Coleção Livro de Bolso

Programação básica de microcontroladores PIC

Rogério Laureano Martins

Programação **5**

PIC para substituir CIs digitais **25**
Portas lógicas *26*
Temporizadores *42*
Multivibrador astável com delay_ms 43
Multivibrador monoestável com delay_ms 44
Multivibrador astável com overflow do timer 0 (zero) 45
Multivibrador monoestável com overflow do timer 0 (zero) 47
Multivibrador biestável (Flip-Flop RS) 50
Flip-flop *51*
Contadores binários *62*
Codificadores e Decodificadores *102*
Comparador binário *110*
Multiplexador *111*
Outros exemplos de circuitos digitais programados *113*

Circuitos de potência **130**
Opticoacoplador *130*
4N25 130

4N30 131
PC817 132
6N136 133
6N138 134
Interface de potência 134

Fontes e reguladores 135
Reguladores positivos 136
Reguladores negativos 137

Montagem de placas 140
Materiais e equipamentos 140
Como será a placa 142
Desenho do projeto no computador 143
Impressão dos esquemas 154
Impressão das trilhas nas placas 167
Corrosão da placa 170
Retirada da tinta 171
Impressão das legendas nas placas 171
Proteção das trilhas e exposição das ilhas ... 172
Furação da placa 173
Colocada dos componentes 173
Soldagem 173

Conclusão 174

Minha gratidão a Deus por me permitir compartilhar o que tenho aprendido. Minha gratidão à minha esposa Beatriz e ao meu filho João por terem paciência. Minha gratidão aos professores e youtubers que de alguma forma procura passar o conhecimento pessoal às pessoas.

Rogério Laureano Martins

Programação

Um programa feito para microcontroladores normalmente seguem um padrão. O primeiro passo é configurar as funcionalidades dos pinos. Como descrito no começo deste livro, os pinos podem ter mais de uma funcionalidade. Por exemplo o pino 17 é um bit do número binário RA mas também é uma entrada analógica "A0". Junto com a configuração dos pinos, também é necessário configurar os fusíveis e recursos especiais. Mais à frente serão detalhados estes fusíveis e recursos especiais.

A série de PICs PIC16F628A, que estamos utilizando como exemplo inicial, possui o pino 7 como o bit 1 do número binário RB. A identificação do pino 7 como bit do número RB é dado como RB1_bit.

Um número digital pode ser de entrada ou saída. Ou seja, para utilização de bits digitais do

microcontrolador PIC16F628A, é necessário definir se estes bits receberão sinais digitais ou enviarão sinais digitais. A variável reservada para o byte RA é o TRISA e para o byte RB é o TRISB.

TRISA → Determina byte A ou seus bits como entrada ou saída
0 parece "o" de output e o 1 lembra o "i" de input

TRISB → Determina byte B ou seus bits como entrada ou saída
0 parece "o" de output e o 1 lembra o "i" de input

Para atribuir ou recuperar um valor de saída em RA utiliza-se a variável reservada PORTA. Para a saída RB PORTB.

PORTA → Permite atribuir ou recuperar valores ao byte A ou seus bits

PORTB → Permite atribuir ou recuperar valores ao byte B ou seus bits

Para atribuir ou recuperar um valor de um bit em específico por exemplo RB1 (pino 7) deve-se utilizar a variável reservada RB1_bit. No caso de RA0 (pino 17) deve-se usar RA0_bit.

Um bom programa para iniciar o estudo dos microcontroladores PIC, é fazer um led ficar piscando.

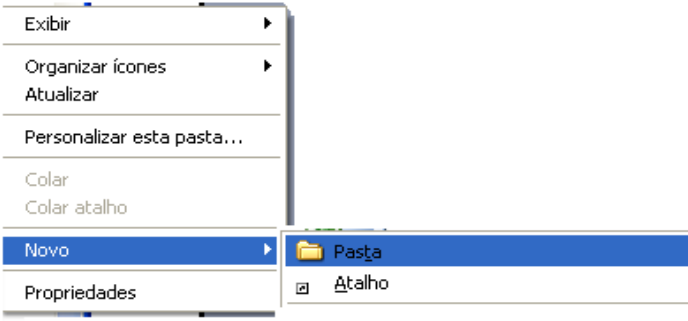
O programa terá duas ações do sistema. A primeira será acender o led e a segunda apagá-lo. Este ciclo se repetirá enquanto o circuito estiver ligado.

