
   ADDLW   0X30                    ; CONVERTE EM ASCII
   CALL    ESCREVE                 ; ESCREVE A DEZENA DO MINUTO NO DISPLAY
   MOVF    MINUTO,W
   ANDLW   B'00001111'             ; MASCARA O MINUTO ATUAL (UNIDADE)
   ADDLW   0X30                    ; CONVERTE EM ASCII
   CALL    ESCREVE                 ; ESCREVE A UNIDADE DO MINUTO NO DISPLAY
   MOVLW   ':'
   CALL    ESCREVE                 ; ENVIA : AO LCD
   SWAPF   SEGUNDO,W
```

```
   ANDLW   B'00001111'            ; MASCARA O SEGUNDO (DEZENA)
   ADDLW   0X30                   ; CONVERTE EM ASCII
   CALL    ESCREVE                ; ESCREVE A DEZENA DO SEGUNDO NO DISPLAY
   MOVF    SEGUNDO,W
   ANDLW   B'00001111'            ; MASCARA O SEGUNDO (UNIDADE)
   ADDLW   0X30                   ; CONVERTE EM ASCII
   CALL    ESCREVE                ; ESCREVE A UNIDADE DO SEGUNDO NO DISPLAY

   GOTO    LOOP                   ; VOLTA AO LOOP

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            FIM DO PROGRAMA                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                            ; FIM DO PROGRAMA
```

Dicas e Comentários

Além do relógio RTC o componente PCF8583 da Philips apresenta também uma região da memória que pode ser utilizada com uma RAM convencional, porém com acesso via I^2C. Como geralmente o relógio sempre é utilizado junto com uma bateria, como no MCMASTER, este recurso da RAM pode ser útil para armazenar informações enquanto o microcontrolador não está energizado.

Exercícios Propostos

1. Altere o software para mostrar a data e a hora no display LCD.
2. Crie um software para ajustar a data e hora do relógio utilizando o teclado matricial.

Capítulo 22 - Experiência 20 – Sistema de temperatura e tacômetro

Objetivo

O objetivo desta experiência é mostrar a maioria os recursos disponíveis na placa de experiências EXP01.

Descrição

O monitoramento da temperatura é um dos problemas mais clássicos enfrentado pela maioria dos projetistas. Como o MCMASTER possui um sensor de temperatura (diodo) e dois atuadores: aquecimento (resistência) e resfriamento (ventilador), nada melhor que implementar um sistema capaz de obter a temperatura atual para mostrá-la no LCD.

O sensor de temperatura

Para obter a temperatura ambiente, fez-se uso de um diodo. Como o diodo é um componente que apresenta uma queda de tensão sobre ele proporcional a temperatura do mesmo, basta monitorar a tensão para encontrar a temperatura.

Para isso, o circuito eletrônico faz uso de um diodo de sinal convencional (1N4148) ligado a um amplificador e a uma porta analógica do PIC. Também está ligado ao amplificador um potenciômetro a fim de alterar o off-set da curva, ajustando assim a temperatura com uma referência externa.

Internamente, o sistema trabalha com uma conversão A/D de 8 bits, gerando 256 possíveis valores de tensão para o diodo. Para cada valor obtido, tem-se uma temperatura relacionada. A rotina TABELA_TEMP, que se encontra no final do código apresentado neste capítulo, efetua a conversão entre a tensão lida (unidades de A/D) e a temperatura real. Nada mais é que uma tabela de conversão/linearização.

Esta tabela foi construída com base na curva de resposta do diodo utilizado em função da temperatura. Caso seja construído um sensor de temperatura com outro tipo de componente que gere uma tensão variável, basta refazer a tabela de conversão.

Uma vez convertida, a temperatura é então mostrada no LCD, na unidade de Celsius [°C] (lado direito).

O aquecimento

O software possibilita que o usuário aumente a temperatura sobre o diodo através do controle manual da resistência existente na placa. Isso é feito por intermédio de um dos PWMs do PIC, que se encontra ligado ao resistor.

Através dos botões das colunas 1 e 2 pode-se aumentar e diminuir o duty cycle do PWM, variando de 0 a 100%. Mantendo-se os botões pressionados o incremento/decremento é automático. O valor atual para o aquecimento é mostrado no LCD (lado esquerdo).

O resfriamento

Inversamente ao aquecimento, o software possibilita também o resfriamento do sistema através do ventilador, que é controlado pelo outro canal de PWM do PIC. Obviamente o sistema só é capaz de obter temperaturas levemente abaixo do valor ambiente, mas a intenção é poder criar variáveis diferentes de aquecimento e resfriamento.

Controla-se o ventilador pelo duty cycle do PWM, mas para o sistema ficar um pouco mais completo, mostra-se no LCD (centro) o valor da rotação do motor, em RPS (rotações por segundo). Isto é feito através do sensor óptico que se encontra instalado na base das hélices do ventilador. Os botões das colunas 3 e 4 são usados para aumentar e diminuir o duty cycle do PWM, variando de 0 a 100%. Mantendo-se os botões pressionados o incremento/decremento é automático.

Cada vez que uma das pás da hélice passa em frente ao sensor óptico, um pulso é transmitido ao PIC. Como este sinal está ligado ao pino RC1, utilizou-se o TMR1 com incremento externo para contabilizar a quantidade de pulsos gerados. A cada segundo (base de tempo gerada pela interrupção de TMR2), o total de pulsos é transferido para a variável CONT_VENT. Antes de mostrar o valor correto no LCD, deve-se dividir o total de pulsos durante 1 segundo (CONT_VENT) pelo número de paletas (pulsos por volta). Neste caso CONT_VENT é dividido por 7.

Comunicação serial

O sistema possibilita ainda que maiores implementações sejam feitas com o tratamento externo da temperatura, pois os valores obtidos são enviados automaticamente pela porta serial, a cada segundo. Com isso, é possível criar um pequeno programa de computador capaz de plotar esta temperatura no decorrer do tempo. Os dados também podem ser armazenados para cálculos ou consultas futuras.

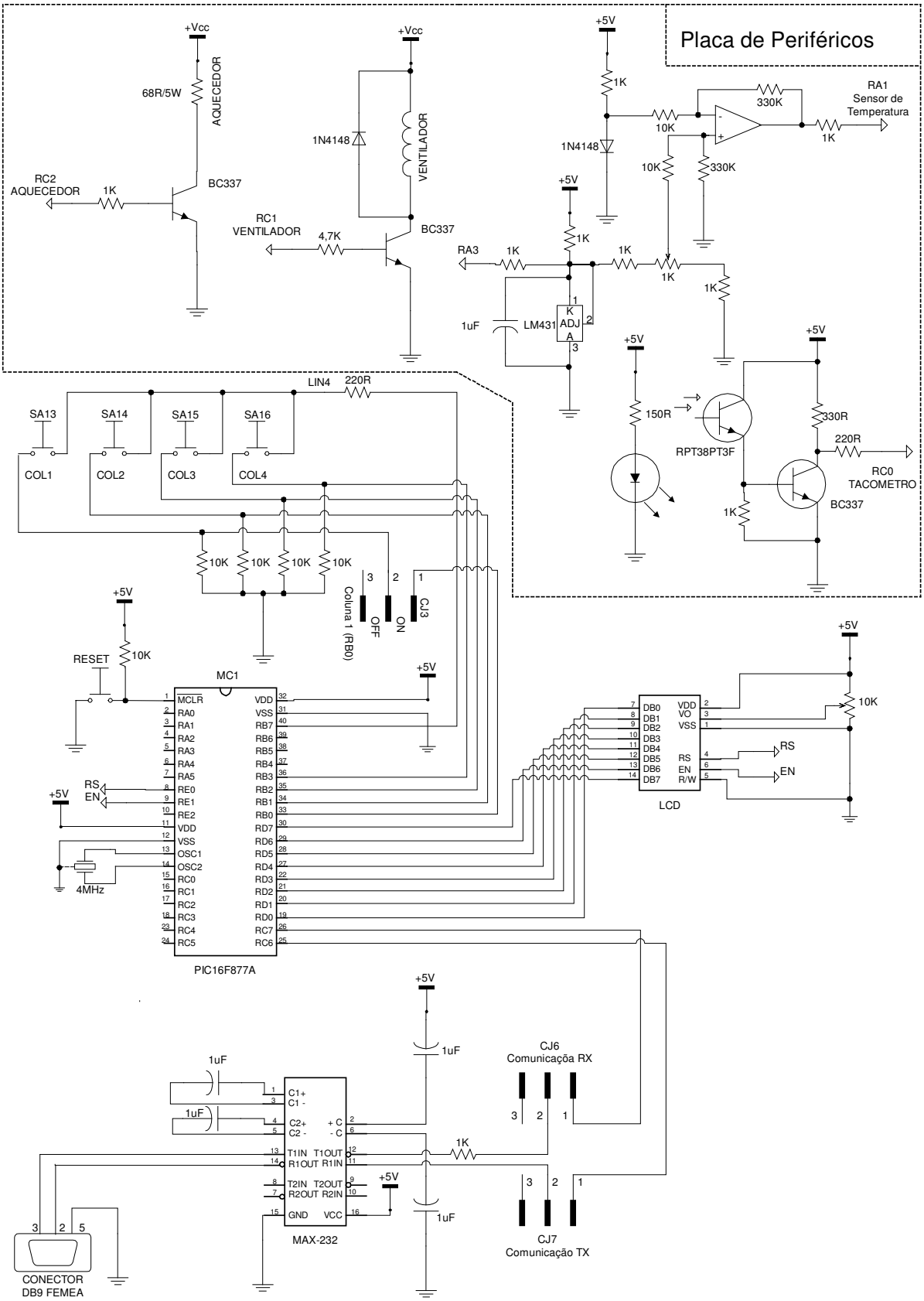
Os dados são transmitidos pela USART através do conector DB-9, respeitando-se o padrão RS-232 com 8N1 e baud rate de 9.600bps. A cada 1 segundo é transmitido um byte com o valor da temperatura já convertido para Celsius [°C].

Considerações gerais

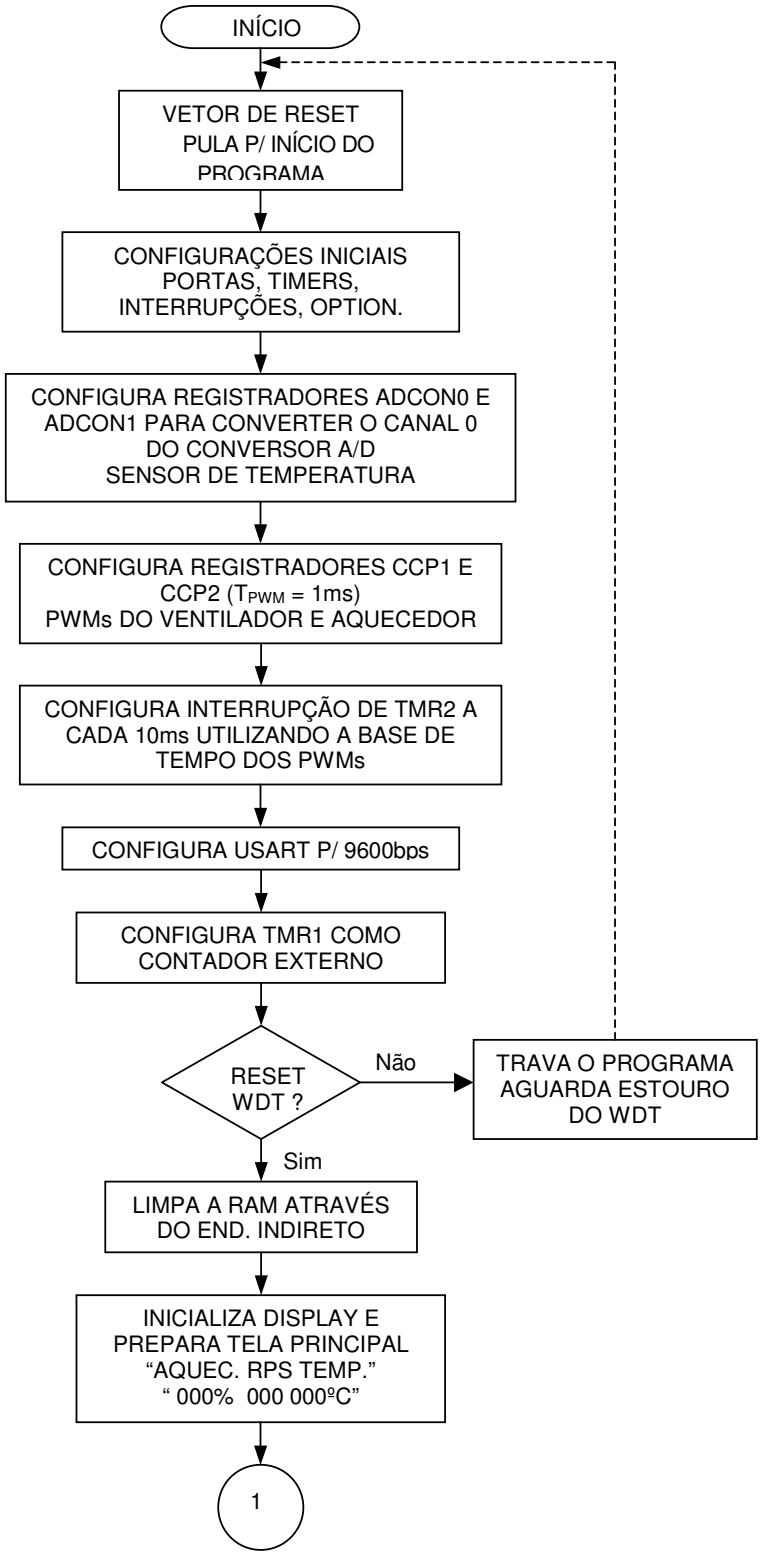
Ambos os PWMs são controlados pelo TMR2, que está regulado para 1ms. Por isso, o período dos 2 PWMs é de 1ms, ou seja, eles operam a 1kHz.

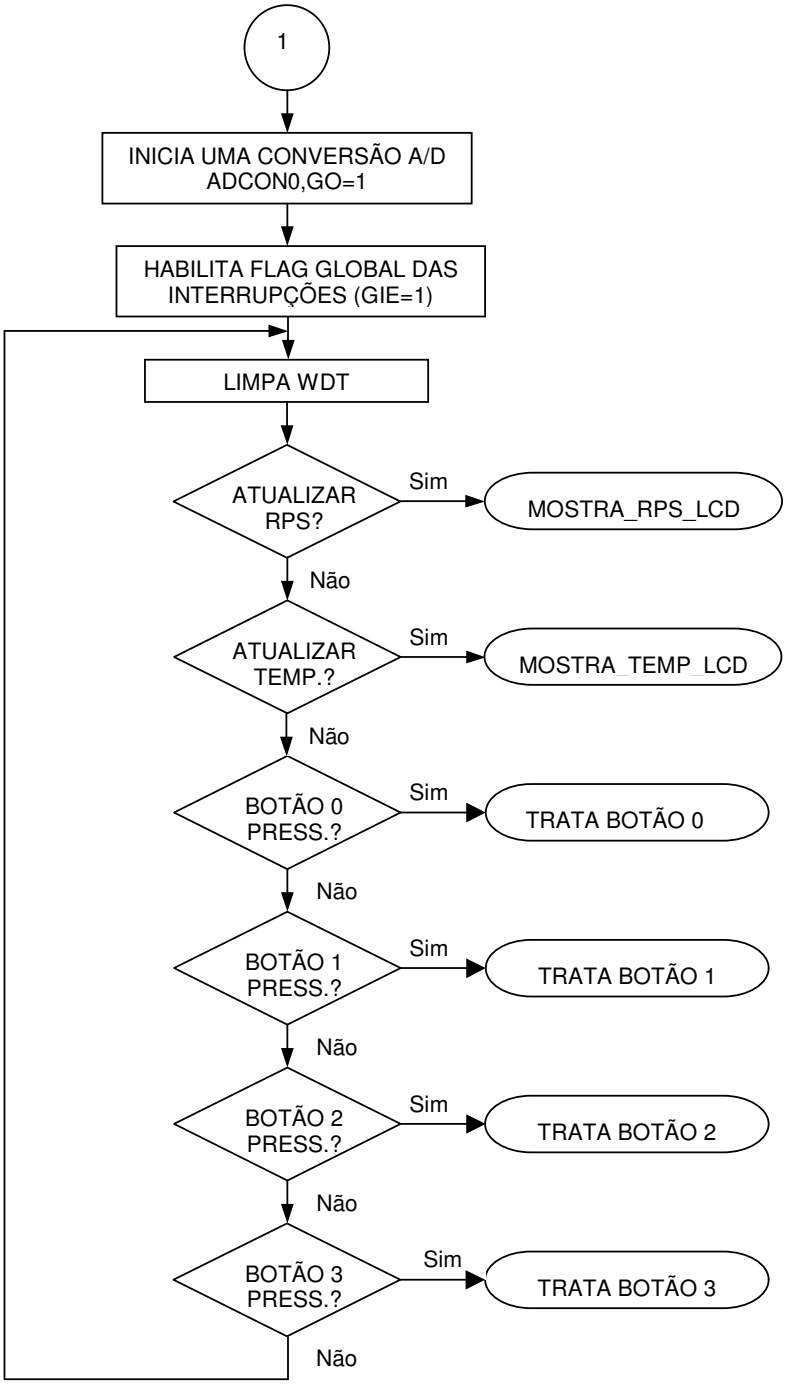
O postscale do TMR2 foi regulado em 1:10, gerando uma interrupção a cada 10ms. Utilizou-se um contador auxiliar (TEMPO_1S) para contabilizar 100 interrupções, gerando a base de tempo de 1 segundo. Esta base é utilizada para capturar a rotação do ventilador, efetuar uma conversão de temperatura e transmitir dados pela USART.

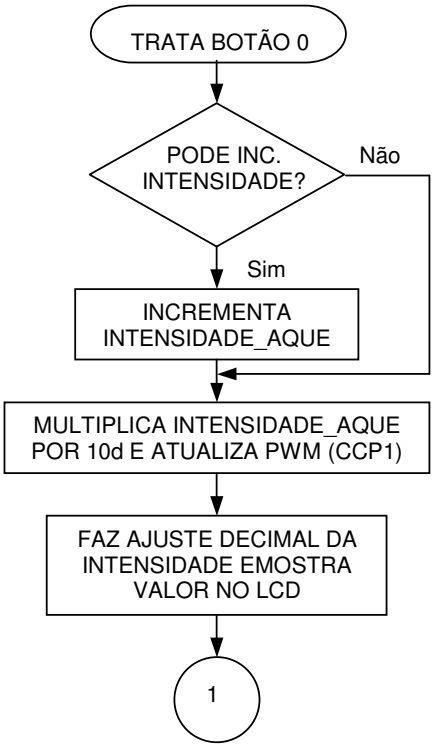
Esquema Elétrico

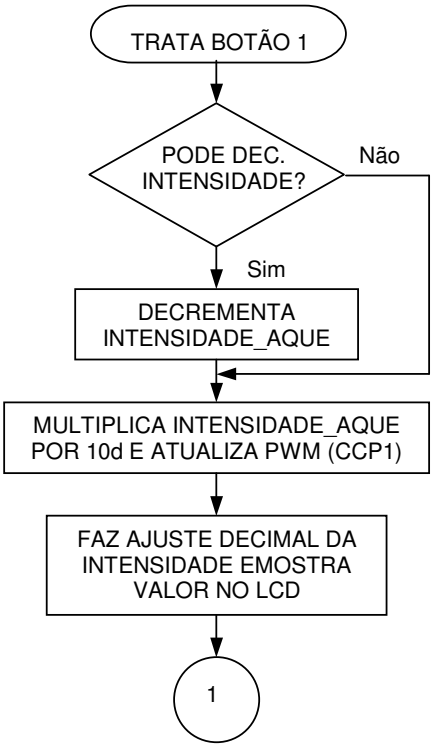


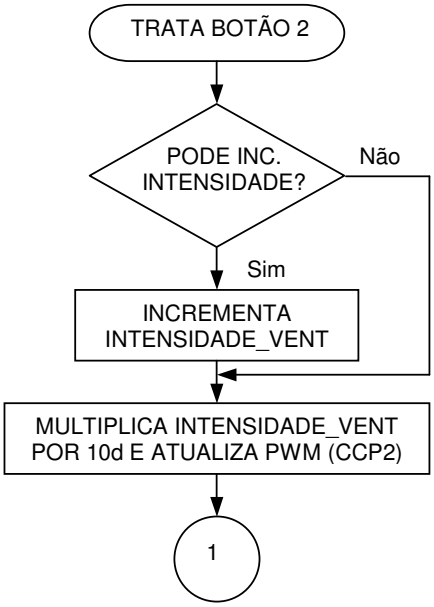
Fluxograma

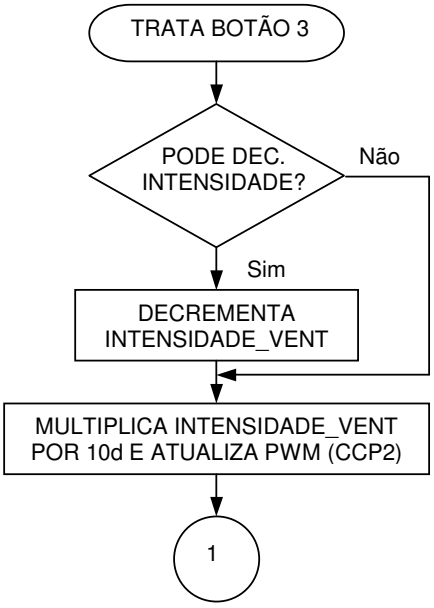

INÍCIO
VETOR DE RESET
PULA P/ INÍCIO DO
PROGRAMA
CONFIGURAÇÕES INICIAIS
PORTAS, TIMERS,
INTERRUPÇÕES, OPTION.
CONFIGURA REGISTRADORES ADCON0 E
ADCON1 PARA CONVERTER O CANAL 0
DO CONVERSOR A/D
SENSOR DE TEMPERATURA
CONFIGURA REGISTRADORES CCP1 E
CCP2 (T_{PWM} = 1ms)
PWMs DO VENTILADOR E AQUECEDOR
CONFIGURA INTERRUPÇÃO DE TMR2 A
CADA 10ms UTILIZANDO A BASE DE
TEMPO DOS PWMs
CONFIGURA USART P/ 9600bps
CONFIGURA TMR1 COMO
CONTADOR EXTERNO
RESET
WDT ?
Não
TRAVA O PROGRAMA
AGUARDA ESTOURO
DO WDT
Sim
LIMPA A RAM ATRAVÉS
DO END. INDIRETO
INICIALIZA DISPLAY E
PREPARA TELA PRINCIPAL
“AQUEC. RPS TEMP.”
“ 000% 000 000ºC”
1

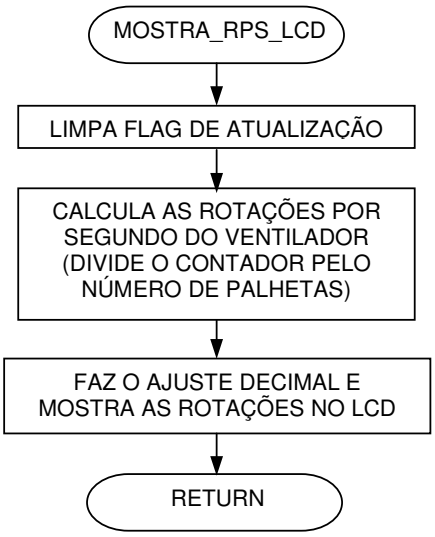

1
INICIA UMA CONVERSÃO A/D
ADCON0,GO=1
HABILITA FLAG GLOBAL DAS
INTERRUPÇÕES (GIE=1)
LIMPA WDT
ATUALIZAR
RPS?
Sim
MOSTRA_RPS_LCD
Não
ATUALIZAR
TEMP.?
Sim
MOSTRA_TEMP_LCD
Não
BOTÃO 0
PRESS.?
Sim
TRATA BOTÃO 0
Não
BOTÃO 1
PRESS.?
Sim
TRATA BOTÃO 1
Não
BOTÃO 2
PRESS.?
Sim
TRATA BOTÃO 2
Não
BOTÃO 3
PRESS.?
Sim
TRATA BOTÃO 3
Não

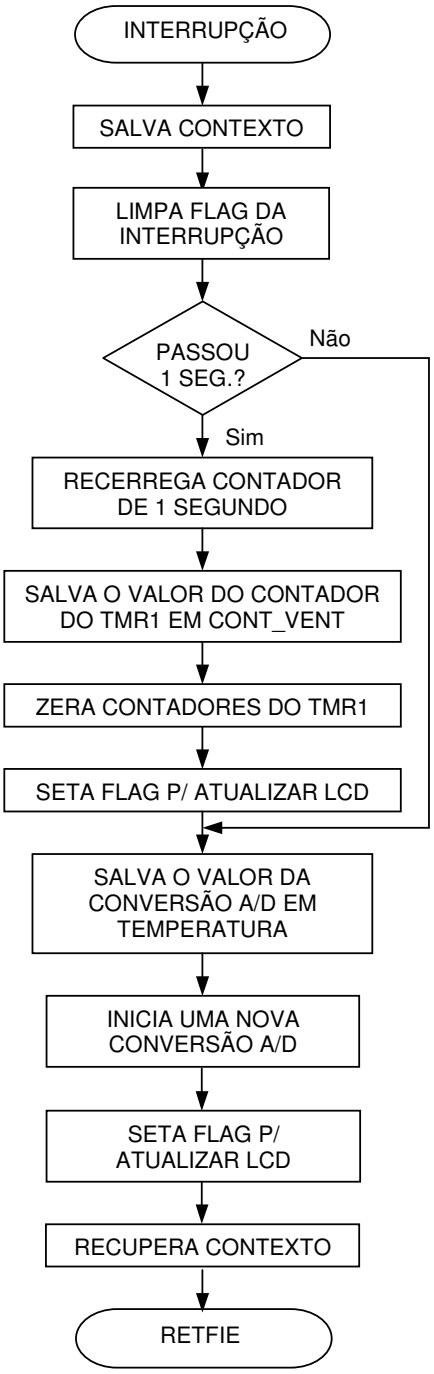

TRATA BOTÃO 0
PODE INC. INTENSIDADE?
Não
Sim
INCREMENTA INTENSIDADE_AQUE
MULTIPLICA INTENSIDADE_AQUE POR 10d E ATUALIZA PWM (CCP1)
FAZ AJUSTE DECIMAL DA INTENSIDADE EMOSTRA VALOR NO LCD
1

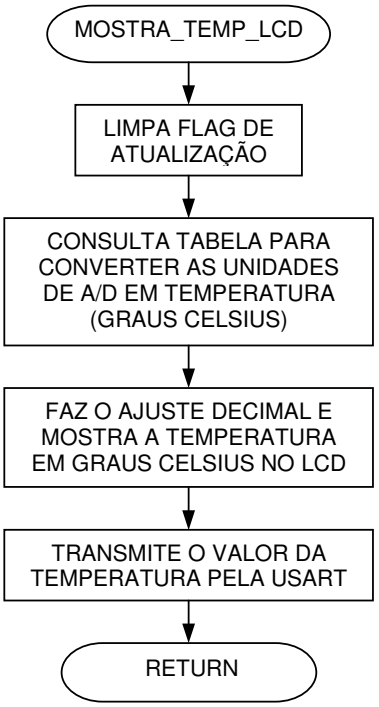

TRATA BOTÃO 1
PODE DEC. INTENSIDADE?
Não
Sim
DECREMENTA INTENSIDADE_AQUE
MULTIPLICA INTENSIDADE_AQUE POR 10d E ATUALIZA PWM (CCP1)
FAZ AJUSTE DECIMAL DA INTENSIDADE EMOSTRA VALOR NO LCD
1


TRATA BOTÃO 2
PODE INC. INTENSIDADE?
Não
Sim
INCREMENTA INTENSIDADE_VENT
MULTIPLICA INTENSIDADE_VENT POR 10d E ATUALIZA PWM (CCP2)
1


TRATA BOTÃO 3
PODE DEC. INTENSIDADE?
Não
Sim
DECREMENTA INTENSIDADE_VENT
MULTIPLICA INTENSIDADE_VENT POR 10d E ATUALIZA PWM (CCP2)
1


MOSTRA_RPS_LCD
LIMPA FLAG DE ATUALIZAÇÃO
CALCULA AS ROTAÇÕES POR SEGUNDO DO VENTILADOR (DIVIDE O CONTADOR PELO NÚMERO DE PALHETAS)
FAZ O AJUSTE DECIMAL E MOSTRA AS ROTAÇÕES NO LCD
RETURN


INTERRUPÇÃO
SALVA CONTEXTO
LIMPA FLAG DA INTERRUPÇÃO
PASSOU 1 SEG.?
Não
Sim
RECERREGA CONTADOR DE 1 SEGUNDO
SALVA O VALOR DO CONTADOR DO TMR1 EM CONT_VENT
ZERA CONTADORES DO TMR1
SETA FLAG P/ ATUALIZAR LCD
SALVA O VALOR DA CONVERSÃO A/D EM TEMPERATURA
INICIA UMA NOVA CONVERSÃO A/D
SETA FLAG P/ ATUALIZAR LCD
RECUPERA CONTEXTO
RETFIE


MOSTRA_TEMP_LCD
LIMPA FLAG DE ATUALIZAÇÃO
CONSULTA TABELA PARA CONVERTER AS UNIDADES DE A/D EM TEMPERATURA (GRAUS CELSIUS)
