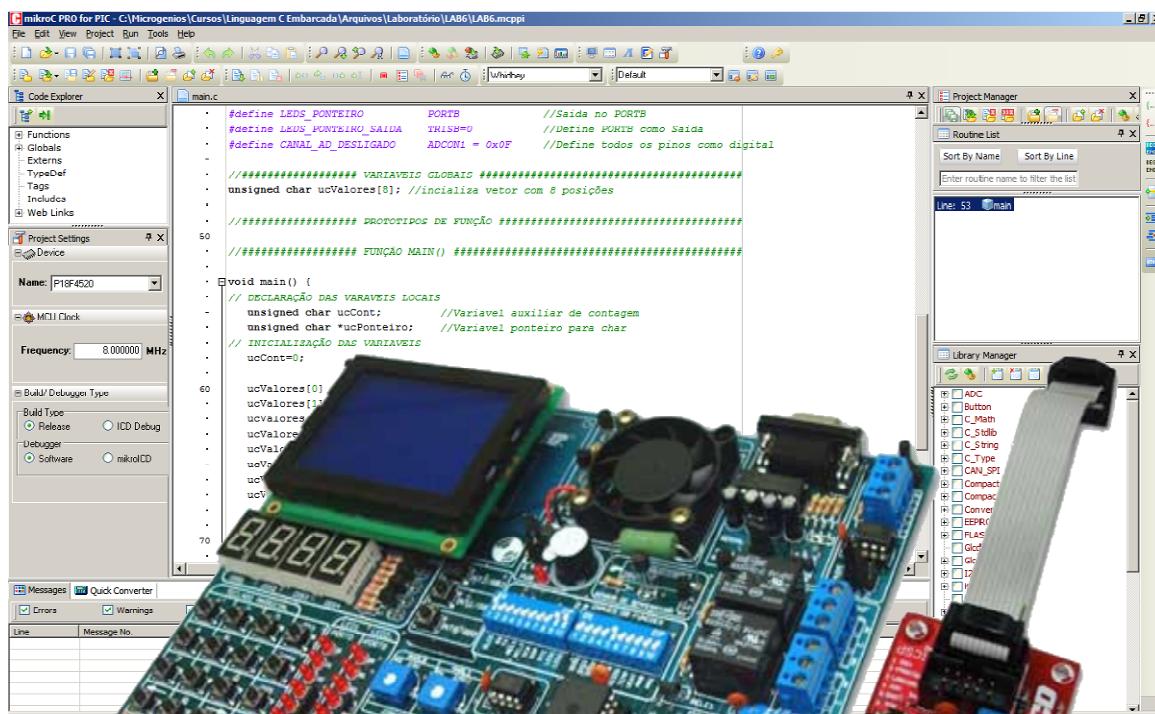


Microcontroladores PIC Família PIC18F Microchip



Autor: Fernando Simplicio de Sousa
 Fabio Perkowitsch Mulero
 Equipe Microgenios

Cursos e Treinamentos sobre Microcontroladores.

www.microgenios.com.br
www.portalwebaula.com.br

Fone: 11 5084-4518 | 3384-5598

Livro:
Microcontroladores PIC - Família PIC18F Microchip

Todos os direitos reservados. Proibida a reprodução total ou parcial, por qualquer meio ou processo, especialmente por sistemas gráficos, microfílmicos, fotográficos, reprodutivos, fonográficos, videográficos, internet, e-books. Vedada a memorização e/ou recuperação total ou parcial em qualquer sistema de processamento de dados e a inclusão de qualquer parte da obra em qualquer programa juscibernético. Essas proibições aplicam-se também às características gráficas da obra e a sua editoração. A violação dos direitos autorais é punível como crime (art. 184 e parágrafos, do código penal, cf. Lei nº 6. 895, de 17.12.80) com pena de prisão e multa, conjuntamente com busca e apreensão e indenizações diversas (artigos 102, 103 parágrafo único, 104, 105, 106 e 107 itens 1, 2, 3 da lei nº. 9.610, de 19/06/98, lei dos direitos autorais).

Eventuais erratas estarão disponíveis no site da Microgenios para download.

Dedicatória:

Dedico esse livro a Minha família e a equipe Microgenios

Advertência:

As informações e o material contido neste livro são fornecidos sem nenhuma garantia quer explícita, ou implícita, de que o uso de tais informações conduzirá sempre ao resultado desejado. Tanto o editor quanto o autor não podem ser responsabilizados por qualquer tipo de reivindicação atribuída a erros, omissões ou qualquer outra imprecisão na informação ou material fornecido neste livro, e em nenhuma hipótese podem ser incriminados direta ou indiretamente por qualquer dano, perda, lucros cessantes, etc., devido ao uso destas informações.

Prefácio

Esta obra foi concebida com o intuito de preparar os estudantes, professores e profissionais da área técnica para a criação de projetos com os microcontroladores da família PIC, utilizando como ferramenta uma linguagem de programação de alto nível, neste material escolhemos para abordar a linguagem C, que é uma das linguagens mais poderosas e portáveis, fato este que a tornou amplamente utilizada, primeiramente para a criação de programas aplicativos para PC e mais tarde em sistemas embarcados microcontrolados.

Trabalhar com uma linguagem de alto nível, como C, para criar programas para microcontroladores, exige do profissional além de um bom conhecimento de lógica de programação e habilidade com a linguagem, um sólido conhecimento da estrutura de hardware do microcontrolador utilizado, de forma a extrair deste o máximo de funcionalidade de seus periféricos internos.

Esta obra estuda paralelamente hardware e software, propiciando um conhecimento completo ao profissional e tornando-o apto a desenvolver suas próprias aplicações, além disso, vale a pena ressaltar a preocupação, por parte do Centro de Tecnologia Microgenios, em priorizar um estudo gradual e prático, para isso usamos os kits de desenvolvimento PICgenios PIC18F, como base para a realização de diversas experiências que complementam e fixam o aprendizado.

Um ponto de destaque da abordagem do treinamento é o uso e detalhamento da IDE de desenvolvimento MikroC PRO (www.mikroe.com) a qual apesar das limitações da versão de demonstração gratuita mostra-se uma excelente ferramenta de desenvolvimento e simulação.

De maneira alguma este material é apresentado como única fonte de estudo sobre o assunto, devendo aqueles que necessitarem se aprofundar nos tópicos aqui estudados buscar outras fontes de pesquisa.

Por fim a equipe Microgenios agradece a atenção de todos e deseja bons estudos e projetos.

Fernando Simplicio de Sousa
Fabio Perkowitsch Mulero

Equipe Microgenios

Cursos e Treinamentos sobre Microcontroladores.

www.microgenios.com.br
www.portalwebaula.com.br

Fone: 11 5084-4518 | 3384-5598

Deus seja louvado!

UNIDADE 1 - O MICROCONTROLADOR PIC	6
Desempenho da família PIC	6
Tipos de Memória de Programa	7
Tipos de memórias de programas disponíveis nos PICs:	7
O PIC18F4520.....	7
Veja em seguida as principais características do PI18F4520	7
Tipos de encapsulamentos:	7
Estrutura interna do PIC18F4520	9
Descrição das funções dos pinos do PI18F4520	10
Tipos de Memórias	11
Vamos exemplificar um processo de gravação na memória de programa do PIC	13
Introdução as portas de I/O.....	16
0 PORTA.....	17
Registrador PORTA	18
PORTA	18
Registrador TRISA	18
Leitura do LATCH de um pino do PIC	18
0 PORTB.....	19
0 PORTC.....	19
0 PORTD.....	20
0 PORTE.....	20
Reset.....	21
Os PICs possuem diversos tipos de reset:	21
Ciclos de máquina.....	21
Qual é a freqüência real de execução das instruções do nosso microcontrolador?	21
Os bits de configuração (fusíveis)	22
Mapa dos bits de configuração do PIC18F4520	23
Configuração de Clock.....	24
Vamos conhecer cada um dos tipos de osciladores:	25
Configuração dos Fusíveis de Energia	27
UNIDADE 2 - CANAIS A/D	30
Conversor A/D do PIC.....	30
Registrador ADCON0:	30
ADCON0:	30
Registrador ADCON1:	31
ADCON1:	31
Registrador ADCON2:	32
ADCON2:	32
Trabalhando com AD no MikroC	33
UNIDADE 3 – CANAL PWM DO PIC	34
Introdução	34
Trabalhando com PWM no PIC	36
UNIDADE 4 - TIMERS/COUNTERS	38
Os Timers/Counters	38
TIMER0.....	38
Os Registradores relacionados ao TIMER0 são:	39
T0CON: (TIMER0 Counter Register)	40
O registrador INTCON.....	41
INTCON (Interrupt Control)	41
TIMER1.....	44
O TIMER1:	44
T1CON: (TIMER1 CONTROL REGISTER):	45
Programando o TIMER1 do PIC	47
Configuração do TIMER1:	49
Calculo de Estouro do TIMER1:	49
TIMER2 :	49
T2CON (TIMER2 CONTROL REGISTER): Configura o setup do TIMER2;	50
Registradores de configuração do TIMER2:	50
TIMER3.....	51
O TIMER3 :	51
Registradores de configuração do TIMER3:	52

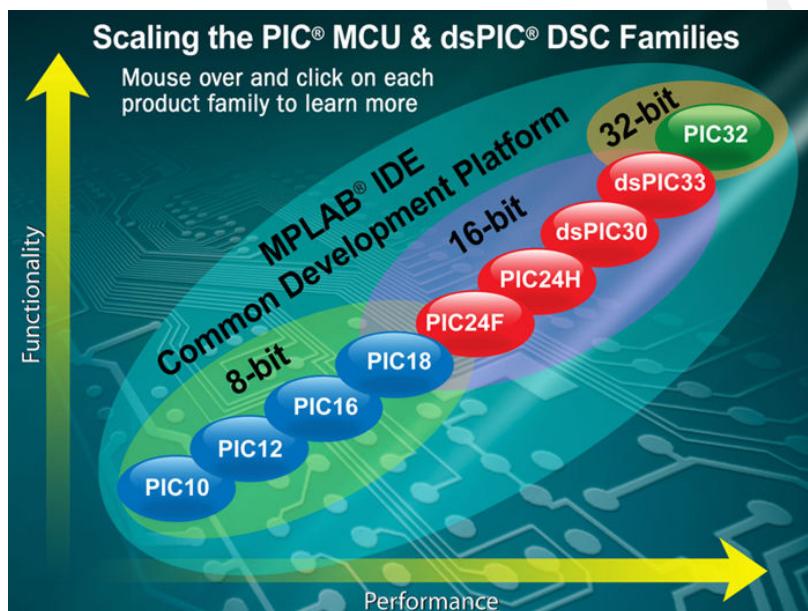
UNIDADE 5 – PROGRAMANDO AS INTERRUPÇÕES DO PIC	53
As Interrupções do PIC18F4520	53
Introdução.....	54
Como funciona as interrupções?	55
Registradores que são usados para controlar as interrupções;	56
UNIDADE 6 – PROGRAMANDO A INTERRUPÇÃO EXTERNA	61
INT0	61
Vamos conhecer os registradores relacionados a INT0:	62
Registradores responsáveis pelas configurações gerais das interrupções:	62
Registradores responsáveis pela habilitação da interrupção externa INT0	62
Registrador responsável pelo tipo de disparo de INT0:	62
INT1	62
Programando a interrupção externa INT1	62
Vamos conhecer os registradores relacionados a INT1:	63
Registradores responsáveis pelas configurações gerais das interrupções:	63
Registradores responsáveis pela habilitação da interrupção externa INT1	63
INT2	63
Registrador responsável pelo tipo de disparo de INT2:	63
Vamos conhecer os registradores relacionados a INT2:	64
Registradores responsáveis pela habilitação da interrupção externa INT2	64
Registrador responsável pelo tipo de disparo de INT2:	64
Interrupção por mudança de estado	64
Vamos conhecer os registradores relacionados a interrupção RB:	65
Registradores responsáveis pelas configurações gerais das interrupções:	65
Registradores responsáveis pela habilitação da interrupção externa RB	65
UNIDADE 7 – EXEMPLOS DE PERIFÉRICOS	66
Controle de display LCD	66
Projetos com displays LCD:	66
Varredura de displays de 7 segmentos	67
Projetos com displays de 7 segmentos:	67
Varredura de Teclado matricial	67
Projetos com teclados matriciais:	68
Acionamento de Leds	68
Projetos com Leds	68
Conversor Analógico digital (A/D)	68
Projetos com os conversores A/D do PIC	69
Controle PWM	69
Projetos com os PWM	69
UNIDADE 8 — ANEXOS	70
Descrição das Pinagens(LCD)	70
Lista de códigos dos Caracteres	70
Tabela com o conjunto completo de instruções:	71
Tabela com as instruções mais comuns:	72
Resumo com as instruções mais úteis:	72
Comandos LCD do mikroC PRO (LCD)	73
UNIDADE 9 — FIGURAS	73
UNIDADE 10 — TABELAS	74

Unidade 1 - O Microcontrolador PIC

Os microcontroladores PIC são fabricados pela empresa Microchip. Existem basicamente quatro famílias de PICs diferenciadas pelo tamanho da palavra de memória de programa: 12, 14, 16 e 32 bits. Atualmente a Microchip lançou nesses últimos anos uma nova família de microcontroladores chamada dsPIC que possui barramento interno de 16 bits e 24 bits, contrário da tradicional família de 8 bits, e também a família PIC32, com barramento de dados em 32 bits.

O PIC possui uma arquitetura interna do tipo Harvard. A diferença entre essa arquitetura e as tradicionais, do tipo Von-Neumann, é que ela possui um barramento para o programa e outro para os dados, diferente da arquitetura tradicional em que um barramento é tanto de dados como de endereço.

O aumento no tamanho da palavra de programa possibilita um aumento no número de instruções: os PICs de 12 bits (12C508, 12C509, 12CE518, 16C54, 16C55) possuem apenas 33 instruções, os de 14 bits (12C671, 12C672, 12CE673, 12C674, 14000, 16C55x) possuem 35 instruções e os de 16 bits (17C4x, 17C75x, 17C76x, 18C2xx, 18C4xx) possuem 77 instruções.



Nota:

As famílias PIC14 e PIC17 estão obsoletas.

Figura 1.1 – Evolução da Família PIC

Os PICs foram otimizados para trabalharem com execução de pequeno conjunto de instruções a grandes velocidades de processamento.

Desempenho da família PIC

Capacidade de pipeline (enquanto executa uma instrução, o processador busca a próxima instrução na memória, de forma a acelerar a execução do programa)

Execução de uma instrução por ciclo de máquina, com exceção das instruções de desvios que consomem dois ciclos de máquinas para serem executadas.

Um ciclo de máquina no PIC equivale a 4 ciclos de clock, ou seja, o sinal de clock é dividido por 4 antes de executar a instrução (falaremos mais sobre esse tópico adiante).

Cada Instrução ocupa uma posição de memória de programa (FLASH).

Tempo de execução das instruções fixa, com exceção das instruções de desvios que consomem dois ciclos de máquina.

Outra característica importante da arquitetura PIC reside na semelhança e compatibilidade entre os diversos microcontroladores membros de sua família. Isto facilita grandemente a migração de microcontrolador para outro, bastando mudar, em alguns casos, apenas alguns comandos no programa, pois partes dos registradores internos não se diferem muito entre si.

Tipos de Memória de Programa

Os PICs da série 12 e 16 armazenam o programa em sua memória interna. Membros da família 18 podem funcionar com memória de programa interna ou externa.

Tipos de memórias de programas disponíveis nos PICs:

ROM: Memória do tipo não volátil gravadas na fábrica pelo processo conhecido como máscara. Os chips com esse tipo de memória normalmente possuem custos menores, mas somente são viáveis na fabricação de grandes quantidades.

OTP: Memória fabricadas do tipo PROM. Saem da fábrica "virgens" e permitem uma única gravação. São inviáveis nas fases de implantação e desenvolvimento de equipamentos. Esses chips são identificados pelo sufixo "C".

EEROM: Podemos encontrar chips com memória do tipo EEPROM. Normalmente são mais caros que os dispositivos ROM e OTP e podem ser identificados através do seu sufixo "JW" para os dispositivos com encapsulamento DIP, ou "CL" para os dispositivos com encapsulamento do tipo PLCC.

FLASH: Os microcontroladores PIC que utilizam este tipo de memória são indicados para etapas de desenvolvimento e testes até mesmo para implantações finais. Permitem no máximo 1000 ciclos de gravações/apagamento, possuem um custo relativamente mediano com relação aos outros chips.

O PIC18F4520

O PIC18F4520 é um microcontrolador que possui memória do tipo FLASH, que nos representa uma grande facilidade em desenvolvimentos de projetos e protótipos pois não requer apagá-lo através de luz-ultravioleta como as versões antigas que utilizavam EEPROM, com tecnologia CMOS (baixíssimo consumo) fabricado pela empresa Microchip Technology.

Veja em seguida as principais características do PI18F4520

- Microcontrolador de 40 pinos;
- Memória de programa FLASH de 32K (16384 bytes words)
- Memória de dados RAM de 1536 bytes;
- Memória EEPROM de 256 byte;
- Processamento de até 10MIPS (milhões de instruções por segundo)
- 2 canais capture/compare/PWM - módulo CCP
- Master synchronous Serial Port (MSSP) module.
- Usart
- 13 canais A/D de 10 bits
- Permite até 100 000 ciclos de escrita e leitura na memória de programa Flash
- Permite 1.000.000 ciclos de leitura e escrita na E2PROM
- Retenção de dados na Flash de 40 anos
- Detector de baixa voltagem programável
- Watchdog timer com oscilador próprio e programável
- Três pinos de interrupção externa.
- 4 Temporizadores/contadores (TIMER0, TIMER1, TIMER2, TIMER3)

Tipos de encapsulamentos:

As primeiras versões do PIC eram baseadas em encapsulamentos do tipo DIP40, hoje os dispositivos de 40 pinos ainda são muito comuns, porém de acordo com a aplicação e os periféricos internos presentes no chip eles podem ser encontrados em diversos encapsulamentos como:

- DIP → Dual In-line Pin
- PLCC → Leadless Chip Carrier.
- TQFP → Thin Quad Flat Pack.

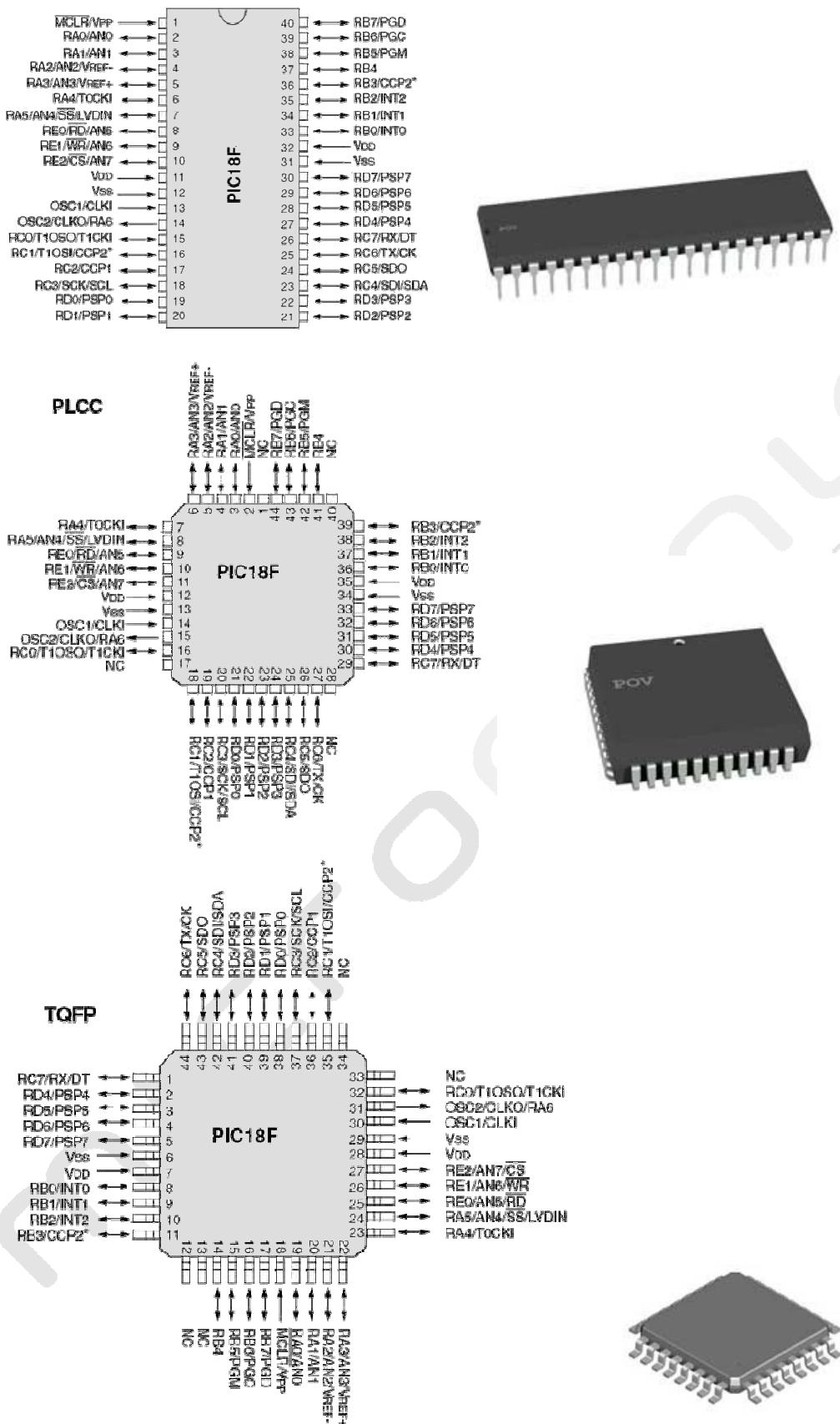


Figura 1.2 - Encapsulamentos

Nosso estudo será baseado em microcontroladores com o encapsulamento DIP (Dual In-line Pin), devido à facilidade de utilização e disponibilidade no mercado, porém não há grandes dificuldades para se migrar para outros encapsulamentos, basta analisar o datasheet do microcontrolador.

Estrutura interna do PIC18F4520

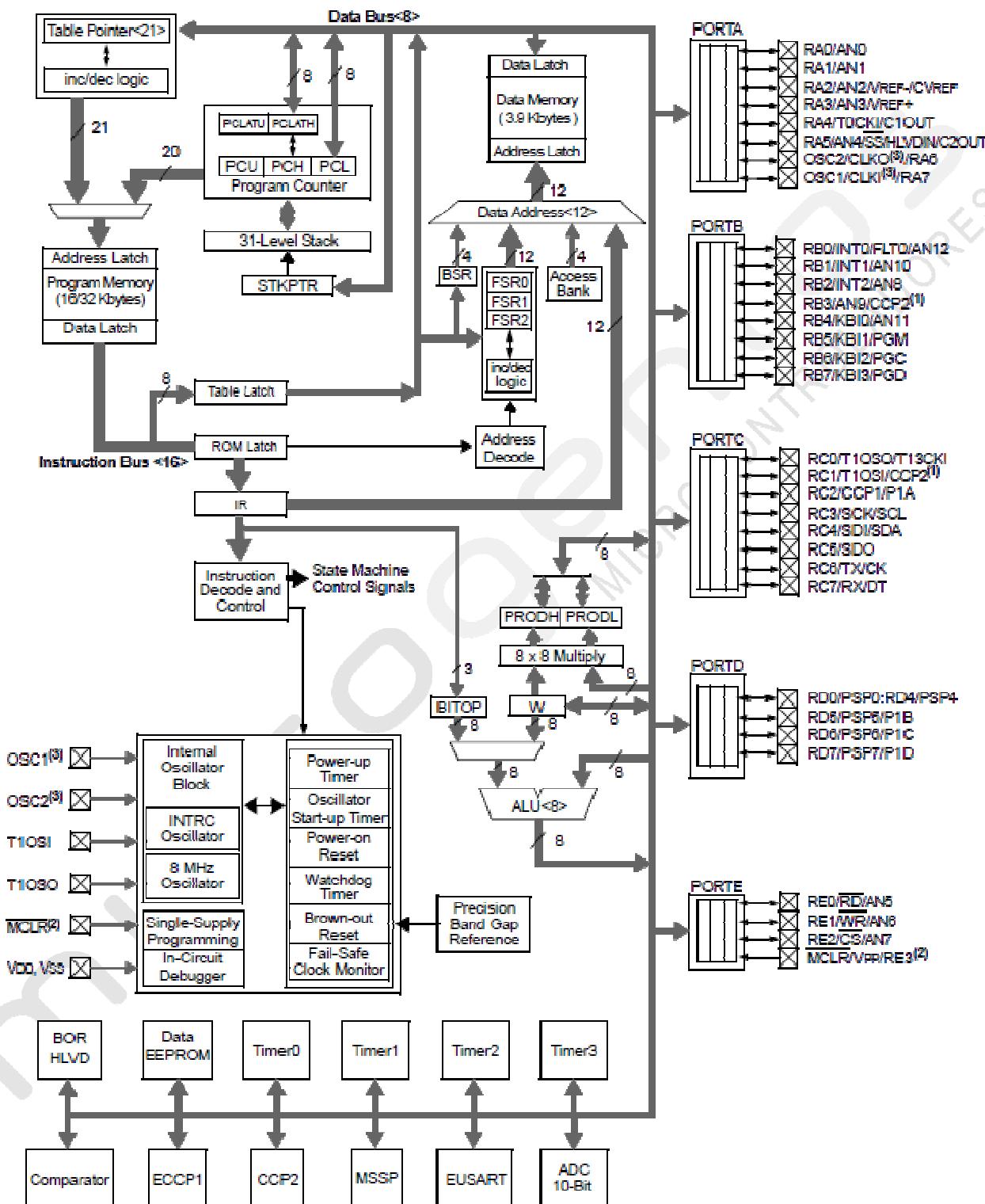


Figura 1.3 – Diagrama interno do PIC18F4520

Descrição das funções dos pinos do PI18F4520

O PI18F4520 possui no total de 34 pinos de I/O divididos entre as PORTA, PORTB, PORTC, PORTD e PORTE.

PORTA: encontramos 7 pinos físicos intitulados de RA0 a RA7 que podem ser utilizados como I/O de uso geral ou como conversor analógico/ digital A/D, além de possuir também a função de detecção de baixa tensão (LVD), referência analógica do A/D e contador externo.

PORTB: encontramos 8 pinos intitulados de RB0 a RB7 configuráveis como I/O de uso geral. Nesse port podemos trabalhar com três interrupções externas, módulo CCP e pinos de gravação e debugger.

PORTC: encontramos 8 pinos intitulados de RC0 a RC7 configuráveis como I/O de uso geral, saída do oscilador do timer, módulo CCP, Clock e data para os modos SPI, I2C e UART.

PORTD: encontramos 8 pinos intitulados de RD0 a RD7 que pode ser configurado como I/O de uso geral ou ser configurado como PSP para ter saída TTL (por exemplo: interfaceamento com microprocessadores).

PORTE: podemos utilizá-lo como PORT de I/O de uso geral ou utilizar os pinos de WR e CS para acesso ao modo paralelo Slave Port (acesso a memória externa por exemplo).

Abaixo segue com detalhes a função de cada pino do microcontrolador PIC18F4520.

Pino(DIP)	Função	Tipo	Descrição
2	RA0 AN0	I/O Entrada	Entrada e saída de uso geral Entrada do conversor AD0
3	RA1 AN1	I/O Entrada	Entrada e saída de uso geral Entrada do conversor AD1
4	RA2 AN2 Vref- CVref	I/O Entrada Entrada Saída	Entrada e saída de uso geral Entrada do conversor AD2 Entrada de referência baixa do A/D Saída de tensão da referência do comparador
5	RA3 AN3 Vref+	I/O Entrada Entrada	Entrada e saída de uso geral Entrada do conversor AD3 Entrada de referência alta do A/D
6	RA4 T0CKI C1OUT	I/O Entrada Saída	Entrada e saída de uso geral Entrada de clock timer0 Saída do comparador 1
7	RA5 AN4 /SS HLVDIN C2OUT	I/O Entrada Entrada Entrada Saída	Entrada e saída de uso geral Entrada do conversor AD4 Entrada de seleção SPI Detector de alta/baixa voltagem. Saída do comparador 2
13	OSC1 CLKI RA7	Entrada Entrada I/O	Entrada do cristal oscilador Entrada do clock externo Entrada e saída de uso geral
14	OSC2 CLKO RA6	Saída Saída I/O	Saída do cristal oscilador Entrada do clock exteno Entrada e saída de uso geral
33	RB0 INT0 FLT0 AN12	I/O Entrada Entrada Entrada	Entrada e saída de uso geral Interrupção externa 0 Entrada de erro PWM para CCP1 Entrada analógica 12
34	RB1 INT1 AN10	I/O Entrada Entrada	Entrada e saída de uso geral Interrupção externa 1 Entrada analógica 10
35	RB2 INT2 AN8	I/O Entrada Entrada	Entrada e saída de uso geral Interrupção externa 2 Entrada analógica 8
36	RB3 AN9 CCP2**	I/O Entrada I/O	Entrada e saída de uso geral Entrada analógica 9 Entrada de captura 2, saída de comparação 2 e saída PWM 2
37	RB4 KBI0 AN11	I/O Entrada I/O	Entrada e saída de uso geral Pino de interrupção por mudança de nível Entrada analógica 11
38	RB5 KBI1 PGM	I/O Entrada I/O	Entrada e saída de uso geral Interrupção por mudança de estado Pino de habilitação ICSP baixa voltagem (In-Circuit Debugger)
39	RB6 KBI2 PGC	I/O Entrada I/O	Entrada e saída de uso geral Interrupção por mudança de estado Pino de clock do ICSP
40	RB7	I/O	Entrada e saída de uso geral

Pino(DIP)	Função	Tipo	Descrição
	KBI3 PGD	Entrada I/O	Interrupção por mudança de estado Pino de dados do ICSP
15	RC0 T10S0 T1CKI	I/O Saída Entrada	Entrada e saída de uso geral Saída do clock do Timer 1 Clock externo Timer1 / Timer3
16	RC1 T10S1 CCP2**	I/O Entrada Saída	Entrada e saída de uso geral Entrada de clock do Timer1 Módulo CCP2(multiplexado com RB3)
17	RC2 CCP1 P1A	I/O I/O Saída	Entrada e saída de uso geral Módulo CCP1 Saída ECCP1
18	RC3 SCK SCL	I/O I/O I/O	Entrada e saída de uso geral Entrada e saída do clock serial para o modo SPI Entrada e saída do clock serial para o modo I2C
23	RC4 SDI SDA	I/O Entrada I/O	Entrada e saída de uso geral Entrada de dados SPI Entrada e saída de dados I2C
24	RC5 SD0	I/O Saída	Entrada e saída de uso geral Saída de dados SPI
25	RC6 TX CK	I/O Saída I/O	Entrada e saída de uso geral Canal de transmissão UART Clock de sincronismo UART
26	RC7 RX DT	I/O Entrada I/O	Entrada e saída de uso geral Canal de recepção UART Clock de sincronismo UART
19	RD0 PSP0	I/O I/O	Entrada e saída de uso geral Porta de comunicação paralela
20	RD1 PSP1	I/O I/O	Entrada e saída de uso geral Porta de comunicação paralela
21	RD2 PSP2	I/O I/O	Entrada e saída de uso geral Porta de comunicação paralela
22	RD3 PSP3	I/O I/O	Entrada e saída de uso geral Porta de comunicação paralela
27	RD4 PSP4	I/O I/O	Entrada e saída de uso geral Porta de comunicação paralela
28	RD5 PSP5 P1B	I/O I/O Saída	Entrada e saída de uso geral Porta de comunicação paralela Saída ECCP1
29	RD6 PSP6 P1C	I/O I/O Saída	Entrada e saída de uso geral Porta de comunicação paralela Saída ECCP1
30	RD7 PSP7 P1D	I/O I/O Saída	Entrada e saída de uso geral Porta de comunicação paralela Saída ECCP1
8	RE0 /RD AN5	I/O Entrada Entrada	Entrada e saída de uso geral Controle de leitura do port paralelo Entrada analógica AD5
9	RE1 /WR AN6	I/O Entrada Entrada	Entrada e saída de uso geral Controle de escrita do port paralelo Entrada analógica AD6
10	RE2 /CS AN7	I/O Entrada Entrada	Entrada e saída de uso geral Controle de seleção do port paralelo Entrada analógica AD7
1	/MCLR VPP RE3	Entrada Entrada Entrada	Entrada do RESET externo. Este pino é ativo em nível baixo Pino de habilitação de alta voltagem ICSP Entrada Digital
12, 31	GND	Alimentação	Negativo
11, 32	VCC	Alimentação	positivo

Tabela 1.1 – Descrição dos pinos do Microcontrolador PIC18F4520

Tipos de Memórias

No PIC18F4520 encontramos três tipos de memórias:

- Memória de Programa - FLASH
- Memória de dados - RAM
- Memória de dados EEPROM

Como nós sabemos, memória de programa (ROM, EPROM, FLASH) é do tipo não volátil, ou seja, podemos desenergizar nosso sistema microcontrolado e ele não perderá o programa gravado.

