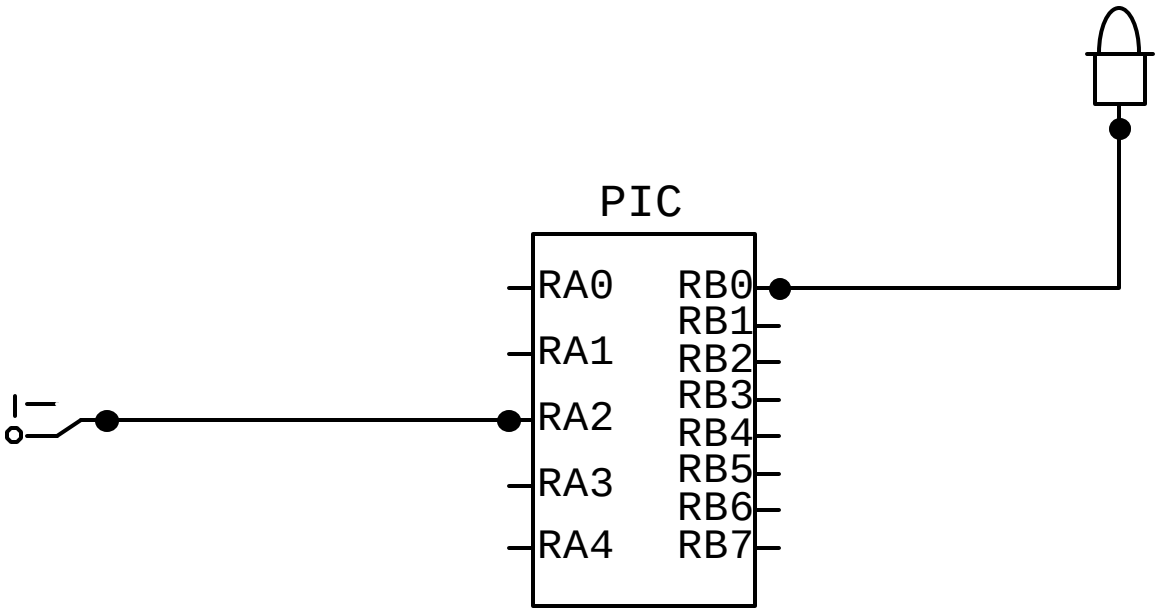
CURSO DE MICROCONTROLADORES

EXERCÍCIOS – ALUNO

Programa 1:

Programa que faz com que um LED, ligado à saída RB0, seja aceso, caso uma chave C, ligada à entrada RA2, esteja em “1”. Caso contrário, o diodo se apaga.

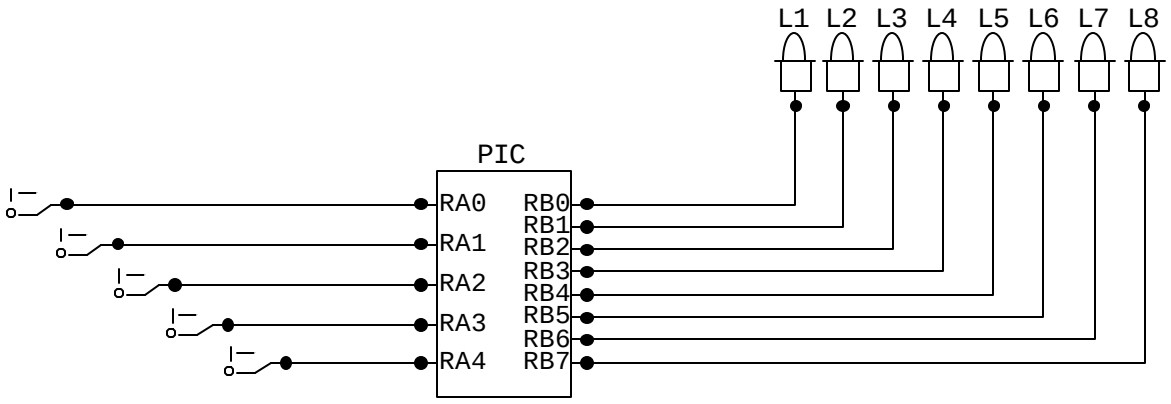
Diagrama Esquemático:



Programa 2:

Programa que 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 combinação deve ser 1-0-1-0-1.