FAZ O AJUSTE DECIMAL E MOSTRA A TEMPERATURA EM GRAUS CELSIUS NO LCD
TRANSMITE O VALOR DA TEMPERATURA PELA USART
RETURN

Código

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              EXEMPLO DE CÓDIGO PARA UTILIZAÇÃO COM O MCMASTER            *
; *                                                                         *
; *            EXPERIÊNCIA 20 - SISTEMA DE TEMPERATURA E TACÔMETRO          *
; *                                                                         *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *   VERSÃO : 1.0                                                          *
; *     DATA : 14/04/2003                                                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                            DESCRIÇÃO GERAL                              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE EXEMPLO FOI ELABORADO PARA DEMOSTRAR A MAIORIA DOS RECURSOS
;  DISPONÍVEIS NA PLACA DE EXPERIÊNCIAS EXP01.
;  DOIS PWMs FORAM UTILIZADOS, UM PARA MODULAR A RESISTÊNCIA DE AQUECIMENTO
;  E OUTRO PARA A VELOCIDADE DO VENTILADOR.
;  O SOFTWARE CONVERTE O CANAL 1 DO CONVERSOR A/D PARA MEDIR A TEMPERATURA.
;  O SISTEMA ÓTICO FOI UTILIZADO PARA MEDIR AS ROTAÇÕES DO VENTILADOR.
;  COM AS TECLAS DAs COLUNAS 1 E 2 PODE-SE VARIAR O PWM DO AQUECEDOR E COM
;  AS TECLAS DAS COLUNAS 3 E 4 O PWM DO VENTILADOR.
;  A LINHA ATIVADA É A 4 DO TECLADO MATRICIAL
;  NO LCD SÃO MOSTRADOS OS VALORES DO PWM DO AQUECEDOR, O NÚMERO DE ROTAÇÕES
;  POR SEGUNDO DO VENTILADOR E A TEMPERATURA DO DIODO JÁ CONVERTIDA EM GRAUS
;  CELSIUS. ALÉM DISSO, O VALOR ATUAL DA TEMPERATURA DO DIODO É TRANSMITIDO
;  PERIODICAMENTE ATRAVÉS DA USART.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                       CONFIGURAÇÕES PARA GRAVAÇÃO                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 __CONFIG _CP_OFF & _CPD_OFF & _DEBUG_OFF & _LVP_OFF & _WRT_OFF & _BODEN_OFF &
_PWRTE_ON & _WDT_ON & _XT_OSC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           DEFINIÇÃO DAS VARIÁVEIS                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTE BLOCO DE VARIÁVEIS ESTÁ LOCALIZADO LOGO NO INÍCIO DO BANCO 0

   CBLOCK  0X20                    ; POSIÇÃO INICIAL DA RAM

           TEMPO0
           TEMPO1                  ; TEMPORIZADORES P/ ROTINA DE DELAY

           AUX                     ; REGISTRADOR AUXILIAR DE USO GERAL

           UNIDADE                 ; ARMAZENA VALOR DA UNIDADE
           DEZENA                  ; ARMAZENA VALOR DA DEZENA
           CENTENA                 ; ARMAZENA VALOR DA CENTENA

           INTENSIDADE_VENT        ; INTENSIDADE DO VENTILADOR
           INTENSIDADE_AQUE        ; INTENSIDADE DO AQUECEDOR

           ACCaHI                  ; ACUMULADOR a DE 16 BITS UTILIZADO
           ACCaLO                  ; NA ROTINA DE DIVISÃO

           ACCbHI                  ; ACUMULADOR b DE 16 BITS UTILIZADO
           ACCbLO                  ; NA ROTINA DE DIVISÃO

           ACCcHI                  ; ACUMULADOR c DE 16 BITS UTILIZADO
           ACCcLO                  ; NA ROTINA DE DIVISÃO

           ACCdHI                  ; ACUMULADOR d DE 16 BITS UTILIZADO
           ACCdLO                  ; NA ROTINA DE DIVISÃO

           temp                    ; CONTADOR TEMPORÁRIO UTILIZADO
                                   ; NA ROTINA DE DIVISÃO
```

```
          H_byte                ; ACUMULADOR DE 16 BITS UTILIZADO
          L_byte                ; P/ RETORNAR O VALOR DA ROTINA
                                ; DE MULTIPLICAÇÃO

          mulplr                ; OPERADOR P/ ROTINA DE MUTIPLICAÇÃO
          mulcnd                ; OPERADOR P/ ROTINA DE MUTIPLICAÇÃO

          TEMPERATURA           ; TEMPERATURA DO DIODO EM UNIDADES DE A/D
          TEMP_CELSIUS          ; TEMPERATURA DO DIODO JÁ CONVERTIDO
                                ; PARA GRAUS CELSIUS

          FILTRO_BOTOES         ; FILTRO P/ DEBOUNCE DOS BOTOÕES
          TEMPO_TURBO           ; TEMPORIZADOR P/ TUBO DO TECLADO

          TEMPO_1S              ; TEMPORIZADOR DE 1 SEGUNDO

          CONT_VENT_HIGH
          CONT_VENT_LOW         ; CONTADORES PARA ROTAÇÃO DO VENTILADOR

          FLAG                  ; FLAG DE USO GERAL

          WORK_TEMP
          STATUS_TEMP
          PCLATH_TEMP
          FSR_TEMP              ; REGISTRADORES UTILIZADOS P/ SALVAR
                                ; O CONTEXTO DURANTE AS INTERRUPÇÕES
   ENDC

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               DEFINIÇÃO DAS VARIÁVEIS INTERNAS DO PIC                   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  O ARQUIVO DE DEFINIÇÕES DO PIC UTILIZADO DEVE SER REFERENCIADO PARA QUE
;  OS NOMES DEFINIDOS PELA MICROCHIP POSSAM SER UTILIZADOS, SEM A NECESSIDADE
;  DE REDIGITAÇÃO.

   #INCLUDE <P16F877A.INC>        ; MICROCONTROLADOR UTILIZADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                      DEFINIÇÃO DOS BANCOS DE RAM                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  OS PSEUDOS-COMANDOS "BANK0" E "BANK1", AQUI DEFINIDOS, AJUDAM A COMUTAR
;  ENTRE OS BANCOS DE MEMÓRIA.

#DEFINE    BANK1   BSF     STATUS,RP0      ; SELECIONA BANK1 DA MEMORIA RAM
#DEFINE    BANK0   BCF     STATUS,RP0      ; SELECIONA BANK0 DA MEMORIA RAM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                           CONSTANTES INTERNAS                           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE CONSTANTES FACILITA A PROGRAMAÇÃO E A MANUTENÇÃO.

FILTRO_TECLA       EQU     .200            ; FILTRO P/ EVITAR RUIDOS DOS BOTÕES

TURBO_TECLA        EQU     .70             ; TEMPERIZADOR P/ TURBO DO TECLADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    DECLARAÇÃO DOS FLAGs DE SOFTWARE                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A DEFINIÇÃO DE FLAGs AJUDA NA PROGRAMAÇÃO E ECONOMIZA MEMÓRIA RAM.

#DEFINE    MOSTRA_RPS      FLAG,0  ; FLAG PARA MOSTRAR A ROTAÇÃO NO LCD
                                   ; 1 -> DEVE MOSTRAR A ROTAÇÃO
                                   ; 0 -> NAO DEVE MOSTRAR A ROTAÇÃO

#DEFINE    MOSTRA_TEMP     FLAG,1  ; FLAG PARA MOSTRAR A TEMPERATURA NO LCD
                                   ; 1 -> DEVE MOSTRAR A TEMPERATURA
                                   ; 0 -> NAO DEVE MOSTRAR A TEMPERATURA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                ENTRADAS                                 *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS ENTRADAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
```

```
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    BOTAO_0         PORTB,0 ; ESTADO DO BOTÃO 0
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_1         PORTB,1 ; ESTADO DO BOTÃO 1
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_2         PORTB,2 ; ESTADO DO BOTÃO 2
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

#DEFINE    BOTAO_3         PORTB,3 ; ESTADO DO BOTÃO 3
                                   ; 0 -> LIBERADO
                                   ; 1 -> PRESSIONADO

;  ESTE PROGRAMA UTILIZA UMA ENTRADA P/ O CONVERSOR A/D.
;  ESTA ENTRADA NÃO PRECISA SER DECLARADA, POIS O SOFTWARE NUNCA FAZ
;  REFERÊNCIA A ELA DE FORMA DIRETA, POIS O CANAL A/D A SER CONVERTIDO É
;  SELECIONADO NO REGISTRADOS ADCON0 DE FORMA BINÁRIA E NÃO ATRAVÉS DE
;  DEFINES. PORÉM PARA FACILITAR O ENTENDIMENTO DO HARDWARE VAMOS DECLARAR
;  ESTA ENTRADA NORMALMENTE.

#DEFINE    CAD_TEMP        PORTA,1 ; ENTRADA A/D PARA TEMPERATURA

;  ALÉM DA ENTRADA DO CONVERSOR A/D, TEMOS A ENTRADA DA USART (RECEPÇÃO).
;  NOVAMENTE ESTA ENTRADA NÃO NECESSITA SER DECLARADA, PORÉM, PARA
;  FACILITAR O ENTENDIMENTO DO HARDWARE VAMOS DECLARAR ESTA ENTRADA
;  NORMALMENTE.

#DEFINE    RXUSART PORTC,7         ; ENTRADA DE RX DA USART

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                                 SAÍDAS                                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  AS SAÍDAS DEVEM SER ASSOCIADAS A NOMES PARA FACILITAR A PROGRAMAÇÃO E
;  FUTURAS ALTERAÇÕES DO HARDWARE.

#DEFINE    DISPLAY         PORTD   ; BARRAMENTO DE DADOS DO DISPLAY

#DEFINE    RS              PORTE,0 ; INDICA P/ O DISPLAY UM DADO OU COMANDO
                                   ; 1 -> DADO
                                   ; 0 -> COMANDO

#DEFINE    ENABLE          PORTE,1 ; SINAL DE ENABLE P/ DISPLAY
                                   ; ATIVO NA BORDA DE DESCIDA

#DEFINE    TEC_MATRICIAL   PORTB   ; PORT DO MICROCONTROLADOR LIGADO AO
                                   ; TECLADO MATRICIAL
                                   ; <RB4:RB7> LINHAS
                                   ; 1->ATIVADAS 0->DESATIVADAS
                                   ; <RB0:RB3> COLUNAS
                                   ; 1->TECLAS PRESSIONADAS 0->TECLAS LIBERADAS

#DEFINE    LINHA_4         PORTB,7 ; PINO P/ ATIVAR LINHA 4 (TECLADO MATRICIAL)
                                   ; 0 -> LINHA 4 ATIVADA
                                   ; 1 -> LINHA 4 DESATIVADA

;  TEMOS TAMBÉM AS SAÍDAS DE TX DA USART, PWM1 E PWM2.