Nosso microcontrolador estudado possui internamente 16Kbyte de memória de programa (words) e possui um barramento de programa de 16 bits. Além disso, existem 21 bits para endereçamento da memória de programa, o que permite que até 2 Mbyte sejam endereçados pelo microcontrolador da família PIC18.

Memória de Programa

A memória de programa utilizada em nosso microcontrolador é do tipo FLASH (sufixo "F") que permite ser gravada/apagada no mínimo 1000 vezes. Este tipo de memória utilizada é ideal para ser utilizada em desenvolvimento de projetos e até mesmo em produtos finais.

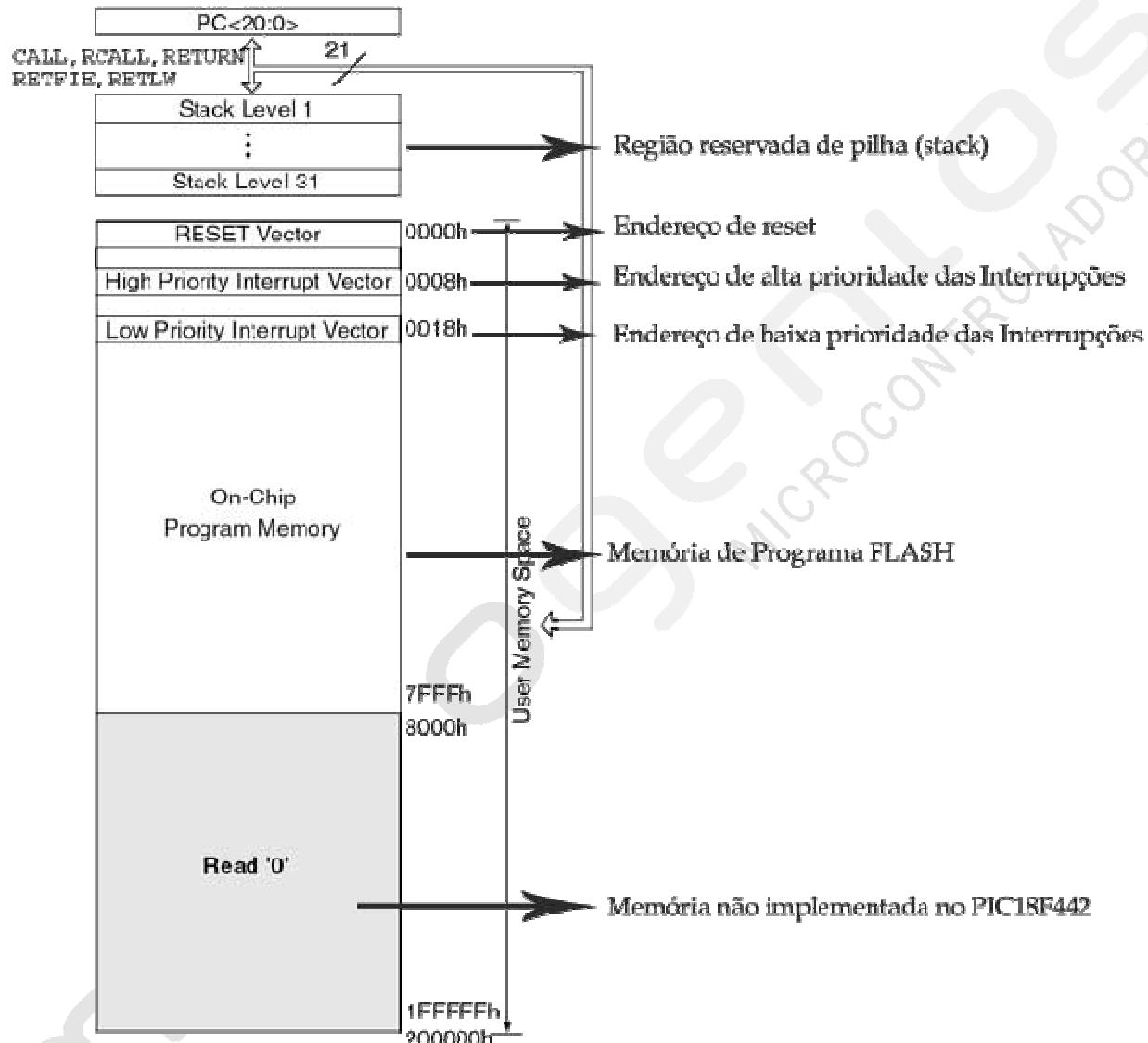


Figura 1.4 – Memória de Programa FLASH do PIC18F4520

Podemos perceber que a memória de programa do PIC18F4520 inicia-se no endereço 0000h e atinge o valor máximo de 7FFFH, ou seja , 32767K byte (32Kbyte).

Nota:

Não podemos confundir de forma alguma memória de programa e memória de dados. Sabemos que memória de programa é onde nosso programa estará gravado, enquanto memória de dados é onde os valores de nossas variáveis serão salvas temporariamente.

O endereço inicial 0000h é chamado de vetor de reset. A instrução de programa que estiver nesse endereço de memória será a primeira a ser executada pelo processador do PIC. Sempre que energizarmos ou resetarmos o microcontrolador, o contador de programa PC apontará sempre para este primeiro endereço da memória.

Em seguida temos os endereços 0008h e 0018h. Estes endereços são chamados de vetores de interrupção. (veremos mais adiante em nosso curso sobre esses vetores de interrupção).

Vamos exemplificar um processo de gravação na memória de programa do PIC.

Acompanhe o programa exemplo de programa abaixo:

```
/******//*****  
'Microgenios | MicroControladores  
'Site: www.Microgenios.com.br  
'Autor: Fernando Simplicio de Sousa  
'*****//*****  
'Programa exemplo: PISCA_PISCA.c  
'Cristal: 8MHz - modo HS  
'Microcontrolador PIC18F4520 Microchip  
'Tools: Kit PICgenios PIC18F Módulo Profissional Microgenios  
'Configuração: DIP1 - Chave 09 -ON  
'*****//*****  
'Objetivo:  
'Este simples programa inverte todo os pinos do PORTD em intervalos  
'de 1 segundo  
'*****//*****  
  
void main(){ //função principal do programa  
    TRISD = 0; //configura portd como saída  
    PORTD = 0; //zera todos os pinos do portd  
  
    while(1){  
        PORTD = 0B11111111; //seta todos os pinos do PORTD  
        Delay_ms(1000); //delay de 1 segundo  
        PORTD = 0; //zera todo o portd  
        Delay_ms(1000); //delay de 1 segundo  
    }  
}
```

Ao compilarmos o programa acima com o Mikroc PRO teremos como resultado o seguinte código de máquina (descrito somente algumas linhas do código gerado):

; ADDRESS OPCODE ASM	
\$0000	\$EF04 F009 GOTC _main
\$0008	\$6A93 CLRF TRISB, 0
\$000A	\$0EFF MOVLW 255
\$000C	\$6E81 MOVWF PORTB, 0
\$000E	\$0E06 MOVLW 6
\$0010	\$6E0C MOVWF STACK_12, 0
\$0012	\$0EFF MOVLW 255
\$0014	\$6E0B MOVWF STACK_11, 0
\$0016	\$0EFF MOVLW 255
\$0018	\$GE0A MOVWF STACK_10, 0
\$001A	\$2E0C DECFSZ STACK_12, F, 0
\$001C	\$D001 BRA \$+2
\$001E	\$D007 BRA \$+8
\$0020	\$2E0B DECFSZ STACK_11, F, 0
\$0022	\$D001 BRA \$+2
\$0024	\$D003 BRA \$+4
\$0026	\$2E0A DECFSZ STACK_10, F, 0
... (Sextante parte do código)	

Endereço físico na memoria de programa

Código em Assembly

Opcode

Figura 1.5 - Opcode gerado após compilação do programa

A partir da imagem acima, podemos verificar os valores que serão gravados na memória de programa após gravação do programa exemplo acima. Esses valores são úteis para o programador experiente, pois permite visualizar e acompanhar a execução por completo da compilação dos arquivos.

Nota:

Os códigos em assembly visualizado na figura acima foi gerado pelo compilador MikroC PRO após a compilação de um programa escrito em C.

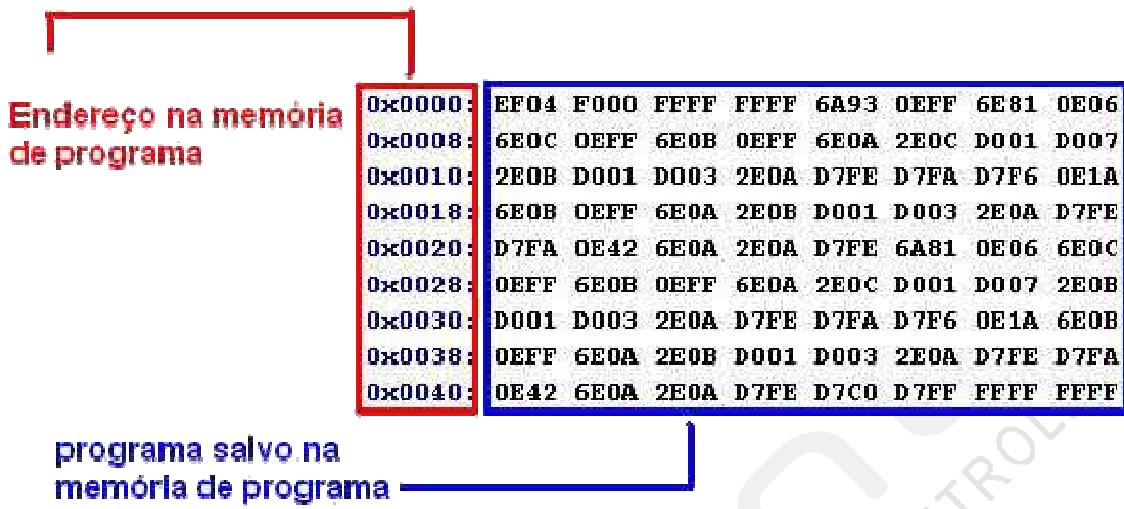


Figura 1.6 – Código Hexadecimal

Memória de dados - EEPROM

Como estudamos anteriormente, sabemos que as memórias do tipo EEPROM são muito parecidos com a memória RAM, sua diferença que as distingue é devido ao fato da memória EEPROM reter as informações salvas quando a desenergizarmos. Podemos salvar nesta memória dados não-voláteis durante o decorrer da execução de nosso programa.

O PIC18F4520 possui internamente 256 bytes de memória de dados EEPROM. O endereço inicial da memória de dados EEPROM é 00h seu último endereço é FFh.

Memória de dados - RAM

Chegamos a uma das partes mais importantes do estudo referente ao nosso microcontrolador PIC, neste tópico apresentaremos toda a estrutura de memória de dados RAM.

Aprenderemos a trabalhar com ela e também onde estão localizados os bits que ligam/desligam, configuram e controlam os periféricos internos do PIC, ou seja, os SFR's (Registradores de Funções Especiais).

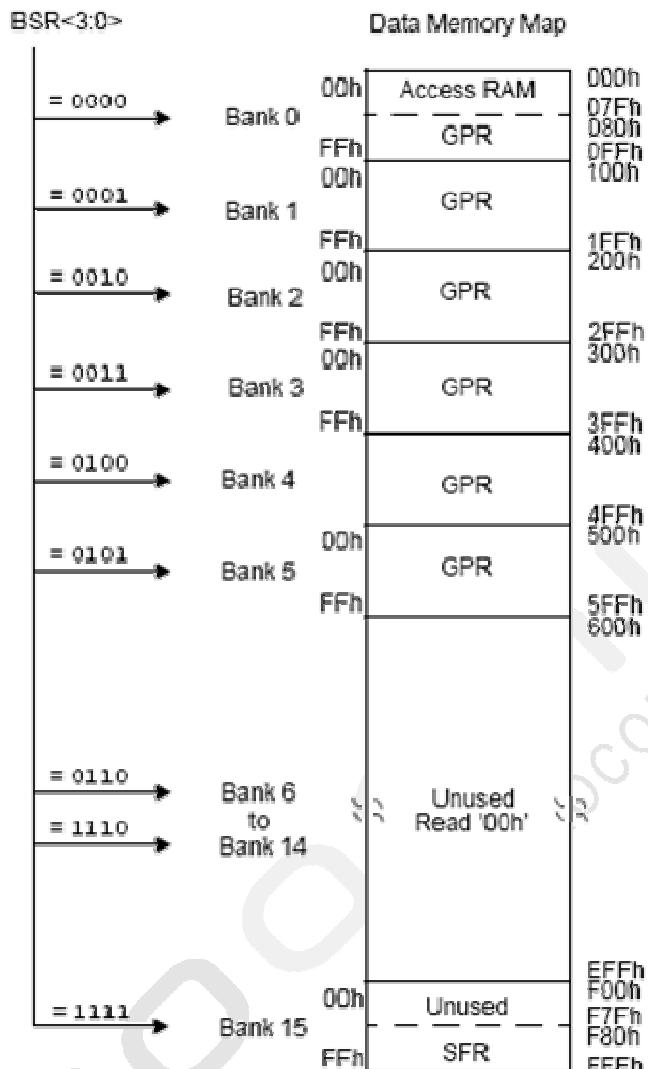
A memória de dados RAM é dividida em dois grupos: os chamados GPR's (Registradores de Propósito Geral) e os SFR's (Registradores de Funções Especiais).

Os GPR's tem a função de armazenar dados de uso geral que utilizamos e criamos durante a execução do nosso programa. Podemos guardar nesse região da memória RAM dados voláteis de variáveis, tabelas, constantes, entre outras.

Os SFR's são a parte principal do microcontrolador, é nesta área da memória RAM que estão armazenados todos os setups de funcionamento do PIC. A partir desses registradores podemos configurar o modo de trabalho dos timers/counters, USART, conversores analógicos digitais, etc.

O PIC18F4520 em estudo possui internamente 1536 bytes de memória de dados RAM.

O banco de memória RAM da família PIC18F é dividido em 16 bancos que contém 256 bytes de memória cada, pois a forma de implementação das instruções limita o endereçamento a um máximo de 7 bits ou 128 registradores. Por esse motivo, a Microchip usa a filosofia de paginação de memória, criando bancos de memória a cada 128 posições. Acompanhe a figura abaixo:



When $a = 1$,
the BSR is used to specify the
RAM location that the
instruction uses.

Figura 1.7 – Diagrama da memória RAM interna

Vamos examinar o mapa de registradores dos SFR's. (endereço 0F80h à FFFh):

[Registradores de Funções Especiais – SFR's](#)

Address	Name	Address	Name	Address	Name	Address	Name
FFFh	TOSU	FDFh	INDF2 ⁽¹⁾	FBFh	CCPR1H	F9Fh	IPR1
FFEh	TOSH	FDEh	POSTINC2 ⁽¹⁾	FBEh	CCPR1L	F9Eh	PIR1
FFDh	TOSL	FDDh	POSTDEC2 ⁽¹⁾	FBDh	CCP1CON	F9Dh	PIE1
FFCh	STKPTR	FDCh	PREINC2 ⁽¹⁾	FBCh	CCPR2H	F9Ch	__(2)
FFBh	PCLATU	FDBh	PLUSW2 ⁽¹⁾	FBBh	CCPR2L	F9Bh	OSCTUNE
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	__(2)
FF9h	PCL	FD9h	FSR2L	FB9h	__(2)	F99h	__(2)
FF8h	TBLPTRU	FD8h	STATUS	FB8h	BAUDCON	F98h	__(2)
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	PWM1CON ⁽³⁾	F97h	__(2)
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	ECCP1AS ⁽³⁾	F96h	TRISE ⁽³⁾
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD ⁽³⁾
FF4h	PRODH	FD4h	__(2)	FB4h	CMCON	F94h	TRISC
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB
FF2h	INTCON	FD2h	HLVDCON	FB2h	TMR3L	F92h	TRISA
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	__(2)
FF0h	INTCON3	FD0h	RCON	FB0h	SPBRGH	F90h	__(2)
FEFh	INDF0 ⁽¹⁾	FCFh	TMR1H	FAFh	SPBRG	F8Fh	__(2)
FEEh	POSTINC0 ⁽¹⁾	FCEh	TMR1L	FAEh	RCREG	F8Eh	__(2)
FEDh	POSTDEC0 ⁽¹⁾	FCDh	T1CON	FADh	TXREG	F8Dh	LATE ⁽³⁾
FECh	PREINC0 ⁽¹⁾	FCCh	TMR2	FACh	TXSTA	F8Ch	LATD ⁽³⁾
FEBh	PLUSW0 ⁽¹⁾	FCBh	PR2	FABh	RCSTA	F8Bh	LATC
FEAh	FSR0H	FCAh	T2CON	FAAh	__(2)	F8Ah	LATB
FE9h	FSR0L	FC9h	SSPBUF	FA9h	EEADR	F89h	LATA
FE8h	WREG	FC8h	SSPADD	FA8h	EEDATA	F88h	__(2)
FE7h	INDF1 ⁽¹⁾	FC7h	SSPSTAT	FA7h	EECON2 ⁽¹⁾	F87h	__(2)
FE6h	POSTINC1 ⁽¹⁾	FC6h	SSPCON1	FA6h	EECON1	F86h	__(2)
FE5h	POSTDEC1 ⁽¹⁾	FC5h	SSPCON2	FA5h	__(2)	F85h	__(2)
FE4h	PREINC1 ⁽¹⁾	FC4h	ADRESH	FA4h	__(2)	F84h	PORTE ⁽³⁾
FE3h	PLUSW1 ⁽¹⁾	FC3h	ADRESL	FA3h	__(2)	F83h	PORTD ⁽³⁾
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA

Figura 1.8 – Registradores de Funções Especiais – SFR's

Através da configuração desses registradores especiais, podemos monitorar o status do microcontrolador em determinados momentos e situações. Estudaremos a função desses registradores quando estivermos trabalhando com projetos, pois dessa maneira ficará mais claro seu funcionamento.

Introdução as portas de I/O

Os microcontroladores PIC possuem pinos físicos destinados à comunicação de dados com os circuitos externos. Através desses pinos podemos enviar níveis lógicos (0 ou 1) para, por exemplo, acender ou apagar um LED, acionar displays LCD, ler botões e sensores, etc.

O microcontrolador PIC18F45200 da Microchip possui 34 pinos de escrita e leitura (I/O), os quais estão divididos em quatro portas chamadas: PORTA, PORTB, PORTC, PORTD e PORTE.

40-Pin PDIP

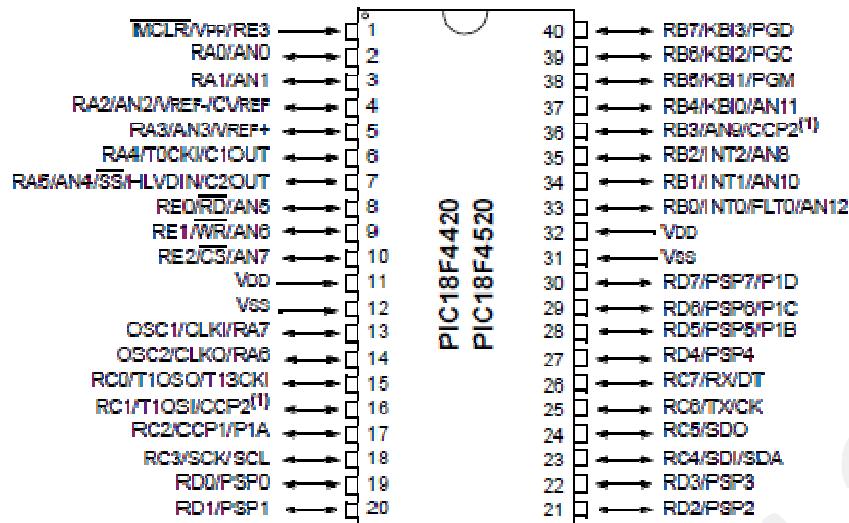


Figura 1.9 – PIC18F4520

Os registradores PORTB, PORTC e PORTD possuem 8 pinos de I/O cada, enquanto o PORTE possui 4 pinos de I/O e PORTA 6 pinos de I/O.

Cada porta de I/O possui três registradores que controlam suas funções: um registrador PORT, LAT e um registrador TRIS.

Os registradores PORT (PORTA, PORTB, PORTC, PORTD, PORTE) pertencem aos SFR's e armazena os dados das portas paralelas (I/O's) do microcontrolador. Qualquer escrita realizada em um desses registros automaticamente altera todo o conteúdo presente na saída do chip.

O registrador TRIS é utilizado para configurar cada pino da respectiva porta como entrada ou saída. Assim, cada bit desse registrador corresponde a um pino da porta. Se o bit estiver em 1 (um), o pino correspondente está configurado como entrada de dados, e se estiver em 0 (zero) configuramos este pino como saída de dados.

As portas de comunicação paralela do PIC são utilizadas para efetuar diversas tarefas:

O PORTA

A primeira porta a ser estudada é o PORTA. Além das funções de entrada e saída de dados encontramos também diversas outras funções multiplexadas aos seus pinos:

Vamos conhecer as funções de cada pino do PORTA:

Tabela 1.2 – Pinos PORTA

Pino(DIP)	Função	Tipo	Descrição
2	RA0 AN0	I/O Entrada	Entrada e saída de uso geral Entrada do conversor AD0
3	RA1 AN1	I/O Entrada	Entrada e saída de uso geral Entrada do conversor AD1
4	RA2 AN2 Vref- CVref	I/O Entrada Entrada Saída	Entrada e saída de uso geral Entrada do conversor AD2 Entrada de referência baixa do A/D Saída de tensão da referência do comparador
5	RA3 AN3 Vref+	I/O Entrada Entrada	Entrada e saída de uso geral Entrada do conversor AD3 Entrada de referência alta do A/D
6	RA4 T0CKI C1OUT	I/O Entrada Saída	Entrada e saída de uso geral Entrada de clock timer0 Saída do comparador 1
7	RA5 AN4 /SS HVLDIN C2OUT	I/O Entrada Entrada Entrada Saída	Entrada e saída de uso geral Entrada do conversor AD4 Entrada de seleção SPI Detector de alta/baixa voltagem. Saída do comparador 2
13	OSC1 CLKI RA7	Entrada Entrada I/O	Entrada do cristal oscilador Entrada do clock externo Entrada e saída de uso geral
14	OSC2 CLKO RA6	Saída Saída I/O	Saída do cristal oscilador Entrada do clock externo Entrada e saída de uso geral

Registrador PORTA

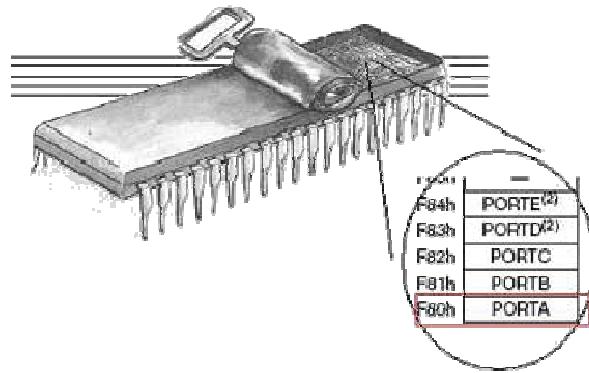


Figura 1.10 - PORTA

O registrador PORTA está localizado na memória RAM do PIC e possui o endereço (0XF80). Este registrador irá acessar os pinos da PORTA da seguinte maneira:

PORTA

RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

Toda a porta de I/O possui internamente um latch de saída para cada pino. Dessa forma, uma escrita no registrador PORTA irá na realidade escrever em cada latch do PORTA. O latch da PORTA é chamado de LATA.

Para ler um dado um dado no PORTA, basta ler o conteúdo do registrador PORTA.

Nota:

O registrador PORTA é o "espelho" dos pinos do PORTA do PIC, para termos ou escrevermos nesse registrador estamos na verdade atuando sobre os pinos do port respectivamente. Não podemos esquecer que o registrador TRISD também afeta a leitura e escrita no port.

Registrador TRISA

O registrador TRISA é utilizado para o controle da direção de atuação de cada pino de I/O do PORTA. Através desse registradores definimos e configuramos o modo de trabalho do pino do microcontrolador como entrada ou saída de dados.

Este registrador define os pinos do PORTA da seguinte maneira:

TRISA

RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

Se o bit do registrador TRISA estiver em nível lógico 1, o pino correspondente a este bit do PORTA estará configurado como entrada.

Se o bit do registrador TRISA estiver em nível lógico 0, o pino correspondente a este bit do PORTA estará configurado como saída.

Leitura do LATCH de um pino do PIC

Para ler o LATCH de saída do microcontrolador ao invés dos pinos utilizamos o registrador LATA, referente ao PORTA, ou LATB, referente ao PORTB. O latch de saída encontra-se antes do buffer de saída do pino, e por esse motivo podemos ler seu estado.

O PORTB

O PORTB, tal qual o PORTA, implementa diversas outras funções multiplexadas aos seus pinos, acompanhe:

Pino(DIP)	Função	Tipo	Descrição
33	RB0 INT0 FLT0 AN12	I/O Entrada Entrada Entrada	Entrada e saída de uso geral Interrupção externa 0 Entrada de erro PWM para CCP1 Entrada analógica 12
34	RB1 INT1 AN10	I/O Entrada Entrada	Entrada e saída de uso geral Interrupção externa 1 Entrada analógica 10
35	RB2 INT2 AN8	I/O Entrada Entrada	Entrada e saída de uso geral Interrupção externa 2 Entrada analógica 8
36	RB3 AN9 CCP2**	I/O Entrada I/O	Entrada e saída de uso geral Entrada analógica 9 Entrada de captura 2, saída de comparação 2 e saída PWM 2
37	RB4 KBI0 AN11	I/O Entrada I/O	Entrada e saída de uso geral Pino de interrupção por mudança de nível Entrada analógica 11
38	RB5 KBI1 PGM	I/O Entrada I/O	Entrada e saída de uso geral Interrupção por mudança de estado Pino de habilitação ICSP baixa voltagem (In-Circuit Debugger)
39	RB6 KBI2 PGC	I/O Entrada I/O	Entrada e saída de uso geral Interrupção por mudança de estado Pino de clock do ICSP
40	RB7 KBI3 PGD	I/O Entrada I/O	Entrada e saída de uso geral Interrupção por mudança de estado Pino de dados do ICSP

Assim como no PORTA, o PORTB possui dois registradores que definem o modo de trabalho dos seus pinos de I/O: são eles: PORTB e TRISB.

Tanto o registrador PORTB quanto o registrador TRISB estão localizados na memória RAM do PIC e pertencem aos SFR'S (Registradores de Funções Especiais).

O registrador TRISB tem as mesmas funções que o registrador TRISA estudado anteriormente, que é programar os pinos do PIC para serem entradas ou saídas para uso geral.

O registrador PORTB funciona da mesma maneira descrita no PORTA.

O PORTC

O PORTC, tal qual o PORTA, implementa diversas outras funções multiplexadas aos seus pinos, acompanhe:

Pino(DIP)	Função	Tipo	Descrição
15	RC0 T10S0 T1CKI	I/O Saída Entrada	Entrada e saída de uso geral Saída do clock do Timer 1 Clock externo Timer1 / Timer3
16	RC1 T10S1 CCP2**	I/O Entrada Saída	Entrada e saída de uso geral Entrada de clock do Timer1 Módulo CCP2(multiplexado com RB3)
17	RC2 CCP1 P1A	I/O I/O Saída	Entrada e saída de uso geral Módulo CCP1 Saída ECCP1
18	RC3 SCK SCL	I/O I/O I/O	Entrada e saída de uso geral Entrada e saída do clock serial para o modo SPI Entrada e saída do clock serial para o modo I2C
23	RC4 SDI SDA	I/O Entrada I/O	Entrada e saída de uso geral Entrada de dados SPI Entrada e saída de dados I2C
24	RC5 SD0	I/O Saída	Entrada e saída de uso geral Saída de dados SPI
25	RC6 TX CK	I/O Saída I/O	Entrada e saída de uso geral Canal de transmissão UART Clock de sincronismo UART
26	RC7 RX DT	I/O Entrada I/O	Entrada e saída de uso geral Canal de recepção UART Clock de sincronismo UART

Assim como no PORTA, o PORTC possui dois registradores que definem o modo de trabalho dos seus pinos de I/O: são eles: PORTC e TRISC.