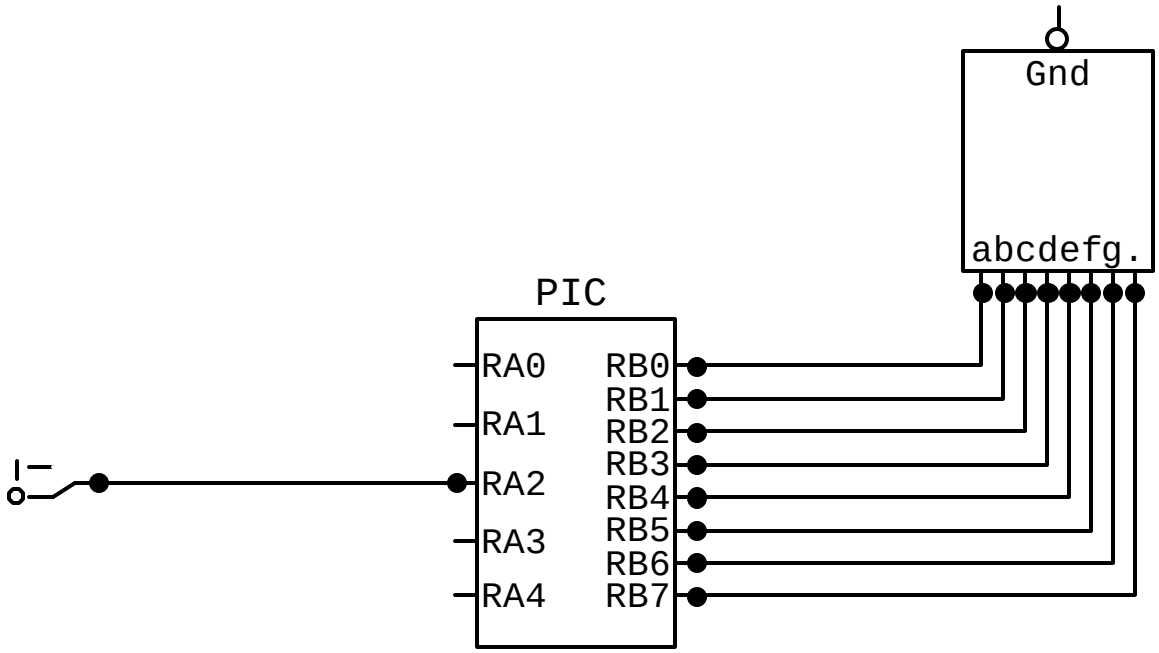
Diagrama Esquemático:



Programa 3:

Programa que 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.

