CURSO DE MICROCONTROLADORES

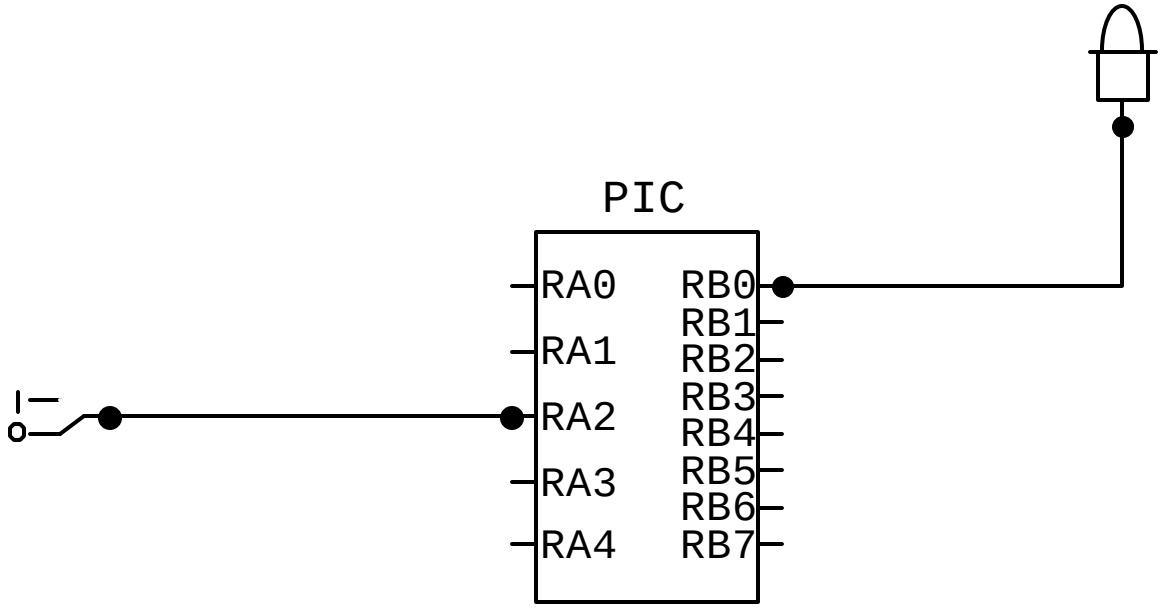
EXERCÍCIOS – PROFESSOR

Programas Exemplo

Programa 1:

Este programa faz com que um LED, ligado àsaída RB0, seja aceso, caso uma chave C, ligada àentrada RA2, esteja em “1”. Caso contrário, o diodo se apaga.

Diagrama Esquemático:



Codificação:

Definição do tipo de dispositivo que será utilizado. Corresponde ao tipo do microcontrolador no qual iremos gravar o programa

```
LIST P=16F84
```

Definição dos comandos de usuário para alteração das páginas de memória (bancos)

```
#DEFINE BANCO0   BCF STATUS, RP0
#DEFINE BANCO1   BSF STATUS, RP0
```

Definições das constantes utilizadas para acessar os registradores do microcontrolador. Cada registrador possui um endereço, então criamos “aliases” para facilitar a codificação. Afinal de contas, é mais fácil se referir ao registrador PORTB pelo seu nome, ao invés de seu endereço na memória (0x06)

```
STATUS      EQU   0x03
PORTA       EQU   0x05
PORTB       EQU   0x06
RP0         EQU   0x05
```

Definições do endereço de início de processamento, ou vetor de reset (0x00), e do vetor de interrupções (0x05)

```
        ORG   0x00
        GOTO Inicio
        ORG   0x05
```

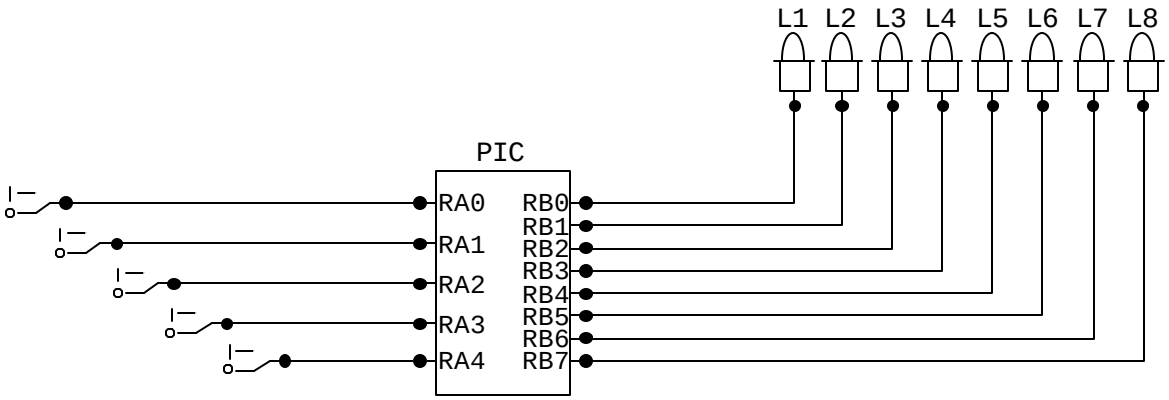
Início do programa

```
Inicio:
        BANCO1                  ; mudança para o banco de memória 1
        CLRF        PORTB       ; limpando os bits de PORTB. Com isso, todos os
                                ; pinos (RB0 a RB7) serão utilizados como saída
        MOVLW       B'00000100' ; utilizando o registrador Work (W) para configurar a
                                ; porta de entrada A. O terceiro bit, que corresponde
                                ; ao pino RA2 está sendo configurado como entrada,
                                ; e o restante dos pinos desta porta como saída
        MOVWF       PORTA       ; concluindo a configuração da porta A
        BANCO0                  ; mudando para o banco de memória 0
        CLRF        PORTB       ; limpando a porta B



Loop:
        BTFSS       PORTA, 2    ; verifica o estado da chave C (pino RA2). Se o valor
                                ; for “1”, a próxima linha é pulada, então o LED irá
                                ; acender. Caso contrário o LED irá apagar
        GOTO        Apagar      ; se RA2 = 0, apaga o LED
        GOTO        Acender     ; se RA2 = 1, acende o LED

Acender:
        BSF         PORTB, 0    ; seta o bit 0 (RB0) da porta B. Isto faz com que o
                                ; LED se acenda. RB0 = 1
        GOTO        Loop        ; volta para verificação da chave C

Apagar:
        BCF         PORTB, 0    ; limpa o bit 0 (RB0) da porta B. Isto faz com que o
                                ; LED se apague. RB0 = 0
        GOTO        Loop        ; volta para verificação da chave C

        END                     ; fim do programa
```

Programa 2:

Este programa funciona como um “segredo eletrônico”. O microcontrolador espera uma combinação de 5 chaves, ligadas na porta A (RA0 – RA5), e, se esta combinação ocorrer, todos os LEDs ligados à porta B (RB0 – RB7) se acendem. Caso contrario, eles permanecem apagados. A combinacao deve ser 1-0-1-0-1.

Diagrama Esquemático:



Codificação:

Definição do tipo de dispositivo que será utilizado. Corresponde ao tipo do microcontrolador no qual iremos gravar o programa

```
LIST P=16F84
```

Definição dos comandos de usuário para alteração das páginas de memória (bancos)

```
#DEFINE BANCO0   BCF STATUS, RP0
#DEFINE BANCO1   BSF STATUS, RP0
```

Definições das constantes utilizadas para acessar os registradores do microcontrolador.

```
STATUS      EQU   0x03
PORTA       EQU   0x05
PORTB       EQU   0x06
RP0         EQU   0x05
Z           EQU   0x02
```

Definições do endereço de início de processamento, ou vetor de reset (0x00), e do vetor de interrupções (0x05)

```
ORG  0
GOTO Inicio
ORG  5
```

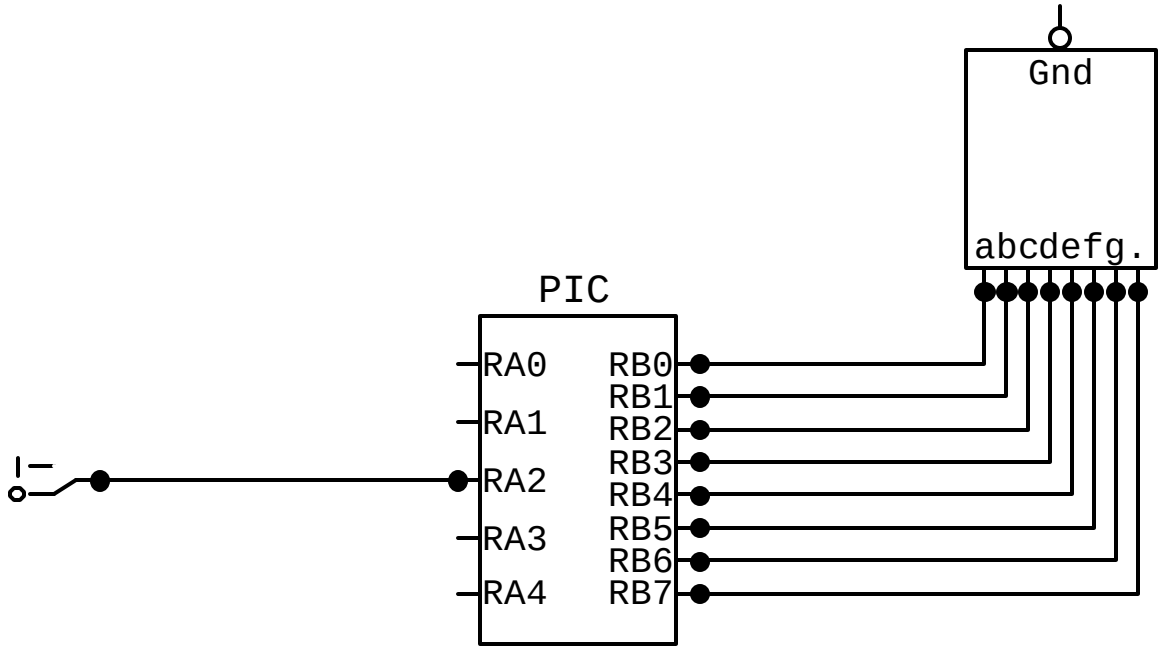
Início do programa

```
Inicio:
        BANCO1                  ; mudança para o banco de memória 1
        CLRF    PORTB           ; limpando os bits de PORTB. Com isso, todos os
                                ; pinos (RB0 a RB7) serão utilizados como saída
        MOVLW   B'00011111'     ; utilizando o registrador Work (W) para configurar a
                                ; porta de entrada A. Todos os pinos da porta (RA0 –
                                ; RA4) estão sendo configurados como entrada
        MOVWF   PORTA           ; concluindo a configuração da porta A
        BANCO0                  ; mudando para o banco de memória 0
        CLRF    PORTB           ; limpando a porta B

Loop:
        MOVF    PORTA, W        ; fazendo uma cópia do status da porta A para o
                                ; registrador Work (W)
        ANDLW   B'00011111'     ; AND lógico entre o registrador Work (W) e o valor
                                ; “00011111”, para que os 5 primeiros bits sejam
                                ; mantidos e os outros 3 passem a ser 0. Isto é
                                ; necessário para realizar a comparação abaixo
        XORLW   B'00010101'     ; verificando a existência da seqüência 1-0-1-0-1,
                                ; através de um XOR.
        BTFSS   STATUS, Z       ; se a operação acima resultou em 0, ou seja se todos
                                ; os bits de W estiverem em 0, significa que a
                                ; sequência 1-0-1-0-1 está presente. Então, a próxima
                                ; linha é pulada
        GOTO    Loop            ; se a sequência não estiver presente, é feita uma
                                ; nova verificação
        MOVLW   0xFF            ; se a sequência estiver presente, todos os bits da
                                ; porta B são setados (valor 0xFF), fazendo com que
                                ; os LEDs se acendam.
        MOVWF   PORTB           ; acendendo os LEDs

Fim:
        GOTO    Fim             ; loop infinito. Para começar o programa novamente,
                                ; depois de uma sequência correta, é necessário
                                ; resetar o microcontrolador
        END                     ; fim do programa
```

Programa 3:

Este programa utiliza um display de 7 segmentos, ligado à porta B (RB0 – RB7) para mostrar o valor de uma chave C ligada à entrada RA2 da porta A. Se a chave estiver ligada, o display exibe o número 1. Caso contrário, é exibido o número 0.