Tanto o registrador PORTC quanto o registrador TRISC estão localizados na memória RAM do PIC e pertencem aos SFR'S (Registradores de Funções Especiais).

O registrador TRISC tem as mesmas funções que o registrador TRISA estudado anteriormente, que é programar os pinos do PIC para serem entradas ou saídas.

O registrador PORTC funciona da mesma maneira descrita no PORTA.

As funções especiais de cada pino serão vistas quando estudarmos suas aplicações reais através dos projetos que vamos desenvolver durante nosso curso.

O PORTD

O PORTD, tal qual o PORT A, PORTB e PORTC implementa diversas outras funções multiplexadas aos seus pinos, acompanhe:

Pino(DIP)	Função	Tipo	Descrição
19	RD0 PSP0	I/O I/O	Entrada e saída de uso geral Porta de comunicação paralela
20	RD1 PSP1	I/O I/O	Entrada e saída de uso geral Porta de comunicação paralela
21	RD2 PSP2	I/O I/O	Entrada e saída de uso geral Porta de comunicação paralela
22	RD3 PSP3	I/O I/O	Entrada e saída de uso geral Porta de comunicação paralela
27	RD4 PSP4	I/O I/O	Entrada e saída de uso geral Porta de comunicação paralela
28	RD5 PSP5 P1B	I/O I/O Saída	Entrada e saída de uso geral Porta de comunicação paralela Saída ECCP1
29	RD6 PSP6 P1C	I/O I/O Saída	Entrada e saída de uso geral Porta de comunicação paralela Saída ECCP1
30	RD7 PSP7 P1D	I/O I/O Saída	Entrada e saída de uso geral Porta de comunicação paralela Saída ECCP1

Assim como nas outras portas, o PORT D possui dois registradores que definem o modo de trabalho dos seus pinos de I/O: são eles: PORTD e TRISD.

Tanto o registrador PORT D quanto o registrador TRISD estão localizados na memória RAM do PIC e pertencem aos SFR'S (Registradores de Funções Especiais).

O registrador TRISD tem a função de programar os pinos do PIC para serem entradas ou saídas.

O registrador PORTD é o "espelho" dos pinos do PORTD do PIC, para termos ou escrevermos nesse registrador estamos na verdade atuando sobre os pinos do port respectivamente. Não podemos esquecer que o registrador TRISD também afeta a leitura e escrita no port.

As funções especiais de cada pino serão vistas quando estudarmos suas aplicações reais através dos projetos que vamos desenvolver durante nosso curso.

O PORTE

O PORTE, tal qual as outras portas, implementa diversas outras funções multiplexadas aos seus pinos, acompanhe:

Pino(DIP)	Função	Tipo	Descrição
8	RE0 /RD AN5	I/O Entrada Entrada	Entrada e saída de uso geral Controle de leitura do port paralelo Entrada analógica AD5
9	RE1 /WR AN6	I/O Entrada Entrada	Entrada e saída de uso geral Controle de escrita do port paralelo Entrada analógica AD6
10	RE2 /CS AN7	I/O Entrada Entrada	Entrada e saída de uso geral Controle de seleção do port paralelo Entrada analógica AD7
1	/MCLR VPP RE3	Entrada Entrada Entrada	Entrada do RESET externo. Este pino é ativo em nível baixo Pino de habilitação de alta voltagem ICSP Entrada Digital

Assim como nas outras portas, o PORTE possui dois registradores que definem o modo de trabalho dos seus pinos de I/O: são eles: PORTE e TRISE.

Tanto o registrador PORTE quanto o registrador TRISE estão localizados na memória RAM do PIC e pertencem aos SFR'S (Registradores de Funções Especiais).

O registrador TRISE tem a função de programar os pinos do PIC para serem entradas ou saídas.

O registrador PORTE é o "espelho" dos pinos do PORTE do PIC, para termos ou escrevermos nesse registrador estamos na verdade atuando sobre os pinos do port respectivamente. Não podemos esquecer que o registrador TRISE também afeta a leitura e escrita no port.

As funções especiais de cada pino serão vistas quando estudarmos suas aplicações reais através dos projetos que vamos desenvolver durante nosso curso.

Reset

Sempre que energizamos nosso circuito microcontrolado é interessante que o PIC seja resetado, para isso é necessário a inserção de um circuito eletrônico bem simples, mas capaz de realizar esse procedimento que costumamos chamar de **POWER ON RESET**.

Além disso é interessante que exista um botão para que o usuário possa reiniciar o sistema sempre que for necessário (reset manual), a seguir indicamos um exemplo de circuito para o reset.

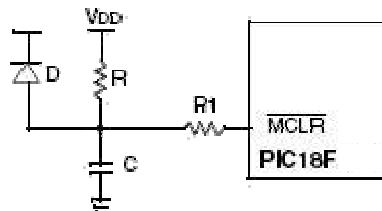


Figura 1.11 – Circuito de Reset

OS PICs possuem diversos tipos de reset:

- POR - Power On Reset - é o reset que ocorre quando o chip é ligado.
- Reset do MCLR durante a operação normal
- Reset do MCLR durante o modo SLEEP
- Reset do watchdog durante operação normal
- Reset do detector de Brown-out (BOR)
- Instrução de Reset
- Reset do Stack Pointer

Ciclos de máquina

A contagem de tempo não é medida diretamente pela freqüência de oscilação do cristal e sim através do que chamamos de **CICLO DE MÁQUINA**.

Internamente no microcontroladores PIC a freqüência do cristal é dividida por 4, o que nos resulta que a freqüência real de trabalho é:

$$\text{freqüência de trabalho real} = \text{freqüência oscilador} / 4$$

Concluímos então que: a cada 1 ciclo de máquina corresponde a 4 pulsos do oscilador.

Para exemplificar vamos supor que temos conectado ao microcontrolador um cristal de quartzo de 8 MHz.

Qual é a freqüência real de execução das instruções do nosso microcontrolador?

É muito simples, acompanhe:

Sabemos que a freqüência do cristal utilizado é de 8 MHz, e que cada instrução leva exatamente 1 ciclo de máquina para ser executada; Basta dividir o valor do cristal por 4 para sabermos o valor real de trabalho do PIC.

$$\text{freqüência de trabalho real} = 8 \text{ MHz} / 4 \Rightarrow 2 \text{ MHz}$$

Concluímos então que nosso microcontrolador PIC com cristal de 8 MHz esta trabalhando efetivamente a 2 MHz, ou seja , cada instrução de programa leva 0.5 us para ser executada.

Pois:

$$\text{Frequencia} = 1 / \text{periodo}$$

$$\text{Logo: Periodo} = 1 / 2\text{MHz} \Rightarrow 0.5\text{us}$$

Nota:

Temos que lembrar que nosso microcontrolador em estudo pode operar em 10MHz (ciclo de máquina)

os bits de configuração (fusíveis)

Os microcontroladores PIC possuem internamente regiões de memórias não voláteis (Flash) que são responsáveis por determinar o modo de trabalho do chip. São chamados de bits de configuração ou fusíveis. É de grande importância que venhamos entender o funcionamento desses bits, caso contrário teremos sérios problemas no funcionamento do nosso projeto.

os bits de configuração do PIC estão armazenados em uma área reservada do PIC. Podemos configurar diversas funções através desses bits:

- OSC Selection (seleção do oscilador)
- RESET (tipos de acionamento de reset)
- POWER -on Reset (POR)
- Power -up Timer (PWRT)
- Oscillator Start-up Timer (OSC)
- Brown-out Reset (BOR) (verificador de tensão de alimentação)
- Interrupts
- Watchdog Timer (WDT)
- SLEEP
- Code Protection
- ID Locations
- In-Circuit Serial Programming

Mapa dos bits de configuração do PIC18F4520

Tabela 1.3 – Bits de configuração do PIC18F4520

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default/ Unprogrammed Value
300001h CONFIG1H	IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0	00-- 0111
300002h CONFIG2L	—	—	—	BORV1	BORV0	BOREN1	BOREN0	PWRTE	---1 1111
300003h CONFIG2H	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN	---1 1111
300005h CONFIG3H	MCLRE	—	—	—	—	LPT1OSC	PBADEN	CCP2MX	1--- -011
300006h CONFIG4L	DEBUG	XINST	—	—	—	LVP	—	STVREN	10-- -1-1
300008h CONFIG5L	—	—	—	—	CP3 ⁽¹⁾	CP2 ⁽¹⁾	CP1	CP0	---- 1111
300009h CONFIG5H	CPD	CPB	—	—	—	—	—	—	11-- ----
30000Ah CONFIG6L	—	—	—	—	WRT3 ⁽¹⁾	WRT2 ⁽¹⁾	WRT1	WRT0	---- 1111
30000Bh CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—	111- ----
30000Ch CONFIG7L	—	—	—	—	EBTR3 ⁽¹⁾	EBTR2 ⁽¹⁾	EBTR1	EBTR0	---- 1111
30000Dh CONFIG7H	—	EBTRB	—	—	—	—	—	—	-1-- ----
3FFFFEh DEVID1	DEV2	DEV1	DEVO	REV4	REV3	REV2	REV1	REV0	xxxx xxxx ⁽²⁾
3FFFFFFh DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	xxxx xxxx ⁽²⁾

Tabela 1.4 – Bits de configuração do PIC18F4520(Detalhado)

Apresentamos a seguir uma tabela com a descrição de todos os bits de configuração do PIC18F4520:

Oscilador	RC, EC, LP, XT, HS, HS-PLL, RC-SOC2 com RA6, EC-OSC2 com RA6
Fail-Safe Clock Monitor Enable	Habilitado ou Desabilitado
Internal External Switch Over Mode	Habilitado ou Desabilitado
Power Up Timer	Habilitado ou Desabilitado
Brown Out Detect	Habilitado ou Desabilitado
Brown Out Voltage	2.5V, 2.7V, 4.5V ou 4.7V
Watchdog Timer	Habilitado ou Desabilitado
Watchdog Postscaler	1:1 , 1:2, 1:4, 1:8, 1:16, 1:32, 1:64, ou 1:128
CCP2MX	Pino RC1 ou RB3
PortB A/D Enable	PortB configurado como I/O Digital no reset ou configurado como A/D no reset
Low Power Timer1 Osc Enable	Habilitado ou Desabilitado
Máster Clear Enable	MCLR Habilitado ou RE3 Habilitado
Stack Overflow Reset	Habilitado ou Desabilitado
Low Voltage Program	Habilitado ou Desabilitado
Extended Instruction Enable Bit	Habilitado ou Desabilitado
Background Debug	Habilitado ou Desabilitado
Code Protect 0x0800 A 0xffff	Habilitado ou Desabilitado
Code Protect 0x2000 A 0x3FFF	Habilitado ou Desabilitado
Code Protect 0x4000 A 0x5FFF	Habilitado ou Desabilitado
Code Protect 0x6000 A 0x7FFF	Habilitado ou Desabilitado
Code Protect Boot	Habilitado ou Desabilitado
Data EEPROM Code Protect	Habilitado ou Desabilitado
Table Write Protect 0x2000 A 0x1fff	Habilitado ou Desabilitado
Table write Protect 0x2000 A 0x3fff	Habilitado ou Desabilitado
Table write Protect 0x4000 A 0x5fff	Habilitado ou Desabilitado
Table write Protect 0x6000 A 0x7fff	Habilitado ou Desabilitado
Config Write Protect	Habilitado ou Desabilitado
Table Write Protect Boot	Habilitado ou Desabilitado
Data Ee Write Protect	Habilitado ou Desabilitado
Table Read Protect 0x0800 A 0x1fff	Habilitado ou Desabilitado
Table Read Protect 0x2000 A 0x3fff	Habilitado ou Desabilitado
Table Read Protect 0x4000 A 0x5fff	Habilitado ou Desabilitado
Table Read Protect 0x6000 A 0x7fff	Habilitado ou Desabilitado
Table Read Protect Boot	Habilitado ou Desabilitado

Acompanhe:

No painel de configuração do MikroC PRO, podemos visualizar e configurar os bits de configuração referente ao nosso microcontrolador PIC18F4520.

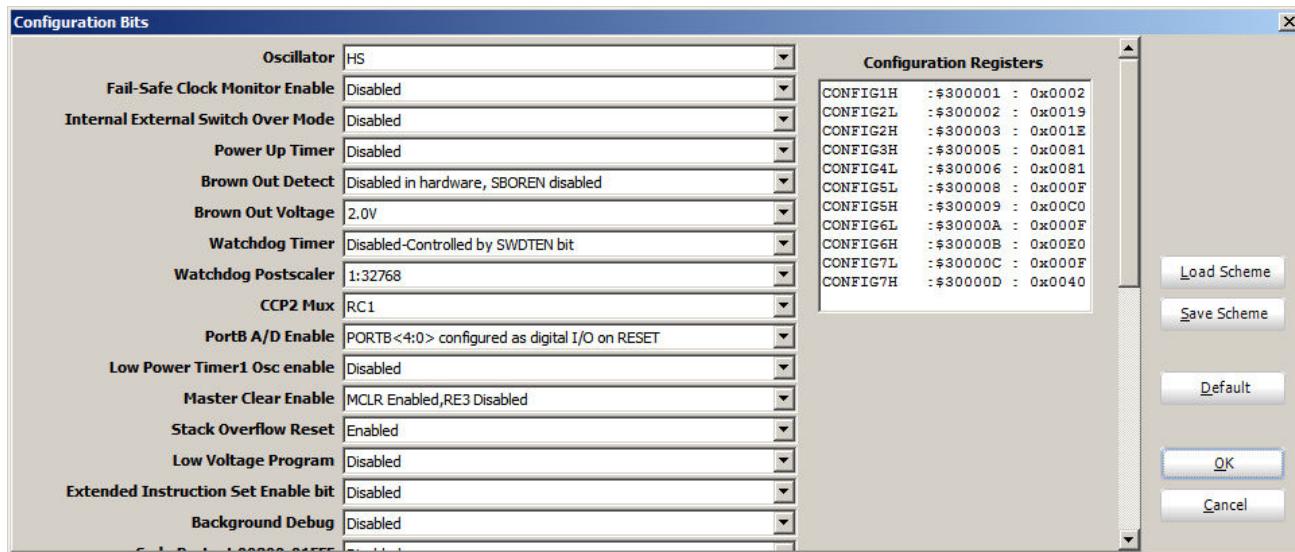


Figura 1.12 - Configuração dos fusíveis no MikroC PRO

Podemos configurar os bits de configuração de duas maneiras:

Através do MikroC PRO, no momento da criação ou desenvolvimento do projeto ou através dos programas de gravação dos gravadores de PIC (WinPIC800, IC-Prog etc.)

No primeiro campo de configuração dos fusíveis do MikroC PRO encontramos o campo de seleção de clock. Através desses campos podemos informar vários tipos de osciladores e circuitos de oscilação:

Configuração de Clock

Todo microcontrolador requer um circuito de oscilação pois é quem dá o sinal de "partida" e "sincronismo" para qualquer atividade interna da chip. A frequência de operação de oscilação é um dos agentes que determinam a velocidade de execução das instruções do microcontrolador.

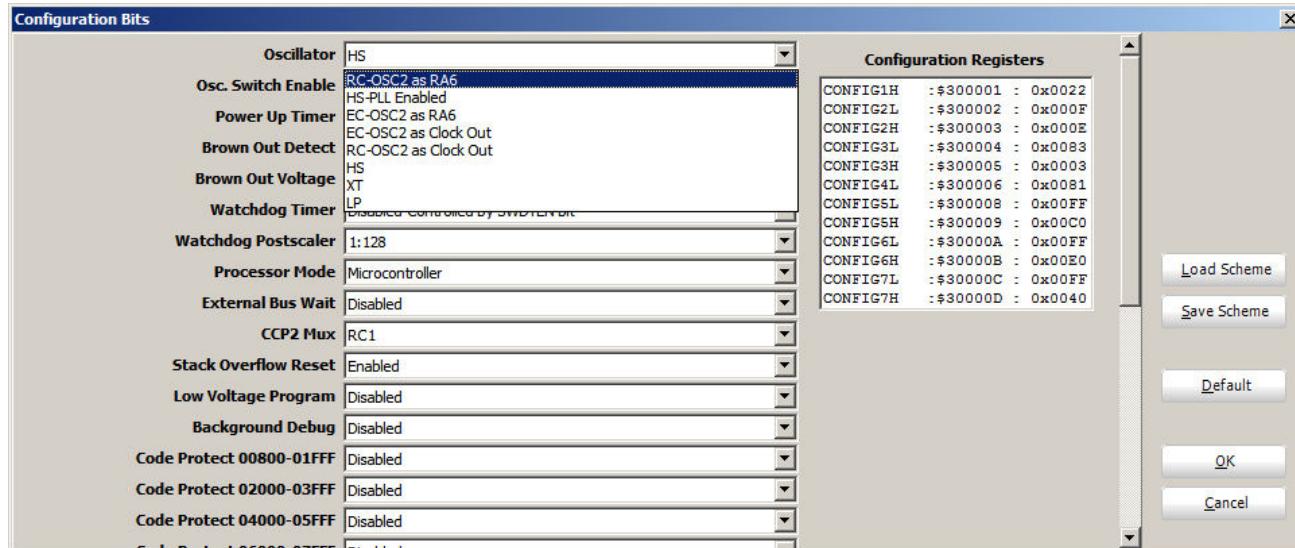
O PIC18F4520 pode funcionar com diversas fontes de osciladores:

LP	Low-power cristal
XT	Cristal ou ressonador
HS	High-Speed cristal (cristal de alta velocidade) ou ressonador
HS + PLL	High-Speed cristal ou ressonador com PLL habilitada
RC	Resistor / Capacitor externo
RCIO	Resistor / Capacitor externo com pino de I/O
EC	Clock externo
ECIO	Clock externo com pino de I/O

Vamos conhecer cada um dos tipos de osciladores:

Oscilador RC – com pino RA6 como saída de clock

Este é o tipo de oscilador mais simples que existe e também o mais barato, mas, por outro lado, é o menos preciso, variando muito a tensão de trabalho, temperatura e tolerâncias. O circuito RC deve ser ligado conforme a figura seguinte:



Oscilador RC – com pino RA6 como I/O de uso geral

Este é o tipo de oscilador mais simples que existe e também o mais barato, mas, por outro lado, é o menos preciso, variando muito a tensão de trabalho, temperatura e tolerâncias. O circuito RC deve ser ligado conforme a figura seguinte:

Nesta configuração o pino RA6 pode ser utilizado como I/O de uso geral

Modo HSPLL

HSPLL é na verdade um modo em que podemos multiplicar o valor da freqüência de entrada do cristal oscilador por 4. É ideal para ambientes em que o dispositivo não pode gerar EMI (interferência Eletromagnética). Este modo deve ser habilitado nos "fusíveis de configurações" no momento em que formos gravar o chip.

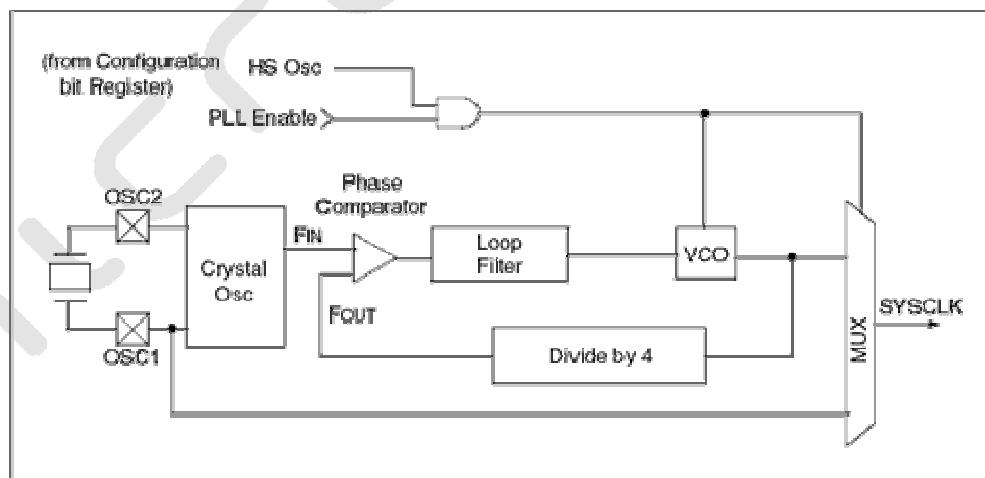


Figura 1.13 - Modo HSPLL

Por exemplo:

Caso venhamos conectar ao PIC um cristal de 10 MHz entre os pinos OSC1 e OSC2 e ativarmos o modo HSPLL no momento em que gravarmos o chip, executando seu ciclo de máquina a 10MHz (10 MIPS - milhões de instruções por segundo).

Exemplo:

Fórmula do ciclo de máquina com HSPLL ativado – cristal de 8MHz

$$\text{Ciclo de máquina} = (\text{Fosc} / 4) * 4$$

Ciclo de máquina = 8 MHz ou 0.125us

Circuito de Oscilação EC - com pino RA6 como I/O de uso geral

Este modo de configuração é utilizado quando temos uma fonte de clock em nosso circuito eletrônico.

Obs: A fonte de clock deverá estar livre de ruidos e deformações.

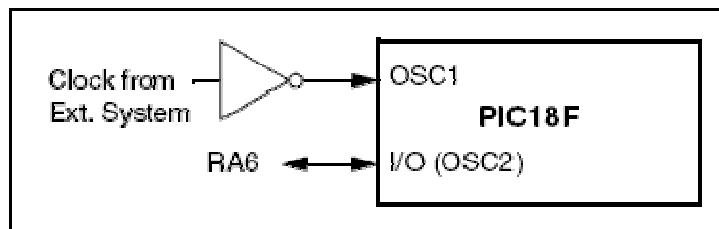


Figura 1.14 – Modo EC – com pino RA6 como I/O de uso geral

Circuito de oscilação EC – com pino RA6 como saída de clock

Este modo de configuração é utilizado quando temos uma fonte de clock em nosso circuito eletrônico.

Percebam que a saída de clock é $F_{osc}/4$, ou seja, o próprio ciclo de máquina do microcontrolador, a partir desse clock de saída, poderemos sincronizar outros microcontroladores no circuito.

Obs: A fonte de clock deverá estar livre de ruidos e deformações.

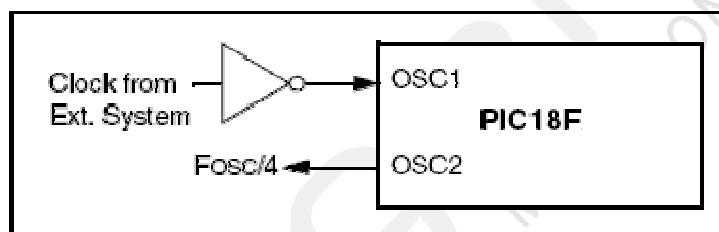


Figura 1.15 – Modo EC – com pino RA6 como saída de clock

Osciladores LP , XT ou HS – Cristal de quartzo ou Ressonador

Para utilizarmos cristal ou ressonadores em nosso chip, devemos conectá-lo da seguinte maneira:

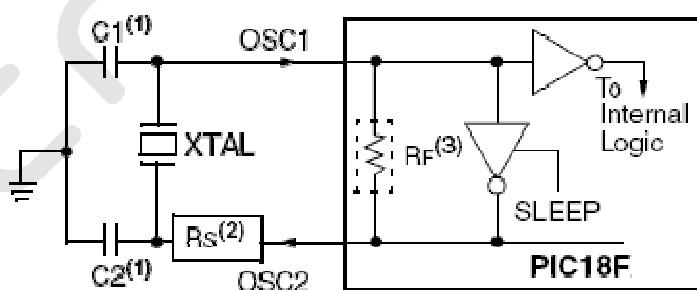


Figura 1.16 – Clock gerado a partir de um cristal de quartzo

Nota:

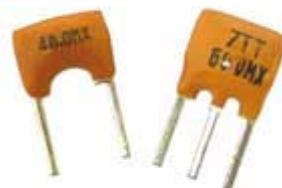
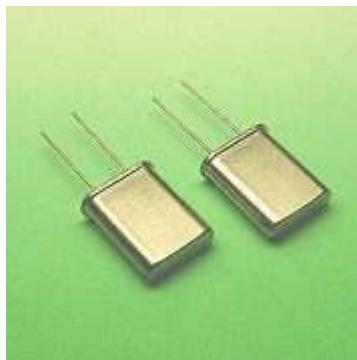
Quando projetamos um circuito com microcontrolador, a regra é colocar o oscilador tão perto quanto possível do microcontrolador, de modo a evitar qualquer interferência nas linhas que ligam o oscilador ao chip.

Note que existe dois capacitores ligados em paralelo com o cristal. O valor desses capacitores variam de acordo com a frequência do cristal utilizado. Segue uma tabela apresentando os valores dos capacitores:

Tipo de oscilador	Freqüência do cristal	Valores típicos para o capacitor	
		C1	C2
LP (cristal de baixa freqüência)	32kHz	33pF	33pF
	200KHz	15pF	15pF

XT (cristal)	200KHz 1.0MHz 4.0MHz	22-68pF 15pF 15pF	22-68pF 15pF 15pF
HS (cristal de alta frequência)	8.0 MHz 20.0MHz 25MHz	15-33pF 15-33pF 15-33pF	15-33pF 15-33pF 15-33pF

Para quem não se recorda de como é um cristal de quartzo ou cerâmico, segue suas fotos:

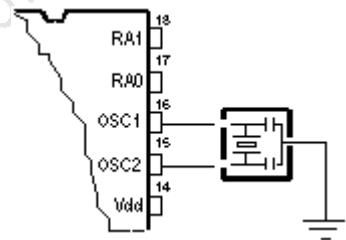


Ressonador cerâmico
cristal de quartzo

Figura 1.17 – Osciladores

Os ressonadores cerâmicos são comercializados em dois tipos: com dois terminais e com três terminais. Os mais utilizados são os ressonadores com três terminais pois não precisam de capacitores externos. O ressonador com dois terminais obedecem a ligação da figura 5.4, enquanto no de três o pino central deve ser aterrado. Segue abaixo o esquema de ligação do ressonador com três terminais:

Os ressonadores cerâmicos é uma segunda opção. Não é tão barato quanto um RC mas é bem mais estável e preciso.



Nota:

Antes de gravarmos um programa no microcontrolador PIC devemos "queimar" os fusíveis de configuração informando qual o tipo de oscilador estamos utilizando em nosso hardware.

Nos projetos do nosso curso iremos utilizar cristal de quartzo de 8 MHz pois garantimos maior precisão nas rotinas de tempo nas execuções das instruções.

OSCSEN (SYSTEM CLOCK SWITCH BIT)

Através desse bit de configuração, podemos selecionar o modo de acionamento do segundo oscilador de baixa frequência do PIC, mas para isso faz necessário habilitarmos o T1OSCEN no TIMER1.

Obs: Caso desejemos utilizar o segundo oscilador (T1OSO e T1OSI), e não venhamos a programar o TIMER1 corretamente, automaticamente o bit de configuração do OSCSEN será desabilitado.

Configuração dos Fusíveis de Energia

PWRTE (SYSTEM CLOCK SWITCH BIT)

Através desse bit, podemos configurar o modo POWER para o acionamento do PIC, quando este bit estiver acionado o modo POWER-UP TIMER estará habilitado. O temporizador de power-up pode ser utilizado para fazer com que o chip permaneça em reset por aproximadamente 72ms após o chip ter sido ligado. E deve ser utilizado nos casos em que o chip não inicializa corretamente devido a um tempo muito elevado para a estabilização da fonte de alimentação ou do circuito de reset externo.

BROWN-Out

O detector de brown-out é utilizado para provocar o reset da CPU no caso de a tensão de alimentação cair abaixo de um determinado limite. Uma vez que a tensão retorne ao seu valor nominal, o circuito de reset é liberado e o programa é reiniciado.

No PIC18F4520 podemos selecionar 4 tipos de tensões para o BROWN-OUT: 2.0V, 2.7V, 4.2V ou 4.5V.

Podemos habilitar ou desabilitar o detector de brown-out, basta selecionar o bit de configuração caso desejemos habilitar o brown-out.

WDT - Watchdog Timer Enable (cão de guarda)

O watchdog timer ou cão de guarda é um sistema interno ao microcontrolador que evita que o programa pare em um determinado ponto. Imagine que você desenvolveu uma aplicação e que por um erro de software em certo momento o seu programa pára (entra em loop infinito). Caso o watchdog dog não seja resetado de tempos em tempos ele irá "estourar", ou seja, chegará ao final da sua contagem fazendo com que o microcontrolador复位 e reinicie todo o processo.

O tempo mínimo para estouro do watchdog é de 18 ms, porém podemos estender esse tempo através do postscale. Com essa função podemos multiplicar o tempo mínimo de 18 ms com os seguintes multiplicadores do prescaler: 1:1; 1:2, 1:4, 1:8, 1:16, 1:32, 1:64, 1:128.

CCP2MX

Podemos através desse bit definir se queremos multiplexar o módulo CCP2 com o pino RC1 (CCP2MX = 0) ou com o pino RB3 (CCP2MX = 1)

Debug

Através desse bit, podemos habilitar o modo DEBUGGER no PIC. Nesse modo é possível emular um programa na própria placa de testes. Emular consiste no processo de, junto com o computador, testar passo a passo o funcionamento do programa que está rodando no microcontrolador. Se esta opção estiver ativa, os pinos de gravação RB6 e RB7 deixam de funcionar como I/O's; caso contrário, o funcionamento desses pinos fica normal.

LVP (Low Voltage Programming)

Normalmente quando o PIC é gravado, é acionado uma tensão de 13Vcc no pino MCLR. Se a opção LVP estiver ativa, para gravar o PIC basta ter nível lógico 1 no pino MCLR, mas neste modo, após a gravação o pino RA5 do PIC não poderá ser utilizado como I/O.

STVREN (Stack Full/Underflow Reset Enable Bit)

O PIC18F4520 possui internamente 31 endereços de pilha. Quando habilitamos esse bit STVREN, toda vez que ultrapassarmos o limite da pilha o microcontrolador será resetado.