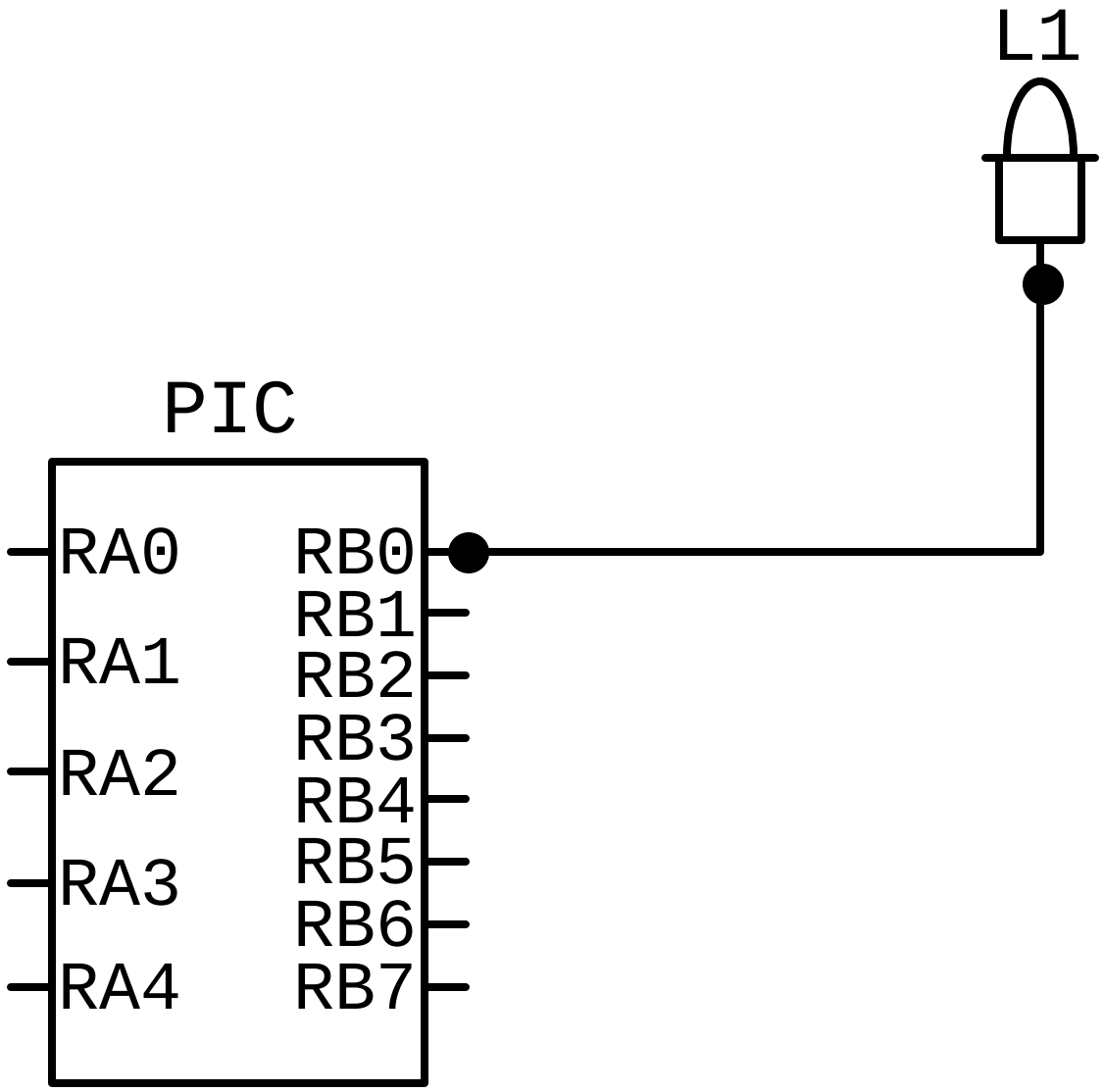
Diagrama Esquemático:



Programa 4:

Programa que 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 atrasos para que as transições do LED possam ser visualizadas. Estes atrasos funcionam da seguinte maneira: uma rotina, chamada Atraso1S, que utiliza a subrotina Del10, que gera atrasos 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 à rotina Del10, gerando um atraso total de 1 segundo.