;  PARA FACILITAR O ENTENDIMENTO DO HARDWARE VAMOS DECLARAR ESTAS SAÍDAS
;  NORMALMENTE APESAR DE NÃO SEREM UTILIZADAS.

#DEFINE    TXUSART         PORTC,6 ; SAÍDA DE TX DA USART

#DEFINE    VENTILADOR      PORTC,1 ; SAÍDA P/ VENTILADOR

#DEFINE    AQUECEDOR       PORTC,2 ; SAÍDA P/ AQUECEDOR

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                   VETOR DE RESET DO MICROCONTROLADOR                  *
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO INICIAL PARA EXECUÇÃO DO PROGRAMA

   ORG     0X0000                  ; ENDEREÇO DO VETOR DE RESET
   GOTO    CONFIG                  ; PULA PARA CONFIG DEVIDO A REGIÃO
                                   ; DESTINADA AS ROTINAS SEGUINTES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          VETOR DE INTERRUPÇÃO                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  POSIÇÃO DE DESVIO DO PROGRAMA QUANDO UMA INTERRUPÇÃO ACONTECE

   ORG     0X0004                  ; ENDEREÇO DO VETOR DE INTERRUPÇÃO

;  É MUITO IMPORTANTE QUE OS REGISTRADORES PRIORITÁRIOS AO FUNCIONAMENTO DA
;  MÁQUINA, E QUE PODEM SER ALTERADOS TANTO DENTRO QUANTO FORA DAS INTS SEJAM
;  SALVOS EM REGISTRADORES TEMPORÁRIOS PARA PODEREM SER POSTERIORMENTE
;  RECUPERADOS.

SALVA_CONTEXTO
   MOVWF   WORK_TEMP               ; SALVA REGISTRADOR DE TRABALHO E
   SWAPF   STATUS,W                ; DE STATUS DURANTE O TRATAMENTO
   MOVWF   STATUS_TEMP             ; DA INTERRUPÇÃO.
   MOVF    FSR,W
   MOVWF   FSR_TEMP                ; SALVA REGISTRADOR FSR
   MOVF    PCLATH,W
   MOVWF   PCLATH_TEMP             ; SALVA REGISTRADOR PCLATH

   CLRF    PCLATH                  ; LIMPA REGISTRADOR PCLATH
                                   ; (SELECIONA PÁGINA 0)
   CLRF    STATUS                  ; LIMPA REGISTRADOR STATUS
                                   ; (SELECIONA BANCO 0)

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               TRATAMENTO DA INTERRUPÇÃO DE TIMER 2                        *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; A INTERRUPÇÃO DE TMR2 É UTILIZADA PARA FORNECER A BASE DE TEMPO PARA AS
; MEDIDAS DAS ROTAÇÕES POR SEGUNDO DO VENTILADOR E DA TEMPERATURA DO DIODO.
; ALÉM DISSO, ELA SETA OS FLAGS PARA QUE ESTES SEJAM ATUALIZADOS NO LCD.
; O TMR2 ESTÁ CONFIGURADO PARA POSTSCALE DE 1:10 E PORTANTO A CADA 10ms A
; INTERRUPÇÃO É GERADA.
; O CONVERSOR A/D É LIDO A CADA INTERRUPÇÃO, OU SEJA, A CADA 10ms.
; A CADA 100 INTERRUPÇÕES, OU SEJA, A CADA 1 SEGUNDO, O VALOR DO CONTADOR DO
; TMR1 É SALVO NA VARIÁVEL CONT_VENT (HIGH E LOW), DESTA FORMA, O VALOR DE
; CONT_VENT É O NÚMERO DE ROTAÇÕES DO VENTILADOR POR SEGUNDO. NA VERDADE ESTE
; VALOR ENCONTRA-SE MULTIPLICADO PELO NÚMERO DE PALHETAS DO VENTILADOR.

INT_TMR2
   BCF     PIR1,TMR2IF             ; LIMPA FLAG DA INTERRUPÇÃO

   DECFSZ  TEMPO_1S,F              ; FIM DO 1 SEGUNDO ?
   GOTO    INT_TMR2_2              ; NÃO - PULA P/ INT_TMR2_2
                                   ; SIM
   MOVLW   .100
   MOVWF   TEMPO_1S                ; RECARREGA TEMPORIZADOR DE 1 SEGUNDO

   BCF     T1CON,TMR1ON            ; PARALIZA CONTADOR DO TMR1

   MOVF    TMR1H,W
   MOVWF   CONT_VENT_HIGH
   MOVF    TMR1L,W
   MOVWF   CONT_VENT_LOW           ; SALVA VALOR DO TMR1 EM CONT_VENT

   CLRF    TMR1H
   CLRF    TMR1L                   ; RESETA CONTADORES

   BSF     T1CON,TMR1ON            ; LIBERA CONTADORES DO TMR1

   BSF     MOSTRA_RPS              ; SETA FLAG P/ MOSTRAR VALOR
                                   ; DAS RPS DO VENTILADOR

INT_TMR2_2
```

```
   MOVF    ADRESH,W
   MOVWF   TEMPERATURA             ; SALVA VALOR DA CONVERSÃO A/D
                                   ; NA VARIÁVEL TEMPERATURA
   BSF     ADCON0,GO               ; INICIA UMA NOVA CONVERSÃO

   BSF     MOSTRA_TEMP             ; SETA FLAG P/ ATUALIZAR VALOR
                                   ; DA TEMPERATURA NO LCD

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          SAÍDA DA INTERRUPÇÃO                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ANTES DE SAIR DA INTERRUPÇÃO, O CONTEXTO SALVO NO INÍCIO DEVE SER
;  RECUPERADO PARA QUE O PROGRAMA NÃO SOFRA ALTERAÇÕES INDESEJADAS.

SAI_INT
   MOVF    PCLATH_TEMP,W
   MOVWF   PCLATH                  ; RECUPERA REG. PCLATH (PAGINAÇÃO)
   MOVF    FSR_TEMP,W
   MOVWF   FSR                     ; RECUPERA REG. FSR (END. INDIRETO)
   SWAPF   STATUS_TEMP,W
   MOVWF   STATUS                  ; RECUPERA REG. STATUS
   SWAPF   WORK_TEMP,F
   SWAPF   WORK_TEMP,W             ; RECUPERA REG. WORK
   RETFIE                          ; RETORNA DA INTERRUPÇÃO (HABILITA GIE)

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          ROTINA DE DIVISÃO                                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;*************************************************************************
;                        Double Precision Division
;*************************************************************************
;   Division : ACCb(16 bits) / ACCa(16 bits) -> ACCb(16 bits) with
;                                               Remainder in ACCc (16 bits)
;      (a) Load the Denominator in location ACCaHI & ACCaLO ( 16 bits )
;      (b) Load the Numerator in location ACCbHI & ACCbLO ( 16 bits )
;      (c) CALL D_divF
;      (d) The 16 bit result is in location ACCbHI & ACCbLO
;      (e) The 16 bit Remainder is in locations ACCcHI & ACCcLO
;*************************************************************************

D_divF
   MOVLW   .16
   MOVWF   temp                    ; CARREGA CONTADOR PARA DIVISÃO

   MOVF    ACCbHI,W
   MOVWF   ACCdHI
   MOVF    ACCbLO,W
   MOVWF   ACCdLO                  ; SALVA ACCb EM ACCd

   CLRF    ACCbHI
   CLRF    ACCbLO                  ; LIMPA ACCb

   CLRF    ACCcHI
   CLRF    ACCcLO                  ; LIMPA ACCc

DIV
   BCF     STATUS,C
   RLF     ACCdLO,F
   RLF     ACCdHI,F
   RLF     ACCcLO,F
   RLF     ACCcHI,F
   MOVF    ACCaHI,W
   SUBWF   ACCcHI,W                ;check if a>c
   BTFSS   STATUS,Z
   GOTO    NOCHK
   MOVF    ACCaLO,W
   SUBWF   ACCcLO,W                ;if msb equal then check lsb
NOCHK
   BTFSS   STATUS,C                ;carry set if c>a
   GOTO    NOGO
   MOVF    ACCaLO,W                ;c-a into c
   SUBWF   ACCcLO,F
```

```
   BTFSS   STATUS,C
   DECF    ACCcHI,F
   MOVF    ACCaHI,W
   SUBWF   ACCcHI,F
   BSF     STATUS,C                ;shift a 1 into b (result)
NOGO
   RLF     ACCbLO,F
   RLF     ACCbHI,F

   DECFSZ  temp,F                  ; FIM DA DIVISÃO ?
   GOTO    DIV                     ; NÃO - VOLTA P/ DIV
                                   ; SIM
   RETURN                          ; RETORNA


; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          ROTINA DE MULTIPLICAÇÃO                            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;*****************************************************************************
;                   8x8 Software Multiplier
;               ( Fast Version : Straight Line Code )
;*****************************************************************************
;
;   The 16 bit result is stored in 2 bytes
; Before calling the subroutine " mpy ", the multiplier should
; be loaded in location " mulplr ", and the multiplicand in
; " mulcnd " . The 16 bit result is stored in locations
; H_byte & L_byte.
;       Performance :
;                       Program Memory  :  37 locations
;                       # of cycles     :  38
;                       Scratch RAM     :   0 locations
;*****************************************************************

; ******************************************
;  Define a macro for adding & right shifting
; ******************************************

mult    MACRO   bit                       ; Begin macro

   BTFSC   mulplr,bit
   ADDWF   H_byte,F
   RRF     H_byte,F
   RRF     L_byte,F

   ENDM                            ; End of macro

; ****************************
;   Begin Multiplier Routine
; ****************************

mpy_F
   CLRF    H_byte
   CLRF    L_byte
   MOVF    mulcnd,W                ; move the multiplicand to W reg.
   BCF     STATUS,C                ; Clear carry bit in the status Reg.

   mult    0
   mult    1
   mult    2
   mult    3
   mult    4
   mult    5
   mult    6
   mult    7

   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                    ROTINA DE DELAY (DE 1MS ATÉ 256MS)                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA É UMA ROTINA DE DELAY VARIÁVEL, COM DURAÇÃO DE 1MS X O VALOR PASSADO
```

```
;  EM WORK (W).

DELAY_MS
   MOVWF   TEMPO1                ; CARREGA TEMPO1 (UNIDADES DE MS)
   MOVLW   .250
   MOVWF   TEMPO0                ; CARREGA TEMPO0 (P/ CONTAR 1MS)

   CLRWDT                        ; LIMPA WDT (PERDE TEMPO)
   DECFSZ  TEMPO0,F              ; FIM DE TEMPO0 ?
   GOTO    $-2                   ; NÃO - VOLTA 2 INSTRUÇÕES
                                 ; SIM - PASSOU-SE 1MS
   DECFSZ  TEMPO1,F              ; FIM DE TEMPO1 ?
   GOTO    $-6                   ; NÃO - VOLTA 6 INSTRUÇÕES
                                 ; SIM