Área de Proteções contra Leitura e escrita

Code Protect 0x0800 a 0x1FFF

Caso este bit esteja habilitado, protege contra leitura a memória de programa de 0x0800 a 0x1FFF.

Code Protect 0x2000 a 0x3FFF

Caso este bit esteja habilitado, protege contra leitura a memória de programa de 0x2000 a 0x3FFF.

Code Protect 0x4000 a 0x5FFF

Caso este bit esteja habilitado, protege contra leitura a memória de programa de 0x4000 a 0x5FFF.

Code Protect 0x6000 a 0x7FFF

Caso este bit esteja habilitado, protege contra leitura a memória de programa de 0x6000 a 0x7FFF.

Data EE Read Protect

Caso este bit esteja habilitado, protege contra leitura os 256 bytes da memória EEPROM (memória não volátil).

Code Protect Boot

Caso este bit esteja habilitado, protege contra leitura a região de memória Boot (área da memória de programa) 0x0000 até 0x0FF0.

Table Write Protect 0x0200 a 0x1FFF

Caso este bit esteja habilitado, protege a memória de programa contra escrita por tabela no endereço especificado.

Table Write Protect 0x2000 a 0x3FFF

Caso este bit esteja habilitado, protege a memória de programa contra escrita por tabela no endereço especificado.

Table Write Protect 0x4000 a 0x5FFF

Caso este bit esteja habilitado, protege a memória de programa contra escrita por tabela no endereço especificado.

Table Write Protect 0x6000 a 0x7FFF

Caso este bit esteja habilitado, protege a memória de programa contra escrita por tabela no endereço especificado.

■ Data Write Protect

Caso este bit esteja habilitado, protege contra escrita os 256 bytes da memória EEPROM (memória não volátil).

■ Table Write Protect Boot

Caso este bit esteja habilitado, protege contra escrita por tabela a área de boot.

■ Config Write Protect

Caso este bit esteja habilitado, protege contra escrita a área de configuração do microcontrolador.

■ Table Read Protect 0x0800 a 0x1FFF

Caso este bit esteja habilitado, protege a memória de programa contra leitura por tabela no endereço especificado.

■ Table Read Protect 0x2000 a 0x3FFF

Caso este bit esteja habilitado, protege a memória de programa contra leitura por tabela no endereço especificado.

■ Table Read Protect 0x4000 a 0x5FFF

Caso este bit esteja habilitado, protege a memória de programa contra leitura por tabela no endereço especificado.

■ Table Read Protect 0x6000 a 0x7FFF

Caso este bit esteja habilitado, protege a memória de programa contra leitura por tabela no endereço especificado.

Unidade 2 - Canais A/D

Conversor A/D do PIC

O PIC18F4520 possui internamente 12 canais de A/D com resolução de 10 bits cada. Através de um conversor A/D com resolução de 10 bits e tensão de referência de 5V, podemos obter um valor de 4,8876... mV, pois $5V / (2^{10} - 1) = 4,8876\dots$ mV.

Os conversores A/D dos PICs utilizam uma técnica de aproximação sucessivas, normalmente com resolução de 10 bits, com clock selecionável pelo usuário e múltiplas entradas multiplexadas.

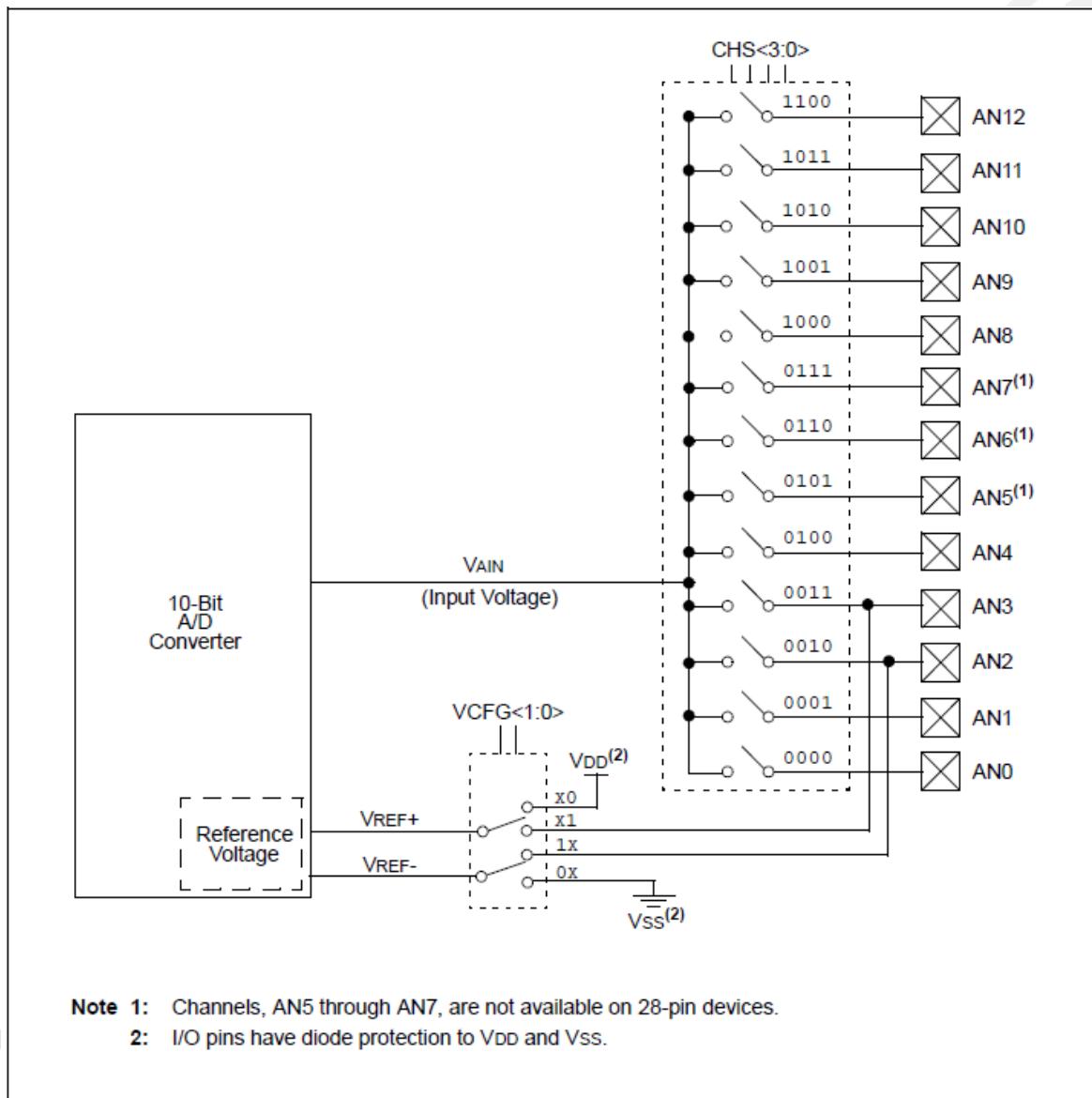


Figura 2.1- Canais AD multiplexados

Os registradores que participam da configuração do conversor A/D são: ADCON0, ADCON1 e ADCON2.

Nosso A/D possui resolução de 10 bits (8 + 2 bits), onde os valores das conversões são armazenados nos registradores ADRESH e ADRESL.

Registrador ADCON0:

Vejamos o funcionamento de cada bit desse registrador:

ADCON0:

Tabela 2.1 – ADCONO

-	-	CHS3	CHS2	CHS1	CHS0	GO_DONE	ADON
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

CHS3 ... CHS0: Bit de seleção de canal A/D

- 0000 = Channel 0 (AN0)
- 0001 = Channel 1 (AN1)
- 0010 = Channel 2 (AN2)
- 0011 = Channel 3 (AN3)
- 0100 = Channel 4 (AN4)
- 0101 = Channel 5 (AN5)^(1,2)
- 0110 = Channel 6 (AN6)^(1,2)
- 0111 = Channel 7 (AN7)^(1,2)
- 1000 = Channel 8 (AN8)
- 1001 = Channel 9 (AN9)
- 1010 = Channel 10 (AN10)
- 1011 = Channel 11 (AN11)
- 1100 = Channel 12 (AN12)
- 1101 = Unimplemented⁽²⁾
- 1110 = Unimplemented⁽²⁾
- 1111 = Unimplemented⁽²⁾

Figura 2.2 – Seleção A/D

GO/DONE: Bit de status da conversão A/D

0 - Conversão A/D não está sendo realizada

1 - Conversão A/D está sendo realizada

ADON: Liga ou desliga o A/D

0 - Conversor A/D desligado

1 - Conversor A/D ligado

Registrador ADCON1:

Vejamos o funcionamento de cada bit desse registrador:

ADCON1:

	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1

VCFG1: Bit de configuração de tensão de referência (VREF-)

0 - VREF-(AN2)

1 - VSS

VCFG0: Bit de configuração de tensão de referência (VREF+)

0 - VREF+(AN3)

1 - VDD

PCFG3, PCFG2, PCFG1, PCFG0: bits configuração do A/D

Tabela 2.2 – Configuração Pinos A/D

PCFG3: PCFG0	AN12	AN11	AN10	AN9	AN8	AN7 ⁽²⁾	AN6 ⁽²⁾	AN5 ⁽²⁾	AN4	AN3	AN2	AN1	AN0
0000 ⁽¹⁾	A	A	A	A	A	A	A	A	A	A	A	A	A
0001	A	A	A	A	A	A	A	A	A	A	A	A	A
0010	A	A	A	A	A	A	A	A	A	A	A	A	A
0011	D	A	A	A	A	A	A	A	A	A	A	A	A
0100	D	D	A	A	A	A	A	A	A	A	A	A	A
0101	D	D	D	A	A	A	A	A	A	A	A	A	A
0110	D	D	D	D	A	A	A	A	A	A	A	A	A
0111 ⁽¹⁾	D	D	D	D	D	A	A	A	A	A	A	A	A
1000	D	D	D	D	D	D	A	A	A	A	A	A	A
1001	D	D	D	D	D	D	D	A	A	A	A	A	A
1010	D	D	D	D	D	D	D	D	A	A	A	A	A
1011	D	D	D	D	D	D	D	D	D	A	A	A	A
1100	D	D	D	D	D	D	D	D	D	D	A	A	A
1101	D	D	D	D	D	D	D	D	D	D	D	A	A
1110	D	D	D	D	D	D	D	D	D	D	D	D	A
1111	D	D	D	D	D	D	D	D	D	D	D	D	D

Registrador ADCON2:

Vejamos o funcionamento de cada bit desse registrador:

ADCON2:

ADFM	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1

ADFM : Ajusta o formato do resultado da conversão A/D

1 - Justifica a direita

0 - Justifica a esquerda

ACQT2 ... ACQT0: Seleção de tempo de aquisição

111 = 20 TAD

110 = 16 TAD

101 = 12 TAD

100 = 8 TAD

011 = 6 TAD

010 = 4 TAD

001 = 2 TAD

000 = 0 TAD⁽¹⁾

ADCS2 ... ADCS0: Seleção do clock do conversor A/D

111 = FRC (clock derived from A/D RC oscillator)⁽¹⁾

110 = Fosc/64

101 = Fosc/16

100 = Fosc/4

011 = FRC (clock derived from A/D RC oscillator)⁽¹⁾

010 = Fosc/32

001 = Fosc/8

000 = Fosc/2

O bit de ADFM tem a função de organizar o resultado da conversão A/D, de forma que os valores convertidos sejam justificados a direita ou a esquerda nos registradores ADRESH e ADRESL. Caso venhamos configurar ADFM = 1, organizamos o valor da conversão a direita, ou seja, os oitos bits menos significativo serão armazenados em ADRESL, e os 2 bits mais significativo serão armazenados em ADRESH.

Caso ADFM = 0, justificaremos a esquerda os valores de conversão, desta forma os oitos bits mais significativos ficarão em ADRESH e os 2 menos significativo ficará em ADRESL.

Fórmula de conversão analógico para digital:

$$Rad = (Vin - Vref-) * 1023 / (Vref+ - Vref-)$$

onde:

Rad = Resultado da conversão

Vref+ e Vref- = valor de tensão de referência máxima e mínima

Vin = Tensão de entrada no pino A/D

obs: 1023 é o valor máximo de conversão do A/D, lembrando que o valor é de 0 a 1023.

Trabalhando com AD no MikroC

No MikroC utilizamos os seguinte função para leitura do conversor A/D do PIC:

Sintaxe:

Adc_Read (canal_AD)

Na qual 'canal_AD' é o canal do conversor AD do PIC que desejamos ler. O número do canal AD depende do tipo de PIC usado.

Exemplo:

```
unsigned int leitura_ad;           //cria variável leitura_ad com ranger 0 a 65535
leitura_ad = Adc_Read(0);          //lê canal ad0 do PIC e salva na variável leitura_ad
```

A função Adc_read salva o valor da conversão AD (canal 0 - pino RA0) na variável leitura_ad.

Lembre-se que o valor da resolução do AD do PIC18F45200 é de 10 bits, e por esse motivo devemos utilizar uma variável do tipo inteiro para armazenar o valor do A/D.

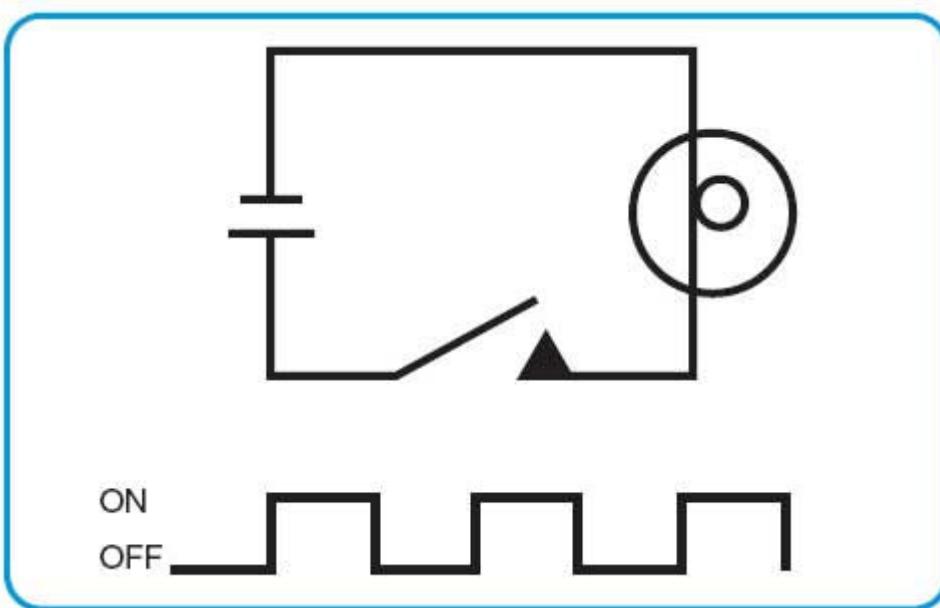
Unidade 3 – Canal PWM do PIC

Introdução

Os controles de potência, inversores de freqüência, conversores para servomotor, fonte chaveadas e muitos outros circuitos utilizam a tecnologia do PWM (Pulse Width Modulation) ou Modulação de Largura de Pulso como base de seu funcionamento.



PWM é a abreviação de Pulse Width Modulation ou Modulação de Largura de Pulso. Para que se entenda como funciona esta tecnologia no controle de potência, partimos de um circuito imaginário formado por um interruptor de ação muito rápida e uma carga que deve ser controlada, conforme a figura abaixo:



Quando o interruptor está aberto não há corrente na lâmpada e a potência aplicada é nula. No instante em que o interruptor é fechado, a carga recebe a tensão total da fonte e a potência aplicada é máxima.

Como fazer para obter uma potência intermediária, digamos 50%, aplicada à carga? Uma idéia é fazermos com que a chave seja aberta e fechada rapidamente de modo a ficar 50% do

tempo aberta e 50% fechada. Isso significa que, em média, teremos metade do tempo com corrente e metade do tempo sem corrente.

A potência média é, portanto, a própria tensão média aplicada à carga é neste caso 50% da tensão de entrada.

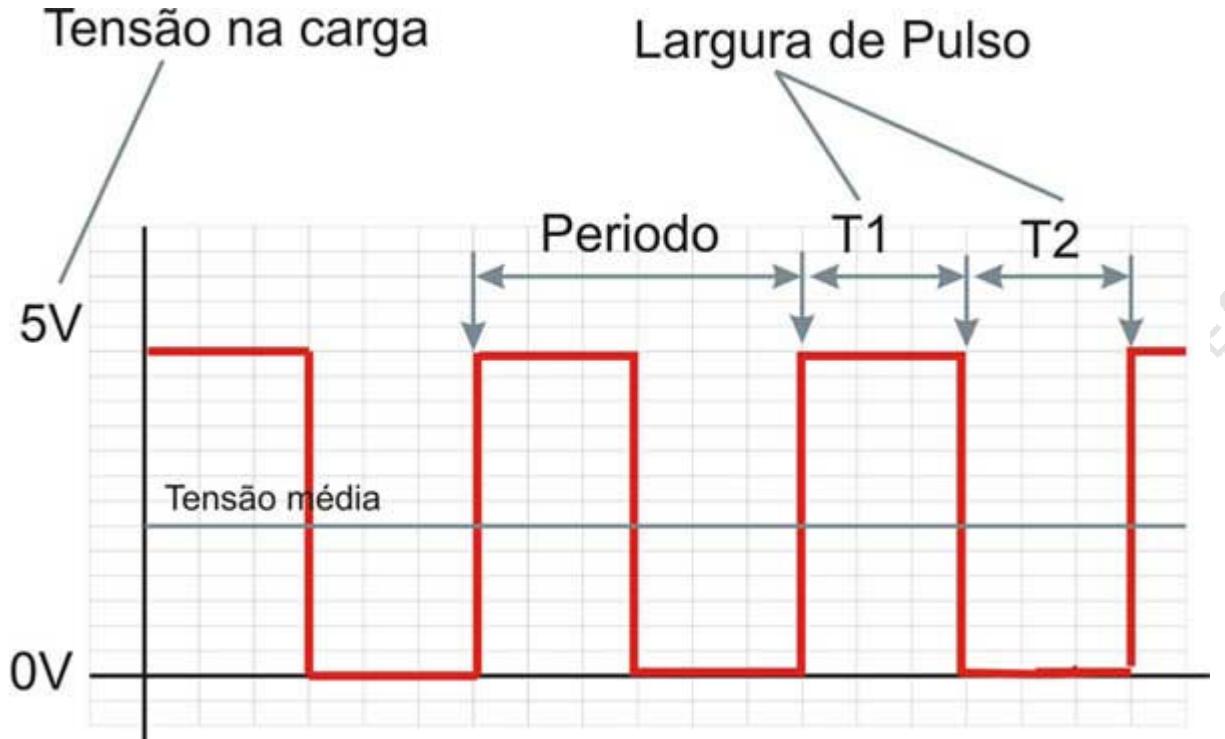


Figura 3.1 – Largura de Pulso

Veja que o interruptor fechado pode definir uma largura de pulso pelo tempo em que ele fica nesta condição, e um intervalo entre pulsos pelo tempo em que ele fica aberto. Os dois tempos juntos definem o período e, portanto, uma freqüência de controle. A relação entre o tempo em que temos o pulso e a duração de um ciclo completo de operação do interruptor nos define ainda o ciclo ativo.

Variando-se a largura do pulso e também o intervalo de modo a termos ciclos ativos diferentes, podemos controlar a potência média aplicada a uma carga. Assim, quando a largura do pulso varia de zero até o máximo, a potência também varia na mesma proporção (duty cycle), conforme está indicado na figura abaixo:

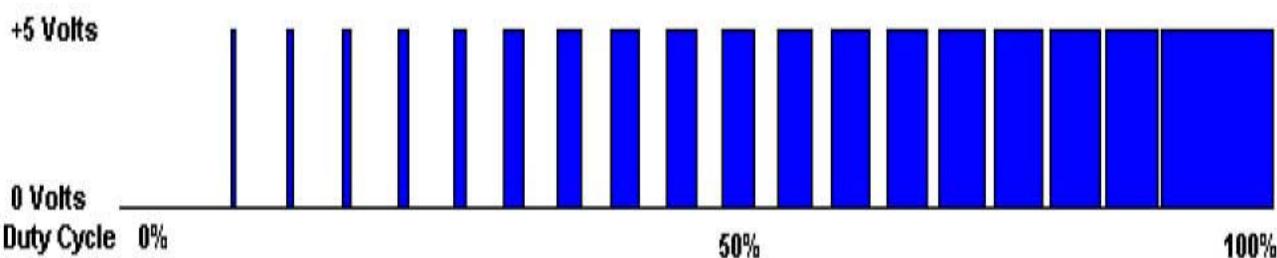


Figura 3.2 – Variação mínima ao máximo do duty Cicle do PWM.

Este princípio é usado justamente no controle PWM: modulamos (variamos) a largura do pulso de modo a controlar o ciclo ativo do sinal aplicado a uma carga e, com isso, a potência aplicada a ela.

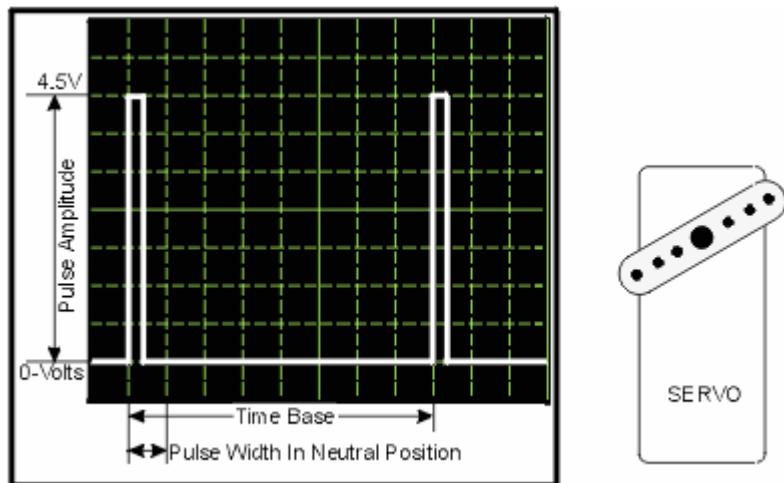


Figura 3.3 – variação do duty cicle do sinal PWM

Trabalhando com PWM no PIC

O microcontrolador PIC18F45200 possui internamente 2 módulos CCP (Capture/Compare/PWM Module), CCP1 e CCP2. Podemos manipular com grande facilidade as instruções para geração de sinal PWM no MikroC. Acompanhe:

Os comandos para manipular o módulo CCP1 são os seguintes:

- PWM1_Init
- PWM1_Set_Duty
- PWM1_Start
- PWM1_Stop

Função de Inicialização da geração do sinal PWM:

No MikroC utilizamos a função `Pwm1_Init()` para informar a freqüência do sinal PWM:

Sintaxe

`PWM1_Init (valor_da_frequência em Hz)`

onde:

`valor_da_frequência`: fator da freqüência em Hz do sinal PWM.

PS.: Consultar datasheet do PIC utilizado para saber a freqüência de oscilação.

Exemplo:

```
PWM1_Init (4000) //inicia pwm com freqüência de 4khz
```

Função de duty Cycle:

Através da função `PWM1_Set_Duty ()` podemos controlar o duty cycle do sinal PWM. O valor do duty cycle varia de 0 a 255, onde "0" é igual a 0%, "127" igual a 50% e "255" igual a 100%.

Sintaxe:

`PWM1_Set_Duty (valor_duty_cycle)`

onde:

`valor_duty_cycle`: valor do tipo char (0 à 255) que determina a porcentagem do duty cycle PWM.

Exemplo:

```
PWM1_Set_Duty (192); // carrega duty cycle para 75%
```

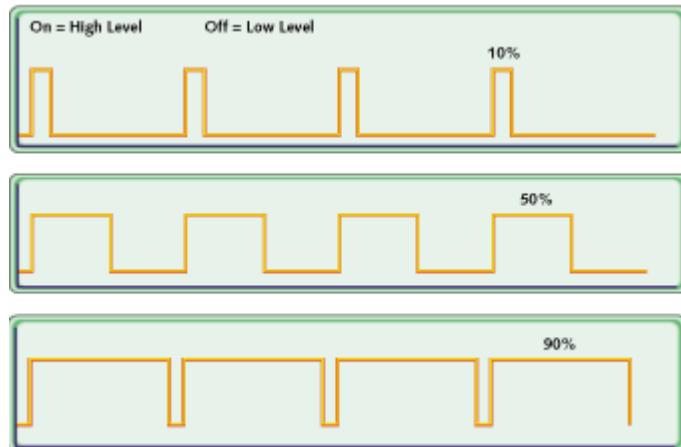


Figura 3.4 – Variação do duty cycle

Para outros valores de duty cycle calculamos através da formula: (Percentual * 255) /100
No caso de configurarmos o duty conforme a figura acima, devemos carregar o seguinte valor para:

Duty de 10%

PWM1_Set_Duty(25); // carrega valor 26 pois : (10% * 255) / 100 = 25,5 , como somente podemos colocar valores inteiros entre 0 a 255, o valor será arredondado para 25

Duty de 50%

PWM1_Set_Duty(127); // carrega valor 127 pois : (50% * 255) / 100 = 127,50 , como somente podemos colocar valores inteiros entre 0 a 255, o valor será arredondado para 127

Duty de 90%

PWM1_Set_Duty(229); // carrega valor 229 pois : (90% * 255) / 100 = 229,50 , como somente podemos colocar valores inteiros entre 0 a 255, o valor será arredondado para 229

Função de Start e Stop:

Através da função Pwm_Start, damos início a geração do sinal PWM no PIC, e através da função Pwm_Stop finalizamos a geração do sinal Pwm.

Sintaxe:

Pwm_Start	' inicial a geração do sinal PWM no módulo CCP1 do PIC
Pwm_Stop	'interrompe a geração do sinal PWM no módulo CCP1 do PIC

Unidade 4 - TIMERS/COUNTERS

Os Timers/Counters

Os timers são ferramentas internas dos microcontroladores em geral que servem para contagem de tempo, eventos, temporização entre outras funções.

O PIC18F4520 possui internamente 4 TIMERS:

- TIMER0
- TIMER1
- TIMER2
- TIMER3

Vamos conhecer cada um desses temporizadores:

TIMER0

O TIMER0 é um temporizador/contador de 8 ou 16 bits que possuem duas funções básicas:

Contagem de eventos externos (quando a entrada de clock é feita por meio do pino RA4/T0CKI);

Temporização (contagem de tempo) quando os pulsos de contagem é proveniente do clock interno do PIC (temporização baseado no ciclo de máquina).

O TIMER0 pode ser configurado para trabalhar com prescaler dedicado. O prescaler nada mais é do que um contador/divisor programável que é utilizado para reduzir a freqüência do sinal de clock aplicado ao TIMER0 por um fator conhecido.

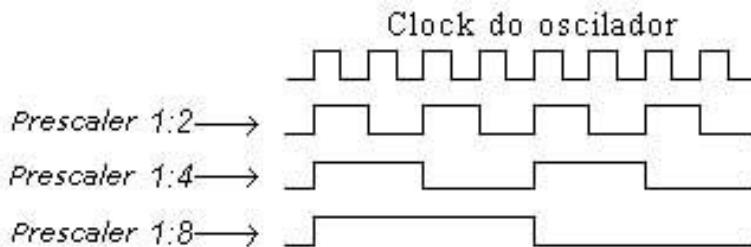


Figura 4.1 – Divisão do Prescaler

O valor do prescaler pode ser configurado a partir dos bits T0PS2, T0PS1 e T0PS0 do registrador T0CON (TIMER0 CONTROL REGISTER);

O prescaler passa a ser uma importantíssima ferramenta do timer, pois através dele conseguimos gerar tempos muito maiores.

Quando o TIMER0 é configurado para operar no modo de 8 bits, podemos efetuar contagens de 0 a 255 (limite da capacidade para 8 bits). Quando a contagem chega até seu valor máximo de 255, o próximo pulso acarretaria o que chamamos de "estouro de contagem", fazendo com que o valor de contagem de início novamente a partir do 0 (zero).

No caso anterior, caso tenhamos a interrupção do TIMER0 habilitada, no momento que ocorre o "estouro de contagem", seria gerado um pedido de interrupção do TIMER0.

No modo 16 bits do TIMER0, seu funcionamento é igual ao modo de 8 bits, porém neste caso a faixa de contagem é de 0 a 65535.

Os valores de iniciais de temporização/contagem devem ser carregados nos registradores especiais intitulado de TMROL (TIMER0 Module Low Byte Register) e TMROH (TIMER0 Module High Byte Register);

Quando programamos os timers para atuarem como temporizadores, estamos considerando que os pulsos que incrementam os registradores de contagem são proveniente do valor do oscilador / 4, ou seja, caso estivermos utilizando um cristal de 4MHz, iremos incrementar em uma unidade os registradores de contagem a cada 1 us, pois $4\text{MHz}/4 = 1\text{MHz} = 1\text{us}$ (microsegundos).

Dica:

Repare que a unidade resultante da divisão da Frequencia do oscilador / 4 (ciclo de máquina) esta em MHz (unidade de freqüência), neste caso para sabermos o tempo (periodo), basta dividir 1 / seu valor.

Ex: $1 / (\text{Fosc} / 4)$

Para um cristal de 8 MHz teremos o seguinte tempo de ciclo de máquina:

$$1 / (8 / 4) = 0,5 \text{ us} \text{ (microsegundos).}$$

Os Registradores relacionados ao TIMERO são:

Tabela 4.1- Registrador relacionados com o TIMERO

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on All Other RESETS
TMR0L	Timer0 Module Low Byte Register								xxxx xxxx	uuuu uuuu
TMR0H	Timer0 Module High Byte Register								0000 0000	0000 0000
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
T0CON	TMR0ON	T08BIT	T0CS	TOSE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	1111 1111
TRISA	—	PORTA Data Direction Register							-111 1111	-111 1111

TMR0L é um registrador de contagem de 8 bits que possui a função de armazenar a parte baixa do valor de contagem programada do TIMERO.

TMR0H é um registrador de contagem de 8 bits que possui a função de armazenar a parte alta do valor de contagem programada do TIMERO.

Os registradores TMR0L e TMR0H juntos formam um único registrador de 16 bits, que nos permite uma contagem máxima de 0 a 65535.

INTCON é um registrador de 8 bits responsável pela configuração do modo de operação do TIMERO. Podemos definir através desse registrador o valor do prescaler que desejamos acionar, o modo de operação de contagem de 8 ou 16 bits, seleção da fonte de clock (interna ou externa) para o timer, seleção de disparo do timer através de borda de subida ou borda de descida do clock externo no pino RA4/T0CK1.