Diagrama Esquemático:



Codificação:

Definição do tipo de dispositivo que será utilizado. Corresponde ao tipo do microcontrolador no qual iremos gravar o programa

```
LIST P=16F84
```

Definição dos comandos de usuário para alteração das páginas de memória (bancos)

```
#DEFINE BANCO0   BCF STATUS, RP0
#DEFINE BANCO1   BSF STATUS, RP0
```

Definições das constantes utilizadas para acessar os registradores do microcontrolador.

```
STATUS      EQU   0x03
PORTA       EQU   0x05
PORTB       EQU   0x06
RP0         EQU   0x05
```

Definições do endereço de início de processamento, ou vetor de reset (0x00), e do vetor de interrupções (0x05)

```
        ORG   0
        GOTO Inicio
        ORG   5
```

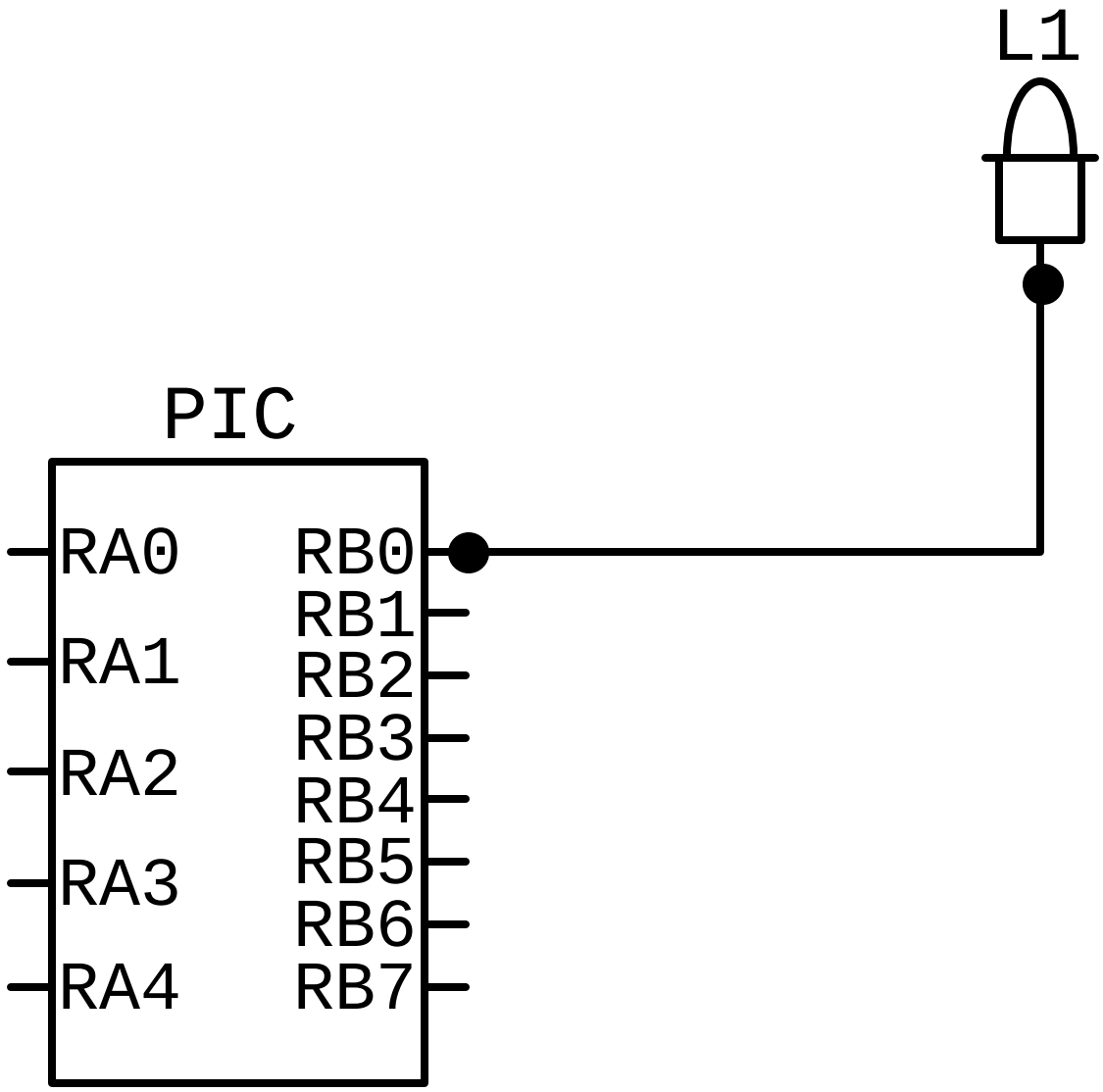
Início do programa:

```
Inicio:
        BANCO1                  ; mudança para o banco de memória 1
        CLRF        PORTB       ; limpando os bits de PORTB. Com isso, todos os
                                ; pinos (RB0 a RB7) serão utilizados como saída
        MOVLW       B'00000100' ; utilizando o registrador Work (W) para configurar a
                                ; porta de entrada A. O terceiro bit, que corresponde
                                ; ao pino RA2 está sendo configurado como entrada,
                                ; e o restante dos pinos desta porta como saída
        MOVWF       PORTA       ; concluindo a configuração da porta A
        BANCO0                  ; mudando para o banco de memória 0

Loop:
        BTFSS       PORTA, 2    ; verifica o estado da chave C (pino RA2). Se o valor
                                ; for “1”, a próxima linha é pulada, então o display
                                ; irá exibir o número 1. Caso contrário o display irá
                                ; exibir o número 0
        GOTO        Zero        ; desvia para linha “Zero” se o valor de RA2 for 0
        GOTO        Um          ; desvia para linha “Um” se o valor de RA2 for 1

Zero:
        MOVLW       B'00111111' ; como o valor a ser exibido deve ser Zero, o código
                                ; binário correspondente àeste valor no display é
                                ; colocado no registrador Work (W)
        MOVWF       PORTB       ; exibindo o valor Zero no display físico
        GOTO        Loop        ; volta para verificação da chave C

Um:
        MOVLW       B'00000110' ; como o valor a ser exibido deve ser Um, o código
                                ; binário correspondente àeste valor no display é
                                ; colocado no registrador Work (W)
        MOVWF       PORTB       ; exibindo o valor Um no display físico
        GOTO        Loop        ; volta para verificação da chave C

        END                     ; fim do programa
```

Programa 4:

Este programa faz com que o LED, ligado ao pino RB0 da porta B acenda e apague a cada segundo. Como o ciclo de clock do microcontrolador é muito pequeno, é necessário introduzir *delays* para que as transições do LED possam ser visualizadas. Estes *delays* funcionam da seguinte maneira: uma rotina, chamada Delay1S, que utiliza a subrotina Del10, que gera *delays* de 10 ms a cada chamada. Dentro desta rotina existe uma variável auxiliar (TEMPO1), que começa com o valor 100 e vai sendo decrementada. Cada vez que ela é decrementada, a rotina Del10 é chamada novamente. Ou seja, serão 100 chamadas àrotina Del10, gerando um *delay* total de 1 segundo.

Diagrama Esquemático:



Codificação:

Definição do tipo de dispositivo que será utilizado. Corresponde ao tipo do microcontrolador no qual iremos gravar o programa

```
LIST P=16F84
```

Definição dos comandos de usuário para alteração das páginas de memória (bancos)

```
#DEFINE BANCO0   BCF STATUS, RP0
#DEFINE BANCO1   BSF STATUS, RP0
```

Definições das constantes utilizadas para acessar os registradores do microcontrolador e variáveis auxiliares utilizadas no programa

```
STATUS          EQU   0x03
PORTA           EQU   0x05
PORTB           EQU   0x06
RP0             EQU   0x05
OPTION_REG      EQU   0x01
INTCON          EQU   0x0B
TEMPO1          EQU   0x0C
```

Definições do endereço de início de processamento, ou vetor de reset (0x00), e do vetor de interrupções (0x05)

```
ORG   0
GOTO Inicio
ORG   5
```

Início do programa:

```
Inicio:
        BANCO1                  ; mudança para o banco de memória 1
        CLRF    PORTB           ; limpando os bits de PORTB. Com isso, todos
                                ; os pinos (RB0 a RB7) serão utilizados como
                                ; saída
        MOVLW   B'00000111'     ; utilizando o registrador Work (W) para
                                ; configurar o registrador de operação do
                                ; microcontrolador. Os três primeiros bits, que
                                ; correspondem àconfiguração do prescaler,
                                ; estão sendo setados, o que significa que o
                                ; prescaler assumirá o valor de 1:256. Por
                                ; exemplo, se tivermos um clock de 4Mhz,
                                ; cada instrução é executada em 1µs. Com o
                                ; prescaler de 1:256, o TMR0 é incrementado
                                ; a cada 256 ciclos, consequentemente ele irá
                                ; estourar em, aproximadamente, 6,5ms.
        MOVWF   OPTION_REG      ; concluindo a configuração do registrador de
                                ; operações
        BANCO0                  ; mudando para o banco de memória 0
```

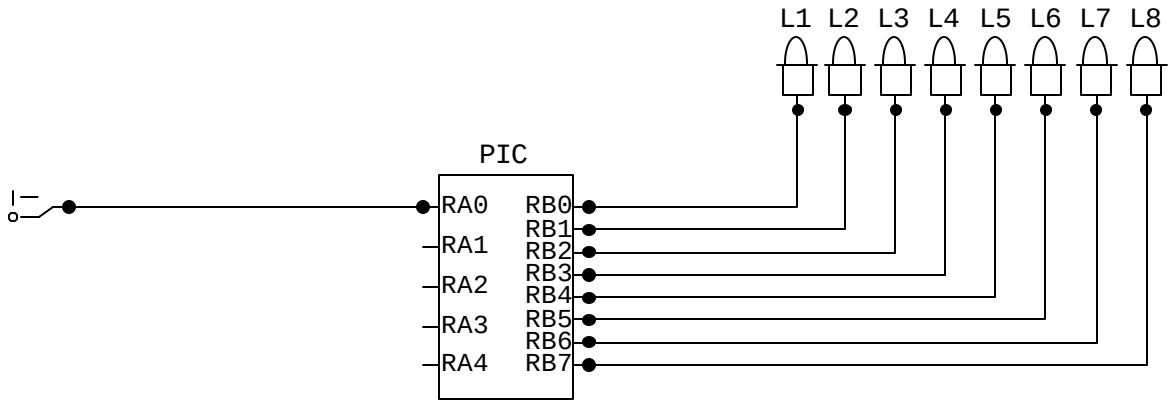
```
        CLRF    PORTB           ; limpando a porta B

Loop:
        BSF     PORTB, 0        ; setando o pino RB0 da porta B. Isto faz com
                                ; que o LED se acenda
        CALL    Delay1S         ; chamando a rotina que introduz um delay de
                                ; 1 segundo no processamento, fazendo com
                                ; que o LED fique aceso por 1 segundo
        BCF     PORTB, 0        ; limpando o pino RB0 da porta B. Isto faz com
                                ; que o LED se apague
        CALL    Delay1S         ; novamente a rotina de delay é chamada,
                                ; para que o LED fique apagado por 1 segundo
        GOTO    Loop            ; volta para o laço principal

Delay1S
        BANCO0                  ; mudando para o banco de memória 0
        MOVLW   0x64            ; colocando o valor 64h (100 em decimal) no
                                ; registrador Work (W)
        MOVWF   TEMPO1          ; movendo o valor anterior para a variável
                                ; temporária TEMPO1, que será utilizada na
                                ; rotina Del10 (delay de 10ms)
        CALL    Del10           ; chamando a rotina Del10
        RETURN                  ; retornando

Del10
        BCF     INTCON, 2       ; setando o bit 2 do registrador de
                                ; configuração de interrupções. Isto faz com
                                ; que a interrupção de estouro de timer
                                ; (registrador TMR0) seja habilitada
        MOVLW   0xD8            ; colocando o valor D8h (11011000 em binário)
                                ; no registrador Work (W)
        MOVWF   OPTION_REG      ; movendo o valor anterior para o registrador
                                ; de operações. Isto faz com que os bits do
                                ; registrador sejam configurados da seguinte
                                ; maneira:
                                ; - Pull-Ups Internos (/RBPU) => habilitados
                                ; - Borda que irá gerar a interrupção externa
                                ; no RB0 (INTEDG) => subida
                                ; - Incremento de TMR0 (T0CS) => clock
                                ; interno
                                ; - Aplicação do prescaler (PSA) => WDT
                                ; - Valor do prescaler: 1:1

Del10_1
        BTFSS   INTCON, 2       ; se o bit 2 do registrador de interrupções
                                ; (T0IF) estiver setado, significa que ocorreu
                                ; um estouro do TMR0. Então, a variável
                                ; auxiliar TEMPO1 (que inicialmente tem o
                                ; valor 100), é decrementada. Quando esta
                                ; variável chegar a zero, ou seja, quando
                                ; ocorrerem 100 vezes a chamada da rotina
```

```
                                ; Del10 (delay de 10 ms), a rotina retorna, pois
                                ; o delay já terá atingido 1 segundo. Caso
                                ; contrário, o programa fica em loop até
                                ; ocorrer a interrução de estouro do TMR0
GOTO        Del10_1             ; volta para verificar se ocorreu interrução
DECFSZ      TEMPO1, 1           ; decrementa a variável auxiliar TEMPO1
GOTO        Del10               ; delay de 10ms
RETURN                          ; retornando

END                             ; fim do programa
```