   RETURN                        ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *              ROTINA DE ESCRITA DE UM CARACTER NO DISPLAY              *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ENVIA UM CARACTER PARA O MÓDULO DE LCD. O CARACTER A SER
;  ESCRITO DEVE SER COLOCADO EM WORK (W) ANTES DE CHAMAR A ROTINA.

ESCREVE
   MOVWF   DISPLAY               ; ATUALIZA DISPLAY (PORTD)
   NOP                           ; PERDE 1US PARA ESTABILIZAÇÃO
   BSF     ENABLE                ; ENVIA UM PULSO DE ENABLE AO DISPLAY
   GOTO    $+1                   ; .
   BCF     ENABLE                ; .

   MOVLW   .1
   CALL    DELAY_MS              ; DELAY DE 1MS
   RETURN                        ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              AJUSTE DECIMAL                           *
; *           W [HEX] =  CENTENA [DEC] : DEZENA [DEC] ; UNIDADE [DEC]     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA RECEBE UM ARGUMENTO PASSADO PELO WORK E RETORNA NAS VARIÁVEIS
;  CENTENA, DEZENA E UNIDADE O NÚMERO BCD CORRESPONDÊNTE AO PARÂMETRO PASSADO.

AJUSTE_DECIMAL
   MOVWF   AUX                   ; SALVA VALOR A CONVERTER EM AUX

   CLRF    UNIDADE
   CLRF    DEZENA
   CLRF    CENTENA               ; RESETA REGISTRADORES

   MOVF    AUX,F
   BTFSC   STATUS,Z              ; VALOR A CONVERTER = 0 ?
   RETURN                        ; SIM - RETORNA
                                 ; NÃO
   INCF    UNIDADE,F             ; INCREMENTA UNIDADE

   MOVF    UNIDADE,W
   XORLW   0X0A
   BTFSS   STATUS,Z              ; UNIDADE = 10d ?
   GOTO    $+3                   ; NÃO
                                 ; SIM
   CLRF    UNIDADE               ; RESETA UNIDADE
   INCF    DEZENA,F              ; INCREMENTA DEZENA

   MOVF    DEZENA,W
   XORLW   0X0A
   BTFSS   STATUS,Z              ; DEZENA = 10d ?
   GOTO    $+3                   ; NÃO
                                 ; SIM
   CLRF    DEZENA                ; RESETA DEZENA
   INCF    CENTENA,F             ; INCREMENTA CENTENA

   DECFSZ  AUX,F                 ; FIM DA CONVERSÃO ?
   GOTO    $-.14                 ; NÃO - VOLTA P/ CONTINUAR CONVERSÃO
   RETURN                        ; SIM
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *            ROTINA PARA MOSTRAR A ROTAÇÃO DO VENTILADOR NO LCD          *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA ATUALIZA O VALOR DAS ROTAÇÕES POR SEGUNDO DO VENTILADOR NO LCD.

MOSTRA_RPS_LCD
   BCF     MOSTRA_RPS              ; LIMPA FLAG DE ATUALIZAÇÃO DA RPS

   CLRF    TEC_MATRICIAL           ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XC7                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 7
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

   MOVF    CONT_VENT_HIGH,W
   MOVWF   ACCbHI
   MOVF    CONT_VENT_LOW,W
   MOVWF   ACCbLO                  ; CARREGA ACCb COM VALOR DO CONTADOR

   CLRF    ACCaHI
   MOVLW   .7
   MOVWF   ACCaLO                  ; CARREGA ACCa COM NÚMERO DE PALHETAS
                                   ; DO VENTILADOR

   CALL    D_divF                  ; CHAMA ROTINA DE DIVISÃO

   MOVF    ACCbLO,W
   CALL    AJUSTE_DECIMAL          ; FAZ O AJUSTE DECIMAL DO RESULTADO
                                   ; (ROTAÇÕES POR SEGUNDO)
   MOVF    CENTENA,W
   ADDLW   0X30                    ; CONVERTE CENTENA EM ASCII
   CALL    ESCREVE                 ; ESCREVE VALOR NO LCD

   MOVF    DEZENA,W
   ADDLW   0X30                    ; CONVERTE DEZENA EM ASCII
   CALL    ESCREVE                 ; ESCREVE VALOR NO LCD

   MOVF    UNIDADE,W
   ADDLW   0X30                    ; CONVERTE UNIDADE EM ASCII
   CALL    ESCREVE                 ; ESCREVE VALOR NO LCD

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                ROTINA PARA MOSTRAR A TEMPERATURA NO LCD                  *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA CONSULTA UMA TABELA P/ CONVERTER O VALOR DO CANAL A/D DO SENSOR
; DE TEMPERATURA EM GRAUS CELSIUS, MOSTRA ESTE NO LCD E TRANSMITE PELA USART.

MOSTRA_TEMP_LCD
   BCF     MOSTRA_TEMP             ; LIMPA FLAG DE ATUALIZAÇÃO DA TEMP.

   CALL    TABELA_TEMPERATURA      ; CONVERTE A/D EM GRAUS CELSIUS
   MOVWF   TEMP_CELSIUS            ; SALVA VALOR EM TEMP_CELSIUS
   CALL    AJUSTE_DECIMAL          ; FAZ O AJUSTE DECIMAL

   CLRF    TEC_MATRICIAL           ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XCB                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 11
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

   MOVF    CENTENA,W
   ADDLW   0X30                    ; CONVERTE CENTENA EM ASCII
   CALL    ESCREVE                 ; ESCREVE VALOR NO LCD

   MOVF    DEZENA,W
```

```
   ADDLW   0X30                    ; CONVERTE DEZENA EM ASCII
   CALL    ESCREVE                 ; ESCREVE VALOR NO LCD

   MOVF    UNIDADE,W
   ADDLW   0X30                    ; CONVERTE UNIDADE EM ASCII
   CALL    ESCREVE                 ; ESCREVE VALOR NO LCD

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   MOVF    TEMP_CELSIUS,W          ; CARREGA EM WORK A TEMPERATURA
   BANK1                           ; ALTERA P/ BANCO 1 DA RAM
   BTFSS   TXSTA,TRMT              ; O BUFFER DE TX ESTÁ VAZIO ?
   GOTO    $-1                     ; NÃO - AGUARDA ESVAZIAR
   BANK0                           ; SIM - VOLTA P/ BANCO 0 DA RAM
   MOVWF   TXREG                   ; CARREGA TXREG COM O VALOR DO WORK
                                   ; TRANSMITE A TEMPERATURA EM GRAUS
                                   ; CELSIUS PELA USART
   RETURN                          ; RETORNA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               CONFIGURAÇÕES INICIAIS DE HARDWARE E SOFTWARE           *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  NESTA ROTINA SÃO INICIALIZADAS AS PORTAS DE I/O DO MICROCONTROLADOR E AS
;  CONFIGURAÇÕES DOS REGISTRADORES ESPECIAIS (SFR). A ROTINA INICIALIZA A
;  MÁQUINA E AGUARDA O ESTOURO DO WDT.

CONFIG
   CLRF    PORTA                   ; LIMPA O PORTA
   CLRF    PORTB                   ; LIMPA O PORTB
   CLRF    PORTC                   ; LIMPA O PORTC
   CLRF    PORTD                   ; LIMPA O PORTD
   CLRF    PORTE                   ; LIMPA O PORTE


   BANK1                           ; ALTERA PARA O BANCO 1 DA RAM
   MOVLW   B'00101111'
   MOVWF   TRISA                   ; CONFIGURA I/O DO PORTA

   MOVLW   B'00001111'
   MOVWF   TRISB                   ; CONFIGURA I/O DO PORTB

   MOVLW   B'10011001'
   MOVWF   TRISC                   ; CONFIGURA I/O DO PORTC

   MOVLW   B'00000000'
   MOVWF   TRISD                   ; CONFIGURA I/O DO PORTD

   MOVLW   B'00000000'
   MOVWF   TRISE                   ; CONFIGURA I/O DO PORTE

   MOVLW   B'11011011'
   MOVWF   OPTION_REG              ; CONFIGURA OPTIONS
                                   ; PULL-UPs DESABILITADOS
                                   ; INTER. NA BORDA DE SUBIDA DO RB0
                                   ; TIMER0 INCREM. PELO CICLO DE MÁQUINA
                                   ; WDT   - 1:8
                                   ; TIMER - 1:1

   MOVLW   B'01000000'
   MOVWF   INTCON                  ; CONFIGURA INTERRUPÇÕES
                                   ; HABILITA INTER. DE PERIFÉRICOS


   MOVLW   B'00000010'             ; CONFIGURA INTER. DE PERIFÉRICOS
   MOVWF   PIE1                    ; HABILITA A INTERRUPÇÃO DE TMR2

   MOVLW   B'00000111'
   MOVWF   CMCON                   ; DESLIGA OS COMPARADORES

   MOVLW   B'00000100'
   MOVWF   ADCON1                  ; CONFIGURA CONVERSOR A/D
                                   ; RA0, RA1 E RA3 COMO ANALÓGICO
```

```
                                ; RA2, RA4 E RA5 COMO I/O DIGITAL
                                ; PORTE COMO I/O DIGITAL
                                ; JUSTIFICADO À ESQUERDA
                                ; 8 BITS EM ADRESH E 2 BITS EM ADRESL
                                ; Vref+ = VDD (+5V)
                                ; Vref- = GND ( 0V)

   MOVLW   B'00100100'
   MOVWF   TXSTA                ; CONFIGURA USART
                                ; HABILITA TX
                                ; MODO ASSINCRONO
                                ; TRANSMISSÃO DE 8 BITS
                                ; HIGH SPEED BAUD RATE
   MOVLW   .25
   MOVWF   SPBRG                ; ACERTA BAUD RATE -> 9600bps

   MOVLW   .249
   MOVWF   PR2                  ; CONFIGURA PERÍODO DO PWM
                                ; T=((PR2)+1)*4*Tosc*TMR2 Prescale
                                ; T=((249)+1)*4*250ns*4
                                ; T=1,000ms -> 1.000Hz = 1KHz

   BANK0                        ; SELECIONA BANCO 0 DA RAM

   MOVLW   B'10010000'
   MOVWF   RCSTA                ; CONFIGURA USART
                                ; HABILITA RX
                                ; RECEPÇÃO DE 8 BITS
                                ; RECEPÇÃO CONTÍNUA
                                ; DESABILITA ADDRESS DETECT

   MOVLW   B'01001001'
   MOVWF   ADCON0               ; CONFIGURA CONVERSOR A/D
                                ; VELOCIDADE -> Fosc/8
                                ; CANAL 1
                                ; MÓDULO LIGADO

   CLRF    TMR1L
   CLRF    TMR1H                ; ZERA CONTADOR DO TMR1