Abaixo segue o diagrama esquemático do Timer0.

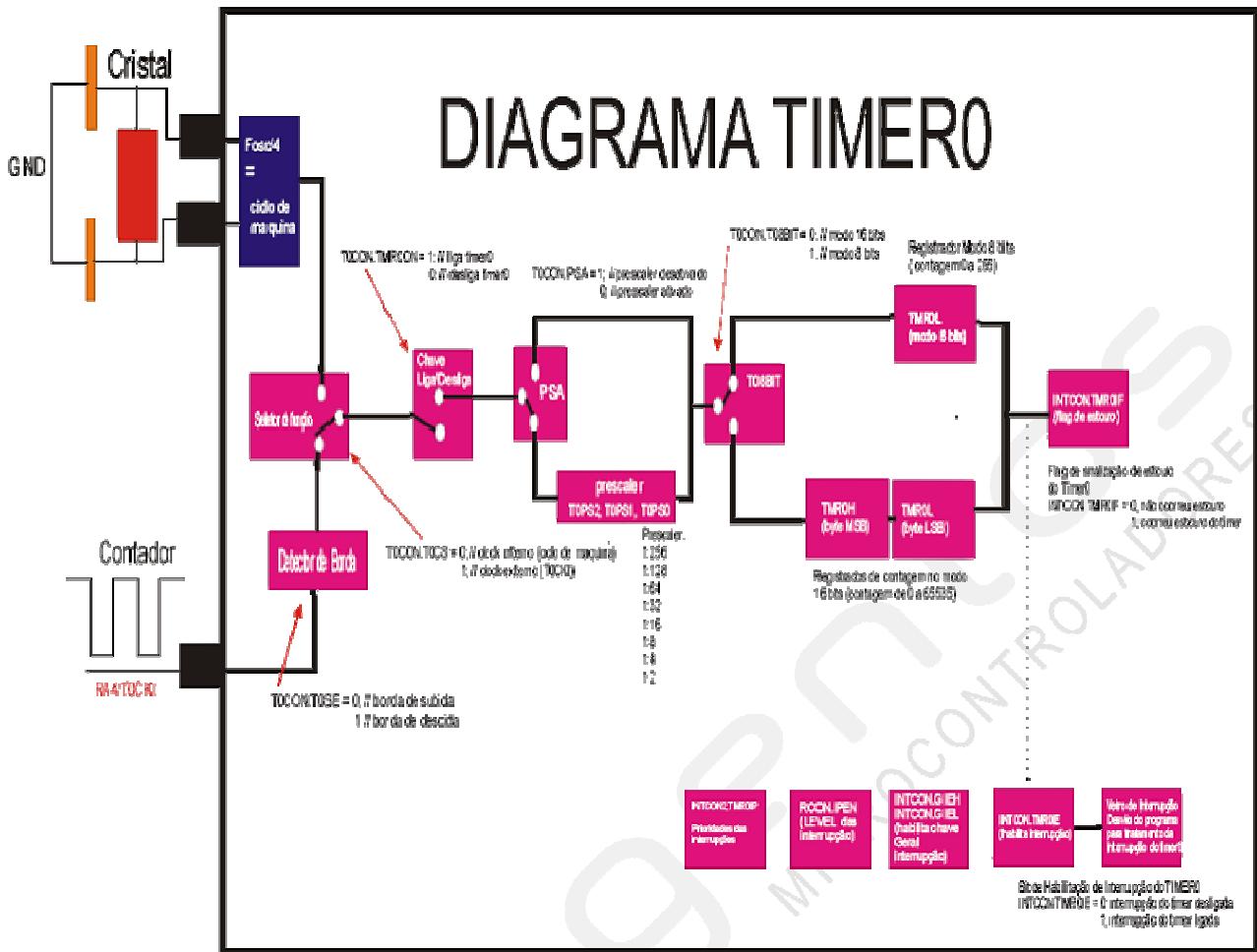


Figura 4.2 – Diagrama interno do TIMER0

O registrador T0CON é responsável pela configuração pelo modo de operação do Timer0, é de vital importância conhecer todas as funções de seus bits.

T0CON: (TIMER0 Counter Register)

Tabela 4.2 – T0CON

TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

TMR0ON: Bit de controle para ligar e desligar a contagem do TIMER0;

0 – desabilita a contagem do TIMER0

1 – Habilita a contagem do TIMER0

T08BIT: Bit de seleção de funcionamento do TIMER0 em 8 ou 16 bits;

0 – TIMER0 configurado para funcionar como contador/temporizador em 16 bits;

1 – TIMER0 esta configurado para funcionar como contador /temporizador de 8 bits;

T0CS: Bit de seleção de fonte de clock para o TIMER0;

0 – A fonte de clock é proveniente do meio interno do chip

1 – A fonte de clock é proveniente do pino T0CKI (meio externo)

T0SE: Bit de seleção de borda (válido somente caso a fonte de clock seja pelo meio externo ao chip)

0 – Contagem do timer ocorre por borda de subida no Pino T0CKI

1 – o incremento do timer ocorre na borda de descida no pino T0CKI

/PSA: Bit de seleção da utilização do prescaler;

1 – TIMER0 não utiliza prescaler. A cada alteração do pulso de clock, corre incremento de uma unidade nos registradores de contagem.

0 - TIMER0 utiliza prescaler.

T0PS2; T0PS1; T0PS0 : Bits de seleção de fonte de prescaler

Tabela 4.3 - Bits de seleção de fonte de prescaler(Timer 0)

T0PS2	T0PS1	T0PS0	Prescaler
1	1	1	1:256
1	1	0	1:128
1	0	1	1:64
1	0	0	1:32
0	1	1	1:16
0	1	0	1:8
0	0	1	1:4
0	0	0	1:2

Configuração do Prescaler:

Como sabemos, através do prescaler conseguimos tempos maiores com os timers, para entendermos melhor sua utilização acompanhe o exemplo abaixo:

Digamos que o ciclo de máquina no PIC seja de 1us e o TIMER0 esteja configurado no modo 8 bits (contagem de 0 a 255) e sem o prescaler. O TIMER0 irá "estourar" sua contagem em 256us.

Agora digamos que para o mesmo anunciado anterior, configurarmos e acionarmos o prescaler para 1:2. Nossa intervalo de "estouro" do TIMER0 não mais será 256us mas sim 512us.

Se ao invés de prescaler de 1:2 utilizarmos prescaler 1:32, o tempo agora será de 256us x 32 = 8192us.

Como calcular o tempo de estouro do TIMER0?

Para calcular o tempo de "estouro" do TIMER com prescaler, utilizamos a seguinte fórmula:

Tempo de estouro: ciclo de máquina x prescaler x (modo 8 /16bits - valor de Contagem).

Em que:

Ciclo de máquina: é o valor da relação entre: 1 / (Fosc / 4), onde Fosc é justamente a frequência do cristal.

Obs: manter modo PLL desabilitado, iremos estudar esta função mais adiante.

Prescaler : é o fator de divisão programado no prescaler.

Modo 8/16bits: é o modo de contagem programado no TIMER0, para 8 bits o valor é 256, e para 16 bits, o valor será de 65536.

Valor de contagem: é o valor de carregagem no registrador de contagem TMROH eTMRL.

Exemplo:

Precisamos ligar e desligar um relé a cada 1 segundo (um segundo ligado e um segundo desligado), estamos utilizando um cristal de 8Mhz, utilizaremos para isso os recursos do Temporizador Timer0, acompanhe:

Calculo do tempo de estouro do Timer0:

```
T0CON = 0b10000110; //modo 16 bits, com prescaler 1:128, fonte de clock interno
TMROL = 0xF7; //carrega valores de contagem C2F7 equivale a
TMROH = 0xC2;
```

Formula:

Ciclo de Máquina x prescaler x (modo 8 /16 bits - valor de carregem nos registradores de contagem)

Tempo de estouro: 0.5us x 128 x (65536 - 49911)

Tempo de estouro: 1.000.000 us (microsegundos) ou 1 segundo.

O registrador INTCON

O registrador INTCON possui diversas funções, entre elas a de habilitar algumas interrupções do PIC (veremos mais adiante no nosso curso) e de armazenar os bits de estatus de estouro do TIMER0:

INTCON (Interrupt Control)

Tabela 4.4 - INTCON

GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

Iremos estudar todos esses bits mais adiante em nosso curso, pois eles tratam das configurações das interrupções, e neste momento não é importante conhecer alguns deles. Vamos apresentar somente os bits que são utilizados nos TIMERS.

INTCON. GIEH = bit de acionamento da "chave geral das interrupções" e habilitação das interrupções de alta prioridade

0 - liga a chave geral das interrupções

1 - desliga a chave geral das interrupções

INTCON. GIEL = bit de habilitação das interrupções de baixa prioridade

0 - TIMER0 não ocorreu estouro de contagem (overflow)

1 - TIMER0 ocorreu estouro de contagem (overflow). Este bit deve ser apagado por software.

INTCON.TMR0IE = bit que habilita interrupção do TIMER0 na ocorrência do estouro de contagem.

0 - Desabilita interrupção do TIMER0 por ocorrência de estouro de contagem (overflow)

1 - Habilita interrupção do TIMER0 por ocorrência de estouro de contagem (overflow).

INTCON. TMR0IF = bit sinalizador de estouro do TIMER0

0 - TIMER0 não ocorreu estouro de contagem (overflow)

1 - TIMER0 ocorreu estouro de contagem (overflow). Este bit deve ser apagado por software.

Aprendendo a programar os TIMER0:

Conhecemos os registradores responsáveis pela configuração do TIMER0 do PIC, agora vamos configurá-los:

Vamos nos recordar da fórmula do TIMER0:

Tempo de estouro: ciclo de máquina x prescaler x (modo 8 /16bits – valor de contagem).

Exemplo 1 :

Se tivermos conectado ao nosso PIC um cristal de 8 MHz e o TIMER0 seja programado como temporizador, com prescaler de 1:4, modo 8 bits e contagem inicial em TMR0L = 0, teremos então:

Tempo de estouro: ciclo de máquina x prescaler x (modo 8 /16bits – valor de contagem).

Tempo de estouro: 0,5us * 4 *(255 - 0)

Tempo de estouro: 510us

Exemplo 2 :

Se tivermos conectado ao nosso PIC um cristal de 8 MHz e o TIMER0 seja programado como temporizador, com prescaler de 1:256, modo 16 bits e contagem inicial em TMR0L = 0 e TMR0H = 0, teremos então:

Tempo de estouro: ciclo de máquina x prescaler x (modo 8 /16bits – valor de contagem).

Tempo de estouro: 0,5us * 256 * (65536 - 0)

Tempo de estouro: 8.388.608us, ou seja, 8, 388 segundos

Obs:

Repare que este é o tempo máximo de estouro do TIMER0

Modo contador do TIMER0 do PIC:

Neste novo exemplo de programa, vamos programar o TIMER0 do PIC para trabalhar como contador de pulsos externos. Neste modo, os pulsos externos, são aplicados no pino RA0/T0CK1 do PIC. Devemos contar apenas 10 pulsos externos, que neste caso, configuraremos o TIMER0 do PIC no modo 8 bits com prescaler 1 : 1, e os pulsos serão lidos por borda de descida. Ao completar a contagem de 10 pulsos, o led conectado ao pino RB1 do PIC deverá acender para sinalizar o fim da contagem.

Esquema elétrico:

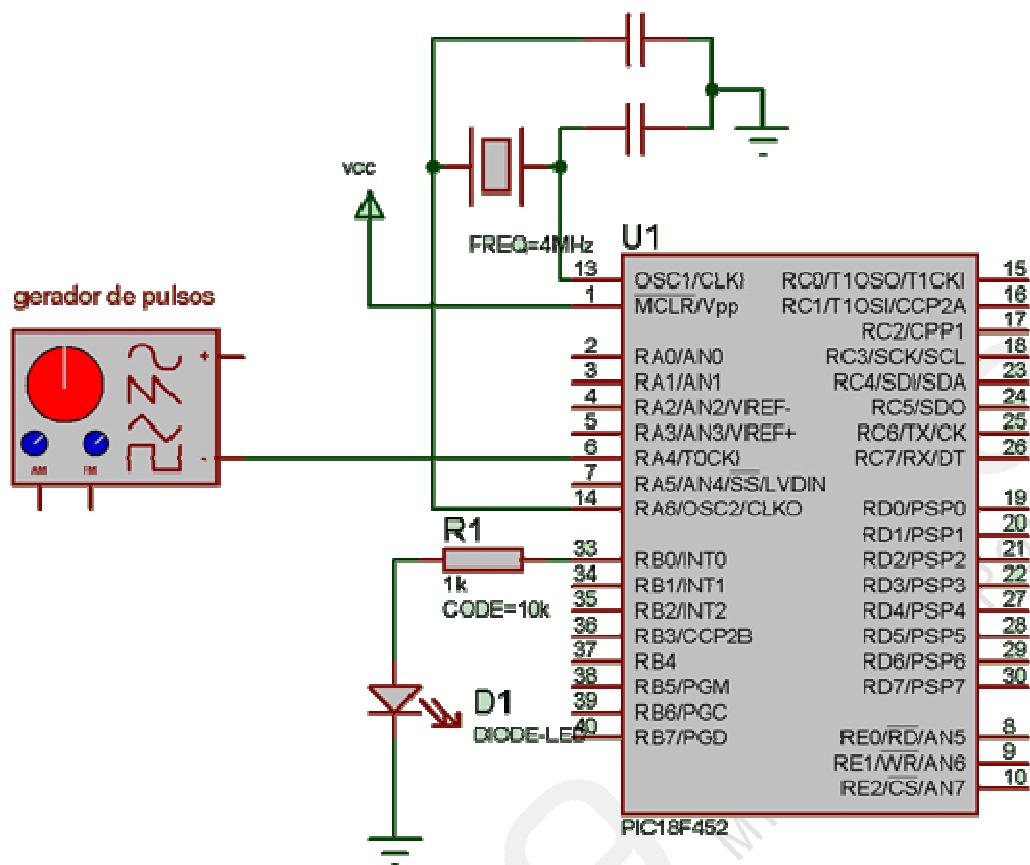


Figura 4.3 – Aplicação com o modo contador do TIMER0

Para realizar o seguinte desafio é necessário configurar os seguintes registradores do TIMER0:

INTCON.TMR0IF (Bit de estouro do TIMER0)

TMR0L (registrador de contagem do TIMER0)

T0CON (registrador de configuração do modo de operação do TIMER0)

Programa:

```
/*
'Microgenios | MicroControladores
'Site: www.Microgenios.com.br
'Autor: Fernando Simplicio de Sousa
' ****
' Programa exemplo: TIMER0_Contador.c
'Cristal: 8MHz - modo HS
'Microcontrolador PIC18F45200 Microchip
'Tools: Kit PICgenios PIC18F Módulo Profissional Microgenios
'Configuração: DIP1 - Chave 09 -ON
' ****
'Objetivo:
'Este programa tem por objetivo ativar o modo contador do Timer0
'OBS: ESTE PROGRAMA É PARA FINS DE ESTUDOS. NÃO HÁ COMO SIMULAR MODO DE
'CONTAGEM COM O TIMER0 NO KIT PICGENIOS, POIS O SENSOR INFRAVERMELHO ESTA LIGADO
'NO PINO T0CKI, AO INVÉS DE T0CKI
' ****
void main(){

    TRISD = 0;           // 'define portd como saída
    PORTD = 0;           // 'zera portd

    TRISA.RA4 = 1; // 'pino contador T0CKI configurado como entrada

    /*'obs: este pino é naturalmente dreno aberto, neste caso é necessário pull-up
    'ou pull-down externo
    ****'calculo temporização 1 segundo ****
    'calculo de temporização de 1 segundo:
    'tempo = ciclo_maquina * prescaler * (modo 8/16 - valor inicial)
    'logo
    '1000.000us = 0.5us * 128 * (65536 - valor_inicial)
    'valor inicial = 49911 ou C2F7h */

    T0CON = 0B11101000;//'modo contador, prescaler off, borda descida, modo 8bits
    TMR0L = 246; //'contar 10 pulsos | 256 - 10 = 246 carrega valores de contagem
    INTCON.TMR0IF = 0; //'apaga flag de estou do timer0

    while (1){
        if (INTCON.TMR0IF==1){
            PORTD.RD0 = ~ PORTD.RD0; // 'inverte o valor do led1
            TMR0L = 246; //'recarga do timer0
            INTCON.TMR0IF = 0; //'apaga flag de estou do timer0
        }
    }
}
```

Estudo detalhado do programa:

Repare que os comandos abaixo que o registrador T0CON foi configurado como contador de pulsos externos, modo de operação do TIMER0 de 8 bits, com prescaler 1:1.

O registrador de contagem TMR0L é o registrador onde será armazenado os valores de contagem dos pulsos externos, e neste caso ele foi pré configurado com o valor 246, devido a necessidade de contarmos apenas 10 pulsos. Devemos lembrar que os registradores de contagem do TIMER0 são incrementados em 1 unidade e contam para cima, neste caso, como desejamos programar o TIMER0 para que a cada 10 pulsos no pino RA0/T0CK1 seja gerado um estouro, devemos carregar em TMR0L o valor da diferença entre o valor máximo de contagem para o modo 8 bits, que é 256, pelo valor da quantidade de pulsos a ser contado, que neste caso é 10. Então teremos:

$$256 - 10 = 246$$

Utilizaremos modo contador no Kit PICgenios quando utilizarmos o TIMER1 nos projetos. Mais adiante iremos utilizar interrupções com nossos TIMERS/COUNTERS.

TIMER1

O TIMER1:

Realizamos nas unidades passadas um estudo não muito aprofundado sobre as funções e características dos TIMERS do PIC. Agora chegou o momento de estudarmos mais

profundamente os recursos e a programação dos registradores e modos de trabalho desses temporizadores e contadores.

O TIMER1 pode operar como temporizador ou como contador de 16 bits, suas características são muito parecida com a do TIMER0.

Dizemos que o timer está operando como temporizador quando a referência do clock de incremento da contagem é realizada pelo clock interno do PIC, e dizemos que o timer esta operando como contador quando a referência do clock de incremento da contagem é proveniente de um clock externo ao PIC. (pulso aplicado no pino RB6/T1OSO/T1CK1).

Registradores de configuração do TIMER1:

Tabela 4.5 – Registrador relacionados com o TIMER1 – (as células em branco)

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on All Other RESETS
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000X	0000 000u
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000	0000 0000
TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
T1CON	RD16	—	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	0-00 0000	u-uu uuuu

P1R1: é um registrador onde é armazenado os bits de status das interrupções e estouro dos timers.

P1E1: é um registrador de 8 bits onde é habilitado as interrupções do PIC.

TMR1L é um registrador de contagem de 8 bits que possui a função de armazenar a parte baixa do valor de contagem programada do TIMER0.

TMR1H é um registrador de contagem de 8 bits que possui a função de armazenar a parte alta do valor de contagem programada do TIMER0.

Os registradores TMROL e TMROH juntos formam um único registrador de 16 bits, que nos permite uma contagem máxima de 0 a 65535.

T1CON é um registrador de 8 bits responsável pela configuração do modo de operação do TIMER1. Podemos definir através desse registrador o valor do prescaler que desejamos acionar, o modo de operação de contagem de 8 ou 16 bits, seleção da fonte de clock (interna ou externa) para o timer, seleção de disparo do timer através de borda de subida ou borda de descida do clock externo no pino RB6/T1OSO/T1CK1.

T1CON: (TIMER1 CONTROL REGISTER):

Tabela 4.6 – T1CON

RD16	---	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

RD16: bit de leitura e escrita em 16bits (utilizado para o modo comparador do canal CCP)

1 – leitura e escrita em 16 bits habilitada

0 – leitura e escrita em 16 bits desabilitada

T1CKPS1; T1CKPS0: Bits de seleção de fonte de prescaler

Tabela 4.7 – Bits de seleção de fonte de prescaler

T1CKPS1	T1CKPS0	Prescaler
1	1	1:8
1	0	1:4
0	1	1:2
0	0	1:1

T1OSCEN: Habilitação do oscilador externo de baixa frequênciia nos pinos T1OSO e T1OSI

0 – Oscilador desligado

1 – Oscilador Ligado

T1SYNC: Controle do sincronismo interno. Caso TMR1CS = 0, esse bit é descartado

0 – Sincronismo ligado

1 - Sincronismo desligado

TMR1CS: Bit de seleção de clock;

0 - A base de clock para o TIMER1 é interna (Fosc/4);

1 - Clock externo no pino RC0/T1CK1;

TMR1ON: Habilita TIMER1;

0 - TIMER1 desligado

1 - TIMER1 ligado

Obs: Bit6 não implementado, lido como "0"

Abaixo o diagrama simplificado do Timer1:

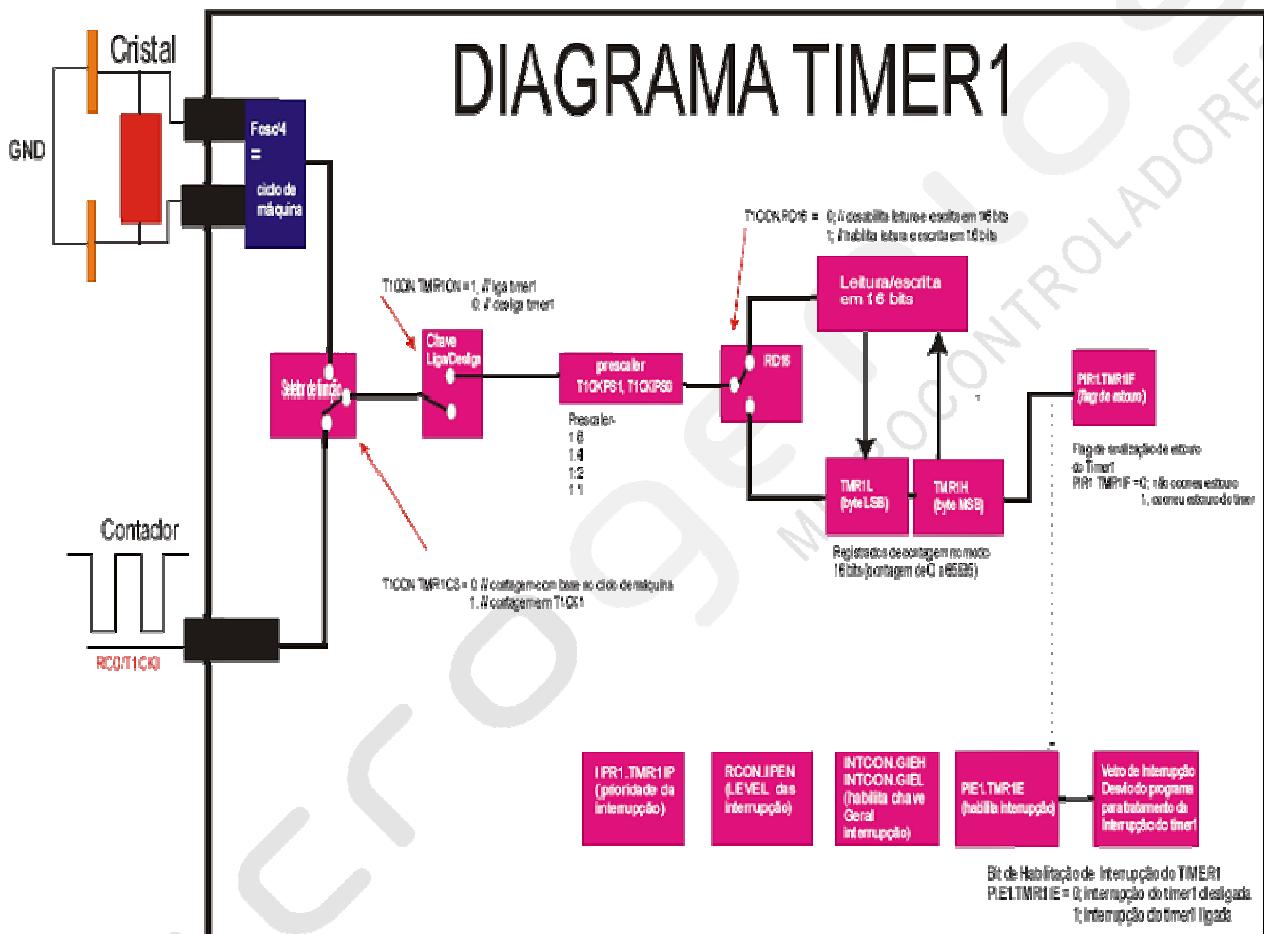


Figura 4.4 – Diagrama interno do TIMER1

O TIMER1 opera de maneira idêntica a do timer anterior, suas diferenças básicas é que este novo timer pode operar como temporizador para o modo Capture/Compare para o modo CCP. Foi implementado neste novo Timer a possibilidade de ligarmos um outro cristal oscilador, de forma que nos permite ligar cristais de baixa freqüência, tais como 32,768 KHz, para maiores precisão na temporização.

Nota:

Obs: Este segundo cristal deverá ser ligado entre os pinos RC0 e RC1. Em hipótese alguma poderemos dispensar o uso do cristal oscilador principal Fosc.

A figura apresenta o diagrama de funcionamento do TIMER1 operando com o oscilador de baixa freqüência.

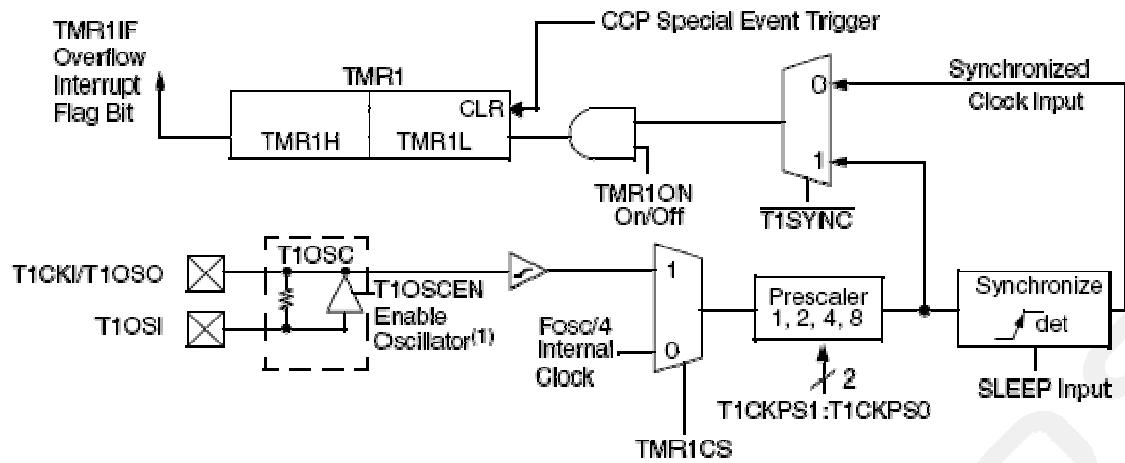


Figura 4.5 – Circuito interno do modo RTC

A figura seguinte mostra para nós claramente uma aplicação em hardware do segundo oscilador conectado ao nosso microcontrolador.

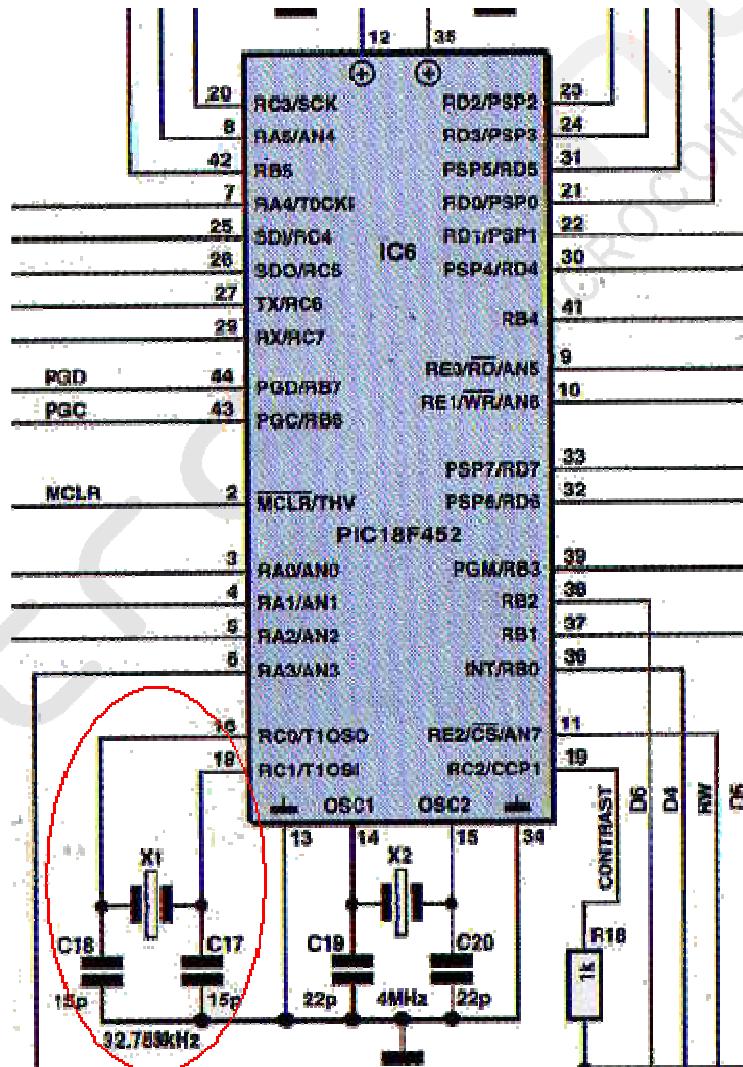


Figura 4.6 – Ligação do segundo oscilador ao microcontrolador PIC

Programando o TIMER1 do PIC.