Programa 5:

Este programa conta as transições (mudanças) de uma chave C, ligada ao pino RA0 da porta A. A quantidade de transições é mostrada por 8 LEDs ligados aos pinos da porta B, em formato binário

Diagrama Esquemático:



Codificação:

Definição do tipo de dispositivo que será utilizado. Corresponde ao tipo do microcontrolador no qual iremos gravar o programa

```
LIST P=16F84
```

Definição dos comandos de usuário para alteração das páginas de memória (bancos)

```
#DEFINE BANCO0   BCF STATUS, RP0
#DEFINE BANCO1   BSF STATUS, RP0
```

Definições das constantes utilizadas para acessar os registradores do microcontrolador e variáveis auxiliares utilizadas no programa

```
STATUS          EQU   0x03
PORTA           EQU   0x05
PORTB           EQU   0x06
RP0             EQU   0x05
OPTION_REG      EQU   0x01
INTCON          EQU   0x0B
CONTADOR        EQU   0x0D
```

Definições do endereço de início de processamento, ou vetor de reset (0x00), e do vetor de interrupções (0x05)

```
ORG   0
GOTO Inicio
ORG   5
```

Início do programa:

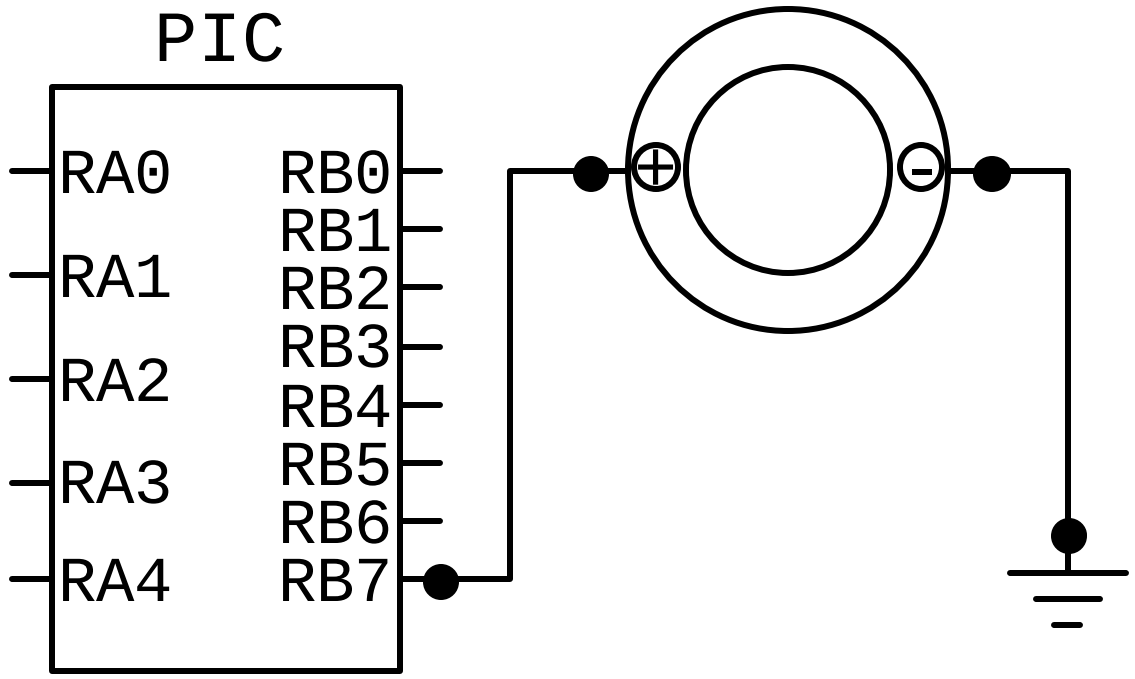
```
Inicio:
        BANCO1                  ; mudança para o banco de memória 1
        MOVLW   0x01            ; utilizando o registrador Work (W) para
                                ; configurar a porta de entrada A. O primeiro
                                ; bit, que corresponde ao pino RA0 está sendo
                                ; configurado como entrada, e o restante dos
                                ; pinos desta porta como saída
        MOVWF   PORTA           ; concluindo a configuração da porta A
        CLRF    PORTB           ; limpando os bits de PORTB. Com isso, todos
                                ; os pinos (RB0 a RB7) serão utilizados como
                                ; saída
        MOVLW   B'00000111'     ; utilizando o registrador Work (W) para
                                ; configurar o registrador de operação do
                                ; microcontrolador. Os três primeiros bits, que
                                ; correspondem àconfiguração do prescaler,
                                ; estão sendo setados, o que significa que o
                                ; prescaler assumirá o valor de 1:256
        MOVWF   OPTION_REG      ; concluindo a configuração do registrador de
                                ; operações
        BANCO0                  ; mudando para o banco de memória 0
        CLRF    PORTB           ; limpando a porta B
        CLRF    CONTADOR        ; limpando a variável auxiliar CONTADOR

Loop:
        BTFSS   PORTA, 0        ; testando o estado da chave C. Enquanto ela
                                ; estiver desligada (valor 0), o programa fica
                                ; em loop. Quando ela for ligada, é introduzido
                                ; um pequeno delay (rotina Anti_Ruídos), para
                                ; evitar interferências entre o acionamento da
                                ; chave C e o acendimento dos LEDs, e o
                                ; programa passa a esperar que a chave C
                                ; seja desligada
        GOTO    Loop            ; espera até a chave C ser ligada
        CALL    Anti_Ruidos     ; quando a chave C for ligada, chama a rotina
                                ; Anti_Ruídos

Loop2:
        BTFSC   PORTA, 0        ; testando o estado da chave C. Enquanto ela
                                ; estiver ligada (valor 1), o programa fica em
                                ; loop. Quando ela for desligada, a rotina
                                ; Anti_Ruídos é chamada, e o programa passa
                                ; a exibir o valor do contador nos LEDs
        GOTO    Loop2           ; espera até a chave C ser desligada
        CALL    Anti_Ruidos     ; quando a chave C for desligada, chama a
                                ; rotina Anti_Ruídos

        INCF    CONTADOR, F     ; neste ponto, uma transição ocorreu. Então, a
                                ; variável auxiliar CONTADOR, que grava a
```

```
                                    ; quantidade de transições feitas na chave C, é
                                    ; incrementada.
        MOVF    CONTADOR, W         ; o valor da variável CONTADOR é gravada no
                                    ; registrador Work (W)
        MOVWF   PORTB               ; o valor de transições é exibido nos LEDs

        GOTO    Loop                ; volta a esperar outra transição

Anti_Ruidos
        BCF     INTCON, 2           ; limpando o bit 2 do vetor de configuração de
                                    ; interrupções. Isto é necessário para que a
                                    ; rotina funcione corretamente, pois se este bit
                                    ; estiver em 1, o delay não será aplicado
        MOVLW   0xB0                ; movendo o valor B0h (10110000 em binário)
                                    ; para o registrador Work (W)
        MOVWF   OPTION_REG          ; modificando as configurações do registrador
                                    ; de operações. As opções escolhidas são:
                                    ; - Pull-Ups Internos (/RBPU) => habilitados
                                    ; - Incremento de TMR0 (T0CS) => externo,
                                    ; pela mudança do pino RA4/T0CKI
                                    ; Borda do incremento => descida
Del:
        BTFSS   INTCON, 2           ; verificação da ocorrência da interrupção de
                                    ; estouro de timer  (TMR0)
        GOTO    Del                 ; espera a ocorrência da interrupção
        RETURN                      ; retornando

        END                         ; fim do programa
```

Programa 6:

Este programa faz com que um speaker, ligado ao pino RB7 da porta B, dê *beeps* periódicos a cada segundo. Cada *beep* dura cerca de 0,5s. Como o ciclo de clock do microcontrolador é muito pequeno, é necessário introduzir *delays* para que os *beeps* do speaker possam ser ouvidos claramente. Estes *delays* funcionam da seguinte maneira: duas rotinas, uma chamada Delay1S (delay de 1 segundo) e outra, DelayMS (delay de 0,5 segundos), utilizam a subrotina Del10, que gera *delays* de 10 ms a cada chamada. Dentro desta rotina existe uma variável auxiliar (TEMPO1), que, para a rotina Delay1S, tem o valor inicial 100 e, para a rotina DelayMS tem o valor inicial de 50. e que vai sendo decrementada. Cada vez que ela é decrementada, a rotina Del10 é chamada novamente.

Diagrama Esquemático:



Codificação:

Definição do tipo de dispositivo que será utilizado. Corresponde ao tipo do microcontrolador no qual iremos gravar o programa

```
LIST P=16F84
```

Definição dos comandos de usuário para alteração das páginas de memória (bancos)

```
#DEFINE BANCO0  BCF STATUS, RP0
#DEFINE BANCO1  BSF STATUS, RP0
```

Definições das constantes utilizadas para acessar os registradores do microcontrolador e variáveis auxiliares utilizadas no programa

```
STATUS          EQU   0x03
PORTA           EQU   0x05
PORTB           EQU   0x06
RP0             EQU   0x05
OPTION_REG      EQU   0x01
INTCON          EQU   0x0B
TEMPO1          EQU   0x0C
```

Definições do endereço de início de processamento, ou vetor de reset (0x00), e do vetor de interrupções (0x05)

```
        ORG   0
        GOTO Inicio
        ORG   5
```

Início do programa:

```
Inicio:
        BANCO1                  ; mudança para o banco de memória 1
        MOVLW   B'01111111'     ; utilizando o registrador Work (W) para
                                ; configurar a porta B. Todos os pinos desta
                                ; porta são configurados como entrada, exceto
                                ; o pino RB7, que funcionará como saída
        MOVWF   PORTB           ; concluindo a configuração da porta B
        MOVLW   B'00000111'     ; utilizando o registrador Work (W) para
                                ; configurar o registrador de operação do
                                ; microcontrolador. Os três primeiros bits, que
                                ; correspondem àconfiguração do prescaler,
                                ; estão sendo setados, o que significa que o
                                ; prescaler assumirá o valor de 1:256.
        MOVWF   OPTION_REG      ; concluindo a configuração do registrador de
                                ; operações
        BANCO0                  ; mudando para o banco de memória 0
        CLRF    PORTB           ; limpando a porta B

Loop:
        BSF     PORTB, 7        ; setando o pino RB0 da porta B. Isto faz com
                                ; que o speaker comece a produzir um som
        CALL    DelayMS         ; chamando a rotina que introduz um delay de
                                ; 0,5 segundos no processamento, fazendo
                                ; com  que o speaker fique ligado por cerca de
                                ; 0,5 segundos
        BCF     PORTB, 7        ; limpando o pino RB0 da porta B. Isto faz com
                                ; que o speaker pare de produzir o som
        CALL    Delay1S         ; chamando a rotina que introduz um delay de
                                ; 1 segundo no processamento, fazendo com
                                ; que o speaker fique desligado por cerca de
                                ; 1 segundo
```