   MOVLW   B'00000111'
   MOVWF   T1CON                ; CONFIGURA TMR1
                                ; PRESCALE DE 1:1
                                ; TMR1 INCREM. PELO PINO T1CKI (RC0)
                                ; NÃO SINCRONIZADO COM CLOCK INTERNO
                                ; CONTADOR HABILITADO

   MOVLW   B'01001101'
   MOVWF   T2CON                ; CONFIGURA TMR2
                                ; TMR2 HABILITADO
                                ; POSTSCALE 1:10
                                ; PRESCALE 1:4

   CLRF    CCPR2L               ; ZERA PWM DO CCP2 (RC1 - VENTILADOR)

   MOVLW   B'00001111'
   MOVWF   CCP2CON              ; CONFIGURA CCP2 P/ PWM

   CLRF    CCPR1L               ; ZERA PWM DO CCP1 (RC2 - AQUECEDOR)

   MOVLW   B'00001111'
   MOVWF   CCP1CON              ; CONFIGURA CCP1 P/ PWM

;  AS INSTRUÇÕES A SEGUIR FAZEM COM QUE O PROGRAMA TRAVE QUANDO HOUVER UM
;  RESET OU POWER-UP, MAS PASSE DIRETO SE O RESET FOR POR WDT. DESTA FORMA,
;  SEMPRE QUE O PIC É LIGADO, O PROGRAMA TRAVA, AGUARDA UM ESTOURO DE WDT
;  E COMEÇA NOVAMENTE. ISTO EVITA PROBLEMAS NO START-UP DO PIC.

   BTFSC   STATUS,NOT_TO        ; RESET POR ESTOURO DE WATCHDOG TIMER ?
   GOTO    $                    ; NÃO - AGUARDA ESTOURO DO WDT
                                ; SIM
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                          INICIALIZAÇÃO DA RAM                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA IRÁ LIMPAR TODA A RAM DO BANCO 0, INDO DE 0X20 A 0X7F

   MOVLW   0X20
   MOVWF   FSR                     ; APONTA O ENDEREÇAMENTO INDIRETO PARA
                                   ; A PRIMEIRA POSIÇÃO DA RAM
LIMPA_RAM
   CLRF    INDF                    ; LIMPA A POSIÇÃO
   INCF    FSR,F                   ; INCREMENTA O PONTEIRO P/ A PRÓX. POS.
   MOVF    FSR,W
   XORLW   0X80                    ; COMPARA O PONTEIRO COM A ÚLT. POS. +1
   BTFSS   STATUS,Z                ; JÁ LIMPOU TODAS AS POSIÇÕES?
   GOTO    LIMPA_RAM               ; NÃO - LIMPA A PRÓXIMA POSIÇÃO
                                   ; SIM

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                     CONFIGURAÇÕES INICIAIS DO DISPLAY                     *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ESTA ROTINA INICIALIZA O DISPLAY P/ COMUNICAÇÃO DE 8 VIAS, DISPLAY PARA 2
; LINHAS, CURSOR APAGADO E DESLOCAMENTO DO CURSOR À DIREITA.

INICIALIZACAO_DISPLAY
   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS

   MOVLW   0X30                    ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   .3
   CALL    DELAY_MS                ; DELAY DE 3MS (EXIGIDO PELO DISPLAY)

   MOVLW   0X30                    ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   0X30                    ; ESCREVE COMANDO 0X30 PARA
   CALL    ESCREVE                 ; INICIALIZAÇÃO

   MOVLW   B'00111000'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; INTERFACE DE 8 VIAS DE DADOS

   MOVLW   B'00000001'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; LIMPAR TODO O DISPLAY

   MOVLW   .1
   CALL    DELAY_MS                ; DELAY DE 1MS

   MOVLW   B'00001100'             ; ESCREVE COMANDO PARA
   CALL    ESCREVE                 ; LIGAR O DISPLAY SEM CURSOR

   MOVLW   B'00000110'             ; ESCREVE COMANDO PARA INCREM.
   CALL    ESCREVE                 ; AUTOMÁTICO À DIREITA

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                 ROTINA DE ESCRITA DA TELA PRINCIPAL                       *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  ESTA ROTINA ESCREVE A TELA PRINCIPAL DO PROGRAMA, COM AS FRASES:
;  LINHA 1 - "AQUEC. RPS TEMP."
;  LINHA 2 - " 000%  000 000°C"

   MOVLW   0X80                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 0 / COLUNA 0
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS
                                   ; COMANDOS PARA ESCREVER AS
                                   ; LETRAS DE "AQUEC. RPS TEMP."
   MOVLW   'A'
   CALL    ESCREVE
   MOVLW   'Q'
   CALL    ESCREVE
   MOVLW   'U'
   CALL    ESCREVE
   MOVLW   'E'
```

```
    CALL    ESCREVE
    MOVLW   'C'
    CALL    ESCREVE
    MOVLW   '.'
    CALL    ESCREVE
    MOVLW   ' '
    CALL    ESCREVE
    MOVLW   'R'
    CALL    ESCREVE
    MOVLW   'P'
    CALL    ESCREVE
    MOVLW   'S'
    CALL    ESCREVE
    MOVLW   ' '
    CALL    ESCREVE
    MOVLW   'T'
    CALL    ESCREVE
    MOVLW   'E'
    CALL    ESCREVE
    MOVLW   'M'
    CALL    ESCREVE
    MOVLW   'P'
    CALL    ESCREVE
    MOVLW   '.'
    CALL    ESCREVE

    BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
    MOVLW   0XC1                    ; COMANDO PARA POSICIONAR O CURSOR
    CALL    ESCREVE                 ; LINHA 1 / COLUNA 1
    BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS
                                    ; COMANDOS PARA ESCREVER AS
                                    ; LETRAS DE "000%"
    MOVLW   '0'
    CALL    ESCREVE
    MOVLW   '0'
    CALL    ESCREVE
    MOVLW   '0'
    CALL    ESCREVE
    MOVLW   '%'
    CALL    ESCREVE


    BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
    MOVLW   0XC7                    ; COMANDO PARA POSICIONAR O CURSOR
    CALL    ESCREVE                 ; LINHA 1 / COLUNA 7
    BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS
                                    ; COMANDOS PARA ESCREVER AS
                                    ; LETRAS DE "000"
    MOVLW   '0'
    CALL    ESCREVE
    MOVLW   '0'
    CALL    ESCREVE
    MOVLW   '0'
    CALL    ESCREVE

    BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
    MOVLW   0XCB                    ; COMANDO PARA POSICIONAR O CURSOR
    CALL    ESCREVE                 ; LINHA 1 / COLUNA 7
    BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS
                                    ; COMANDOS PARA ESCREVER AS
                                    ; LETRAS DE "000°C"
    MOVLW   '0'
    CALL    ESCREVE
    MOVLW   '0'
    CALL    ESCREVE
    MOVLW   '0'
    CALL    ESCREVE
    MOVLW   0XDF
    CALL    ESCREVE
    MOVLW   'C'
    CALL    ESCREVE
```

```
   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                               LOOP PRINCIPAL                            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  A ROTINA PRINCIPAL FICA AGUARDANDO O MOMENTO DE ESCREVER O VALOR DAS
;  ROTAÇÕES DO VENTILADOR E A TEMPERATURA NO LCD ALÉM DE VARRER O TECLADO
;  PARA MANIPULAR O VALOR DO PWM.

   BSF     ADCON0,GO               ; INICIA CONVERSÃO A/D
                                   ; EXECUTADA APENAS UMA VEZ

   BSF     INTCON,GIE              ; HABILITA FLAG GLOBAL DAS
                                   ; INTERRUPÇÕES

VARRE
   CLRWDT                          ; LIMPA WATCHDOG TIMER

   BTFSC   MOSTRA_RPS              ; DEVE MOSTRAR RPS NO LCD ?
   CALL    MOSTRA_RPS_LCD          ; SIM - CHAMA ROTINA P/ ATUALIZAR RPS
                                   ; NÃO
   BTFSC   MOSTRA_TEMP             ; DEVE MOSTRAR A TEMP. NO LCD ?
   CALL    MOSTRA_TEMP_LCD         ; SIM - CHAMA ROTINA P/ ATUALIZAR TEMP.
                                   ; NÃO

   BSF     LINHA_4                 ; ATIVA LINHA 4 DO TECLADO MATRICIAL

   GOTO    $+1                     ; DELAY PARA ESTABILIZAÇÃO
                                   ; E LEITURA DO TECLADO

   BTFSC   BOTAO_0                 ; O BOTÃO 0 ESTÁ PRESSIONADO ?
   GOTO    TRATA_BOTAO_0           ; SIM - PULA P/ TRATA_BOTAO_0
                                   ; NÃO
   BTFSC   BOTAO_1                 ; O BOTÃO 1 ESTÁ PRESSIONADO ?
   GOTO    TRATA_BOTAO_1           ; SIM - PULA P/ TRATA_BOTAO_1
                                   ; NÃO
   BTFSC   BOTAO_2                 ; O BOTÃO 2 ESTÁ PRESSIONADO ?
   GOTO    TRATA_BOTAO_2           ; SIM - PULA P/ TRATA_BOTAO_2
                                   ; NÃO
   BTFSC   BOTAO_3                 ; O BOTÃO 3 ESTÁ PRESSIONADO ?
   GOTO    TRATA_BOTAO_3           ; SIM - PULA P/ TRATA_BOTAO_3
                                   ; NÃO

   BCF     LINHA_4                 ; DESATIVA LINHA 4 DO TECLADO MATRICIAL

   MOVLW   FILTRO_TECLA            ; CARREGA NO WORK O VALOR DE FILTRO_TECLA
   MOVWF   FILTRO_BOTOES           ; SALVA EM FILTRO_BOTOES
                                   ; RECARREGA FILTRO P/ EVITAR RUIDOS
   MOVLW   .1
   MOVWF   TEMPO_TURBO             ; CARREGA TEMPO DO TURBO DAS TECLAS
                                   ; COM 1 - IGNORA O TURBO A PRIMEIRA
                                   ; VEZ QUE A TECLA É PRESSIONADA

   GOTO    VARRE                   ; VOLTA PARA VARRER TECLADO

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                        TRATAMENTO DOS BOTÕES                            *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

; ************************ TRATAMENTO DO BOTÃO 0 ***************************

TRATA_BOTAO_0
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO

   DECFSZ  TEMPO_TURBO,F           ; FIM DO TEMPO DE TURBO ?
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM
   MOVLW   TURBO_TECLA
   MOVWF   TEMPO_TURBO             ; RECARREGA TEMPORIZADOR DO TURBO