Para exemplificar a utilização do TIMER1 do PIC vamos analizar um exemplo de programa:
O Hardware:

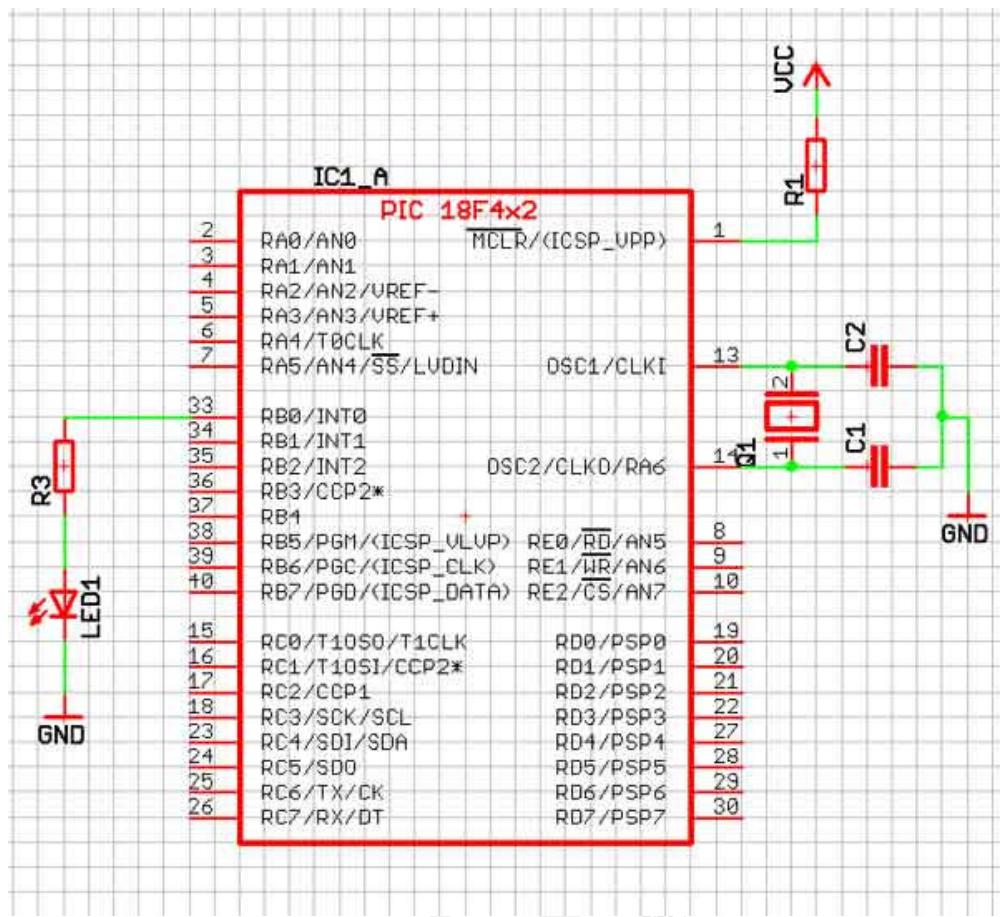


Figura 4.7 – Esquema Eletrônica

Programa:

```
/*
'Microgenios | MicroControladores
'Site: www.Microgenios.com.br
'Autor: Fernando Simplicio de Sousa
' ****
'Programa exemplo: TIMER1.c
'Cristal: 8MHz - modo HS
'Microcontrolador PIC18F45200 Microchip
'Tools: Kit PICgenios PIC18F Módulo Profissional Microgenios
'Configuração: DIP1 - Chave 09 -ON
' ****
'Objetivo:
'Este programa tem por objetivo ativar a temporização máxima do TIMER1
'utilizando cristal de 8Mhz
' ****

void main(){

    /******calculo temporização timer1 *****
    'calculo de temporização máxima do timer1
    'tempo = ciclo_maquina * prescaler * (modo 16 - valor inicial)
    'logo
    'Tempo = 0.5us * 8 * (65536 - 0)
    'Tempo = 262.144 us ou 262ms */

    TRISD = 0;      //define portb como saída
    PORTD = 0;      //apaga todos os leds conectados ao portd

    T1CON = 0b10110001; //liga TIMER1, prescaler 1:8, modo 16bits.
    TMR1L = 0;        //carrega valor de contagem baixa do TIMER1
    TMR1H = 0;        //carrega valor de contagem alta do TIMER1
    PIR1.TMR1IF = 0;  //apaga flag de estouro do TIMER1

    while (1){

        if (PIR1.TMR1IF == 1) {
            PORTD.RD0 = ~PORTD.RD0; //inverte o valor do led1
            TMR1L = 0;             //carrega valor de contagem baixa do TIMER1
            TMR1H = 0;             //carrega valor de contagem alta do TIMER1
            PIR1.TMR1IF = 0;       //apaga flag de estouro do TIMER1
        }
    }
}
```

O programa acima configura o TIMER1 para operar em 16bits com prescaler 1:8, clock interno. A cada estouro do TIMER1, o led conectado o portd.rd0 é invertido.

Análise detalhado do programa:

Sabendo que nosso microcontrolador esta trabalhando com um cristal externo de 8MHz, vamos calcular o tempo de estouro do TIMER1:

Configuração do TIMER1:

Modo de Operação: 16 bits (timer1 somente opera no modo 16 bits)

Fonte de oscilação: Ciclo interno.

Prescaler: 1:8

Calculo de Estouro do TIMER1:

Fórmula:

Tempo: ciclo de máquina * prescaler * (modo 16bits - valor de carga)

Portanto

Tempo: $0.5 \times 8 \times (65536 - 0) = 262.144\text{us}$ ou 262ms

TIMER2 :

O TIMER2 é um timer de 8 bits com recarga automática. Esse TIMER tem um registrador de configuração, um de contagem e outro de comparação. Ele possui um registrador de contagem de 8 bits (0 a 255) chamado TMR2. Diferentemente dos outros timers, o TIMER2 possui um prescale e um postscaler. Os registradores especiais responsável pela configuração de temporização do TIMER2 são:

T2CON (TIMER2 CONTROL REGISTER): Configura o setup do TIMER2;

TMR2: Registrador de contagem do TIMER2 (8 bits);

PR2: Registrador de comparação do TIMER2

Dizemos que o TIMER2 é um timer com recarga automática pois quando o valor carregado em PR2 é igual ao de contagem TMR2, o valor de TMR2 é zerado e inicia-se uma nova contagem, ou melhor dizendo, temporização.

Os registradores relacionados com o TIMER2 são:

Tabela 4.8 – Registradores responsáveis pela configuração do TIMER2

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on All Other RESETS
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000	0000 0000
TMR2	Timer2 Module Register								0000 0000	0000 0000
T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000
PR2	Timer2 Period Register								1111 1111	1111 1111

Através do registrador T2CON, podemos configurar o modo de operação do TIMER2, tais como: valor do prescale e postcale e ligar ou desligar o timer2.

Veja o diagrama esquemático de construção do TIMER2: (procure sempre comparar um TIMER com o outro, pois dessa forma você saberá qual é mais adequado para sua aplicação).

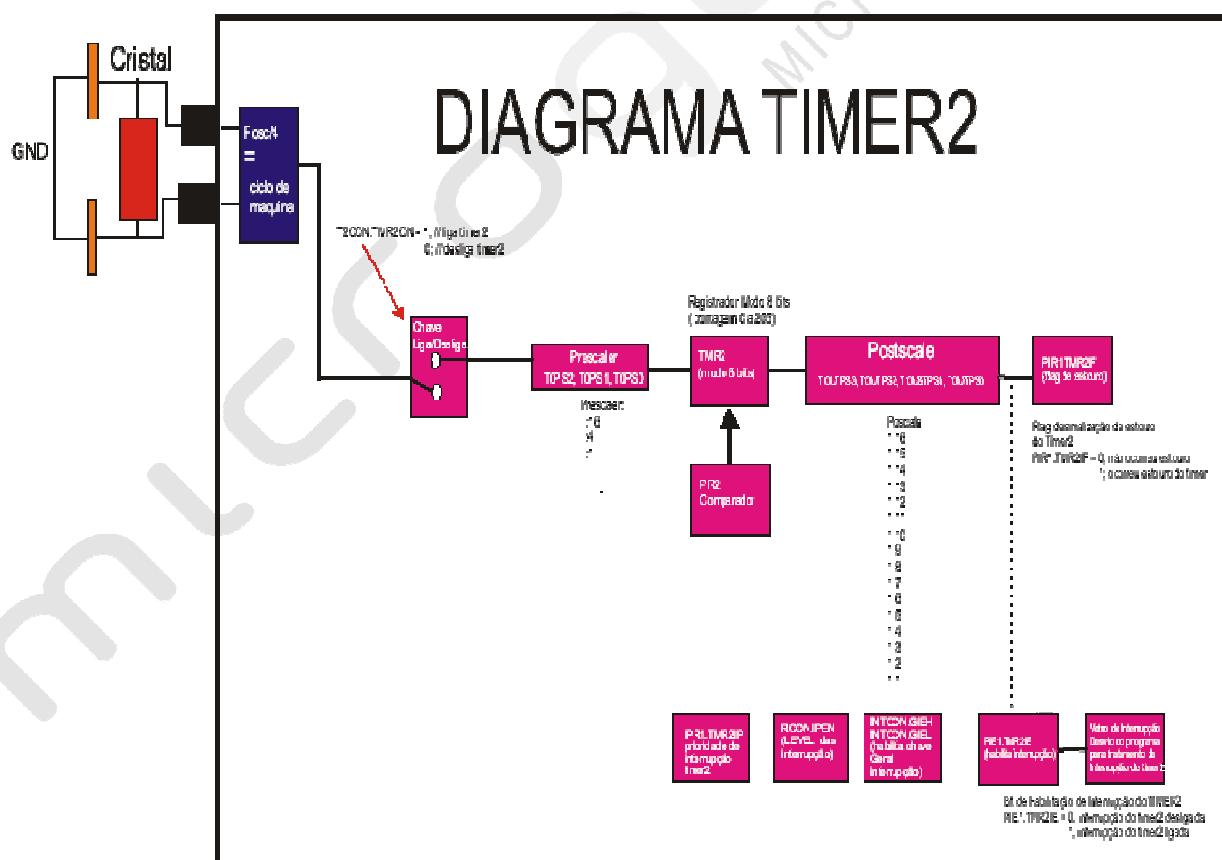


Figura 4.8 – Diagrama de construção interna do TIMER2

Registradores de configuração do TIMER2:

T2CON: (TIMER2 CONTROL REGISTER):

Tabela 4.9 – T2CON

---	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

TOUTPS3 TOUTPS2 TOUTPS1 TOUTPS0: Bits de ajuste do postcale:

Tabela 4.10 – Bits de ajuste do postcale

TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	Postscale
0	0	0	0	1:1
0	0	0	1	1:2
0	0	1	0	1:3
0	0	1	1	1:4
0	1	0	0	1:5
0	1	0	1	1:6
0	1	1	0	1:7
0	1	1	1	1:8
1	0	0	0	1:9
1	0	0	1	1:10
1	0	1	0	1:11
1	0	1	1	1:12
1	1	0	0	1:13
1	1	0	1	1:14
1	1	1	0	1:15
1	1	1	1	1:16

O funcionamento do postcale é muito semelhante ao prescaler, sua diferença básica está na contagem. Em vez de contar pulsos de ciclos de máquina, o postscale conta n comparações do TMR2 com PR2. Após n comparações, o flag de estouro do TIMER2 é sinalizado com nível lógico 1.

TMR2ON: Habilitação do TIMER2;

0 - TIMER2 desligado

1 - TIMER 2 ligado

T2CKPS1 T2CKPS0: Bits responsáveis pelo ajuste de prescaler:

Tabela 4.11 – Bits responsáveis pelo ajuste de prescaler(TIMER 2)

T2CKPS1	T2CKPS0	Prescaler
0	0	1:1
0	1	1:4
1	x	1:16

X pode ser 1 ou 0

Tempo máximo do TIMER2 com cristal de 8Mhz:

Tempo = ciclo de máquina * prescaler * postscale * (PR2 + 1)

Tempo = 0.5us * 16 * 16 * (255 +1)

Tempo = 32.768us ou 32ms

A partir disso você sabe que não tem como programar tempos maiores que 32 milisegundos com seu TIMER2, neste caso é necessário repetir n ciclos de estouros, utilizando variáveis de incrementos.

OBS: Quando utilizamos a função PWM do módulo CCP os registradores de contagem do TIMER2 são comprometidos, por esse motivo não podemos utilizar a função de temporizador do TIMER2 e PWM ao mesmo tempo nos nossos projetos.

TIMER3

O TIMER3 :

O TIMER3 é um temporizador e contador de 16 bits. Possui internamente dois registradores de 8 bits, TMR3L e TMR3H, que juntos formam um registrador de 16 bits (contagem de 0 a 65535). Diferente do TIMER1, o TIMER3 somente pode operar no modo 16 bits.

Os registradores relacionados com o TIMER3 são:

Tabela 4.12 – Registradores relacionados com o TIMER3

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on All Other RESETS
INTCON	GIE/ GIEH	PEIE/ GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
PIR2	—	—	—	EEIF	BCLIF	LVDIF	TMR3IF	CCP2IF	---0 0000	---0 0000
PIE2	—	—	—	EEIE	BCLIE	LVDIE	TMR3IE	CCP2IE	---0 0000	---0 0000
IPR2	—	—	—	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP	---1 1111	---1 1111
TMR3L	Holding Register for the Least Significant Byte of the 16-bit TMR3 Register								xxxx xxxx	uuuu uuuu
TMR3H	Holding Register for the Most Significant Byte of the 16-bit TMR3 Register								xxxx xxxx	uuuu uuuu
T1CON	RD16	—	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	0-00 0000	u-uu uuuu
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	T3SYNC	TMR3CS	TMR3ON	0000 0000	uuuu uuuu

Através do registrador T3CON, podemos configurar o modo de operação do TIMER3, tais como: valor do prescale, ligar ou desligar o TIMER3, seleção de clock interno ou externo (contador) e fonte do timer para o módulo CCP do PIC.

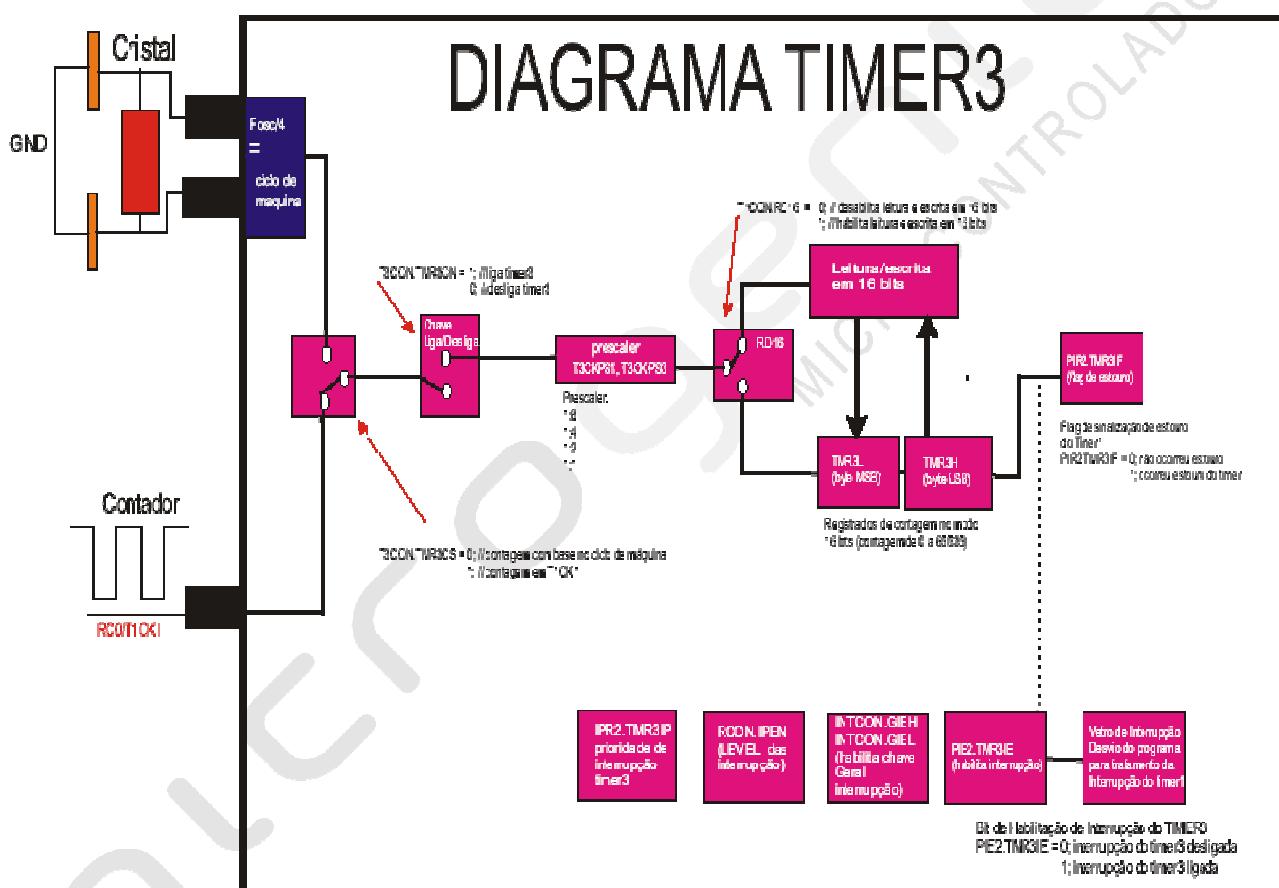


Figura 4.9 – Estrutura interna do TIMER3

Registradores de configuração do TIMER3:

O registrador de controle do TIMER3;

T3CON: (TIMER3 CONTROL REGISTER);

Tabela 4.13 – T3CON

RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	T3SYNC	TMR3CS	TMR3ON
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

RD16: bit de leitura e escrita em 16bits

0 – leitura e escrita em 16 bits desabilitada (utilizado no modo CCP)

1 – leitura e escrita em 16 bits habilitada (utilizado no modo CCP)

T3CCP2 T3CCP1: Habilita a fonte de timer para o módulo CCP;

Tabela 4.14 – Habilita a fonte de timer para o módulo CCP

T3CCP2	T3CCP1	Prescaler
0	0	TIMER1 fonte de timer para os módulos CCP
0	1	TIMER1 fonte de timer para o módulo CCP1
1	0	TIMER3 fonte de timer para o módulo CCP2
1	1	TIMER3 fonte de timer para os módulos CCP

T3CKPS1 T3CKPS0: bits de ajuste do prescaler do TIMER3;

Tabela 4.15 – bits de ajuste do prescaler do TIMER3

T3CKPS1	T3CKPS0	Prescaler
0	0	1:1
0	1	1:2
1	0	1:4
1	1	1:8

/T3SYNC: Controle de sincronismo interno. Caso TMR3CS = 0, este bit é descartado;

obs: este bit somente é utilizado para trabalhar com o periférico RTC (segundo oscilador ligado aos pinos T1OSO e T1OSI).

0 - Sincronismo ligado

1 - Sincronismo desligado

TMR3CS: Seleção de Clock

0 - Clock Interno -TIMER3 programado como Temporizador (clock baseado no ciclo de máquina)

1 - clock externo no pino T1CK1 - Habilitar modo contador do TIMER1

TMR3ON: Habilitação do TIMER3;

0 - TIMER3 desligado

1 - TIMER3 ligado

Para exemplificar a utilização do TIMER3, vamos efetuar algumas pequenas modificações no programa exemplo anterior, cuja finalidade é piscar um Led conectado ao pino RB0 do PIC em intervalos de 1 segundo. (ligado e desligado).

Primeiramente devemos programar o TIMER3:

Para este exemplo, vamos programar o TIMER3 no modo 16 bits (contagem de 0 a 65535 - lembre-se que TIMER3 somente opera em 16 bits), prescaler de 1:8, clock interno, cristal de 8Mhz e modo HSPLL desativado (multiplicador do ciclo de máquina), temos então a seguinte fórmula de temporização:

Tempo de estouro do TIMER3 = ciclo de máquina * prescaler * (modo 16bits - valor de carga inicial)

Antes de começarmos a programar nossa temporização de 1 segundo, vamos descobrir a temporização máxima suportada pelo TIMER3. (com cristal de 8MHz)

Tempo TIMER3 = 0.5us * 8 * (65536 - 0)

Logo

Tempo TIMER3 = 262.144us ou 262ms (valor igual ao do TIMER1)

Nota:

Faça a comparação entre o TIMER1 com o TIMER3 e repare que suas estruturas são muito semelhantes, inclusive seus modos contadores compartilham o mesmo pino RC0 (T1CKI) para leitura dos pulsos externos.

Como desejamos que o estado do Led pisque a cada 1 segundo, somos obrigados a criar uma rotina em C para multiplicar n vezes o estouro do TIMER3, pois nossa temporização máxima é de 262ms (com cristal de 8MHz).

Vamos prograr o TIMER3 para estourar a cada 250ms, e através de alguns comandos em C vamos multiplicar esta temporização em 4x para que tenhamos 1 segundo ($4 * 250\text{ms} = 1\text{ segundo}$).

Unidade 5 – Programando as Interrupções do PIC

As Interrupções do PIC18F4520

Introdução

Diferente da família PIC16F da Microchip, essa nova família PIC18F implementa prioridades nas interrupções.

Na memória FLASH do PIC18F4520 possui dois endereços de interrupção: um chamado de vetor de alta prioridade, cujo endereço é 0008h, e outro chamado de vetor de baixa prioridade, cujo endereço é 0018h.

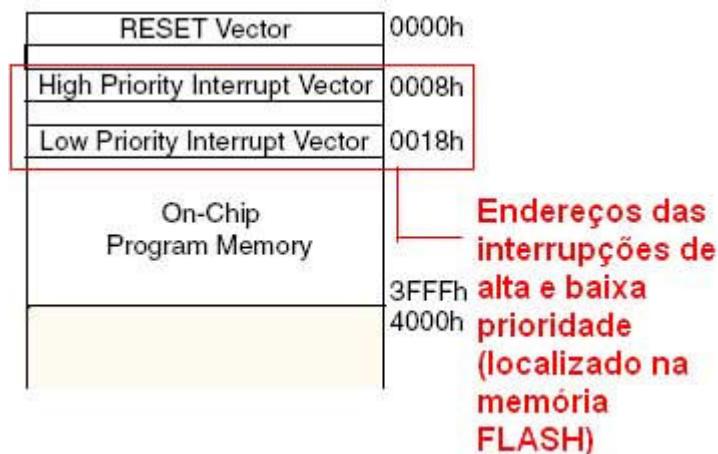


Figura 5.1 – VETOR DE ALTA PRIORIDADE E BAIXA PRIORIDADE DE INTERRUPÇÃO SÉRIE PIC18F

No PIC18F4520 existem diversos tipos de interrupções, segue algumas:

- Interrupção do TIMER0
- Interrupção do TIMER1
- Interrupção do TIMER2
- Interrupção do TIMER3
- Interrupção por mudança de estado (pinos RB4, RB5, RB6, RB7)
- Interrupção externa INT0
- Interrupção externa INT1
- Interrupção externa INT2
- Interrupção na conversão AD
- Interrupção na recepção serial
- Interrupção na transmissão serial
- Interrupção do módulo CCP
- Interrupção de escrita na EEPROM/FLASH
- Interrupção por queda de tensão.

As prioridades de interrupção funcionam da seguinte maneira: Caso venhamos programar interrupções de alta e baixa prioridade em nosso programa, e por um determinado momento seja solicitado ao mesmo tempo as duas interrupções, a interrupção de alta prioridade será atendida primeiro. Ao término da execução da rotina de interrupção de alta prioridade, automaticamente a rotina de baixa prioridade começa o seu tratamento. Quando uma rotina de baixa prioridade estiver sendo executada e ocorrer uma interrupção de alta prioridade, a interrupção de alta prioridade será atendida primeiramente, e ao seu término, a interrupção de baixa prioridade continuará a sua execução do ponto seguinte onde parou.

Se formos traçar um gráfico sobre as prioridades de interrupção e seus acionamentos, teremos:



Figura 5.2 – Diagrama de Interrupção x Tempo

Através do gráfico ficou fácil perceber que uma rotina de interrupção de alta prioridade pode interromper a qualquer momento qualquer rotina de menor prioridade de interrupção ou execução.

Como funciona as interrupções?

Podemos resumidamente comparar as interrupções do nosso microcontrolador com uma instalação elétrica de uma residência qualquer.

Repare no diagrama:

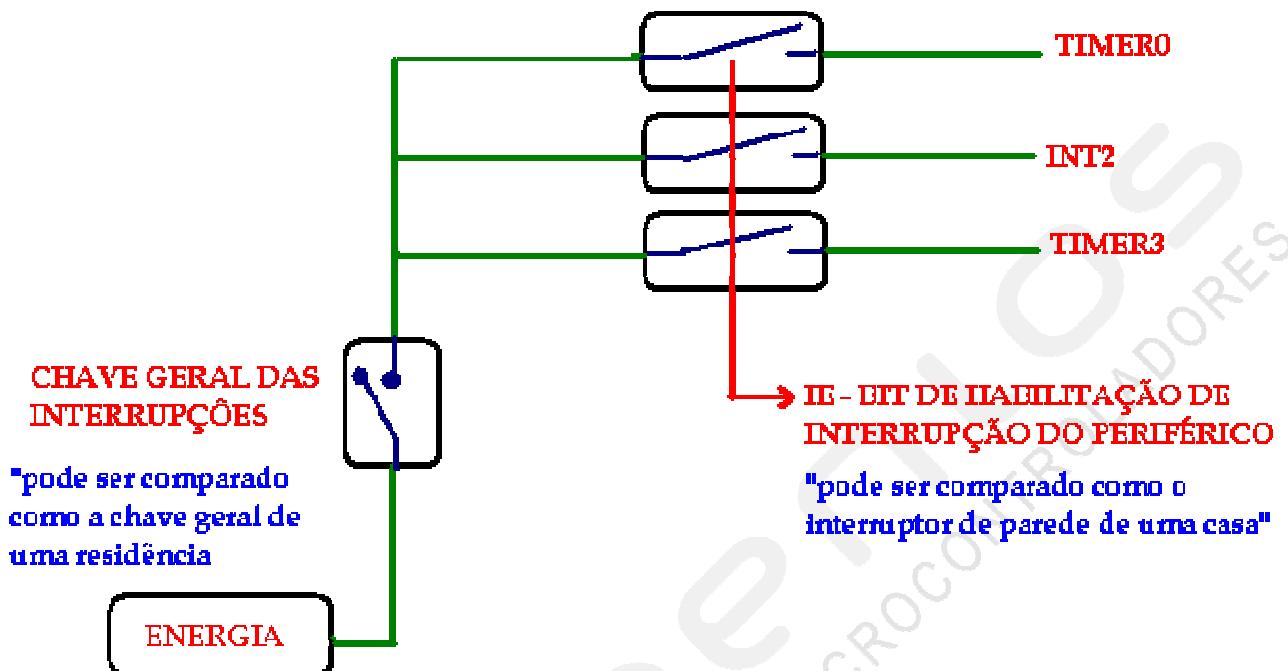


Figura 5.3 – Exemplo para compreensão das Interrupções do PIC

Repare que temos em nosso exemplo uma chave geral, esta chave geral habilita ou desabilita todas as interrupções interna do nosso microcontrolador, mas o fato dela estar ativado não representa que as interrupções dos periféricos estão habilitadas. Caso necessitarmos habilitar a interrupção de algum periférico interno do PIC, como exemplo, o TIMER0, faz necessário ligar a chave geral e o “interruptor de parede”, ou seja, o IE do TIMER0.

Iremos apartir de agora apresentar os comandos responsáveis pela configuração geral das interrupções.

```
//Configuração Geral das interrupções
INTCON.GIEH = 1; /*habilita a chave geral das interrupções e habilita o vetor de alta prioridade de interrupção*/
INTCON.GIEL = 1; //habilita o vetor de baixa prioridade de interrupção
RCON.IPEN = 1; //define característica da família PIC18F com 2 level de prioridade de interrupção
```

Diferente da família PIC16F, a família PIC18F possui dois vetores de interrupção. Pensando nos projetos da família PIC16F, a Microchip criou uma estrutura interna nos PIC18F de forma a permitir a micração de software com facilidade.

Caso venhamos a colocar nível lógico zero em rcon.ipen herdamos características da família PIC16F que possui apenas 1 vetor de interrupção (endereço 0004h da Flash).

Caso necessitarmos trabalhar com dois níveis de prioridade de interrupção em nosso programa somos obrigados a habilitar o bit rcon.ipen = 1 e intcon.giel = 1.

Bom, agora que já programamos as configurações gerais de interrupção do nosso microcontrolador, podemos habilitar a interrupção particular de um determinado periférico. Para exemplificar vamos programar a interrupção do TIMER0:

Primeiro de tudo devemos conhecer a nomenclatura utilizado pela Microchip.

Nomenclatura:

Para habilitarmos a interrupção de qualquer periférico precisamos configurar 3 bits: (além das configurações gerais que já conhecemos)

Bit com final: IF, IP e IE:

IF – Flag de interrupção

IP – Prioridade de interrupção

IE – Habilitação de Interrupção

Logo a interrupção do TIMER0 será:

_____ .TMR0IF = 0 'configuração inicial, flag de estouro do TIMER0 apagado (você não quer programar um timer com seu alarme já tocando, não é?)

_____ .TMR0IP = 1 'define a prioridade de interrupção do TIMER0, neste caso, alta prioridade.

_____ .TMR0IE = 0 'habilita a chave individual de interrupção do TIMER0 – "lembre-se do interruptor de parede"

O que foi nos apresentado é o nome do BIT, falta ainda colocar o nome do byte, mas antes disso vamos habilitar mais uma interrupção, neste caso INT3. (interrupção externa 3);

_____ .INT3IF = 0 'configuração inicial, flag de sinalização de INT3 apagado

_____ .INT3IP = 0 'define a prioridade de interrupção de INT3, neste caso, BAIXA prioridade.

_____ .INT3IE = 0 'habilita a chave individual de interrupção do INT3

Nota parte 1/2:

Para habilitar qualquer interrupção precisamos garantir as configurações gerais das interrupções, que são:

```
//Configuração Geral das interrupções
```

```
INTCON.GIE = 1
```

```
INTCON.GIEL = 1
```

```
RCON.IPEN = 1
```

E habilitar os 3 bits individuais de interrupção de qualquer periférico, por exemplo: INT2

Então teremos:

_____ .INT2IF = 0 'configuração inicial, flag de sinalização de INT2 apagado

_____ .INT2IP = 1 'define alta prioridade de interrupção de INT2

_____ .INT2IE = 0 'habilita a chave individual de interrupção do INT2

Para descobrirmos o nome do byte apresentado com um "_____ ", basta consultar os registradores seguintes

Registradores que são usados para controlar as interrupções;

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1, PIR2
- PIE1 , PIE2
- IPR1, IPR2

Estudo dos Registradores de Configuração das Interrupções do PIC

Uma das função do registrador RCON na interrupção é definir as prioridades altas e baixas das interrupções (0x0008 e 0x0018), vetor único (modo normal, compatível com a família PIC16 em que somente há um vetor de interrupção) do PIC ou modo com dois níveis de prioridade, presente na família PIC18F.