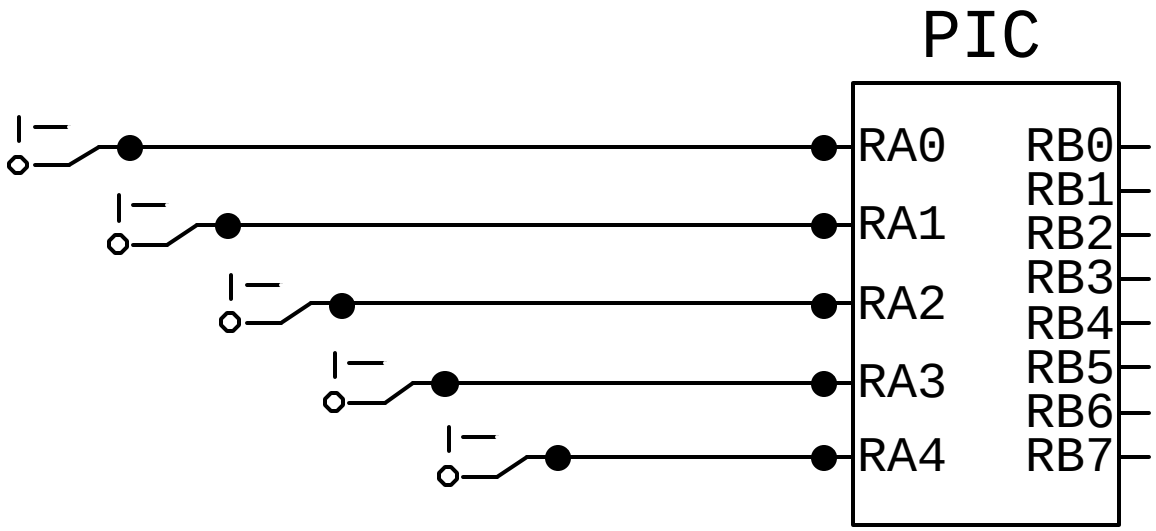
```
        GOTO    Loop            ; volta para o laço principal

Delay1S
        BANCO0                  ; mudando para o banco de memória 0
        MOVLW   .100            ; colocando o valor 100 (em decimal) no
                                ; registrador Work (W)
        MOVWF   TEMPO1          ; movendo o valor anterior para a variável
                                ; temporária TEMPO1, que será utilizada na
                                ; rotina Del10 (delay de 10ms)
        CALL    Del10           ; chamando a rotina Del10
        RETURN                  ; retornando

DelayMS
        BANCO0                  ; mudando para o banco de memória 0
        MOVLW   .50             ; colocando o valor 50 (em decimal) no
                                ; registrador Work (W)
        MOVWF   TEMPO1          ; movendo o valor anterior para a variável
                                ; temporária TEMPO1, que será utilizada na
                                ; rotina Del10 (delay de 10ms)
        CALL    Del10           ; chamando a rotina Del10
        RETURN                  ; retornando

Del10
        BCF     INTCON, 2       ; setando o bit 2 do registrador de
                                ; configuração de interrupções. Isto faz com
                                ; que a interrupção de estouro de timer
                                ; (registrador TMR0) seja habilitada
        MOVLW   0xD8            ; colocando o valor D8h (11011000 em binário)
                                ; no registrador Work (W)
        MOVWF   OPTION_REG      ; movendo o valor anterior para o registrador
                                ; de operações. Isto faz com que os bits do
                                ; registrador sejam configurados da seguinte
                                ; maneira:
                                ; - Pull-Ups Internos (/RBPU) => habilitados
                                ; - Borda que irá gerar a interrupção externa
                                ; no RB0 (INTEDG) => subida
                                ; - Incremento de TMR0 (T0CS) => clock
                                ; interno
                                ; - Aplicação do prescaler (PSA) => WDT
                                ; - Valor do prescaler: 1:1

Del10_1
        BTFSS   INTCON, 2       ; se o bit 2 do registrador de interrupções
                                ; (T0IF) estiver setado, significa que ocorreu
                                ; um estouro do TMR0. Então, a variável
                                ; auxiliar TEMPO1 (que inicialmente tem o
                                ; valor 100 ou 50), é decrementada. Quando
                                ; esta variável chegar a zero, ou seja, quando
                                ; ocorrerem 100 (ou 50) vezes a chamada da
                                ; rotina Del10 (delay de 10 ms), a rotina
                                ; retorna, pois o delay já terá atingido 1
```

```
                                ; segundo (ou 0,5 segundos). Caso contrário,
                                ; o programa fica em loop até ocorrer a
                                ; interrução de estouro do TMR0
GOTO        Del10_1             ; volta para verificar se ocorreu interrução
DECFSZ      TEMPO1, 1           ; decrementa a variável auxiliar TEMPO1
GOTO        Del10               ; delay de 10ms
RETURN                          ; retornando

END                             ; fim do programa;
```

Programa 7:

Este programa escreve dados na memória EEPROM do microcontrolador. A posição 0 desta memória irá armazenar o valor das chaves ligadas aos pinos RA0 àRA4 da porta A (em binário). A verificação do valor armazenado na memória pode ser visualizado pelo programa de gravação do PIC, já que esta memória não é apagada até ser subscrita. A gravação da memória EEPROM leva um certo tempo. Devemos esperar este tempo antes de continuar a execução do programa. Temos duas opções: ou esperamos que o bit WR do registrador EECON1 seja limpado pelo hardware, indicando o fim da escrita, ou introduzirmos um delay, que espera uma determinada quantidade de tempo antes de continuar a execução. Esta implementação utiliza um delay de, aproximadamente, 2,6 segundos, para a espera da gravação da memória EEPROM.

Diagrama Esquemático:



Codificação:

Definição do tipo de dispositivo que será utilizado. Corresponde ao tipo do microcontrolador no qual iremos gravar o programa

```
LIST P=16F84
```

Definição dos comandos de usuário para alteração das páginas de memória (bancos)

```
#DEFINE BANCO0    BCF STATUS, RP0
#DEFINE BANCO1    BSF STATUS, RP0
```

Definições das constantes utilizadas para acessar os registradores do microcontrolador e variáveis auxiliares utilizadas no programa

```
STATUS              EQU   0x03
```

```
PORTA           EQU   0x05
PORTB           EQU   0x06
RP0             EQU   0x05
OPTION_REG      EQU   0x01
INTCON          EQU   0x0B
EECON1          EQU   0x88
EECON2          EQU   0x89
EEDATA          EQU   0x08
EEADDR          EQU   0x09
TEMPO1          EQU   0x0D
RP0B            EQU   5
EEIE            EQU   6
WREN            EQU   2
WR              EQU   1
RD              EQU   0
```

Definições do endereço de início de processamento, ou vetor de reset (0x00), e do vetor de interrupções (0x05)

```
        ORG   0
        GOTO Inicio
        ORG   5
```

Início do programa:

```
Inicio:
        BANCO1                  ; mudando para o banco de memória 1
        MOVLW     B'00011111'   ; utilizando o registrador Work (W) para
                                ; configurar a porta de entrada A. Todos os
                                ; pinos da porta (RA0 – RA4) estão sendo
                                ; configurados como entrada
        MOVWF     PORTA         ; concluindo a configuração da porta A
        MOVLW     B'00000111'   ; utilizando o registrador Work (W) para
                                ; configurar o registrador de operação do
                                ; microcontrolador. Os três primeiros bits, que
                                ; correspondem àconfiguração do prescaler,
                                ; estão sendo setados, o que significa que o
                                ; prescaler assumirá o valor de 1:256.
        MOVWF     OPTION_REG    ; concluindo a configuração do registrador de
                                ; operações
        BANCO0                  ; mudando para o banco de memória 0
        CALL      Dado          ; rotina que obtém o dado a ser escrito
        CALL      Write         ; rotina que escreve o dado na EEPROM
        GOTO      Fim           ; finaliza programa

Dado
        BANCO0                  ; mudando para o banco de memória 0
        MOVLW     B'00000000'   ; colocando o número 0 no registrador Work
        MOVWF     EEADDR        ; movendo o valor 0 para o registrador de
                                ; endereços da EEPROM. Este corresponde
                                ; ao endereço da memória EEPROM no qual o
```

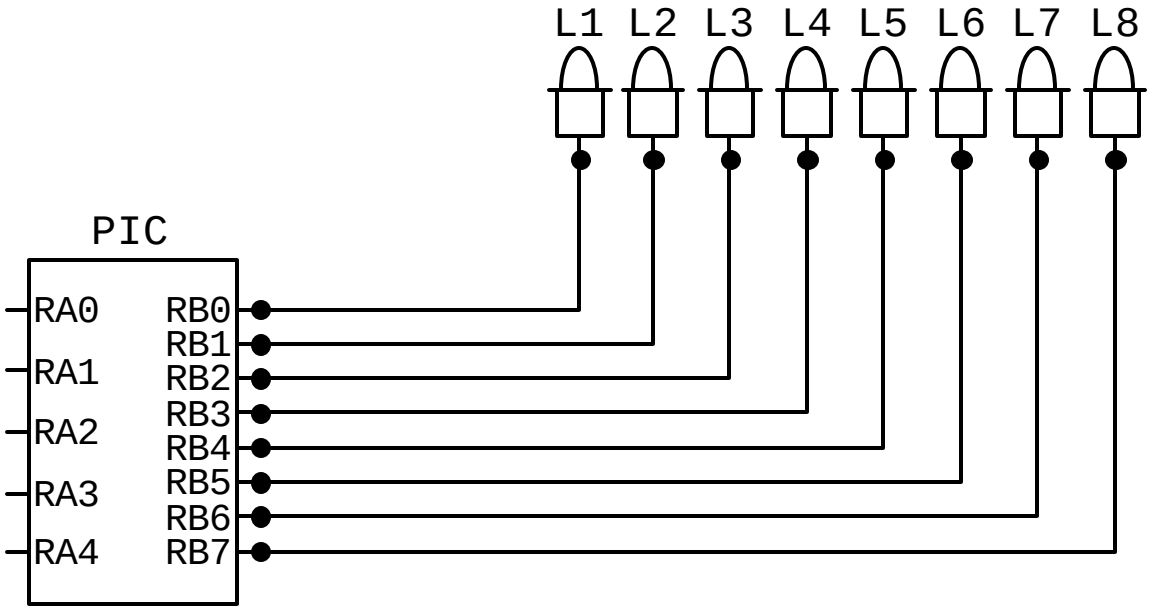
```
                                    ; dado será armazenado
        MOVF    PORTA, W            ; colocando o valor das chaves de entrada
                                    ; ligadas na porta A no registrador Work (W),
                                    ; que será o valor em binário armazenado no
                                    ; endereço 0 da EEPROM
        MOVWF   EEDATA              ; colocando o valor de W no registrador
                                    ; EEDATA, que serve para armazenar o dado
                                    ; que será gravado na memória EEPROM
        RETURN                      ; retornando

Write
        BANCO1                      ; mudando para o banco de memória 1
        BSF     EECON1, WREN        ; setando o bit WREN do registrador de
                                    ; configurações EECON1. Este bit habilita a
                                    ; escrita na memória, e é utilizado como bit de
                                    ; segurança
        MOVLW   0x55                ; também utilizado por questões de proteção,
                                    ; evitando que a memória EEPROM seja
                                    ; subscrita acidentalmente, o registrador
                                    ; EECON2 deve, primeiramente, receber o
                                    ; valor 0x55, e, posteriormente, o valor 0xAA.
                                    ; Este procedimento é obrigatório em todos os
                                    ; processos de escrita na memória EEPROM
        MOVWF   EECON2
        MOVLW   0xAA
        MOVWF   EECON2
        BSF     EECON1, WR          ; setando o bit WR do registrador de
                                    ; configurações EECON1. Quanto este bit é
                                    ; setado, o ciclo de escrita é iniciado. Quando
                                    ; terminada a escrita, o hardware,
                                    ; automaticamente, limpa este bit.
        BANCO0                      ; mudando para o banco de memória 0
        CALL    Del10               ; espera a memória ser escrita
        RETURN                      ; retornando

Del10
        BCF     INTCON, 2           ; setando o bit 2 do registrador de
                                    ; configuração de interrupções. Isto faz com
                                    ; que a interrupção de estouro de timer
                                    ; (registrador TMR0) seja habilitada
        MOVLW   0xD8                ; colocando o valor D8h (11011000 em binário)
                                    ; no registrador Work (W)
        MOVWF   OPTION_REG          ; movendo o valor anterior para o registrador
                                    ; de operações. Isto faz com que os bits do
                                    ; registrador sejam configurados da seguinte
                                    ; maneira:
                                    ; - Pull-Ups Internos (/RBPU) => habilitados
                                    ; - Borda que irá gerar a interrupção externa
                                    ; no RB0 (INTEDG) => subida
                                    ; - Incremento de TMR0 (T0CS) => clock
                                    ; interno
```

```
                                    ; - Aplicação do prescaler (PSA) => WDT
                                    ; - Valor do prescaler: 1:1

Del10_1
        BTFSS       INTCON, 2       ; se o bit 2 do registrador de interrupções
                                    ; (T0IF) estiver setado, significa que ocorreu
                                    ; um estouro do TMR0. Então, a variável
                                    ; auxiliar TEMPO1 (que inicialmente tem o
                                    ; valor 0), é decrementada. Como o valor
                                    ; inicial desta variável é 0, quando
                                    ; decrementada pela primeira vez, ela passa a
                                    ; ter o valor 255. Quando esta variável chegar
                                    ; a zero, ou seja, quando ocorrerem 256 vezes
                                    ; a chamada da rotina Del10 (delay de 10 ms),
                                    ; a rotina retorna, pois o delay já terá
                                    ; terminado. Caso contrário, o programa fica
                                    ; em loop até ocorrer a interrução de estouro
                                    ; do TMR0
        GOTO        Del10_1         ; volta para verificar se ocorreu interrução
        DECFSZ      TEMPO1, 1       ; decrementa a variável auxiliar TEMPO1
        GOTO        Del10           ; delay de 10ms
        RETURN                      ; retornando


Fim:
        GOTO        Fim             ; loop infinito. Para começar o programa
                                    ; novamente, depois de uma escrita, é
                                    ; necessário resetar o microcontrolador

        END                         ; fim do programa
```