                                   ; DAS TECLAS
```

```
   MOVLW   .100
   XORWF   INTENSIDADE_AQUE,W
   BTFSS   STATUS,Z                ; PODE INCREMENTAR PWM DO AQUECEDOR ?
   INCF    INTENSIDADE_AQUE,F      ; SIM - INCREMENTA INTENSIDADE_AQUE
                                   ; NÃO

   MOVF    INTENSIDADE_AQUE,W      ; CARREGA INTENSIDADE_AQUE NO WORK
   MOVWF   mulplr                  ; CARREGA WORK EM mulplr

   MOVLW   .10
   MOVWF   mulcnd                  ; CARREGA 10d EM mulcnd

   CALL    mpy_F                   ; CHAMA ROTINA DE MULTIPLICAÇÃO

   SWAPF   L_byte,W
   ANDLW   B'00110000'
   IORLW   B'00001111'
   RRF     H_byte,F
   RRF     L_byte,F
   RRF     H_byte,F
   MOVWF   CCP1CON
   RRF     L_byte,W
   MOVWF   CCPR1L                  ; ATUALIZA REGISTRADORES DO DUTY CYCLE
                                   ; DO MÓDULO CCP1 - PWM DO AQUECEDOR

   MOVF    INTENSIDADE_AQUE,W      ; FAZ O AJUSTE DECIMAL DA INTENSIDADE
   CALL    AJUSTE_DECIMAL          ; DO PWM DO AQUECEDOR

   CLRF    TEC_MATRICIAL           ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XC1                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 1
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

   MOVF    CENTENA,W
   ADDLW   0X30                    ; FAZ AJUSTE ASCII DA CENTENA
   CALL    ESCREVE                 ; ESCREVE VALOR NO LCD

   MOVF    DEZENA,W
   ADDLW   0X30                    ; FAZ AJUSTE ASCII DA DEZENA
   CALL    ESCREVE                 ; ESCREVE VALOR NO LCD

   MOVF    UNIDADE,W
   ADDLW   0X30                    ; FAZ AJUSTE ASCII DA UNIDADE
   CALL    ESCREVE                 ; ESCREVE VALOR NO LCD

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************* TRATAMENTO DO BOTÃO 1 ***************************

TRATA_BOTAO_1
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO

   DECFSZ  TEMPO_TURBO,F           ; FIM DO TEMPO DE TURBO ?
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM
   MOVLW   TURBO_TECLA
   MOVWF   TEMPO_TURBO             ; RECARREGA TEMPORIZADOR DO TURBO
                                   ; DAS TECLAS

   MOVF    INTENSIDADE_AQUE,F
   BTFSS   STATUS,Z                ; PODE DECREMENTAR PWM DO AQUECEDOR ?
   DECF    INTENSIDADE_AQUE,F      ; SIM - DECREMENTA INTENSIDADE_AQUE
                                   ; NÃO

   MOVF    INTENSIDADE_AQUE,W      ; CARREGA INTENSIDADE_AQUE NO WORK
```

```
   MOVWF   mulplr                  ; CARREGA WORK EM mulplr

   MOVLW   .10
   MOVWF   mulcnd                  ; CARREGA 10d EM mulcnd

   CALL    mpy_F                   ; CHAMA ROTINA DE MULTIPLICAÇÃO

   SWAPF   L_byte,W
   ANDLW   B'00110000'
   IORLW   B'00001111'
   RRF     H_byte,F
   RRF     L_byte,F
   RRF     H_byte,F
   MOVWF   CCP1CON
   RRF     L_byte,W
   MOVWF   CCPR1L                  ; ATUALIZA REGISTRADORES DO DUTY CYCLE
                                   ; DO MÓDULO CCP1 – PWM DO AQUECEDOR

   MOVF    INTENSIDADE_AQUE,W      ; FAZ O AJUSTE DECIMAL DA INTENSIDADE
   CALL    AJUSTE_DECIMAL          ; DO PWM DO AQUECEDOR

   CLRF    TEC_MATRICIAL           ; DESATIVA TODAS AS LINHAS DO TECLADO

   BCF     RS                      ; SELECIONA O DISPLAY P/ COMANDOS
   MOVLW   0XC1                    ; COMANDO PARA POSICIONAR O CURSOR
   CALL    ESCREVE                 ; LINHA 1 / COLUNA 1
   BSF     RS                      ; SELECIONA O DISPLAY P/ DADOS

   MOVF    CENTENA,W
   ADDLW   0X30                    ; FAZ AJUSTE ASCII DA CENTENA
   CALL    ESCREVE                 ; ESCREVE VALOR NO LCD

   MOVF    DEZENA,W
   ADDLW   0X30                    ; FAZ AJUSTE ASCII DA DEZENA
   CALL    ESCREVE                 ; ESCREVE VALOR NO LCD

   MOVF    UNIDADE,W
   ADDLW   0X30                    ; FAZ AJUSTE ASCII DA UNIDADE
   CALL    ESCREVE                 ; ESCREVE VALOR NO LCD

   CLRF    DISPLAY                 ; LIMPA BARRAMENTO DE DADOS

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************* TRATAMENTO DO BOTÃO 2 ***************************

TRATA_BOTAO_2
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO – VOLTA P/ VARRE
                                   ; SIM – BOTÃO PRESSIONADO

   DECFSZ  TEMPO_TURBO,F           ; FIM DO TEMPO DE TURBO ?
   GOTO    VARRE                   ; NÃO – VOLTA P/ VARRE
                                   ; SIM
   MOVLW   TURBO_TECLA
   MOVWF   TEMPO_TURBO             ; RECARREGA TEMPORIZADOR DO TURBO
                                   ; DAS TECLAS
   MOVLW   .100
   XORWF   INTENSIDADE_VENT,W
   BTFSS   STATUS,Z                ; PODE INCREMENTAR PWM DO VENTILADOR ?
   INCF    INTENSIDADE_VENT,F      ; SIM – INCREMENTA INTENSIDADE_VENT
                                   ; NÃO

   MOVF    INTENSIDADE_VENT,W      ; CARREGA INTENSIDADE_VENT NO WORK
   MOVWF   mulplr                  ; CARREGA WORK EM mulplr

   MOVLW   .10
   MOVWF   mulcnd                  ; CARREGA 10d EM mulcnd

   CALL    mpy_F                   ; CHAMA ROTINA DE MULTIPLICAÇÃO

   SWAPF   L_byte,W
```

```
   ANDLW   B'00110000'
   IORLW   B'00001111'
   RRF     H_byte,F
   RRF     L_byte,F
   RRF     H_byte,F
   MOVWF   CCP2CON
   RRF     L_byte,W
   MOVWF   CCPR2L                  ; ATUALIZA REGISTRADORES DO DUTY CYCLE
                                   ; DO MÓDULO CCP2 - PWM DO VENTILADOR

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTÕES

; ************************ TRATAMENTO DO BOTÃO 3 ***************************

TRATA_BOTAO_3
   DECFSZ  FILTRO_BOTOES,F         ; FIM DO FILTRO ? (RUIDO?)
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM - BOTÃO PRESSIONADO

   DECFSZ  TEMPO_TURBO,F           ; FIM DO TEMPO DE TURBO ?
   GOTO    VARRE                   ; NÃO - VOLTA P/ VARRE
                                   ; SIM
   MOVLW   TURBO_TECLA
   MOVWF   TEMPO_TURBO             ; RECARREGA TEMPORIZADOR DO TURBO
                                   ; DAS TECLAS

   MOVF    INTENSIDADE_VENT,F
   BTFSS   STATUS,Z                ; PODE DECREMENTAR PWM DO VENTILADOR ?
   DECF    INTENSIDADE_VENT,F      ; SIM - DECREMENTA INTENSIDADE_VENT
                                   ; NÃO

   MOVF    INTENSIDADE_VENT,W      ; CARREGA INTENSIDADE_VENT NO WORK
   MOVWF   mulplr                  ; CARREGA WORK EM mulplr

   MOVLW   .10
   MOVWF   mulcnd                  ; CARREGA 50d EM mulcnd

   CALL    mpy_F                   ; CHAMA ROTINA DE MULTIPLICAÇÃO

   SWAPF   L_byte,W
   ANDLW   B'00110000'
   IORLW   B'00001111'
   RRF     H_byte,F
   RRF     L_byte,F
   RRF     H_byte,F
   MOVWF   CCP2CON
   RRF     L_byte,W
   MOVWF   CCPR2L                  ; ATUALIZA REGISTRADORES DO DUTY CYCLE
                                   ; DO MÓDULO CCP2 - PWM DO VENTILADOR

   GOTO    VARRE                   ; VOLTA P/ VARREDURA DOS BOTOES

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *               TABELAS DE CONVERSAO P/ TEMPERATURA DO DIODO             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   ORG     0X200                   ; POSICIONA O INICIO DA TABELA EM 0X400
   RADIX   DEC                     ; CONFIGURA RADIX EM DECIMAL

TABELA_TEMPERATURA
   MOVLW   HIGH TABELA_TEMP
   MOVWF   PCLATH                  ; ACERTA VALOR DO PCLATH
   MOVLW   LOW TABELA_TEMP         ; CARREGA NO WORK PARTE BAIXA DO PC
   ADDWF   TEMPERATURA,W           ; SOMA AO DESLOCAMENTO (TEMPERATURA)
   BTFSC   STATUS,C                ; HOUVE ESTOURO ?
   INCF    PCLATH,F                ; SIM - INCREMENTA PCLATH (PARTE ALTA DO PC)
   MOVWF   PCL                     ; MOVE WORK P/ PCL
                                   ; PROVOCA UM SALTO

TABELA_TEMP
     DT      000,000,000,000,000,000,000,000,000,000,000,000,000,000,010,000 ;15
     DT      000,000,000,000,000,000,000,000,000,000,000,000,000,000,001,001 ;31
```

```
     DT      002,002,003,003,004,004,005,005,006,006,007,007,008,008,009,009 ;47
     DT      010,010,011,011,012,012,013,013,014,014,015,015,016,016,017,017 ;63

     DT      018,018,019,019,020,020,021,021,022,022,023,023,023,024,024,025 ;79
     DT      025,026,026,027,027,028,028,029,029,030,030,031,031,032,032,033 ;95
     DT      033,034,034,035,035,036,036,037,037,038,038,039,039,040,040,041 ;111
     DT      041,042,042,043,043,044,044,045,045,046,046,047,047,048,048,049 ;127

     DT      049,050,050,051,051,052,052,053,053,054,054,055,055,056,056,057 ;143
     DT      057,058,058,059,059,060,060,061,061,062,062,063,063,064,064,065 ;159
     DT      065,066,066,067,067,068,068,069,069,070,070,071,071,072,072,073 ;175
     DT      073,074,074,075,075,076,076,077,077,078,078,079,079,080,080,081 ;191

     DT      081,082,082,083,083,084,084,085,085,086,086,087,087,088,088,089 ;207
     DT      089,090,090,091,091,092,092,093,093,094,094,095,095,096,096,097 ;223
     DT      097,098,098,099,099,100,100,101,101,102,102,103,103,104,104,104 ;239
     DT      105,105,106,106,107,107,108,108,109,109,110,110,111,111,112,112 ;255

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *                              FIM DO PROGRAMA                                *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   END                              ; FIM DO PROGRAMA
```