O bit do registrador RCON responsável pela seleção de habilitação do modo de interrupção chama-se: IPEN.

RCON (CONTROL REGISTER)

Tabela 5.1 - RCON

IPEN	---	----	/RI	/T0	/PD	/POR	/BOR
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

IPEN: Habilita as prioridades de interrupções do PIC

1 - Habilitação do modo de alta e baixa prioridades do PIC (presente na família PIC18)

0 - Desabilita o Modo de alta e baixa prioridade (modo compatível com família PIC16)

Demais bits consultar datasheet do microcontrolador PIC18F45200.

Os registradores INTCON, INTCON2 e INTCON3 são responsáveis pela habilitação das interrupções de baixa e alta prioridade, além de definir se algumas interrupções externas serão utilizadas.

INTCON (Interrupt Control)

Tabela 5.2 – INTCON

Bit	Evento
INTCON.GIE (INTCON <7>) = 1	Habilitação geral das interrupções
INTCON.PEIE (INTCON <6>) = 1	Habilitação Interrupção dos periféricos
INTCON.TMR0IE (INTCON <5>) = 1	Habilitação Interrupção do TIMER0
INTCON.INT0IE (INTCON <4>) = 1	Habilitação Interrupção INT0 externa
INTCON.RBIE (INTCON <3>) = 1	Habilitação Interrupção por mudança de estado
INTCON.TMR0IF (INTCON <2>) = 1	Bit de sinalização de estouro do TIMER0
INTCON.INT0IF (INTCON <1>) = 1	Bit de sinalização de interrupção externa INT0
INTCON.RBIF (INTCON <0>) = 1	Bit de sinalização de interrupção por mudança de estado

No registrador INTCON, os registradores marcados em amarelo na tabela acima são bits de sinalização de interrupção

INTCON2 (Interrupt Control 2)

Tabela 5.3 – INTCON2

Bit	Evento
INTCON2.RBPU (INTCON2 <7>) = 0	Habilita resistores de pull-up do portb
INTCON2.RBPU (INTCON2 <7>) = 1	desabilita resistores de pull-up do portb
INTCON2.INTEDG0 (INTCON2 <6>) = 1	Configura interrupção externa INT0 por borda de Subida
INTCON2.INTEDG0 (INTCON2 <6>) = 0	Configura interrupção externa INT0 por borda de descida
INTCON2.INTEDG1 (INTCON2 <5>) = 1	Configura interrupção externa INT1 por borda de Subida
INTCON2.INTEDG1 (INTCON2 <5>) = 0	Configura interrupção externa INT1 por borda de descida
INTCON2.INTEDG2 (INTCON2 <4>) = 1	Configura interrupção externa INT2 por borda de Subida
INTCON2.INTEDG2 (INTCON2 <4>) = 0	Configura interrupção externa INT2 por borda de descida
INTCON2.TMR0IP (INTCON2 <2>) = 1	Alta prioridade da Interrupção do TIMER0
INTCON2.TMR0IP (INTCON2 <2>) = 0	Baixa prioridade da Interrupção do TIMER0
INTCON2.RBIP (INTCON2 <0>) = 1	Alta prioridade da Interrupção por mudança de estado
INTCON2.RBIP (INTCON2 <0>) = 0	Baixa prioridade da Interrupção por mudança de estado

INTCON3 (Interrupt Control 3)

Tabela 5.4 – INTCON3

Bit	Evento
INTCON3.INT2IP (INTCON3 <7>) = 1	Alta prioridade na Interrupção externa INT2
INTCON3.INT2IP (INTCON3 <7>) = 0	Baixa prioridade na Interrupção externa INT2
INTCON3.INT1IP (INTCON3 <6>) = 1	Alta prioridade na Interrupção externa INT1
INTCON3.INT1IP (INTCON3 <6>) = 0	Baixa prioridade na Interrupção externa INT1
INTCON3.INT2IE (INTCON3 <4>) = 1	Habilita interrupção externa INT2
INTCON3.INT2IE (INTCON3 <4>) = 0	desabilita interrupção externa INT2
INTCON3.INT1IE (INTCON3 <3>) = 1	Habilita interrupção externa INT1
INTCON3.INT1IE (INTCON3 <3>) = 0	desabilita interrupção externa INT1
INTCON3.INT2IF (INTCON3 <1>) = 1	Bit de sinalização de Interrupção INT2
INTCON3.INT2IF (INTCON3 <1>) = 0	Bit de sinalização de Interrupção INT2
INTCON3.INT1IF (INTCON3 <0>) = 1	Bit de sinalização de Interrupção INT1
INTCON3.INT1IE (INTCON3 <0>) = 0	Bit de sinalização de Interrupção INT1

Dica:

Sempre quando você estiver estudando e se deparar Bits finalizados com IF, IP e IE, já interprete eles da seguinte maneira:

IF = bit de sinalização de algum periférico

IE = bit de habilitação de interrupção de algum periférico

IP = bit de habilitação da prioridade de algum periférico.

Não se esqueça disso!!!

Os registradores P1R1 e P1R2 são responsáveis em armazenar os estados das interrupções. Quando uma interrupção foi programada, é através dos bits desse registrador que poderemos monitorar o estado da interrupção (acionada ou não acionada).

PIR1 (sinaliza Interrupção dos Periféricos 1)

Tabela 5.5 – PIR1

Bit	Evento
PIR1.PSPIF (PIR1 < 7 >) = 1	bit de status da Interrupção da porta Parallel Port Slave
PIR1.ADIF (PIR1 < 6 >) = 1	bit de status da Interrupção do conversor A/D
PIR1.RCIF (PIR1 < 5 >) = 1	bit de status da Dado recebido pela USART
PIR1.TXIF (PIR1 < 4 >) = 1	bit de status da Dado transmitido pela USART
PIR1.SSPIF (PIR1 < 3 >) = 1	bit de status da interrupção do módulo SSP
PIR1.CCP1IF (PIR1 < 2 >) = 1	bit de status da Interrupção do módulo CCP
PIR1.TMR2IF (PIR1 < 1 >) = 1	bit de status do estouro do TIMER2
PIR1.TMR1IF (PIR1 < 0 >) = 1	bit de status do estouro do TIMER1

PIR2 (sinaliza Interrupção dos Periféricos 2)

Tabela 5.6 – PIR2

Bit	Evento
PIR2.EEIF (PIR2 < 4 >) = 1	Bit de sinalização da Interrupção de escrita na EEPROM/Flash
PIR2.BCLIF (PIR2 < 3 >) = 1	Bit de sinalização da Interrupção por colisão de dados
PIR2.LCDIF (PIR2 < 2 >) = 1	Bit de sinalização da Interrupção do módulo de baixa voltagem LDV
PIR2.TMR3IF (PIR2 < 1 >) = 1	Bit de sinalização da Interrupção de estouro do TIMER3
PIR2.CCP2IF (PIR2 < 0 >) = 1	Bit de sinalização da Interrupção do módulo CCP2

PIE1 (Habilitação das interrupção dos periféricos)

Tabela 5.7 – PIE1

Bit	Evento
PIE1.PSPIE (PIE1 < 7 >) = 1	habilita Interrupção da porta Parallel port Slave
PIE1.ADIE (PIE1 < 6 >) = 1	habilita Interrupção do conversor A/D
PIE1.RCIE (PIE1 < 5 >) = 1	habilita Interrupção da recepção serial
PIE1.TXIE (PIE1 < 4 >) = 1	habilita Interrupção da transmissão serial
PIE1.SSPIE (PIE1 < 3 >) = 1	habilita interrupção do módulo SSP
PIE1.CCP1IE (PIE1 < 2 >) = 1	habilita Interrupção do módulo CCP
PIE1.TMR2IE (PIE1 < 1 >) = 1	habilita interrupção do TIMER2
PIE1.TMR1IE (PIE1 < 0 >) = 1	habilita interrupção do TIMER1

PIE2 (Habilitação das Interrupção dos Periféricos)

Tabela 5.8 – PIE2

Bit	Evento
PIE2.EEIF (PIE2 < 4 >) = 1	Bit de sinalização de Interrupção de escrita na EEPROM/Flash
PIE2.BCLIF (PIE2 < 3 >) = 1	Bit de sinalização de Interrupção por colisão de dados
PIE2.LCDIF (PIE2 < 2 >) = 1	Bit de sinalização de Interrupção do módulo de baixa voltagem LDV
PIE2.TMR3IF (PIE2 < 1 >) = 1	Bit de sinalização de Interrupção do TIMER3
PIE2.CCP2IF (PIE2 < 0 >) = 1	Bit de sinalização de Interrupção do módulo CCP2

IPR1 (registrador de prioridade de interrupções)

Através dos registradores IPR1 e IPR2 definimos as prioridades altas e baixas das interrupções, caso esse sistema seja utilizado.

Tabela 5.9 – IPR1

Bit	Evento
IPR1.PSPIP (BIT < 7 >) = 1	Alta prioridade na interrupção paralela
IPR1.PSPIP (BIT < 7 >) = 0	Baixa prioridade na interrupção paralela
IPR1.ADIP (BIT < 6 >) = 1	Alta prioridade na interrupção conversão AD
IPR1.ADIP (BIT < 6 >) = 0	Baixa prioridade na interrupção conversão AD

IPR1.RCIP (BIT < 5 >) = 1	Alta prioridade na interrupção de recepção serial
IPR1.RCIP (BIT < 5 >) = 0	Baixa prioridade na interrupção de recepção serial
IPR1.TXIP (BIT < 4 >) = 1	Alta prioridade na interrupção de transmissão serial
IPR1.TXIP (BIT < 4 >) = 0	Baixa prioridade na interrupção de transmissão serial
IPR1.SSPIP (BIT < 3 >) = 1	Alta prioridade na interrupção MSSP
IPR1.SSPIP (BIT < 3 >) = 0	Baixa prioridade na interrupção MSSP
IPR1.CCPIP (BIT < 2 >) = 1	Alta prioridade na interrupção do módulo CCP
IPR1.CCPIP (BIT < 2 >) = 0	Baixa prioridade na interrupção do módulo CCP
IPR1.TMR2IP (BIT < 1 >) = 1	Alta prioridade na interrupção do TIMER2
IPR1.TMR2IP (BIT < 1 >) = 0	Baixa prioridade na interrupção do TIMER2
IPR1.TMR1IP (BIT < 0 >) = 1	Alta prioridade na interrupção do TIMER1
IPR1.TMR1IP (BIT < 0 >) = 0	Baixa prioridade na interrupção do TIMER1

IPR2 (registraror de prioridade de interrupções 2)

Através dos registradores IPR1 e IPR2 definimos as prioridades altas e baixas das interrupções, caso esse sistema seja utilizado.

Tabela 5.10 – IPR2

Bit	Evento
IPR2.EEIP (BIT < 4 >) = 1	Alta prioridade na interrupção EEPROM
IPR2.EEIP (BIT < 4 >) = 0	Baixa prioridade na interrupção EPROM
IPR2.BCLIP (BIT < 3 >) = 1	Alta prioridade na interrupção por colisão no barramento
IPR2.BCLIP (BIT < 3 >) = 0	Baixa prioridade na interrupção por colisão no barramento
IPR2.LVDIP (BIT < 2 >) = 1	Alta prioridade na interrupção por queda de tensão
IPR2.LVDIP (BIT < 2 >) = 0	Baixa prioridade na interrupção por queda de tensão
IPR2.TMR3IP (BIT < 1 >) = 1	Alta prioridade na interrupção do TIMER3
IPR2.TMR3IP (BIT < 1 >) = 0	Baixa prioridade na interrupção do TIMER3
IPR1.CCP2IP (BIT < 0 >) = 1	Alta prioridade na interrupção CCP2IP
IPR1.CCP2IP (BIT < 0 >) = 0	Baixa prioridade na interrupção CCP2IP

Nota parte 2/2:

Consultando as tabelas descobrimos que para habilitar a interrupção INT2 é necessário programar:

```
//Configuração Geral das interrupções
INTCON.GIE = 1;
INTCON.GIEL = 1;
RCON.IPEN = 1;
//Configuração de habilitação individual da interrupção externa INT2
INTCON3.INT2IF = 0; //configuração inicial, flag de sinalização de INT2 apagado
INTCON3.INT2IP = 1; //define alta prioridade de interrupção de INT2
INTCON3.INT2IE = 0; //habilita a chave individual de interrupção do INT2
```

Resumo:

1 - bit para habilitar a interrupção desejada, finalizadas com o nome IE.

Ex: INTCON3.INT2IE = 1; // habilita interrupção externa INT2

1 - bit para monitorar o estado da interrupção, também chamado de bit de sinalização ou flag, finalizadas com o nome IF.

Ex: INTCON3.INT2IF = 1; // flag de sinalização da interrupção externa INT2 (setada por hardware do microcontrolador após o acionamento de INT2).

1 - bit de definição de prioridade de interrupção (alta ou baixa prioridade), finalizadas com o nome IP

Ex: INTCON3.INT2IP = 1; // bit de habilitação de prioridade de interrupção externa INT2.

microgenios
MICROCONTROLADORES

Unidade 6 – Programando a Interrupção externa

INT0

A interrupção interna INT0 do PIC18F4520 está multiplexada com o pino RB0. Segue um diagrama estrutural de INT0, acompanhe:

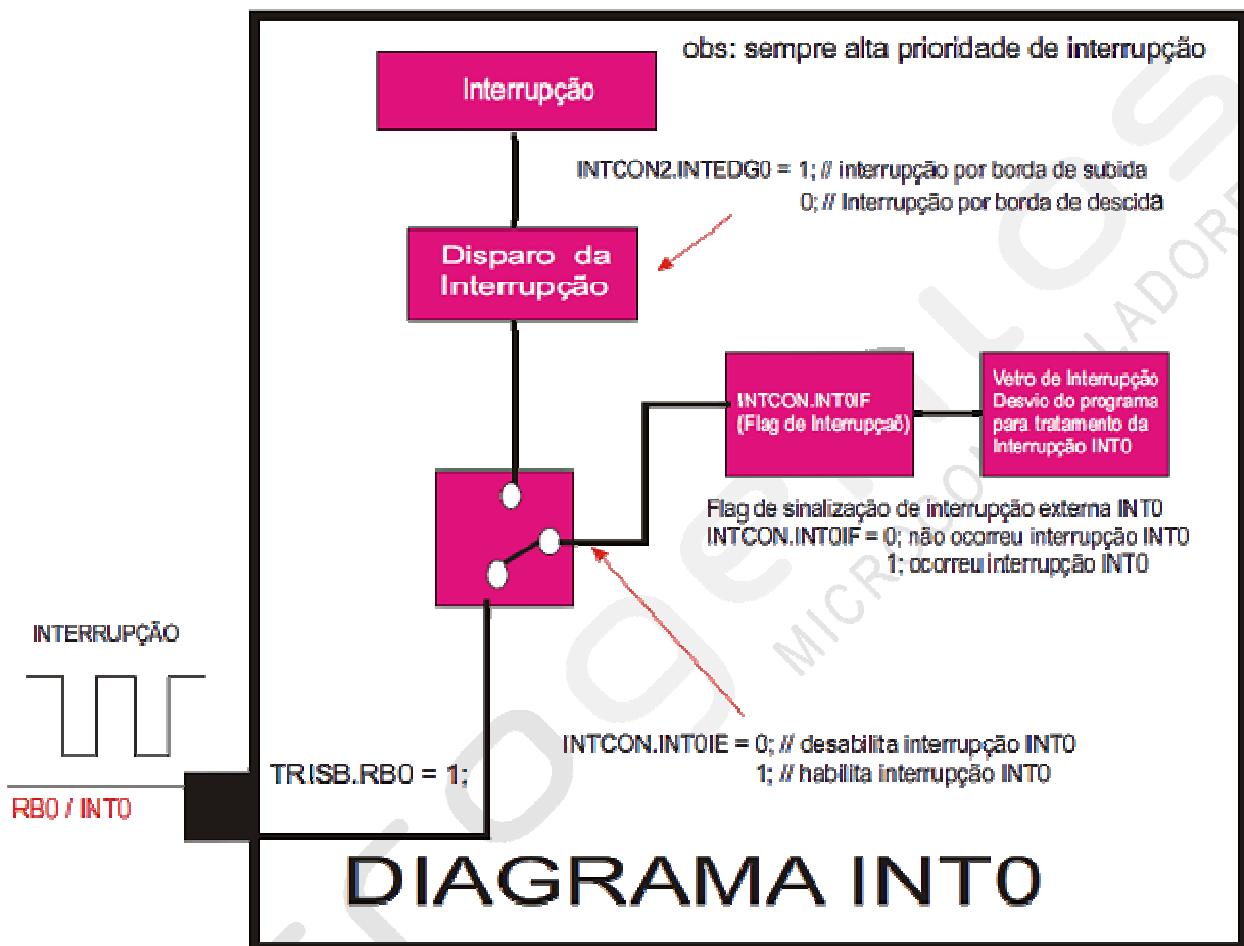


Figura 6.1 – Diagrama estrutural da interrupção externa INT0

Sabemos que todas as interrupções que ocorre dentro do nosso microcontrolador são geradas por causa dos flags de sinalização (bit com final IF na Microchip). A interrupção externa INT0 não é diferente dos demais periféricos, ela internamente possui um bit de sinalização de status que informa ao microcontrolador que ocorreu uma mudança de nível lógico no pino de interrupção.

Podemos programar a interrupção externa por borda de descida ou por borda de subida:

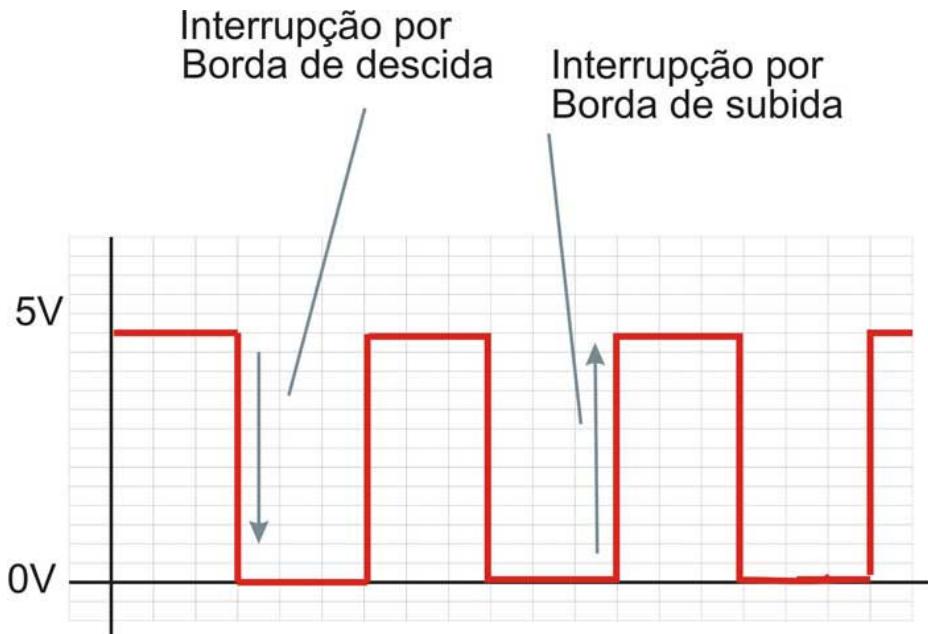


Figura 6.2 – Tipos de disparo das interrupções externa INT0, INT1 e INT2

Importante:

A interrupção externa INT0 somente pode operar como alta prioridade de interrupção, ou seja, não existe registrador INT0IP para INT0, as demais interrupções externa INT1 e INT2 permitem serem programadas em baixa prioridade.

Vamos conhecer os registradores relacionados a INT0:

Registradores responsáveis pelas configurações gerais das interrupções:

INTCON.GIEH: habilita ou desabilita a chave geral das interrupções e as interrupções de alta prioridade

INTCON.GIEL: habilita ou desabilita as interrupções de baixa prioridades

RCON.IPEN : habilita ou desabilita as interrupções de baixa ou alta prioridades pertencente a família PIC18F

Registradores responsáveis pela habilitação da interrupção externa INT0

INTCON.INT0IE : bit de habilitação da interrupção externa INT0

INTCON.INT0IF : bit de sinalização da interrupção externa INT0

TRISB.RB0 = 1 : somos obrigados a programar pino RB0/INT0 como entrada

Registrador responsável pelo tipo de disparo de INT0:

INTCON2.INTEDG0 : bit de configuração do modo de disparo da interrupção externa INT0

1 = Aciona interrupção externa por borda de subida

0 = aciona interrupção externa por borda de descida

Obs: A interrupção INT0 somente pode ser programada em alta prioridade, ou seja, não existe o registrador TMR2IP. Esta limitação está presente no microcontrolador PIC18F4520 e PIC18F45200.

INT1

Programando a interrupção externa INT1

O funcionamento da interrupção externa INT1 é parecida com a interrupção INT0, sua única diferença esta no fato de INT1 permitir ser acionado por baixa prioridade de interrupção.

Segue um diagrama estrutural de INT1, acompanhe:

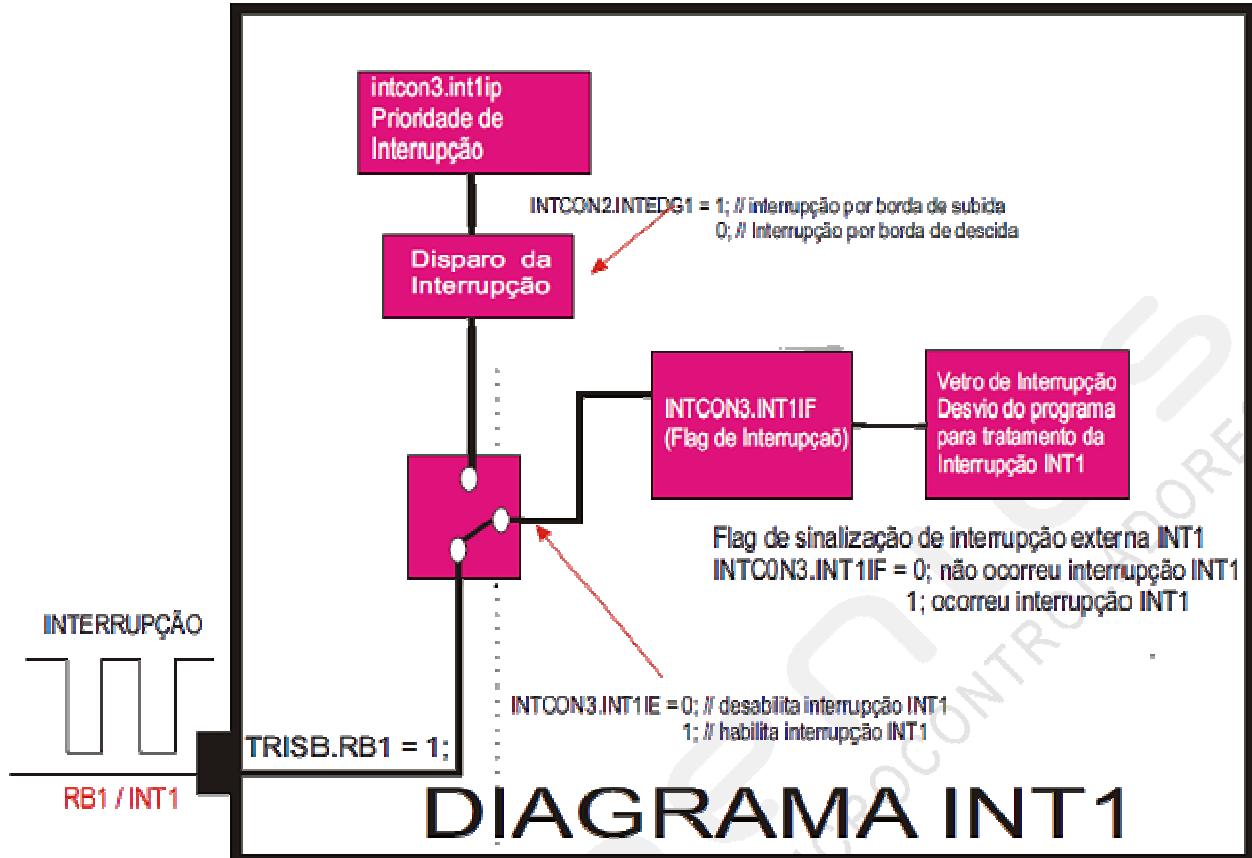


Figura 6.3 – Diagrama estrutural da interrupção externa INT1

Vamos conhecer os registradores relacionados a INT1:

Registradores responsáveis pelas configurações gerais das interrupções:

INTCON.GIEH: habilita ou desabilita a chave geral das interrupções e as interrupções de alta prioridade

INTCON.GIEL: habilita ou desabilita as interrupções de baixa prioridades

RCON.IPEN : habilita ou desabilita as interrupções de baixa ou alta prioridades pertencente a família PIC18F

Registradores responsáveis pela habilitação da interrupção externa INT1

INTCON3.INT1IE : bit de habilitação da interrupção externa INT1

INTCON3.INT1IF : bit de sinalização da interrupção externa INT1

INTCON3.INT1IP : bit de habilitação de alta ou baixa prioridade de interrupção externa INT1

TRISB.RB1 = 1 : somos obrigados a programar pino RB1/INT1 como entrada

INT2

Registrador responsável pelo tipo de disparo de INT2:

INTCON2.INTEDG1 : bit de configuração do modo de disparo da interrupção externa INT0

1 = Aciona interrupção externa por borda de subida

0 = aciona interrupção externa por borda de descida

Diagrama estrutural de INT2:

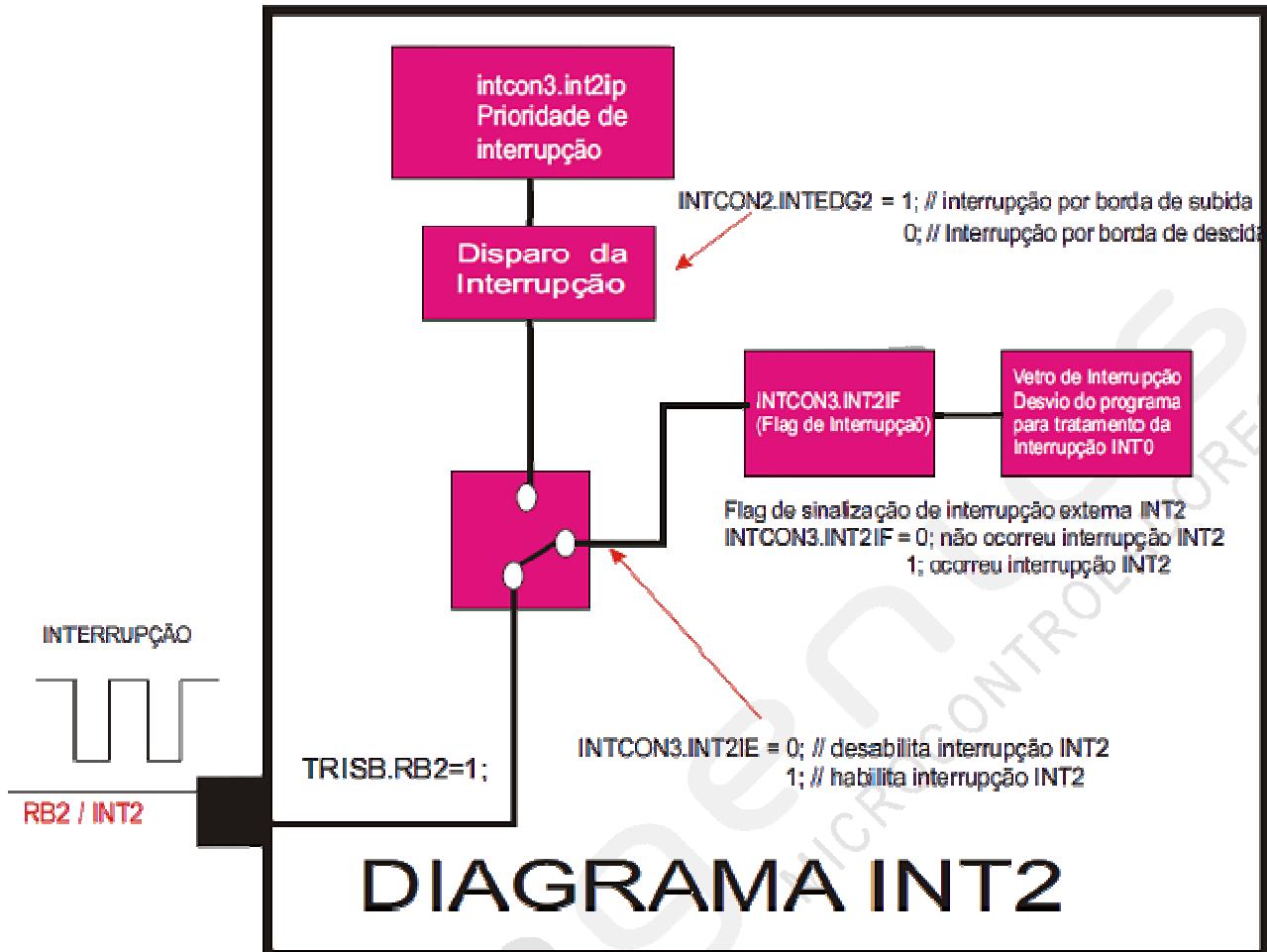


Figura 6.4 – Diagrama estrutural da interrupção externa INT2.

Vamos conhecer os registradores relacionados a INT2:

Registradores responsáveis pelas configurações gerais das interrupções:

INTCON.GIEH: habilita ou desabilita a chave geral das interrupções e as interrupções de alta prioridade

INTCON.GIEL: habilita ou desabilita as interrupções de baixa prioridades

RCON.IPEN : habilita ou desabilita as interrupções de baixa ou alta prioridades pertencente a família PIC18F

Registradores responsáveis pela habilitação da interrupção externa INT2

INTCON3.INT2IE : bit de habilitação da interrupção externa INT2

INTCON3.INT2IF : bit de sinalização da interrupção externa INT2

INTCON3.INT2IP : bit de habilitação de alta ou baixa prioridade de interrupção externa INT2

TRISB.RB2 = 1 : somos obrigados a programar pino RB2/INT2 como entrada

Registrador responsável pelo tipo de disparo de INT2:

INTCON2.INTEDG2 : bit de configuração do modo de disparo da interrupção externa INT2

1 = Aciona interrupção externa por borda de subida

0 = aciona interrupção externa por borda de descida

Interrupção por mudança de estado

Os microcontroladores PIC (alguns modelos) possuem uma interrupção externa chamada Interrupção por mudança de estado (também é chamado de RB ou interrupção de teclado) ligado aos pinos RB4, RB5, RB6 e RB7.