Programa 8:

Este programa lê os dados anteriormente gravados na memória EEPROM, na posição 0. O valor lido é representado, em binário, através de LEDs ligados aos pinos da porta B.

Diagrama Esquemático:



Codificação:

Definição do tipo de dispositivo que será utilizado. Corresponde ao tipo do microcontrolador no qual iremos gravar o programa

```
LIST P=16F84
```

Definição dos comandos de usuário para alteração das páginas de memória (bancos)

```
#DEFINE BANCO0   BCF STATUS, RP0
#DEFINE BANCO1   BSF STATUS, RP0
```

Definições das constantes utilizadas para acessar os registradores do microcontrolador e variáveis auxiliares utilizadas no programa

```
STATUS          EQU   0x03
PORTA           EQU   0x05
PORTB           EQU   0x06
RP0             EQU   0x05
OPTION_REG      EQU   0x01
INTCON          EQU   0x0B

EECON1          EQU   0x88
```

```
EEDATA          EQU   0x08
EEADDR          EQU   0x09

RD              EQU   0
```

Definições do endereço de início de processamento, ou vetor de reset (0x00), e do vetor de interrupções (0x05)

```
        ORG   0
        GOTO Inicio
        ORG   5
```

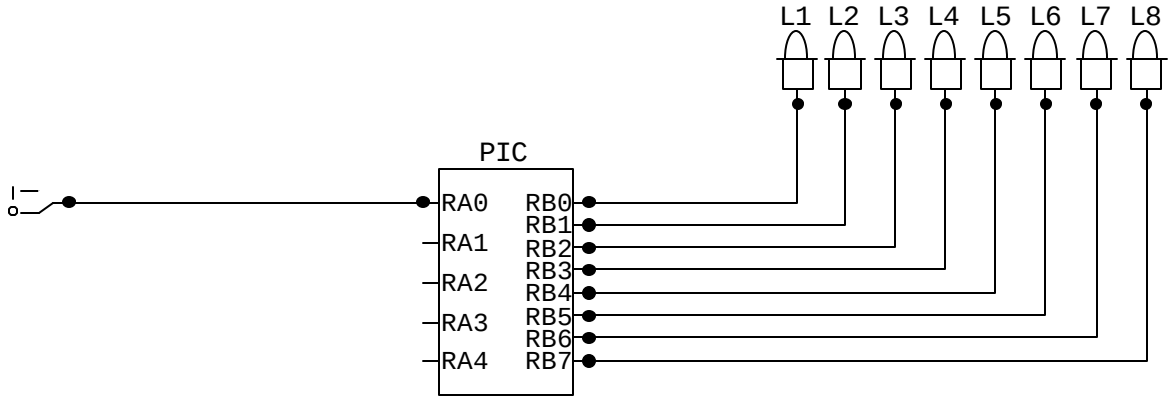
Início do programa:

```
Inicio:
        BANCO1                  ; mudando para o banco de memória 1
        CLRF    PORTB           ; limpando os bits de PORTB. Com isso, todos
                                ; os pinos (RB0 a RB7) serão utilizados como
                                ; saída
        BANCO0                  ; mudando para o banco de memória 0
        CLRF    EEADDR          ; limpando os bits do registrador de endereços
                                ; EEADDR. Isso faz com que o endereço a ser
                                ; lido seja o endereço 0
        CALL    Read            ; chama a rotina de leitura da memória
        MOVF    EEDATA, W       ; move os dados lidos da memória para o
                                ; registrador Work (W)
        MOVWF   PORTB           ; exibe o valor lido da memória, em binário,
                                ; nos LEDs ligados aos pinos da porta B
        GOTO    Fim             ; finaliza o programa

Read
        BANCO1                  ; mudando para o banco de memória 1
        BSF     EECON1, RD      ; setando o bit RD do registrador de
                                ; configurações EECON1. Quanto este bit é
                                ; setado, o ciclo de leitura é iniciado. Quando
                                ; terminada a leitura, o hardware,
                                ; automaticamente, limpa este bit. Como a
                                ; leitura da memória é muito rápida, ou seja, o
                                ; dado é lido quase imediatamente após o bit
                                ; de início do ciclo de leitura ser setado, não é
                                ; necessário introduzir delays para a leitura, ou
                                ; esperar que o bit RD seja limpado pelo
                                ; hardware
        BANCO0                  ; mudando para o banco de memória 0
        RETURN                  ; retornando

Fim:
        GOTO    Fim             ; loop infinito. Para começar o programa
                                ; novamente, depois de uma escrita, é
                                ; necessário resetar o microcontrolador
        END                     ; fim do programa
```

Programa 9:

Este programa é similar ao programa 5, mas explora os recursos de WatchDog e modo Sleep do microcontrolador. A partir deste programa, passaremos a utilizar o arquivo de definições padrão da Microchip para microcontroladores modelo 16F84, onde se encontra as definições dos nomes e endereços de todos os SFRs (registradores especiais) e uma série de outras definições necessárias para a utilização do microcontrolador 16F84. Como curiosidade, você pode consultar este arquivo para visualizar todos os nomes dos SFRs e constantes que podem ser utilizadas na programação do 16F84.

Diagrama Esquemático:



Codificação:

Definição do tipo de dispositivo que será utilizado. Corresponde ao tipo do microcontrolador no qual iremos gravar o programa

```
LIST P=16F84
```

Inclusão do arquivo de definições da Microchip para o modelo 16F84a

```
INCLUDE "P16F84A.INC"
```

Definição dos comandos de usuário para alteração das páginas de memória (bancos)

```
#DEFINE BANCO0  BCF STATUS, RP0
#DEFINE BANCO1  BSF STATUS, RP0
```

Definições das variáveis auxiliares utilizadas no programa

```
CONTADOR        EQU  0x0C
```

Definições do endereço de início de processamento, ou vetor de reset (0x00), e do vetor de interrupções (0x05)

```
ORG  0
GOTO Inicio
```

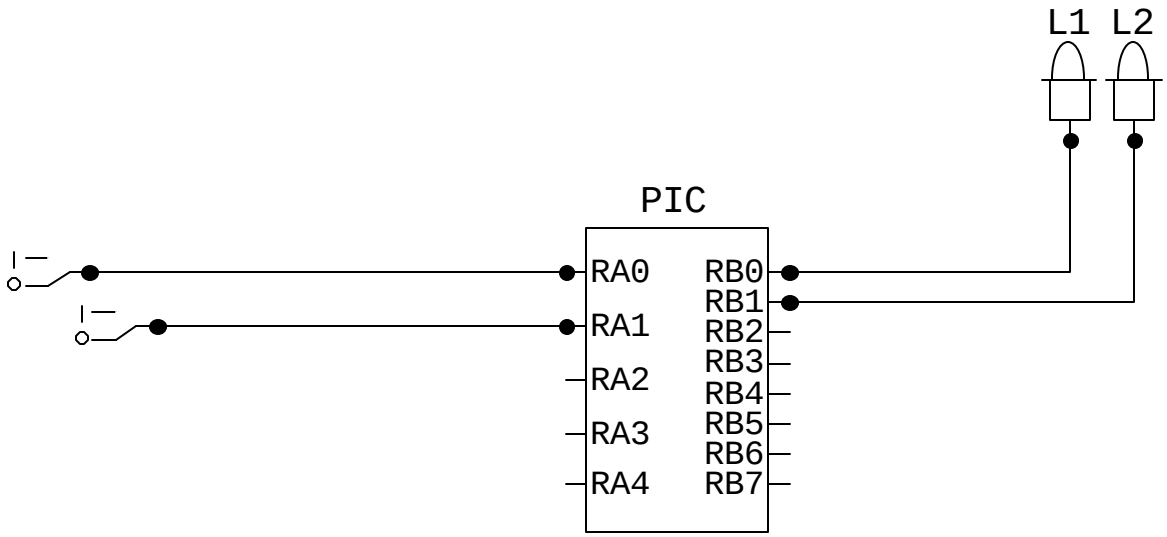
```
        ORG   5

Início do programa:
Inicio:
        BANCO1                  ; mudança para o banco de memória 1
        MOVLW   0x01            ; utilizando o registrador Work (W) para
                                ; configurar a porta de entrada A. O primeiro
                                ; bit, que corresponde ao pino RA0 está sendo
                                ; configurado como entrada, e o restante dos
                                ; pinos desta porta como saída
        MOVWF   PORTA           ; concluindo a configuração da porta A
        CLRF    PORTB           ; limpando os bits de PORTB. Com isso, todos
                                ; os pinos (RB0 a RB7) serão utilizados como
                                ; saída
        MOVLW   B'00001111'     ; utilizando o registrador Work (W) para
                                ; configurar o registrador de operação do
                                ; microcontrolador. Os três primeiros bits, que
                                ; correspondem àconfiguração do prescaler,
                                ; estão sendo setados, o que significa que o
                                ; prescaler assumirá o valor de 1:128, pois,
                                ; como o quarto bit está setado, significa que o
                                ; prescaler será aplicado ao WDT (WatchDog)
                                ; Como o prescaler está acertado para 1:128,
                                ; o WDT irá demorar 18 * 128 = 2,3 segundos
                                ; para estourar. Note que 256 incrementos do
                                ; WDT correspondem a 18 ms
        MOVWF   OPTION_REG      ; concluindo a configuração do registrador de
                                ; operações
        BANCO0                  ; mudando para o banco de memória 0
        CLRF    PORTB           ; limpando a porta B
        CLRF    CONTADOR        ; limpando a variável auxiliar CONTADOR

Loop:
        BTFSS   PORTA, 0        ; testando o estado da chave C. Enquanto ela
                                ; estiver desligada (valor 0), o programa fica
                                ; em loop. Quando ela for ligada, é introduzido
                                ; um delay, através do WDT, para evitar
                                ; interferências entre o acionamento da chave
                                ; C e o acendimento dos LEDs, e o programa
                                ; passa a esperar que a chave C seja
                                ; desligada
        GOTO    Loop            ; espera até a chave C ser ligada
        SLEEP                   ; espera o WDT estourar (cerca de 2,3
                                ; segundos). Neste período de tempo, o
                                ; oscilador é desligado e o processamento é
                                ; paralisado
Loop2:
        BTFSC   PORTA, 0        ; testando o estado da chave C. Enquanto ela
                                ; estiver ligada (valor 1), o programa fica em
                                ; loop. Quando ela for desligada, o delay é
                                ; introduzido, e o programa passa a exibir o
```

```
                                    ; valor do contador nos LEDs
GOTO        Loop2                   ; espera até a chave C ser desligada
SLEEP                               ; espera o WDT estourar (cerca de 2,3
                                    ; segundos). Neste período de tempo, o
                                    ; oscilador é desligado e o processamento é
                                    ; paralisado
INCF        CONTADOR, F             ; neste ponto, uma transição ocorreu. Então, a
                                    ; variável auxiliar CONTADOR, que grava a
                                    ; quantidade de transições feitas na chave C, é
                                    ; incrementada.
MOVF        CONTADOR, W             ; o valor da variável CONTADOR é gravada no
                                    ; registrador Work (W)
MOVWF       PORTB                   ; o valor de transições é exibido nos LEDs

GOTO        Loop                    ; volta a esperar outra transição

END                                 ; fim do programa
```