Dicas e Comentários

Para converter a temperatura do diodo lida pelo conversor A/D na temperatura ambiente em graus Celsius utilizou-se uma tabela de conversão, do mesmo tipo da utilizada para converter um número BCD numa representação gráfica para displays de 7 segmentos. Porém, como a tabela é relativamente extensa (possui 256 posições, relativos aos 8 bits da conversão A/D), para economizar espaço na listagem do software utilizou-se a diretriz DT. O compilador interpreta a diretriz substituindo cada valor que a sucede por uma instrução RETLW. Assim, o código fonte é exatamente o mesmo que se fosse utilizada a instrução RETLW, porém, ao invés de escrever 256 vezes a instrução RETLW e o valor de retorno, utilizou-se a diretriz. Em cada linha foram passados 16 argumentos, desta forma, no exemplo da experiência 20, o compilador substitui cada linha com a instrução DT por 16 instruções RETLW. Este recurso não altera em nada o código fonte e o funcionamento do programa.

Exercícios Propostos

1. Fazer um software para controlar a temperatura do diodo. Este software deve medir a temperatura e alterar o valor do PWM da resistência para manter a temperatura do diodo em constante em 50ºC.
2. Fazer um software para controlar a velocidade de rotação do ventilador. Este software deve medir rotação e alterar o valor do PWM a fim de manter a rotação constante.

Capítulo 23 - Software de comunicação serial SDCom

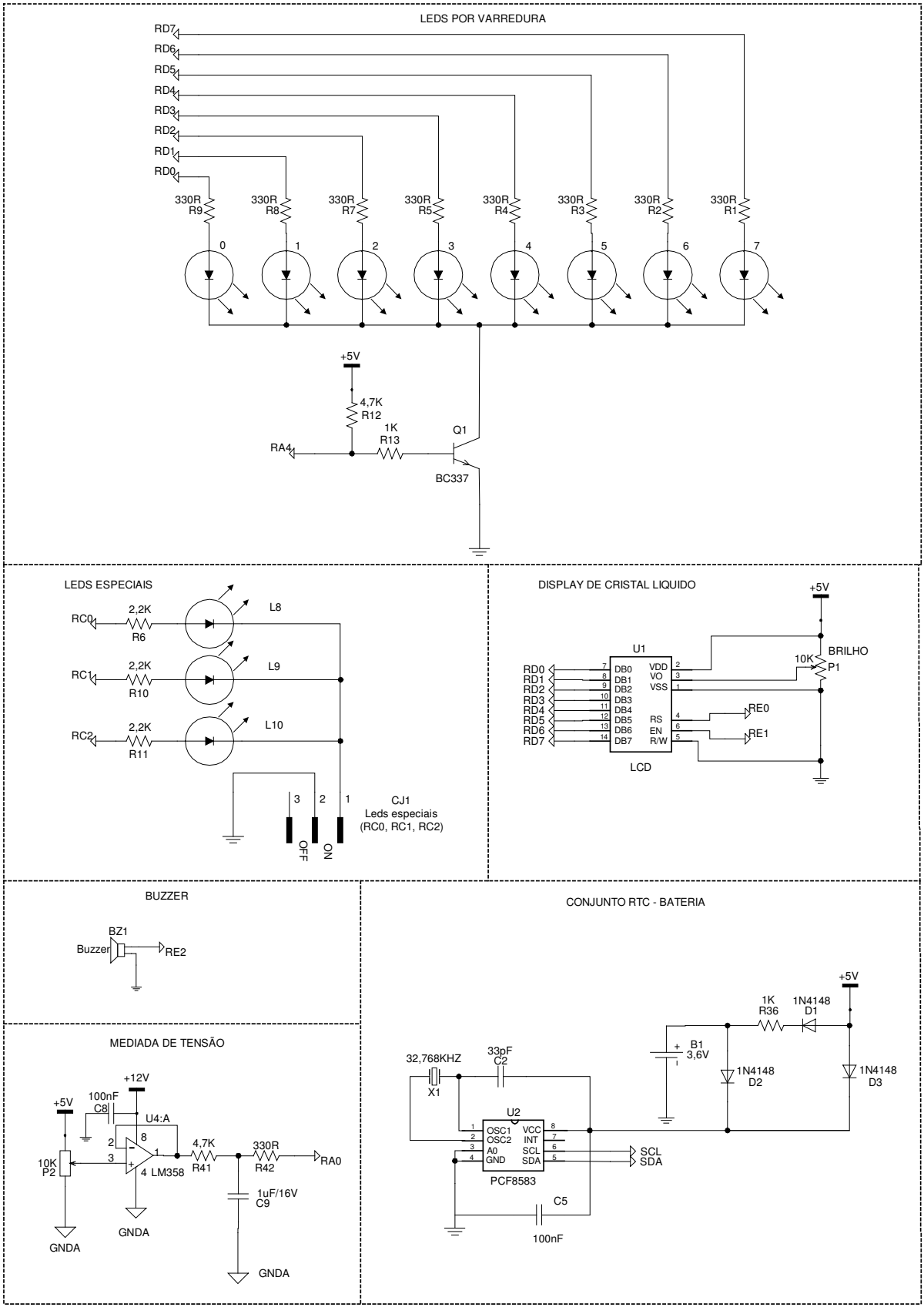
O software SDCom disponível no CD pode ser utilizado para comunicação entre o PC e o MCMASTER. O software é útil para testar as experiências de número 17 e 20.

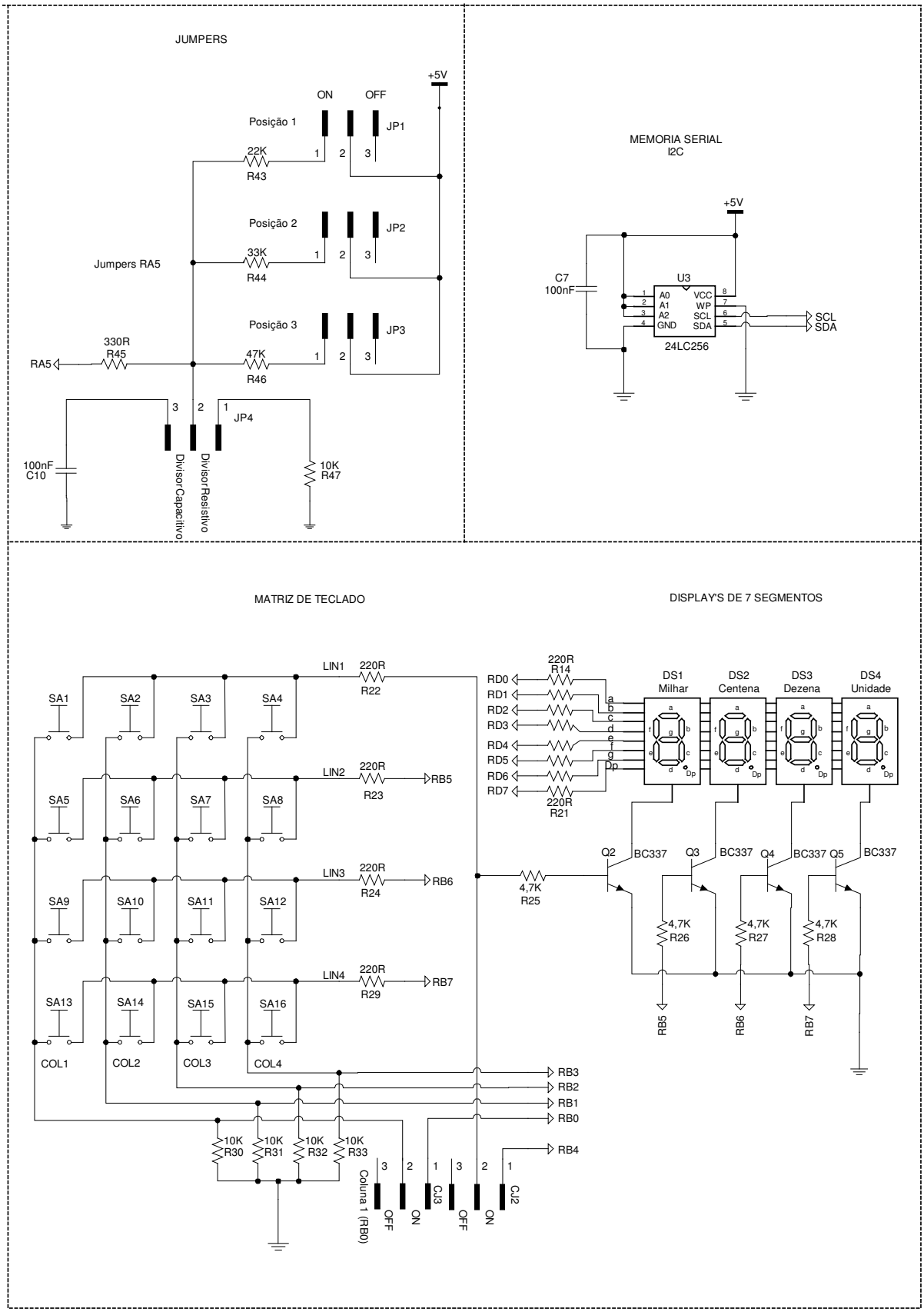
O software foi desenvolvido sob a plataforma windows é deve ser instalado através do arquivo setup.exe. Pode ocorrer dele pedir para atualizar alguns arquivos do sistema. Neste caso, é aconselhável que o próprio software execute esta tarefa. O micro será reiniciado e o arquivo setup.exe deverá ser executado novamente pelo usuário.

Para a utilização do software deve-se inicialmente selecionar a porta COM de comunicação e o baud rate desejado.

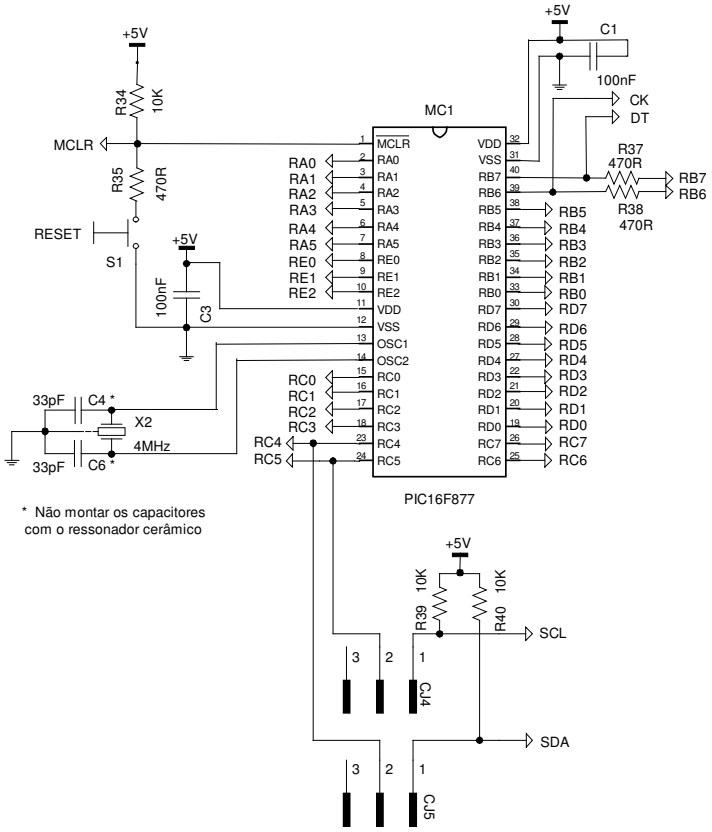
Capítulo 24 - Software demo para teste do hardware

A fim de validar o hardware do MCMASTER, servindo como uma giga de testes, é fornecido também um software que pode ser utilizado para testar a funcionabilidade de quase todos os recursos do sistema. Para este software, não é fornecido o código fonte apenas o arquivo .HEX está disponível no CD. Como padrão este software já vem gravado no microcontrolador, porém a qualquer momento o usuário pode testar o funcionamento do hardware do MCMASTER regravando o arquivo .HEX. O software de teste pode ser executado sem interação com o usuário, porém recomendamos que o usuário faça a interação com o software a fim comprovar o correto funcionamento de todos os componentes do sistema.

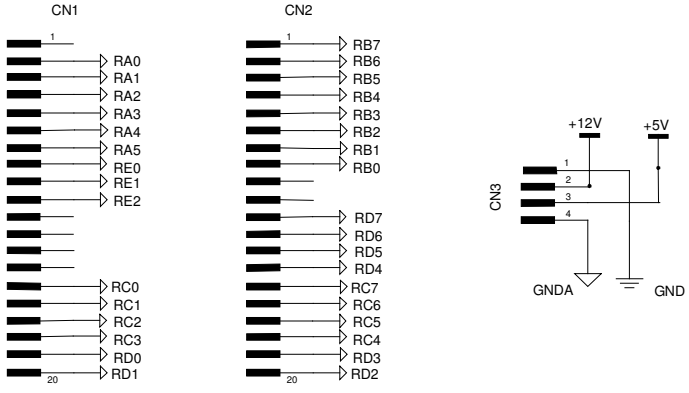

LEDS POR VARREDURA
RD7
RD6
RD5
RD4
RD3
RD2
RD1
RD0
330R R9
330R R8
330R R7
330R R5
330R R4
330R R3
330R R2
330R R1
0
1
2
3
4
5
6
7
+5V
4,7K R12
1K R13
Q1
RA4
BC337
LEDS ESPECIAIS
RC0
RC1
RC2
2,2K R6
2,2K R10
2,2K R11
L8
L9
L10
3
2
1
CJ1
Leds especiais
(RC0, RC1, RC2)
OFF
ON
DISPLAY DE CRISTAL LIQUIDO
+5V
U1
RD0
RD1
RD2
RD3
RD4
RD5
RD6
RD7
DB0
DB1
DB2
DB3
DB4
DB5
DB6
DB7
VDD
VO
VSS
RS
EN
R/W
10K P1
BRILHO
RE0
RE1
LCD
BUZZER
BZ1
Buzzer
RE2
MEDIADA DE TENSÃO
+5V
+12V
100nF C8
U4:A
10K P2
4,7K R41
330R R42
RA0
LM358
1uF/16V C9
GNDA
CONJUNTO RTC - BATERIA
+5V
1K R36
1N4148 D1
B1 3,6V
1N4148 D2
1N4148 D3
32,768KHZ X1
33pF C2
U2
OSC1
OSC2
A0
GND
VCC
INT
SCL
SDA
PCF8583
C5
100nF