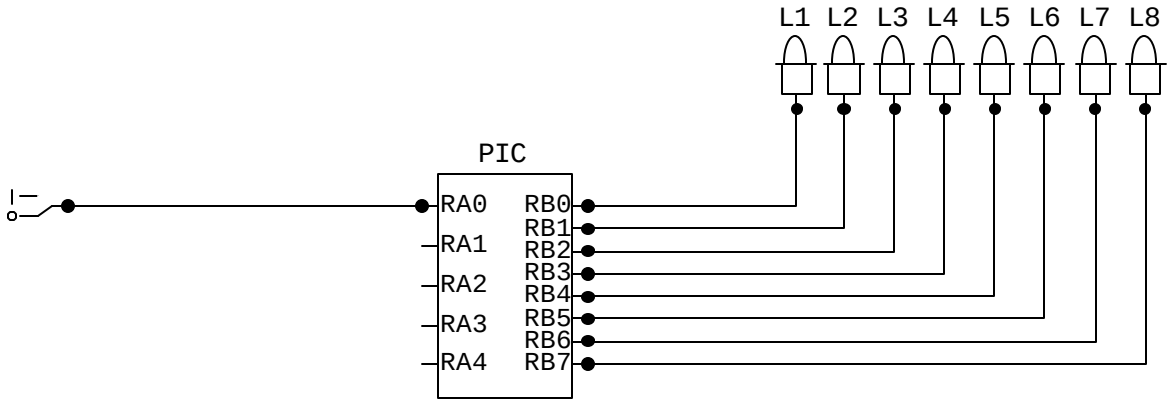
Diagrama Esquemático:



Programa 5:

Programa que 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

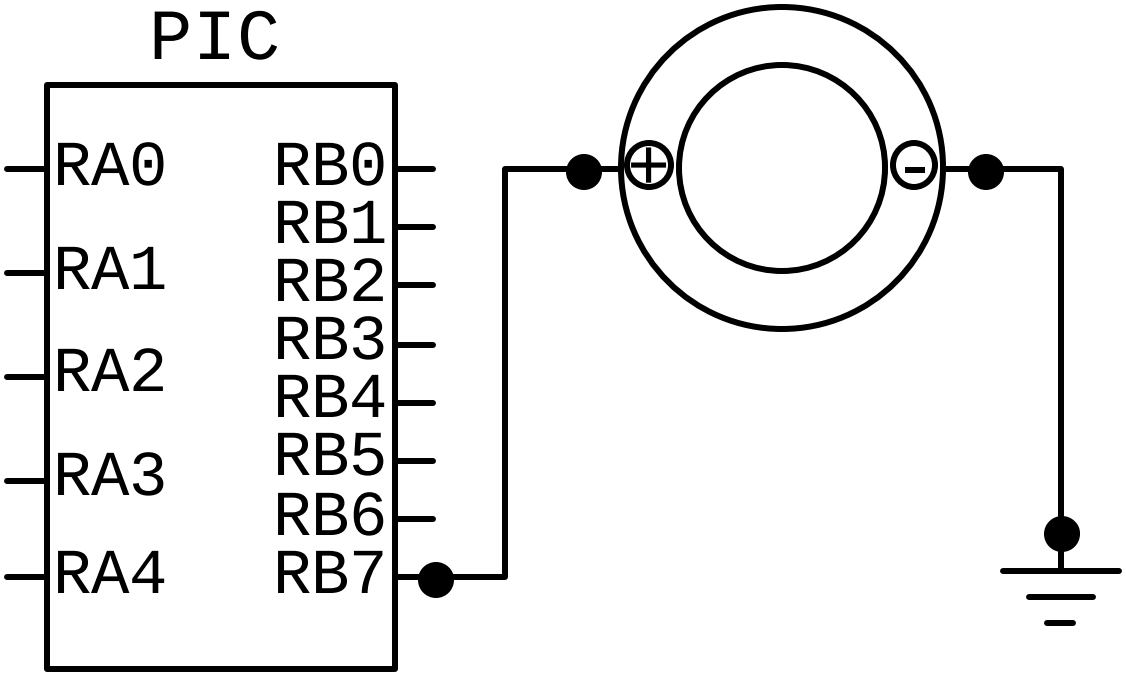
Diagrama Esquemático:



Programa 6:

Programa que 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 atrasos para que os *beeps* do speaker possam ser ouvidos claramente. Estes atrasos funcionam da seguinte maneira: duas rotinas, uma chamada Atraso1S (atraso de 1 segundo) e outra, AtrasoMS (atraso de 0,5 segundos), utilizam a subrotina Del10, que gera atrasos de 10 ms a cada chamada. Dentro desta rotina existe uma variável auxiliar (TEMPO1), que, para a rotina Atraso1S, tem o valor inicial 100 e, para a rotina AtrasoMS tem o valor inicial de 50, e que vai sendo decrementada. Cada vez que ela é decrementada, a rotina Del10 é chamada novamente.