Programa 10:

Este programa simula os movimentos de um robô. Este robô imaginário possui dois dispositivos infravermelhos, que fazem com que ele siga uma trajetória desenhada no chão, e dois motores, um para cada roda de tração. Quando os sensores detectam a necessidade de acertar a rota, os motores devem ser manipulados a fim de realizar o acerto. Por exemplo, se o robô estiver desviando a rota para a esquerda, o robô deve se mover para a direita até que a rota seja acertada. Para isto, o motor da direita deve ser revertido (rodar para trás), e o da esquerda deve continuar ligado (rodando para frente). Quando a rota estiver certa, ambos os motores devem ser ligados para frente, para que o robô continue andando em frente. Os movimentos possíveis são para a esquerda (reverter o motor da esquerda e manter o motor da direita a frente), para a direita (reverter o motor da direita e manter o motor da esquerda a frente), para frente (manter os dois motores a frente), e para trás (reverter os dois motores).
Os sensores serão simulados pelas chaves ligadas aos pinos RA0 e RA1 da porta. Vamos convencionar que o valor 0 significa que o sensor não detectou desvio na trajetória, e o valor 1 significa que o sensor detectou um desvio na trajetória. Os LEDs ligados aos pinos RB0 e RB1 da porta B irão simular as saídas de controle dos dois motores de tração (direito e esquerdo, respectivamente). Vamos convencionar que o valor 1 significa ligar os motores àfrente, e 0 significa reverter os motores.

Diagrama Esquemático:



Codificação:

Definição do tipo de dispositivo que será utilizado. Corresponde ao tipo do microcontrolador no qual iremos gravar o programa

```
LIST P=16F84
```

Inclusão do arquivo de definições da Microchip para o modelo 16F84a

```
        INCLUDE "P16F84A.INC"
```

Definição dos comandos de usuário para alteração das páginas de memória (bancos)

```
#DEFINE BANCO0    BCF STATUS, RP0
#DEFINE BANCO1    BSF STATUS, RP0
```

Definições das variáveis auxiliares utilizadas no programa

```
AUX    EQU    0x0C
```

Definições do endereço de início de processamento, ou vetor de reset (0x00), e do vetor de interrupções (0x05)

```
        ORG   0
        GOTO Inicio
        ORG   5
```

Início do programa:

```
Inicio:
        BANCO1                  ; mudando para o banco de memória 1
        MOVLW     B'00000011'   ; utilizando o registrador Work (W) para
                                ; configurar a porta de entrada A. O dois
                                ; primeiros bits, que correspondem aos pinos
                                ; RA0 e RA1 estão sendo configurados como
                                ; entrada, e o restante dos pinos desta porta
                                ; como saída
        MOVWF     PORTA         ; concluindo a configuração da porta A
        CLRF      PORTB         ; limpando os bits de PORTB. Com isso, todos
                                ; os pinos (RB0 a RB7) serão utilizados como
                                ; saída
        BANCO0                  ; mudando para o banco de memória 0

Loop:
        MOVF      PORTA, W      ; movendo o status atual da porta A para o
                                ; registrador Work (W)
        ANDLW     B'00000011'   ; AND lógico entre o registrador Work (W) e o
                                ; valor “00000011”, para que os 2 primeiros
                                ; bits sejam mantidos e os outros 6 passem a
                                ; ser 0. Isto é necessário para realizar as
                                ; comparações
        MOVWF     AUX           ; coloca o valor da operação anterior na
                                ; variável auxiliar AUX
        XORLW     B'00000000'   ; verifica se a trajetória está correta (valores
                                ; dos sensores = 0. Se estiver correta, mantém
                                ; o movimento àfrente
        BTFSC     STATUS, Z     ; verifica se a operação anterior resultou em
                                ; zero. Se o bit Z do registrador STATUS
                                ; estiver setado, significa que a operação
                                ; resultou em zero. Isto significa que ambos os
```

```
                                    ; sensores estão com os valores 0, e a
                                    ; trajetória está correta. Se isto ocorrer, o robô
                                    ; mantém o movimento á frente. Caso
                                    ; contrário, a próxima linha é pulada e outras
                                    ; verificações serão feitas
        GOTO        Para_Frente     ; faz o robô movimentar-se para frente

        MOVF        AUX, W          ; faz uma cópia do valor da variável AUX no
                                    ; registrador Work (W) para comparação
        XORLW       B'00000001'     ; verifica se a trajetória desviou para a
                                    ; esquerda
        BTFSC       STATUS, Z       ; se a operação anterior resultou em zero,
                                    ; significa que a trajetória deve ser acertada,
                                    ; ou seja, o robô deve movimentar-se para a
                                    ; direita até que retome a trajetória correta
        GOTO        Direita         ; faz o robô movimentar-se para a direita

        MOVF        AUX, W          ; faz uma cópia do valor da variável AUX no
                                    ; registrador Work (W) para comparação
        XORLW       B'00000010'     ; verifica se a trajetória desviou para a
                                    ; direita
        BTFSC       STATUS, Z       ; se a operação anterior resultou em zero,
                                    ; significa que a trajetória deve ser acertada,
                                    ; ou seja, o robô deve movimentar-se para a
                                    ; esquerda até que retome a trajetória correta
        GOTO        Esquerda        ; faz o robô movimentar-se para a direita

        MOVF        AUX, W          ; faz uma cópia do valor da variável AUX no
                                    ; registrador Work (W) para comparação
        XORLW       B'00000011'     ; verifica se o robô saiu totalmente da trajetória
                                    ; (isto ocorre quando ambos os sensores
                                    ; passam a ter o valor 1)
        BTFSC       STATUS, Z       ; se a operação anterior resultou em zero,
                                    ; significa que a trajetória deve ser acertada,
                                    ; ou seja, o robô deve movimentar-se para trás
                                    ; até que retome a trajetória correta
        GOTO        Para_Tras       ; faz o robô movimentar-se para trás

        GOTO        Loop            ; volta para o loop principal

Para_Frente:
        MOVLW       B'00000011'     ; os dois motores serão ligados àfrente
        MOVWF       PORTB           ; liga os motores
        GOTO        Loop            ; volta para o loop principal

Para_Tras:
        MOVLW   B'00000000'         ; os dois motores serão revertidos
        MOVWF   PORTB               ; reverte os motores
        GOTO    Loop                ; volta para o loop principal

Direita:
```

```
        MOVLW   B'00000010'     ; o motor esquerdo será será ligado àfrente e
                                ; o direito, revertido
        MOVWF   PORTB           ; aciona os motores
        GOTO    Loop            ; volta para o loop principal

Esquerda:
        MOVLW   B'00000001'     ; o motor direito será será ligado àfrente e
                                ; o esquerdo, revertido
        MOVWF   PORTB           ; aciona os motores
        GOTO    Loop            ; volta para o loop principal

        END                     ; fim do programa
```

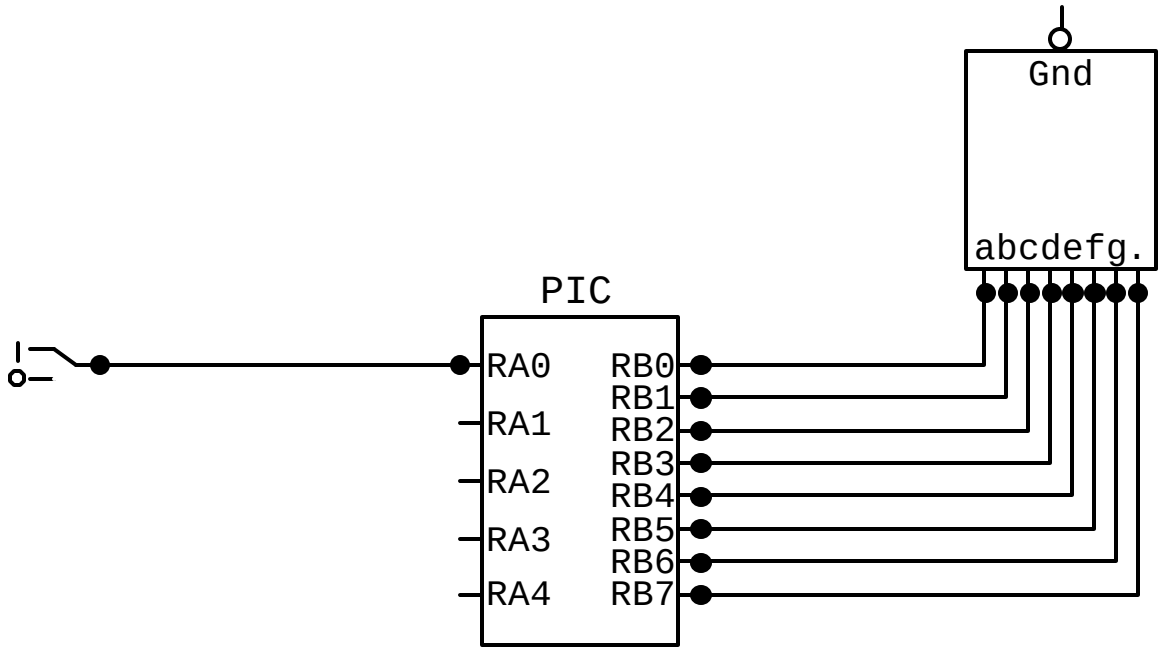
Programa 11:

Este programa servirá para a geração de números aleatórios. Irá funcionar da seguinte maneira:

- Devemos gerar números aleatórios de 0 a 6
- Quando a chave C, ligada ao pino RA0 da porta A, estiver em "1", o display (ligado aos pinos RB0 à RB7 da porta B) irá mostrar, sequencialmente, números de 0 a 6, em intervalos de 0,05 segundos
- Ao passar a chave C para "0", o display mostrará, durante 3 segundos, o número aleatório obtido
- Passados os 3 segundos, o display se apaga e a sequencia se repete

O número representado nos 4 bits menos significativos do registrador Work (W) é transformado em um número equivalente no display de 7 segmentos, como se estivéssemos implementando um conversor. O código referente ao número no display de 7 segmentos também retorna no registrador W.