Para os exercícios de programação, deve-se criar um diretório padrão que será "C:\aulasPIC"
Para tanto, siga os passos:

1 – Pressione o símbolo do windows + “r” do teclado

2 – Na caixa “Executar” informe “c:” e pressione a tecla <ENTER>

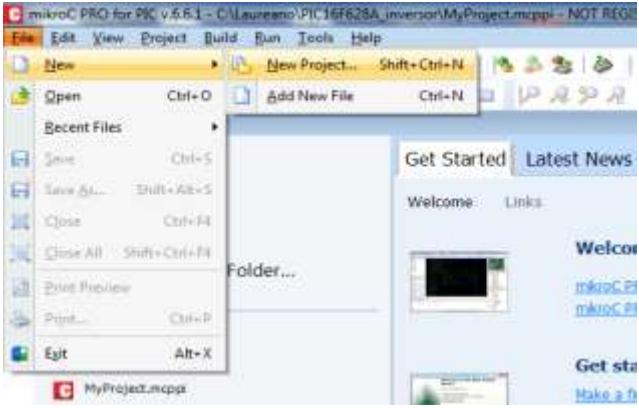
3 – Clique com o botão direito do mouse escolhendo o menu “Novo → Pasta”



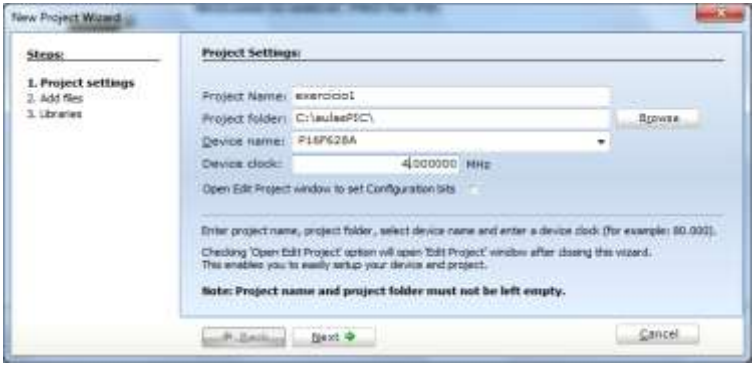
4 – Nomeie a pasta como “aulasPIC”

Nestes primeiro exercício, abra o MikroC (Caso não tenha instalado em sua máquina baixe da internet).

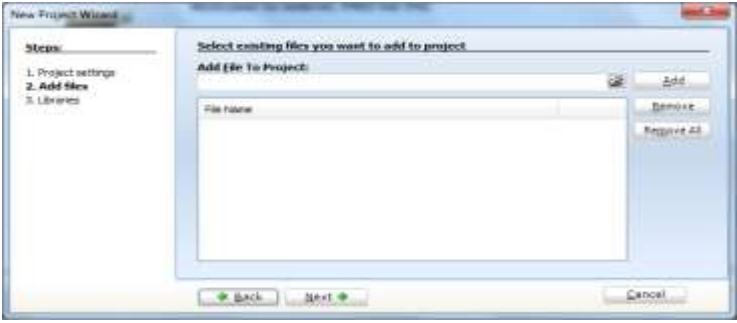
1 – Clique no menu "New → New Project..."



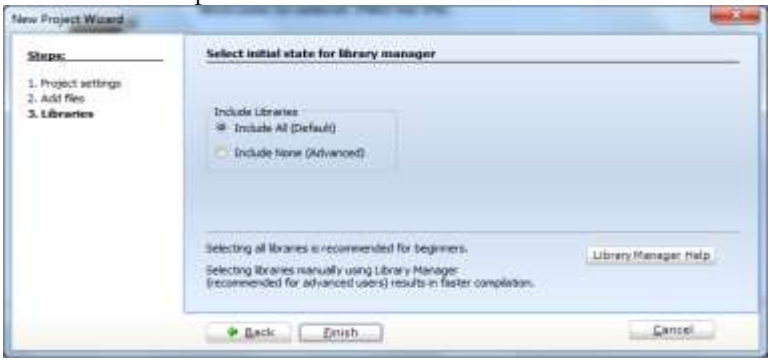
2 – Preencher os dados da janela de projeto como abaixo e clique em "Next"



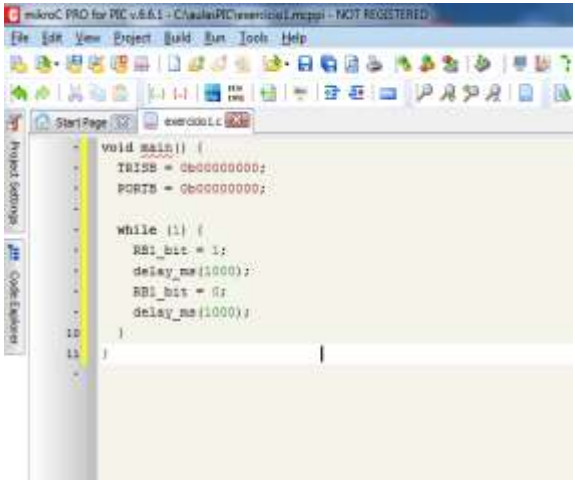
3 – Clicar em next novamente



4 – Clicar pela terceira vez em "Next"



5 – Digite o fonte do arquivo exercicio1.c conforme abaixo:

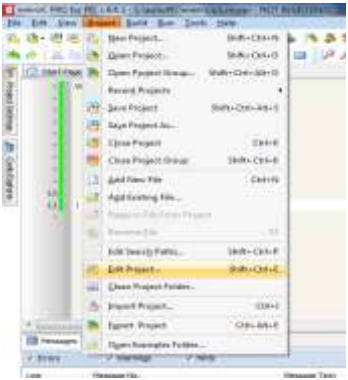


Note que a estrutura básica do programa é o processo "void main()" dentro deste processo, iniciamos as variáveis reservadas TRISB (para indicar que todos os bits do número binário RB serão saída de dados) e PORTB (para indicar que o número RB é igual a 0). A notação tanto para TRISB quanto para PORTB foi "0b00000000". Os dois primeiros dígitos "0b"

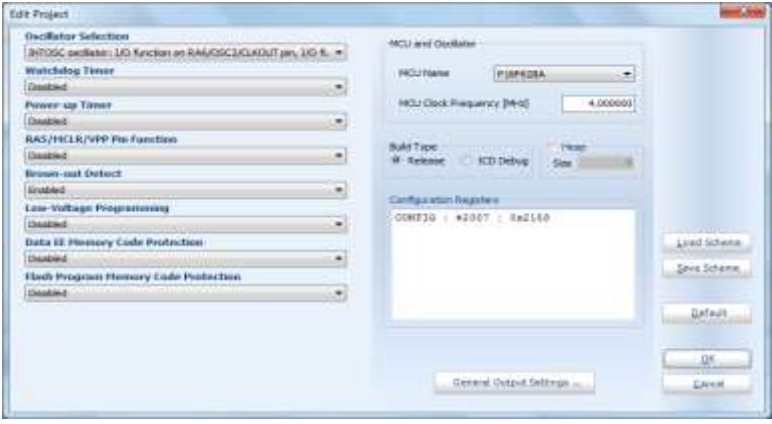
indica a notação binária. Poderíamos alterar a notação para que fosse decimal. Neste caso bastava escrever "0" visto que o número binário 0b00000000 é igual ao decimal 0. Uma terceira notação bastante proveitosa é a notação hexadecimal. Neste caso seria "0x00".

Note também, uma segunda estrutura dentro do programa que é o "while (1)". Esta estrutrura será detalhada mais à frente, mas para um entendimento rápido, podemos traduzir este comando como: "Faça enquanto for verdadeiro" ou "Faça até que se prove ao contrário". Costumamos chamar esta estrutura de "looping infinito". Dentro desta estrutura foi atribuido ao bit RB1 um valor em nível alto (usualmente 5V) e após 1 segundo (1000 milisegundos) foi atribuido um valor em nível baixo (usualmente 0V). Logo após 1 segundo, o programa volta à primeira linha da estrutrura "while (1)"

6 – Clique no menu "Project → Edit Project":

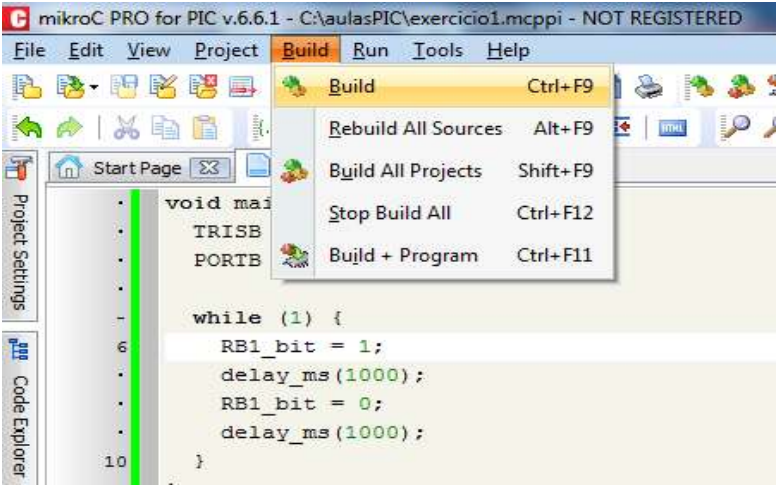


7 – Preencher os dados da janela de fusíveis conforme abaixo:

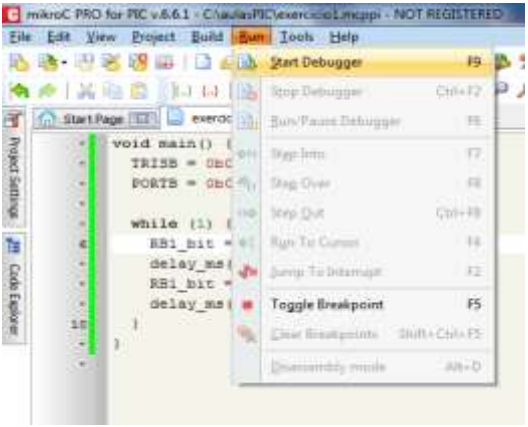


8 – Salve o arquivo exercicio1.c

9 – Clique no menu "Build → Build" para compilar o programa



10 – Clique no menu "Run → Start Debugger" para simular o programa do PIC



11 – Na janela de Debug, digite RB1_bit para que o valor seja monitorado e clique no botão de adição.