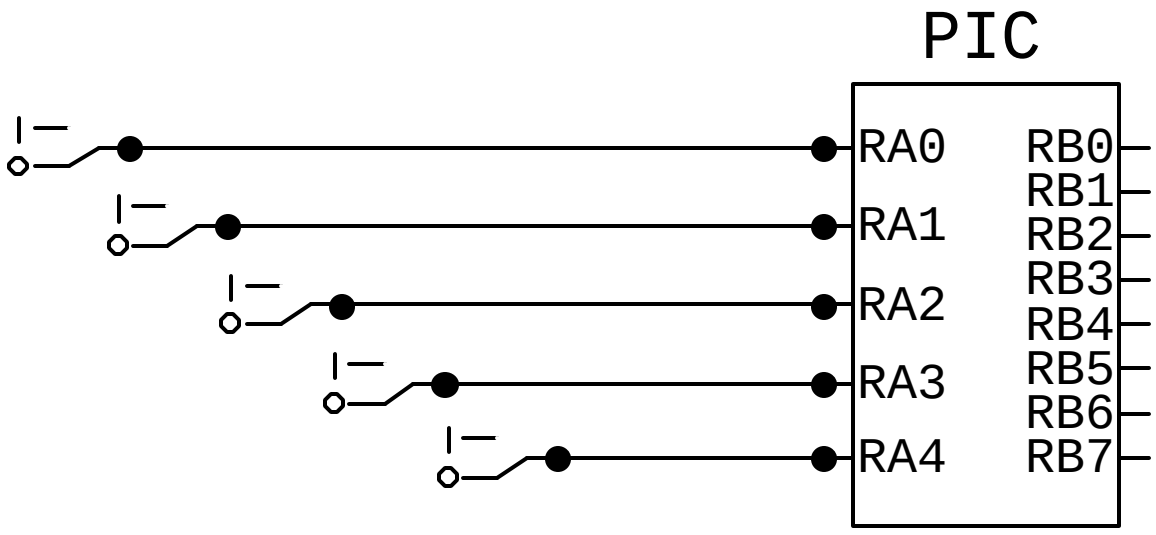
Diagrama Esquemático:



Programa 7:

Programa que escreve dados na memória EEPROM do microcontrolador. A posição 0 desta memória irá armazenar o valor das chaves ligadas aos pinos RA0 à RA4 da porta A (em binário). A verificação do valor armazenado na memória pode ser visualizada 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 limpo pelo hardware, indicando o fim da escrita, ou introduzirmos um atraso, que espera uma determinada quantidade de tempo antes de continuar a execução. Esta implementação utiliza um atraso de, aproximadamente, 2,6 segundos, para a espera da gravação da memória EEPROM.