Diferente das interrupções externas INT0, INT1 e INT2 que somente podem ser acionadas por borda de subida ou por borda de descida, as interrupções por mudança de estado acontece nos dois estados.

Analise o gráfico de acionamento da interrupção:

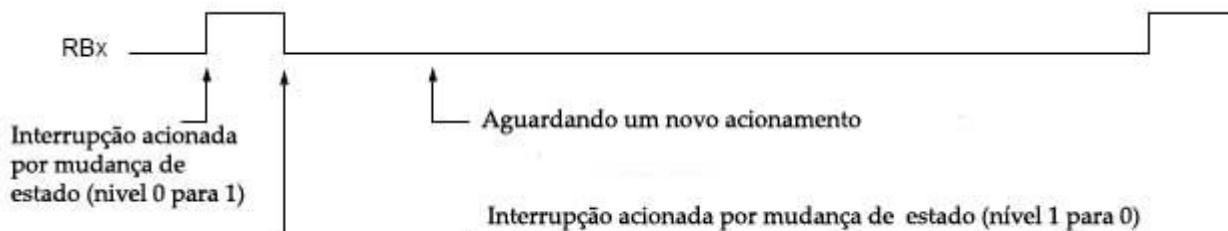


Figura 6.5 - Gráfico de acionamento de interrupção externa RB

Vamos conhecer os registradores relacionados a interrupção RB:

Registradores responsáveis pelas configurações gerais das interrupções:

INTCON.GIEH: habilita ou desabilita a chave geral das interrupções e as interrupções de alta prioridade

INTCON.GIEL: habilita ou desabilita as interrupções de baixa prioridades

RCON.IPEN : habilita ou desabilita as interrupções de baixa ou alta prioridades pertencente a família PIC18F

Registradores responsáveis pela habilitação da interrupção externa RB

INTCON.RBIF : bit responsável pelo flag de status da interrupção por mudança de estado

INTCON.RBIE : bit de habilitação da interrupção por mudança de estado

INTCON2.RBIP : bit de configuração de alta ou baixa prioridade de interrupção

TRISB = \$1111xxxx : somos obrigados a programar os pinos RB4, RB5, RB6 e RB7 como entrada

INTCON.RBPU = 1 : desabilita resistores de pull up interno do PORTB (obrigatório)

Unidade 7 – Exemplos de Periféricos

Controle de display LCD

Iremos no decorrer das unidades programar o PIC para controlar e escrever mensagens publicitárias nos display LCD 16X2 alfanumérico:

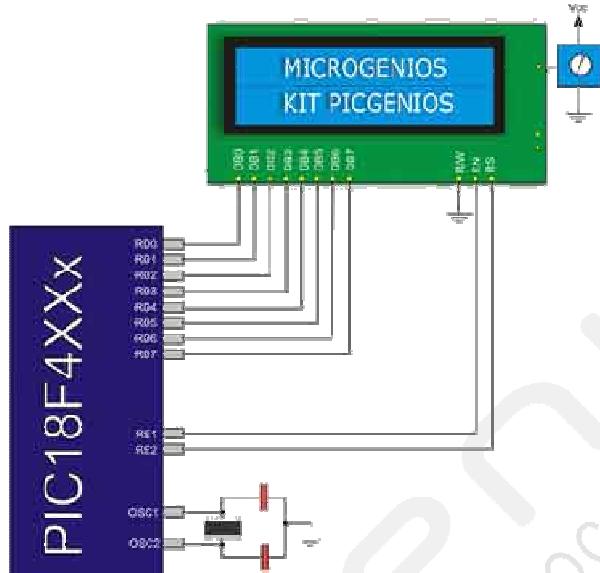


Figura 7.1 – Aplicações com display LCD

Os displays LCD são amplamente utilizados em diversos equipamentos e aparelhos. No decorrer dos estudos iremos explorar as funções do MikroC PRO em linguagem C para controle de displays LCD. Estudaremos passo a passo como escrever mensagens de textos nos modos 4 e 8 bits.

Projetos com displays LCD:



Figura 7.2 – Projetos com displays LCD

Varredura de displays de 7 segmentos

Os displays de 7 segmentos são largamente utilizados em equipamentos como: balança, painéis de máquinas industriais, equipamentos médicos, eletrodomésticos entre outros. Podemos controlar os displays de 7 segmentos através de conversores BCD, como por exemplo o 74HC247 (decodificador BCD) ou desenvolver códigos BCD pelo microcontrolador. Em nosso caso, os displays estão sendo acionados por varredura.

Para acionar os displays de 7 segmentos, iremos utilizar o sistema de varredura, que permite através de um barramento de dados de 8 bits e mais 4 pinos de acionamento, "escrever" o valor correspondente ao dado que deve ser mostrado no visor.

Projetos com displays de 7 segmentos:



Painéis de equipamentos industriais



Balanças

Figura 7.3 – Projetos com displays de 7 segmentos

Varredura de Teclado matricial

O sistema de varredura de teclado matricial permite que o microcontrolador leia muitas teclas ligadas aos seus pinos. O teclado matricial é muito utilizado para economizar pinos físicos do microcontrolador. Equipamentos de diversos tipos usam o teclado matricial para inserir dados ao microcontrolador.

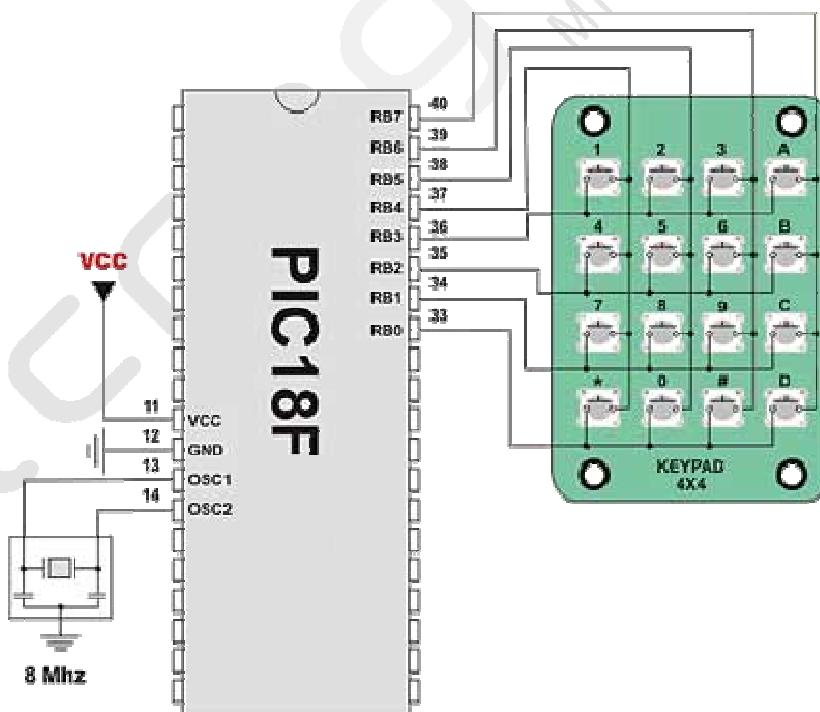


Figura 7.4 – Teclado Matricial

Projetos com teclados matriciais:



Figura 7.5 – Projetos com teclados matriciais

Acionamento de Leds

Os leds são utilizados praticamente em quase todas as aplicações eletrônicas. Através dos leds podemos visualizar o status de uma máquina, "desenhar" mensagens de textos, iluminar objetos, criar animações visuais, entre outras aplicações.

Iremos estudar os recursos de programação em C para controle das portas de saída disponíveis no microcontroladores PIC utilizando os leds como barramento de dados visual.

Projetos com com Leds

os leds são utilizados em diversos equipamentos no mercado para as mais variadas aplicações. Muitas das aplicações é o microcontrolador responsável pelo controle desses leds.



Figura 7.6 – Projetos com Leds

Conversor Analógico digital (A/D)

A aplicação básica do microcontrolador PIC trabalhando com o conversor A/D abaixo é simples, mas possui um grande conteúdo educativo para nós neste momento. No exemplo abaixo utilizamos dois simples trimpots para variarmos o valor de tensão no pino A/D do PIC. Este exemplo na verdade representa inúmeras aplicações práticas de equipamentos do mercado, tais como: aparelhos de medição, leitores de sensores de temperatura, atuadores, entre outros. Criaremos programas para controle e leitores de tensão nas unidades seguintes.

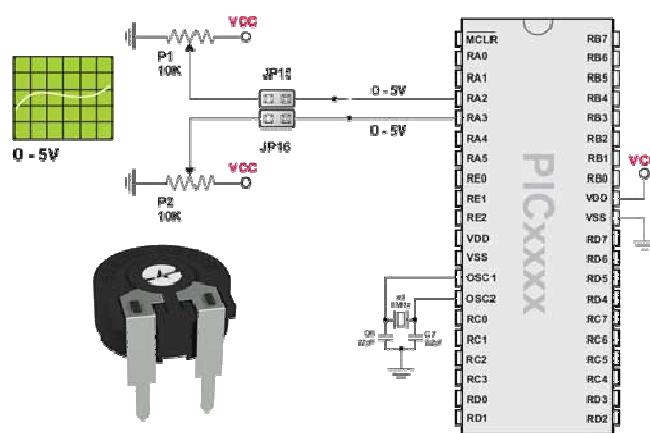


Figura 7.7 – Conversores A/D

Veremos também como ler e interpretar valores analógicos vindo de sensores de temperatura (LM35) utilizando os recursos da linguagem BASIC.

Projetos com os conversores A/D do PIC



Sensores de proximidade



Sondas e termopares



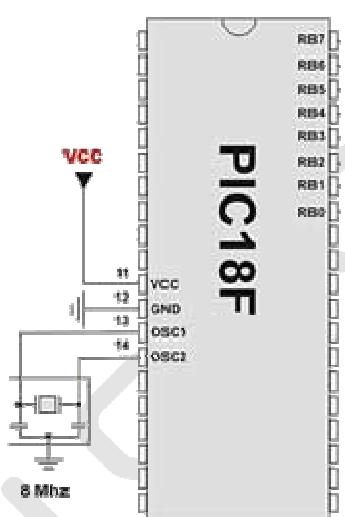
Equipamentos de medição

Figura 7.8 – Projetos com Conversores A/D

Controle PWM

Iremos simular programas de controle de largura de pulsos. Através do canal PWM disponível no PIC, podemos controlar diversos equipamentos, tais como: inversores de freqüência, estabilizadores, fonte chaveada, controle de velocidade de motores DC, entre outras.

Em nossos laboratórios, iremos controlar a velocidade de giro de uma ventoinha e controlar o aquecimento de uma carga resistiva via PWM.



Ventoinha



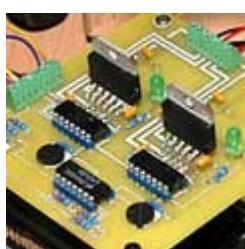
Resistência

Figura 7.9 – Acionamento PWM

Projetos com os PWM



Fontes chaveadas



Drive de Motores



Inversores de freqüência

Figura 7.10 – Projetos com PWM

Unidade 8 -- Anexos

Descrição das Pinagens(LCD)

Pino	Função	Descrição
1	Alimentação	Terra ou GND
2	Alimentação	VCC ou +5V
3	V0	Tensão para ajuste de contraste (ver figura1)
4	RS Seleção:	1 - dado, 0 - instrução
5	R/W seleção	1 - Leitura, 0 - Escrita
6	E chip Select	1 ou (1 - 0) Habilita, 0 - desabilita
7	D0	
8	D1	
9	D2	
10	D3	
11	D4	barramento de dados
12	D5	
12	D6	
14	D7	
15	A (display c/ back)	Anodo p/ LED backlight
16	K (display c/ back)	Catodo p/ LED backlight

Lista de códigos dos Caracteres

A seguir o código que devemos enviar para o LCD a fim de obtermos um determinado carácter:

xxxx	Upper 4 Bits Lower 4 Bits	CG RAM (1)	0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111	00P1P2P3P4P5P6P7P8P9P10P11P12P13P14P15P16P17P18P19P20P21P22P23P24P25P26P27P28P29P30P31P32P33P34P35P36P37P38P39P40P41P42P43P44P45P46P47P48P49P50P51P52P53P54P55P56P57P58P59P60P61P62P63P64P65P66P67P68P69P70P71P72P73P74P75P76P77P78P79P80P81P82P83P84P85P86P87P88P89P90P91P92P93P94P95P96P97P98P99P100P101P102P103P104P105P106P107P108P109P110P111P112P113P114P115P116P117P118P119P120P121P122P123P124P125P126P127P128P129P130P131P132P133P134P135P136P137P138P139P140P141P142P143P144P145P146P147P148P149P150P151P152P153P154P155P156P157P158P159P160P161P162P163P164P165P166P167P168P169P170P171P172P173P174P175P176P177P178P179P180P181P182P183P184P185P186P187P188P189P190P191P192P193P194P195P196P197P198P199P200P201P202P203P204P205P206P207P208P209P210P211P212P213P214P215P216P217P218P219P220P221P222P223P224P225P226P227P228P229P230P231P232P233P234P235P236P237P238P239P240P241P242P243P244P245P246P247P248P249P250P251P252P253P254P255P256P257P258P259P260P261P262P263P264P265P266P267P268P269P270P271P272P273P274P275P276P277P278P279P280P281P282P283P284P285P286P287P288P289P290P291P292P293P294P295P296P297P298P299P300P301P302P303P304P305P306P307P308P309P310P311P312P313P314P315P316P317P318P319P320P321P322P323P324P325P326P327P328P329P330P331P332P333P334P335P336P337P338P339P340P341P342P343P344P345P346P347P348P349P350P351P352P353P354P355P356P357P358P359P360P361P362P363P364P365P366P367P368P369P370P371P372P373P374P375P376P377P378P379P380P381P382P383P384P385P386P387P388P389P390P391P392P393P394P395P396P397P398P399P400P401P402P403P404P405P406P407P408P409P410P411P412P413P414P415P416P417P418P419P420P421P422P423P424P425P426P427P428P429P430P431P432P433P434P435P436P437P438P439P440P441P442P443P444P445P446P447P448P449P450P451P452P453P454P455P456P457P458P459P460P461P462P463P464P465P466P467P468P469P470P471P472P473P474P475P476P477P478P479P480P481P482P483P484P485P486P487P488P489P490P491P492P493P494P495P496P497P498P499P500P501P502P503P504P505P506P507P508P509P510P511P512P513P514P515P516P517P518P519P520P521P522P523P524P525P526P527P528P529P530P531P532P533P534P535P536P537P538P539P540P541P542P543P544P545P546P547P548P549P550P551P552P553P554P555P556P557P558P559P560P561P562P563P564P565P566P567P568P569P570P571P572P573P574P575P576P577P578P579P580P581P582P583P584P585P586P587P588P589P590P591P592P593P594P595P596P597P598P599P600P601P602P603P604P605P606P607P608P609P610P611P612P613P614P615P616P617P618P619P620P621P622P623P624P625P626P627P628P629P630P631P632P633P634P635P636P637P638P639P640P641P642P643P644P645P646P647P648P649P650P651P652P653P654P655P656P657P658P659P660P661P662P663P664P665P666P667P668P669P670P671P672P673P674P675P676P677P678P679P680P681P682P683P684P685P686P687P688P689P690P691P692P693P694P695P696P697P698P699P700P701P702P703P704P705P706P707P708P709P710P711P712P713P714P715P716P717P718P719P720P721P722P723P724P725P726P727P728P729P730P731P732P733P734P735P736P737P738P739P740P741P742P743P744P745P746P747P748P749P750P751P752P753P754P755P756P757P758P759P760P761P762P763P764P765P766P767P768P769P770P771P772P773P774P775P776P777P778P779P780P781P782P783P784P785P786P787P788P789P790P791P792P793P794P795P796P797P798P799P800P801P802P803P804P805P806P807P808P809P810P811P812P813P814P815P816P817P818P819P820P821P822P823P824P825P826P827P828P829P830P831P832P833P834P835P836P837P838P839P840P841P842P843P844P845P846P847P848P849P850P851P852P853P854P855P856P857P858P859P860P861P862P863P864P865P866P867P868P869P870P871P872P873P874P875P876P877P878P879P880P881P882P883P884P885P886P887P888P889P890P891P892P893P894P895P896P897P898P899P900P901P902P903P904P905P906P907P908P909P910P911P912P913P914P915P916P917P918P919P920P921P922P923P924P925P926P927P928P929P930P931P932P933P934P935P936P937P938P939P940P941P942P943P944P945P946P947P948P949P950P951P952P953P954P955P956P957P958P959P960P961P962P963P964P965P966P967P968P969P970P971P972P973P974P975P976P977P978P979P980P981P982P983P984P985P986P987P988P989P990P991P992P993P994P995P996P997P998P999P1000P1001P1002P1003P1004P1005P1006P1007P1008P1009P1010P1011P1012P1013P1014P1015P1016P1017P1018P1019P1020P1021P1022P1023P1024P1025P1026P1027P1028P1029P1030P1031P1032P1033P1034P1035P1036P1037P1038P1039P1040P1041P1042P1043P1044P1045P1046P1047P1048P1049P1050P1051P1052P1053P1054P1055P1056P1057P1058P1059P1060P1061P1062P1063P1064P1065P1066P1067P1068P1069P1070P1071P1072P1073P1074P1075P1076P1077P1078P1079P1080P1081P1082P1083P1084P1085P1086P1087P1088P1089P1090P1091P1092P1093P1094P1095P1096P1097P1098P1099P1100P1101P1102P1103P1104P1105P1106P1107P1108P1109P1110P1111P1112P1113P1114P1115P1116P1117P1118P1119P1120P1121P1122P1123P1124P1125P1126P1127P1128P1129P1130P1131P1132P1133P1134P1135P1136P1137P1138P1139P1140P1141P1142P1143P1144P1145P1146P1147P1148P1149P1150P1151P1152P1153P1154P1155P1156P1157P1158P1159P1160P1161P1162P1163P1164P1165P1166P1167P1168P1169P1170P1171P1172P1173P1174P1175P1176P1177P1178P1179P1180P1181P1182P1183P1184P1185P1186P1187P1188P1189P1190P1191P1192P1193P1194P1195P1196P1197P1198P1199P1200P1201P1202P1203P1204P1205P1206P1207P1208P1209P1210P1211P1212P1213P1214P1215P1216P1217P1218P1219P1220P1221P1222P1223P1224P1225P1226P1227P1228P1229P1230P1231P1232P1233P1234P1235P1236P1237P1238P

Tabela com o conjunto completo de instruções:

INSTRUÇÃO	RS	RW	B7	B6	B5	B4	B3	B2	B1	B0	Descrição e tempo de execução (uS)	t
Limpa Display	0	0	0	0	0	0	0	0	0	1	-Limpa todo o display e retoma o cursor para a primeira posição da primeira linha	1.6 mS
Home p/ Cursor	0	0	0	0	0	0	0	0	1	*	-Retorna o cursor para a 1. coluna da 1. Linha -Retorna a mensagem previamente deslocada a sua posição original	1.6 mS
Fixa o modo de funcionamento	0	0	0	0	0	0	0	1	X	S	-Estabelece o sentido de deslocamento do cursor (X=0 p/ esquerda, X=1 p/ direita) -Estabelece se a mensagem deve ou não ser deslocada com a entrada de um novo caractere (S=1 SIM, X=1 p/ direita) -Esta instrução tem efeito somente durante a leitura e escrita de dados.	40 uS
Controle do Display	0	0	0	0	0	0	1	D	C	B	-Liga (D=1) ou desliga display (D=0) -Liga(C=1) ou desliga cursor (C=0) -Cursor Piscante(B=1) se C=1	40 uS
Desloca cursor ou mensagem	0	0	0	0	0	1	C	R	*	*	-Desloca o cursor (C=0) ou a mensagem (C=1) para a Direita se (R=1) ou esquerda se (R=0) - Desloca sem alterar o conteúdo da DDRAM	40 uS
Fixa o modo de utilização do módulo LCD	0	0	0	0	1	Y	N	F	*	*	-Comunicação do módulo com 8 bits(Y=1) ou 4 bits(Y=0) -Número de linhas: 1 (N=0) e 2 ou mais (N=1) -Matriz do caractere: 5x7(F=0) ou 5x10(F=1) - Esta instrução deve ser ativada durante a inicialização	40 uS
Posiciona no endereço da CGRAM	0	0	0	1	Endereço da CGRAM					-Fixa o endereço na CGRAM para posteriormente enviar ou ler o dado (byte)		
Posiciona no endereço da DDRAM	0	0	1	Endereço da DDRAM					-Fixa o endereço na DDRAM para posteriormente enviar ou ler o dado (byte)			40 uS
Leitura do Flag Busy	0	1	B F	AC					-Lê o conteúdo do contador de endereços (AC) e o BF. O BF (bit 7) indica se a última operação foi concluída (BF=0 <i>concluída</i>) ou está em execução (BF=1).			0
Escreve dado na CGRAM / DDRAM	0	1	Dado a ser gravado no LCD					- Grava o byte presente nos pinos de dados no local apontado pelo contador de endereços (<i>posição do cursor</i>)			40 uS	
Lê Dado na CGRAM / DDRAM	1	1	Dado lido do módulo					- Lê o byte no local apontado pelo contador de endereços (<i>posição do cursor</i>)			40 uS	

Figura 8.2 – Conjunto completo de instruções do LCD

Tabela com as instruções mais comuns:

DESCRIÇÃO	MODO	RS	R/W	Código (Hexa)
Display	Liga (sem cursor)	0	0	0C
	Desliga	0	0	0A / 08
Limpa Display com Home cursor		0	0	01
Controle do Cursor	Liga	0	0	0E
	Desliga	0	0	0C
	Desloca para Esquerda	0	0	10
	Desloca para Direita	0	0	14
	Cursor Home	0	0	02
	Cursor Piscante	0	0	0D
	Cursor com Alternância	0	0	0F
Sentido de deslocamento do cursor ao entrar com caracter	Para a esquerda	0	0	04
	Para a direita	0	0	06
Deslocamento da mensagem ao entrar com caracter	Para a esquerda	0	0	07
	Para a direita	0	0	05
Deslocamento da mensagem sem entrada de caracter	Para a esquerda	0	0	18
	Para a direita	0	0	1C
End. da primeira posição	primeira linha	0	0	80
	segunda linha	0	0	C0

Figura 8.3 – Instruções mais comuns do LCD

Resumo com as instruções mais úteis:

FIXAÇÃO DAS CONDIÇÕES DE UTILIZAÇÃO	Instrução
1 linha 5x7 (8 bits)	30H
2 linha 5x7 (8 bits)	38H
1 linha 5x10 (8 bits)	34H
1 linha 5x7 (4 bits)	20H
2 linha 5x7 (4 bits)	28H
1 linha 5x10 (4 bits)	24H

CONTROLE DISPLAY	Instrução
Display aceso c/ cursor fixo	0EH
Display aceso c/ cursor intermitente	0FH
Display aceso sem cursor	0CH
Display apagado	08H

MODO DE OPERAÇÃO	Instrução
Escreve deslocando a mensagem para esquerda (cursor fixo)	07H
Escreve deslocando a mensagem para a direita (cursor fixo)	05H
Escreve deslocando o cursor para a direita	06H
Escreve deslocando o cursor para a esquerda	04H

OUTROS COMANDOS ÚTEIS	Instrução
Limpa display e retorna o cursor para o inicio	01H
Retorna o cursor para o inicio (sem alterar a DDRAM)	02H
Desloca somente o cursor para a direita	14H
Desloca somente o cursor para a esquerda	10H
Desloca o cursor + mensagem para a direita	1CH
Desloca o cursor + mensagem para a esquerda	18H
Desloca o cursor para posição inicial da segunda linha	C0H
Desloca o cursor para posição inicial da primeira linha	80H

CGRAM (caracteres especiais)	Instrução
Endereço inicial para construir caracteres especiais	40H
Para escrever o primeiro caracter (previamente construídos)	00H
Para escrever o último caracter (previamente construídos)	07H

Comandos LCD do mikroC PRO (LCD)

Comandos	Descrição
_LCD_FIRST_ROW	Move cursor para primeira linha do LCD
_LCD_SECOND_ROW	Move cursor para segunda linha do LCD
_LCD_THIRD_ROW	Move cursor para a terceira linha do LCD
_LCD_FOURTH_ROW	Move cursor para a quarta linha do LCD
_LCD_CLEAR	Apaga todo o display
_LCD_RETURN_HOME	Retorna cursor Home (1º coluna da 1º linha do LCD)
_LCD_CURSOR_OFF	Desliga cursor
_LCD_UNDERLINE_ON	Cursor aparecerá no visor como underline
_LCD_BLINK_CURSOR_ON	Ativa o modo piscante do cursor
_LCD_MOVE_CURSOR_LEFT	Move cursor para a esquerda sem movimentar os textos
_LCD_MOVE_CURSOR_RIGHT	Move cursor para a direita sem movimentar os textos
_LCD_TURN_ON	Liga todo o visor do LCD
_LCD_TURN_OFF	Apaga todo o visor do LCD, sem perder os dados no visor
_LCD_SHIFT_LEFT	Movimenta todos os textos do LCD para a esquerda
_LCD_SHIFT_RIGHT	Movimenta todos os textos do LCD para a direita

Unidade 9 – Figuras

Figura 1.1 – Evolução da Família PIC	6
Figura 1.2 - Encapsulamentos	8
Figura 1.3 – Diagrama interno do PIC18F4520	9
Figura 1.4 – Memória de Programa FLASH do PIC18F4520	12
Figura 1.5 - Opcode gerado após compilação do programa	13
Figura 1.6 – Código Hexadecimal	14
Figura 1.7 – Diagrama da memória RAM interna	15
Figura 1.8 – Registradores de Funções Especiais – SFR's	16
Figura 1.9 – PIC18F4520	17
Figura 1.10 - PORTA	18
Figura 1.11 – Circuito de Reset	21
Figura 1.12 – Configuração dos fusíveis no MikroC PRO	24
Figura 1.13 - Modo HSPLL	25
Figura 1.14 – Modo EC – com pino RA6 como I/O de uso geral	26
Figura 1.15 – Modo EC – com pino RA6 como saída de clock	26
Figura 1.16 – Clock gerado a partir de um cristal de quartzo	26
Figura 1.17 – Osciladores	27
Figura 2.1- Canais AD multiplexados	30
Figura 2.2 – Seleção A/D	31
Figura 3.1 – Largura de Pulso	35
Figura 3.2 – Variação mínima ao máximo do duty Cicle do PWM	35
Figura 3.3 – Variação do duty cicle do sinal PWM	36
Figura 3.4 – Variação do duty Cicle	37
Figura 4.1 – Divisão do Prescaler	38
Figura 4.2 – Diagrama interno do TIMER0	40
Figura 4.3 – Aplicação com o modo contador do TIMER0	43
Figura 4.4 – Diagrama interno do TIMER1	46
Figura 4.5 – Circuito interno do modo RTC	47
Figura 4.6 – Ligação do segundo oscilador ao microcontrolador PIC	47
Figura 4.7 – Esquema Eletrônica	48
Figura 4.8 – Diagrama de construção interna do TIMER2	50
Figura 4.9 – Estrutura interna do TIMER3	52
Figura 5.1 – Vetor de Alta Prioridade e Baixa Prioridade de Interrupção série PIC18F	54
Figura 5.2 – Diagrama de Interrupção x Tempo	54
Figura 5.3 – Exemplo para compreensão das Interrupções do PIC	55
Figura 6.1 – Diagrama estrutural da interrupção externa INT0	61
Figura 6.2 – Tipos de disparo das interrupções externa INT0, INT1 e INT2	62
Figura 6.3 – Diagrama estrutural da interrupção externa INT1	63
Figura 6.4 – Diagrama estrutural da interrupção externa INT2	64
Figura 6.5 - Gráfico de acionamento de interrupção externa RB	65
Figura 7.1 – Aplicações com display LCD	66
Figura 7.2 – Projetos com displays LCD	66
Figura 7.3 – Projetos com displays de 7 segmentos	67
Figura 7.4 – Teclado Matricial	67
Figura 7.5 – Projetos com teclados matriciais	68
Figura 7.6 – Projetos com Leds	68
Figura 7.7 – Conversores A/D	68
Figura 7.8 – Projetos com Conversores A/D	69

Figura 7.9 – Acionamento PWM	69
Figura 7.10 – Projetos com PWM	69
Figura 8.1 – Lista de códigos dos Caracteres no LCD	70
Figura 8.2 – Conjunto completo de instruções do LCD	71
Figura 8.3 – Instruções mais comuns do LCD	72

Unidade 10 – Tabelas

Tabela 1.1 – Descrição dos pinos do Microcontrolador PIC18F4520	11
Tabela 1.2 – Pinos PORTA	17
Tabela 1.3 – Bits de configuração do PIC18F4520	23
Tabela 1.4 – Bits de configuração do PIC18F4520(Detalhado)	23
Tabela 2.1 – ADCON0	31
Tabela 2.2 – Configuração Pinos A/D	31
Tabela 4.1- Registrador relacionados com o TIMER0	39
Tabela 4.2 – T0CON	40
Tabela 4.3 - Bits de seleção de fonte de prescaler(Timer 0)	41
Tabela 4.4 – INTCON	42
Tabela 4.5 – Registrador relacionados com o TIMER1 – (as células em branco)	45
Tabela 4.6 – T1CON	45
Tabela 4.7 – Bits de seleção de fonte de prescaler	45
Tabela 4.8 – Registradores responsáveis pela configuração do TIMER2	50
Tabela 4.9 – T2CON	51
Tabela 4.10 – Bits de ajuste do postcale	51
Tabela 4.11 – Bits responsáveis pelo ajuste de prescaler(TIMER 2)	51
Tabela 4.12 – Registradores relacionados com o TIMER3	51
Tabela 4.13 – T3CON	52
Tabela 4.14 – Habilita a fonte de timer para o módulo CCP	53
Tabela 4.15 – bits de ajuste do prescaler do TIMER3	53
Tabela 5.1 - RCON	56
Tabela 5.2 – INTCON	57
Tabela 5.3 – INTCON2	57
Tabela 5.4 – INTCON3	57
Tabela 5.5 – PIR1	58
Tabela 5.6 – PIR2	58
Tabela 5.7 – PIE1	58
Tabela 5.8 – PIE2	58
Tabela 5.9 – IPR1	58
Tabela 5.10 – IPR2	59