JUMPERS
+5V
ON
OFF
Posição 1
JP1
22K
R43
Posição 2
JP2
33K
R44
Jumpers RA5
Posição 3
JP3
330R
R45
RA5
47K
R46
JP4
100nF
C10
Divisor Capacitivo
Divisor Resistivo
10K
R47
MEMORIA SERIAL
I2C
+5V
C7
100nF
U3
A0
A1
A2
GND
VCC
WP
SCL
SDA
24LC256
SCL
SDA
MATRIZ DE TECLADO
DISPLAY'S DE 7 SEGMENTOS
LIN1
LIN2
LIN3
LIN4
220R
R22
220R
R23
220R
R24
220R
R29
SA1
SA2
SA3
SA4
SA5
SA6
SA7
SA8
SA9
SA10
SA11
SA12
SA13
SA14
SA15
SA16
COL1
COL2
COL3
COL4
RB5
RB6
RB7
RB3
RB2
RB1
RB0
RB4
10K
R30
10K
R31
10K
R32
10K
R33
Coluna 1 (RB0)
CJ3
CJ2
OFF
ON
220R
R14
220R
R21
RD0
RD1
RD2
RD3
RD4
RD5
RD6
RD7
a
b
c
d
e
f
g
Dp
DS1
Milhar
DS2
Centena
DS3
Dezena
DS4
Unidade
Q2
Q3
Q4
Q5
BC337
4,7K
R25
4,7K
R26
4,7K
R27
4,7K
R28

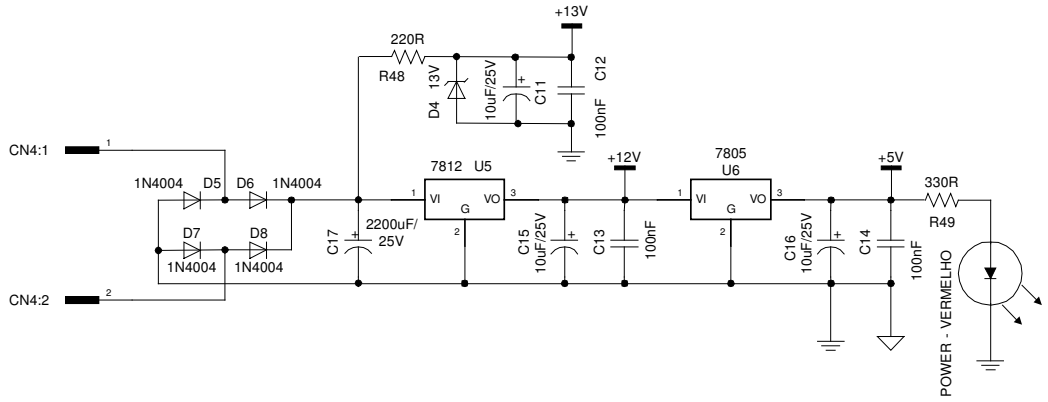
MICROCONTROLADOR



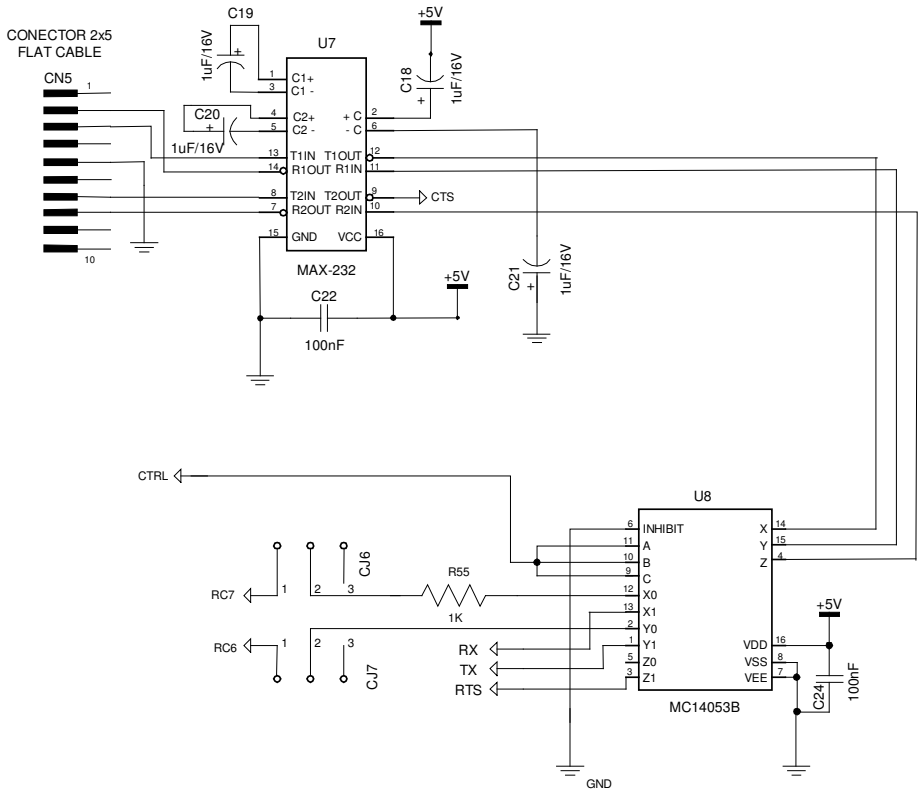
CONECTOR DE EXPANSÃO



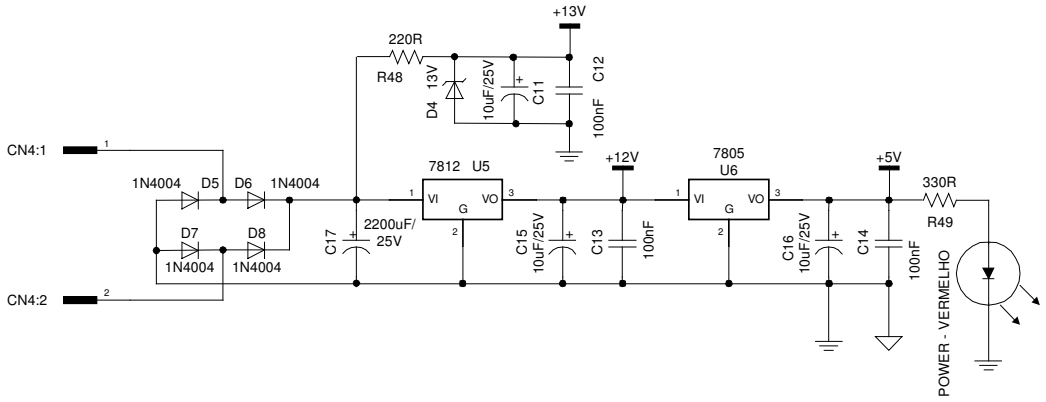
FONTE



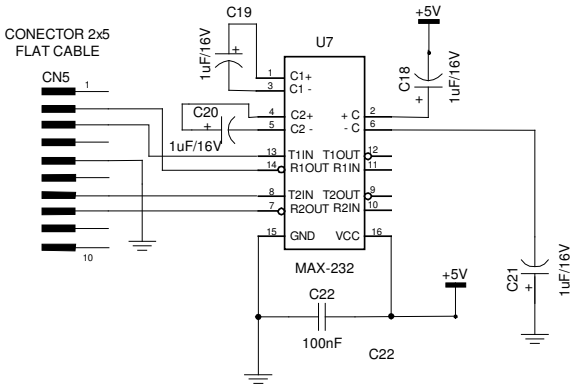
COMUNICAÇÃO RS-232

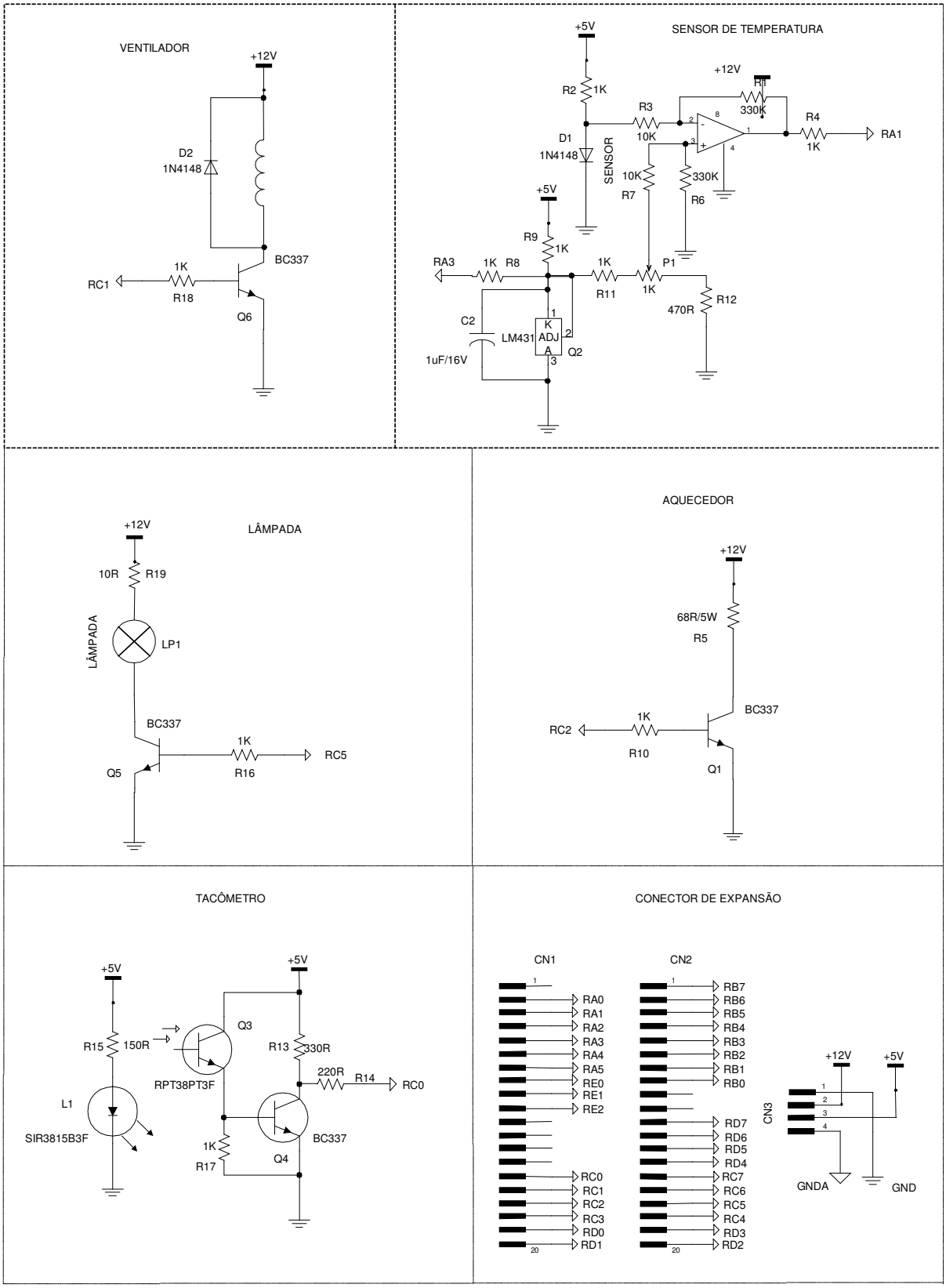


FONTE



COMUNICAÇÃO RS-232




VENTILADOR
+12V
D2
1N4148
BC337
1K
RC1
R18
Q6
SENSOR DE TEMPERATURA
+5V
R2 1K
+12V
RT
330K
R3
10K
R4
1K
RA1
D1
1N4148
SENSOR
10K
R7
330K
R6
+5V
R9
1K
RA3
1K R8
1K
R11
P1
1K
R12
470R
C2
1uF/16V
LM431
K
ADJ
A
Q2
LÂMPADA
+12V
10R R19
LÂMPADA
LP1
BC337
1K
RC5
R16
Q5
AQUECEDOR
+12V
68R/5W
R5
BC337
1K
RC2
R10
Q1
TACÔMETRO
+5V
+5V
R15 150R
Q3
R13 330R
220R
R14
RC0
RPT38PT3F
L1
SIR3815B3F
BC337
1K
R17
Q4
CONECTOR DE EXPANSÃO
CN1
CN2
RA0
RA1
RA2
RA3
RA4
RA5
RE0
RE1
RE2
RC0
RC1
RC2
RC3
RD0
RD1
RB7
RB6
RB5
RB4
RB3
RB2
RB1
RB0
RD7
RD6
RD5
RD4
RC7
RC6
RC5
RC4
RD3
RD2
CN3
+12V
+5V
GNDA
GND

"PARABÉNS; VOCÊ ACABA DE ADQUIRIR O KIT McMASTER PARA MICROCONTROLADORES PIC DA LABTOOLS"

1. Tempo de Garantia

A Labtools garante contra defeitos de fabricação durante 4 meses para mão de obra de conserto.

O prazo de garantia começa a ser contado a partir da emissão do pedido de venda.

2. Condições de Garantia

Durante o prazo coberto pela garantia, à Labtools fará o reparo do defeito apresentado, ou substituirá o produto, se isso for necessário.

Os produtos deverão ser encaminhados à Labtools, devidamente embalados por conta e risco do comprador, e acompanhados deste Certificado de Garantia "sem emendas ou rasuras" e da respectiva Nota Fiscal de aquisição.

O atendimento para reparos dos defeitos nos produtos cobertos por este Certificado de Garantia será feito somente na Labtools, ficando, portanto, excluído o atendimento domiciliar.

3. Exclusões de Garantia

Estão excluídos da garantia os defeitos provenientes de:

Alterações do produto ou dos equipamentos.

Utilização incorreta do produto ou dos equipamentos.

Queda, raio, incêndio ou descarga elétrica.

Manutenção efetuada por pessoal não credenciado pela Labtools.

Obs.: Todas as características de funcionamento dos produtos Labtools estão em seus respectivos manuais.

4. Limitação de Responsabilidade

A presente garantia limita-se apenas ao reparo do defeito apresentado, a substituição do produto ou equipamento defeituoso.Nenhuma outra garantia, implícita

ou explícita, é dada ao comprador.

A Labtools não se responsabiliza por qualquer dano, perda, inconveniência ou prejuízo direto ou indireto que possa advir de uso ou inabilidade de se usarem os produtos cobertos por esta garantia.

A Labtools estabelece o prazo de 30 dias (a ser contado a partir da data da nota Fiscal de Venda) para que seja reclamado qualquer eventual falta de componentes.

Importante: Todas as despesas de frete e seguro são de responsabilidade do usuário, ou seja, em caso de necessidade o Cliente é responsável pelo encaminhamento do equipamento até a Labtools.

jul/2006