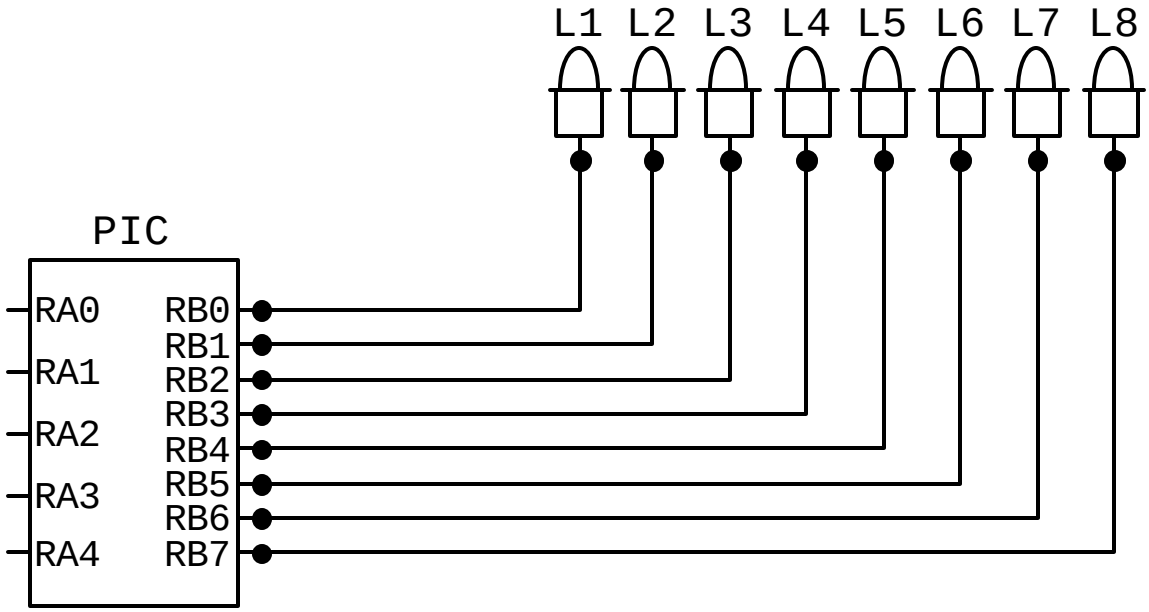
Diagrama Esquemático:



Programa 8:

Programa que 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.

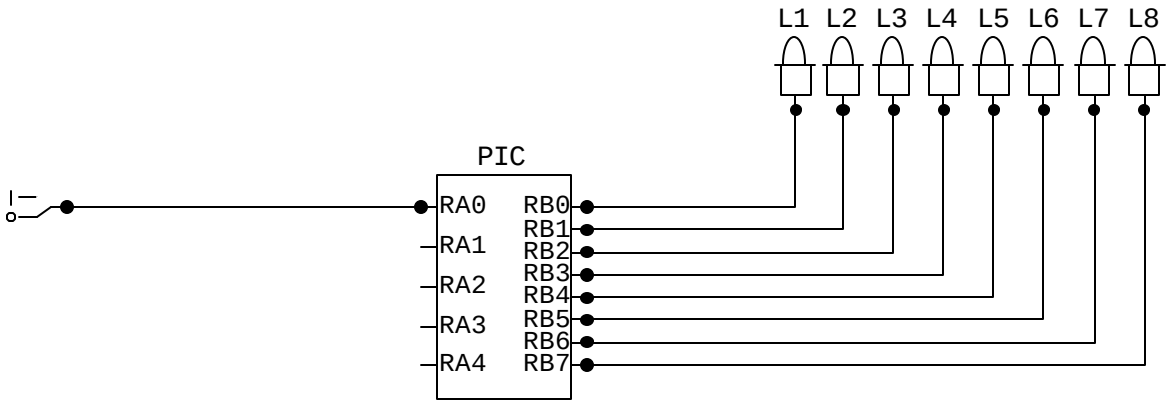
Diagrama Esquemático:



Programa 9:

Programa similar ao programa 5, mas explora os recursos de WatchDog e modo Sleep do microcontrolador. A partir este 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.