Diagrama Esquemático:



Codificação:

Definição do tipo de dispositivo que será utilizado. Corresponde ao tipo do microcontrolador no qual iremos gravar o programa

```
LIST P=16F84
```

Inclusão do arquivo de definições da Microchip para o modelo 16F84a

```
        INCLUDE "P16F84A.INC"
```

Definição dos comandos de usuário para alteração das páginas de memória (bancos)

```
#DEFINE BANCO0    BCF STATUS, RP0
#DEFINE BANCO1    BSF STATUS, RP0
```

Definições das variáveis auxiliares utilizadas no programa

```
        CBLOCK  0x0C            ; início da área de memória do usuário

            NUMERO
            DELAY_CONT
            TEMPORAL

        ENDC
```

Definições do endereço de início de processamento, ou vetor de reset (0x00), e do vetor de interrupções (0x05)

```
        ORG   0
        GOTO Inicio
        ORG   5
```

Início do programa:

Tabela

```
        ADDWF       PCL, F          ; soma o valor do registrador W (que contém o
                                    ; número a ser convertido) com o valor do
                                    ; registrador PCL, para que o program counter
                                    ; seja desviado para o valor que corresponde
                                    ; ao número na tabela de conversão. O
                                    ; número convertido é retornado em W,
                                    ; através da chamada RETLW
        RETLW       B'00111111'     ; representação do número 0
        RETLW       B'00000110'     ; representação do número 1
        RETLW       B'01011011'     ; representação do número 2
        RETLW       B'01001111'     ; representação do número 3
        RETLW       B'01100110'     ; representação do número 4
        RETLW       B'01101101'     ; representação do número 5
        RETLW       B'01111101'     ; representação do número 6
```

Delay_20_ms:

```
        BCF         INTCON, T0IF    ; setando o bit T0IF do registrador de
                                    ; configuração de interrupções. Isto faz com
                                    ; que a interrupção de estouro de timer
                                    ; (registrador TMR0) seja habilitada
        MOVLW       0xB1            ; colocando o valor B1h (177 em decimal) no
                                    ; registrador Work (W)
```

```
        MOVWF   TMR0            ; movendo o valor 177 para o TMR0. Como o
                                ; prescaler foi setado para 1:256, o TMR0 será
                                ; incrementado a cada 256 ciclos. Com um
                                ; clock de 4 Mhz, isto significa que ocorre um
                                ; incremento a cada 256µs. Como o TMR0
                                ; começa em 177, após 79 incrementos ele irá
                                ; estourar. Para ocorrer 79 incrementos, vão
                                ; ser necessários 79 * 256 = 20224µs, ou,
                                ; aproximadamente, 20 ms, que é o delay que
                                ; desejamos

Delay_20_ms_1

        CLRWDT                  ; limpa o registrador WDT para não ocorrer
                                ; um reset
        BTFSS   INTCON, T0IF    ; se o bit T0IF do registrador de interrupções
                                ; estiver setado, significa que ocorreu um
                                ; estouro do TMR0. Ou seja, o delay de 20 ms
                                ; foi atingido, então a rotina retorna. Enquanto
                                ; não ocorrer a interrupção de estouro de
                                ; timer, o programa fica em loop, aguardando
                                ; que esta interrupção ocorra
        GOTO    Delay_20_ms_1   ; espera interrupção de estouro de timer
        RETURN                  ; retornando

Delay_Var

        BCF     INTCON, T0IF    ; setando o bit T0IF do registrador de
                                ; configuração de interrupções. Isto faz com
                                ; que a interrupção de estouro de timer
                                ; (registrador TMR0) seja habilitada
        MOVLW   0x3C            ; colocando o valor 3Ch (60 em decimal) no
                                ; registrador Work (W)
        MOVWF   TMR0            ; movendo o valor 60 para o TMR0. Como o
                                ; prescaler foi setado para 1:256, o TMR0 será
                                ; incrementado a cada 256 ciclos. Com um
                                ; clock de 4 Mhz, isto significa que ocorre um
                                ; incremento a cada 256µs. Como o TMR0
                                ; começa em 60, após 196 incrementos ele irá
                                ; estourar. Para ocorrer 196 incrementos, vão
                                ; ser necessários 196 * 256 = 50176 µs, ou,
                                ; aproximadamente, 50 ms.
        CLRWDT                  ; limpa o registrador WDT para não ocorrer
                                ; um reset

Intervalo

        CLRWDT                  ; limpa o registrador WDT para não ocorrer
                                ; um reset
        BTFSS   INTCON, T0IF    ; se o bit T0IF do registrador de interrupções
                                ; estiver setado, significa que ocorreu um
```

```
                                        ; estouro do TMR0. Ou seja, o delay de 50 ms
                                        ; foi atingido, então a variável DELAY_CONT,
                                        ; que contém a quantidade de vezes que a
                                        ; rotina irá repetir (ou seja, é um parâmetro
                                        ; que corresponde ao tempo desejado de
                                        ; delay, levando em conta que cada iteração
                                        ; da rotina demora 50 ms), é decrementada, e
                                        ; a rotina recomeça. Quando a variável
                                        ; DELAY_CONT chegar a zero, a rotina
                                        ; retorna, sendo que o tempo gasto nesta
                                        ; rotina corresponde ao delay desejado.
                                        ; Enquanto não ocorre uma interrupção de
                                        ; estouro de timer, a rotina permanece em loop
        GOTO    Intervalo               ; espera interrupção de estouro de timer
        DECFSZ  DELAY_CONT, F           ; decrementa a variável DELAY_CONT
        GOTO    Delay_Var               ; se DELAY_CONT não chegou a zero, volta a
                                        ; executar a rotina. Note que esta rotina
                                        ; contém traços de rotina recursiva. Então por
                                        ; que, ao invés de chamar a rotina usando
                                        ; CALL usamos GOTO? A resposta é simples,
                                        ; pois se executamos CALL, o endereço de
                                        ; retorno é armazenado na pilha (stack). Como
                                        ; a pilha possui apenas 8 níveis, não seria
                                        ; possível armazenar todos os endereços de
                                        ; retornos das várias chamadas àmesma
                                        ; rotina (por exemplo, se desejarmos um delay
                                        ; de 3 segundos, a rotina deve ser chamada
                                        ; 60 vezes). Isto acarretaria a perda dos
                                        ; endereços mais antigos da pilha, e o
                                        ; programa não funcionaria corretamente.
                                        ; Então, utilizando GOTO, o programa
                                        ; somente desvia a execução para a linha
                                        ; correspondente ao nome da rotina, não
                                        ; sendo necessário armazenar o endereço de
                                        ; retorno.
        RETURN                          ; retornando

Inicio:
        CLRF    PORTB                   ; limpando a porta B
        BANCO1                          ; mudando para o banco de memória 1
        CLRF    TRISB                   ; limpando os bits de TRISB. Com isso, todos
                                        ; os pinos (RB0 a RB7) serão utilizados como
                                        ; saída
        MOVLW   B'00011111'             ; utilizando o registrador Work (W) para
                                        ; configurar a porta de entrada A. Todos os
                                        ; pinos (RA0 – RA4) serão utilizados como
                                        ; entrada.
        MOVWF   TRISA                   ; concluindo a configuração da porta A
        MOVLW   B'00000111'             ; utilizando o registrador Work (W) para
                                        ; configurar o registrador de operação do
                                        ; microcontrolador. Os três primeiros bits, que
```

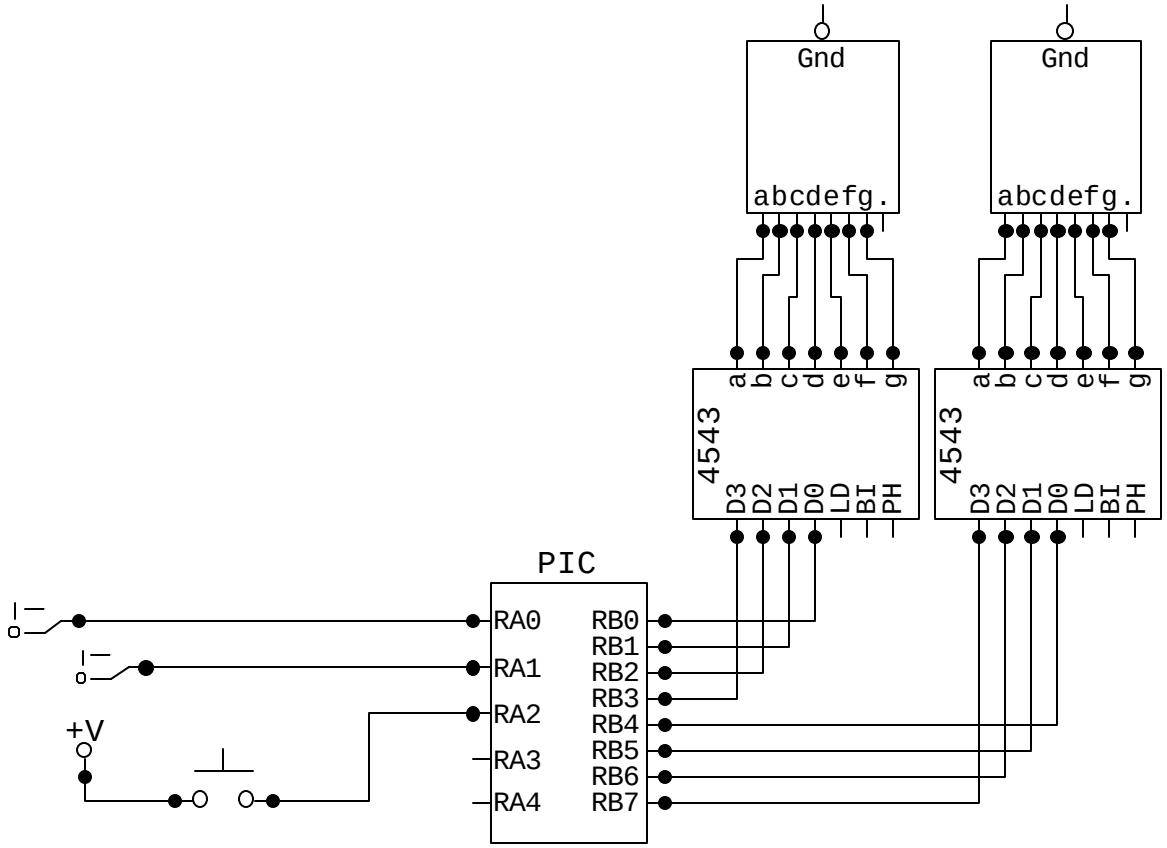
```
                                    ; correspondem àconfiguração do prescaler,
                                    ; estão sendo setados, o que significa que o
                                    ; prescaler assumirá o valor de 1:256
        MOVWF   OPTION_REG          ; concluindo a configuração do registrador de
                                    ; operação
        BANCO0                      ; mudando para o banco de memória 0

Loop:
        CLRWDT                      ; limpa o registrador WDT para não ocorrer
                                    ; um reset
        BTFSS   PORTA, 0            ; testa o status da chave C. Se estiver
                                    ; ligada, espera até que ela seja desligada.
        GOTO    Loop                ; espera a chave C ser desligada
        MOVF    TMR0, W             ; copia o valor corrente do timer (TMR0) para o
                                    ; registrador Work (W)
        MOVWF   NUMERO              ; move o valor de Work (W) para a variável
                                    ; NUMERO
Divide:
        MOVLW   D'6'                ; move o valor 6 para o registrador Work (W)
        SUBWF   NUMERO, W           ; subtrai o valor de W da variável NUMERO
        MOVWF   NUMERO              ; atualiza a variável NUMERO
        SUBLW   D'5'                ; subtrai o valor 5 do registrador W
        BTFSS   STATUS, C           ; testa a ocorrência do carry (negativo)
        GOTO    Divide              ; se não ocorreu carry, continua os cálculos
        INCF    NUMERO, F           ; se ocorreu carry, incrementa NUMERO

Dado:
        MOVLW   D'6'                ; move o valor 6 para o registrador Work (W)
        MOVWF   TEMPORAL            ; copia o conteúdo de W (valor 6) para a
                                    ; variável TEMPORAL

RA0_1:
        CLRWDT                      ; limpa o registrador WDT para não ocorrer
                                    ; um reset
        BTFSS   PORTA, 0            ; testa o status da chave C. Se estiver
                                    ; ligada, vai exibindo os números aleatórios no
                                    ; display, em intervalos de 0,05 segundos. Se
                                    ; estiver desligada, exibe no display o número
                                    ; aleatório gerados, por 3 segundos.
        GOTO    Saida               ; se a chave C estiver desligada, exibe o
                                    ; número aleatório gerado no display
        MOVF    TEMPORAL, W         ; move o número aleatório atual para o
                                    ; registrador Work (W)
        CALL    Tabela              ; converte este número para o formato de 7
                                    ; segmentos
        MOVWF   PORTB               ; exibe o número no display
        MOVLW   D'1'                ; move o valor 1 para o registrador Work (W)
        MOVWF   DELAY_CONT          ; seta o multiplicador da rotina Delay_Var para
                                    ; o valor 1. Ou seja, a rotina irá gerar um delay
                                    ; de 50 ms
        CALL    Delay_Var           ; chama a rotina Delay_Var
```

```
        DECFSZ      TEMPORAL, F     ; decrementa o valor do número aleatório atual
        GOTO        RA0_1           ; se TEMPORAL não chegou a zero, continua
                                    ; exibindo os números aleatórios no display
        GOTO        Dado            ; se TEMPORAL for zero, seu valor deve ser
                                    ; atualizado. No caso, no desvio para a linha
                                    ; Dado, ele voltará ao valor 6
        CALL        Delay_20_ms     ; delay de 20 ms

Saida:
        MOVF        NUMERO, W       ; move o numero aleatório gerado, a ser
                                    ; exibido no display, para o registrador W
        CALL        Tabela          ; converte este número para o formato de 7
                                    ; segmentos
        MOVWF       PORTB           ; exibe o número no display
        MOVLW       D'60'           ; move o valor 60 para o registrador Work (W)
        MOVWF       DELAY_CONT      ; seta o multiplicador da rotina Delay_Var para
                                    ; o valor 60. Ou seja, a rotina irá gerar um
                                    ; delay de 60 * 50 = 3000 ms, ou seja, 3
                                    ; segundos
        CALL        Delay_Var       ; chama a rotina Delay_Var
        CLRF        PORTB           ; apaga o conteúdo do display
        GOTO        Loop            ; retorna para o loop principal

        END                         ; fim do programa
```

Programa 12:

Este programa irá simular um contador de 2 dígitos (00 a 99). Para sua realização, serão utilizados dois displays de 7 segmentos, e dois conversores binário-7segmentos, que recebe um número binário de 4 bits e transforma este número para a representação do display.
Para exibir os números nos displays, utilizaremos a porta B, sendo que os 4 bits menos significativos representarão o número a ser exibido no primeiro display (unidade), e os 4 bits mais significativos representarão o número a ser exibido no segundo display (dezena).
Este contador terá duas chaves e um botão (push buttom). A chave ligada à entrada RA0 da porta A irá controlar o sentido da contagem (0 = Decrescente e 1 = Crescente). A chave ligada à entrada RA1 da porta A irá controlar a parada do contador (0 = Parar e 1 = Contar). E, por último, o botão ligado à RA2 servirá de reset, ou seja, quando pressionado, irá zerar os displays e recomeçar a contagem.

Diagrama Esquemático:



Codificação:

Definição do tipo de dispositivo que será utilizado. Corresponde ao tipo do microcontrolador no qual iremos gravar o programa

```
        LIST P=16F84
```

Inclusão do arquivo de definições da Microchip para o modelo 16F84a

```
        INCLUDE "P16F84A.INC"
```

Definição dos comandos de usuário para alteração das páginas de memória (bancos)

```
#DEFINE BANCO0    BCF STATUS, RP0
#DEFINE BANCO1    BSF STATUS, RP0
```

Definições das variáveis auxiliares utilizadas no programa

```
        CBLOCK  0x0C                ; início da área de memória do usuário

                CONTADOR_D1
                CONTADOR_D2
                CONTADOR_TMP
                CONTADOR_DELAY

        ENDC
```

Definições do endereço de início de processamento, ou vetor de reset (0x00), e do vetor de interrupções (0x05)

```
        ORG   0
        GOTO Inicio
        ORG   5
```

Início do programa:

```
Delay
        MOVLW       .10                 ; move o valor 10 para o registrador
                                        ; Work (W)
        MOVWF       CONTADOR_DELAY      ; move o conteúdo de W para a
                                        ; variável CONTADOR_DELAY
        BCF         INTCON, T0IF        ; limpando o bit de identificação de
                                        ; ocorrência da interrupção de estouro
                                        ; de timer (T0IF)
        MOVLW       0x3C                ; colocando o valor 3Ch (60 em
                                        ; decimal) no registrador Work (W)
        MOVWF       TMR0                ; movendo o valor 60 para o TMR0.
                                        ; Como o prescaler foi setado para
                                        ; 1:64, o TMR0 será incrementado a
                                        ; cada 64 ciclos. Com um clock de
                                        ; 4 Mhz, isto significa que ocorre um
                                        ; incremento a cada 64µs. Como o
                                        ; TMR0 começa em 60, após 196
                                        ; incrementos ele irá estourar.
                                        ; Para ocorrer 196 incrementos, vão ser
```

```
                                    ; necessários 196 * 64 = 12544 µs, ou,
                                    ; aproximadamente, 12,5 ms.
        CLRWDT                      ; limpa o registrador WDT para não
                                    ; ocorrer um reset
Intervalo

        CLRWDT                      ; limpa o registrador WDT para não
                                    ; ocorrer um reset
        BTFSS   INTCON, T0IF        ; se o bit T0IF do registrador de
                                    ; interrupções estiver setado, significa
                                    ; que ocorreu um estouro do TMR0. Ou
                                    ; seja, o delay de 12,5 ms foi atingido,
                                    ; então a variável
                                    ; CONTADOR_DELAY, que contém a
                                    ; quantidade de vezes que a rotina irá
                                    ; repetir (que, no caso do exemplo, é
                                    ; de 10 vezes), é decrementada, e a
                                    ; rotina recomeça. Quando a variável
                                    ; CONTADOR_CONT chegar a zero, a
                                    ; rotina retorna, sendo que o tempo
                                    ; gasto nesta rotina corresponde ao
                                    ; delay desejado (no caso, 125 ms)
                                    ; Enquanto não ocorre uma interrupção
                                    ; de estouro de timer, a rotina
                                    ; permanece em loop
        GOTO    Intervalo           ; espera interrupção de estouro de
                                    ; timer
        DECFSZ  CONTADOR_DELAY, F   ; decrementa a variável DELAY_CONT
        GOTO    Delay_Var           ; se DELAY_CONT não chegou a zero,
                                    ; volta a executar a rotina
Mostrar_Valor
        MOVF    CONTADOR_D2, W      ; move o valor atual do display2 para o
                                    ; registrador Work (W)
        MOVWF   CONTADOR_TMP        ; coloca este valor na variável
                                    ; CONTADOR_TMP
        SWAPF   CONTADOR_TMP, F     ; realiza um swap dos bits menos
                                    ; significativos com os bits mais
                                    ; significativos da variável
                                    ; CONTADOR_TMP. Isto é necessário
                                    ; porque o número a ser exibido no
                                    ; display2 será escrito nos 4 bits mais
                                    ; significativos da porta B, então eles
                                    ; devem estar na posição adequada
        CLRW                        ; limpa o registrador Work (W)
        IORWF   CONTADOR_D1, W      ; realiza um OR lógico entre o valor
                                    ; corrente do display1 e o conteúdo do
                                    ; registrador Work (W). Com isso, os
                                    ; 4 bits menos significativos de W irão
                                    ; conter o número a ser exibido no
                                    ; display1
        IORWF   CONTADOR_TMP, W     ; realiza um OR lógico entre o valor da
```

```
                                        ; variável CONTADOR_TMP (que
                                        ; contém o swap dos bits do número
                                        ; contido na variável CONTADOR_D2)
                                        ; e o conteúdo do registrador Work (W).
                                        ; Com isso, os 4 bits mais significativos
                                        ; de W irão conter o número a ser
                                        ; exibido no display2
        MOVWF   PORTB                   ; exibe os números nos displays 1 e 2
        RETURN                          ; retornando

Inicio:
        CLRF    PORTB                   ; limpando a porta B
        BANCO1                          ; mudando para o banco de memória 1
        CLRF    TRISB                   ; limpando os bits de TRISB. Com isso,
                                        ; todos os pinos (RB0 a RB7) serão
                                        ; utilizados como saída
        MOVLW   B'00011111'             ; utilizando o registrador Work (W) para
                                        ; configurar a porta de entrada A.
                                        ; Todos os pinos (RA0 – RA4) serão
                                        ; utilizados como entrada.
        MOVWF   TRISA                   ; concluindo a configuração da porta A
        MOVLW   B'0000011'              ; utilizando o registrador Work (W) para
                                        ; configurar o registrador de operação
                                        ; do microcontrolador. Os três primeiros
                                        ; bits, que correspondem à
                                        ; configuração do prescaler, estão
                                        ; sendo configurados de maneira a
                                        ; fazer o prescaler assumir o valor de
                                        ; 1:64
        MOVWF   OPTION_REG              ; concluindo a configuração do
                                        ; registrador de operação
        BANCO0                          ; mudando para o banco de memória 0

Resetar:
        CLRF    CONTADOR_D1             ; limpando o display1
        CLRF    CONTADOR_D2             ; limpando o display2
        CLRF    CONTADOR_TMP            ; limpando a variável
                                        ; CONTADOR_TMP
        CALL    Mostrar_Valor           ; exibindo o valor “00” nos displays

Loop:
        CALL    Delay                   ; delay de 0,25 segundos
        BTFSC   PORTA, 2                ; verificando se o botão de reset do
                                        ; cntador foi pressionado.
        GOTO    Resetar                 ; se o botão estiver pressionado, reseta
                                        ; os displays
        BTFSS   PORTA, 1                ; verificando o status da chave de
                                        ; parada do contador. Se estiver
                                        ; desligada, o contador irá parar. Se
                                        ; estiver ligada, a contagem continua
        GOTO    Loop                    ; pára o contador
```

```
        BTFSS   PORTA, 0            ; verificando o sentido da contagem. Se
                                    ; a chave de sentido estiver ligada, a
                                    ; contagem é crescente. Caso
                                    ; contrário, a contagem é decrescente
        GOTO    Decrescente         ; contagem decrescente

Crescente:                          ; contagem crescente
        INCF    CONTADOR_D1, F      ; incrementa o valor atual do display1
        MOVLW   .10                 ; move o valor 10 para o registrador
                                    ; Work (W)
        SUBWF   CONTADOR_D1, W      ; subtrai o valor atual do display1 do
                                    ; valor de W (no caso, 10)
        BTFSS   STATUS, Z           ; verifica se a operação anterior
                                    ; resultou em zero. Caso sim, o
                                    ; display1 deve ser atualizado, voltando
                                    ; para o valor 0, e o display2 deve ser
                                    ; incrementado. Caso contrário, a
                                    ; contagem continua normalmente
        GOTO    Desviar_C           ; continua a contagem
        CLRF    CONTADOR_D1         ; o display1 volta para o valor 0
        INCF    CONTADOR_D2, F      ; o display2 é incrementado
        MOVLW   .10                 ; move o valor 10 para o registrador
                                    ; Work (W)
        SUBWF   CONTADOR_D2, W      ; subtrai o valor atual do display2 do
                                    ; valor de W (no caso, 10)
        BTFSC   STATUS, Z           ; verifica se a operação anterior
                                    ; resultou em zero. Caso sim, o
                                    ; display2 também deve ser atualizado,
                                    ; voltando para o valor 0
        CLRF    CONTADOR_D2         ; o display2 volta para o valor 0

Desviar_C:
        CALL    Mostrar_Valor       ; exibe o valor atual dos contadores de
                                    ; cada display
        GOTO    Loop                ; volta para o loop principal

Decrescente:
        DECF    CONTADOR_D1, F      ; decrementa o valor atual do display1
        MOVLW   0xFF                ; move o valor 0xFF (255 em decimal)
                                    ; para o registrador Work (W)
        SUBWF   CONTADOR_D1, W      ; subtrai o valor atual do display1 de W
        BTFSS   STATUS, Z           ; verifica se a operação anterior
                                    ; resultou em zero. Caso sim, o
                                    ; display1 deve ser atualizado, voltando
                                    ; para o valor 9, e o display2 deve ser
                                    ; decrementado. Caso contrário, a
                                    ; contagem continua normalmente
        GOTO    Desviar_D           ; continua a contagem
        MOVLW   .9                  ; move o valor 9 para o registrador
                                    ; Work (W)
        MOVWF   CONTADOR_D1         ; atualiza o contador do display1
```

```
        DECF    CONTADOR_D2, F      ; decrementa o display2
        MOVLW   0xFF                ; move o valor 0xFF (255 em decimal)
                                    ; para o registrador Work (W)
        SUBWF   CONTADOR_D2, W      ; subtrai o valor atual do display2 de W
        BTFSS   STATUS, Z           ; verifica se a operação anterior
                                    ; resultou em zero. Caso sim, o
                                    ; display2 também deve ser atualizado,
                                    ; voltando para o valor 9
        GOTO    Desviar_D           ; continua a contagem
        MOVLW   .9                  ; move o valor 9 para o registrador
                                    ; Work (W)
        MOVWF   CONTADOR_D2         ; o display2 volta para o valor 9

Desviar_D:
        CALL    Mostrar_Valor       ; exibe o valor atual dos contadores de
                                    ; cada display
        GOTO    Loop                ; volta para o loop principal

        END                         ; fim do programa
```