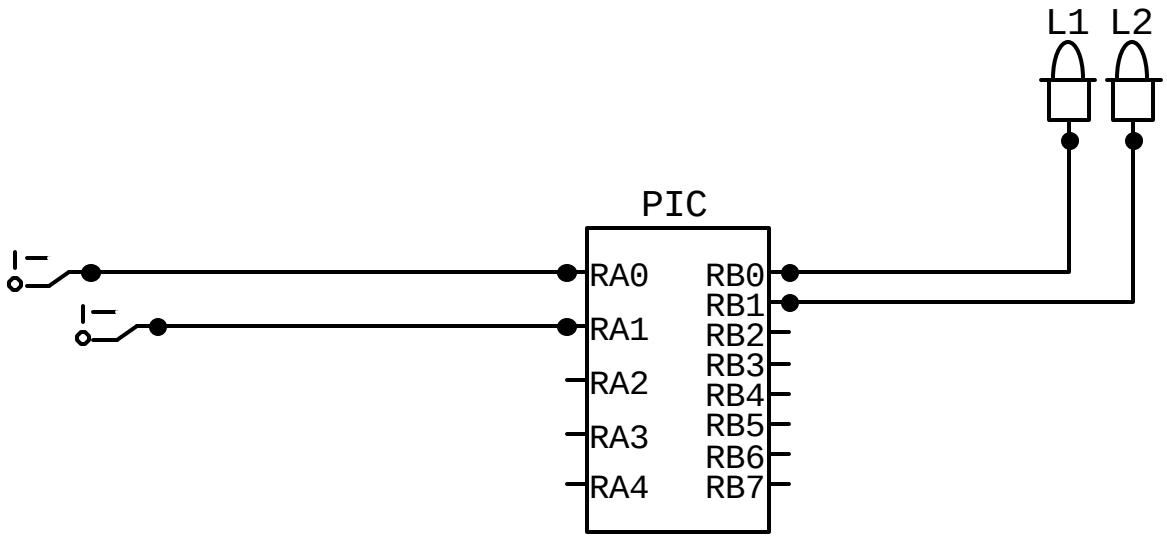
Diagrama Esquemático:



Programa 10:

Programa que 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 à frente, e 0 significa reverter os motores.

Diagrama Esquemático:



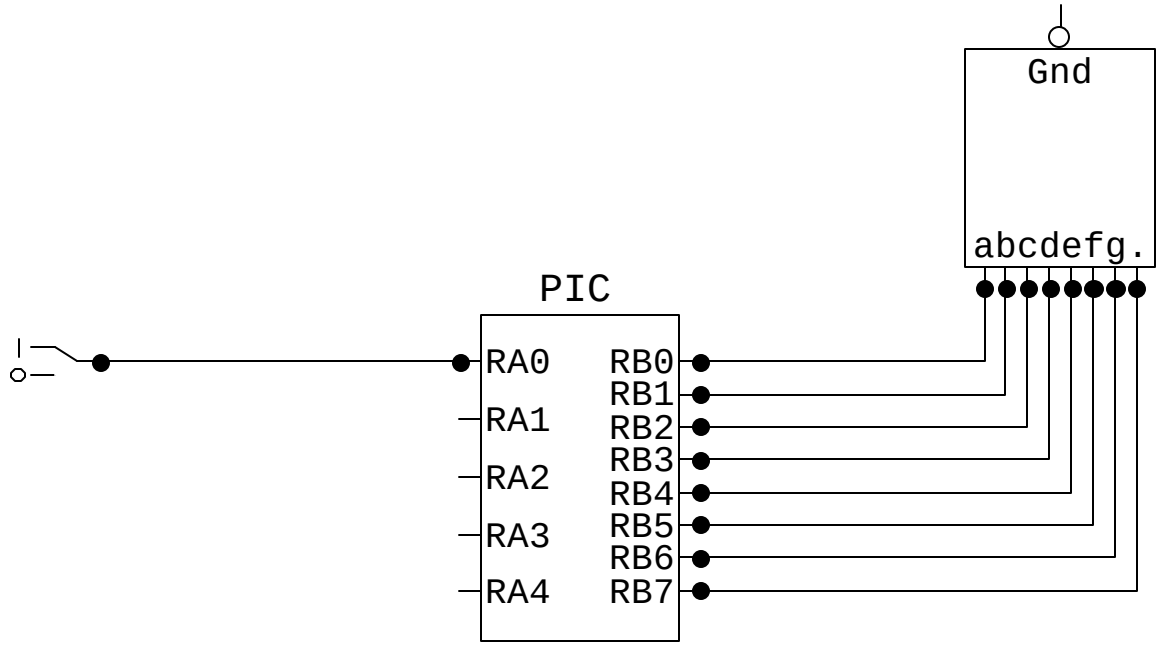
Programa 11:

Programa que gera 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, seqüencialmente, 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 seqüência 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:



Programa 12:

Programa que 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: