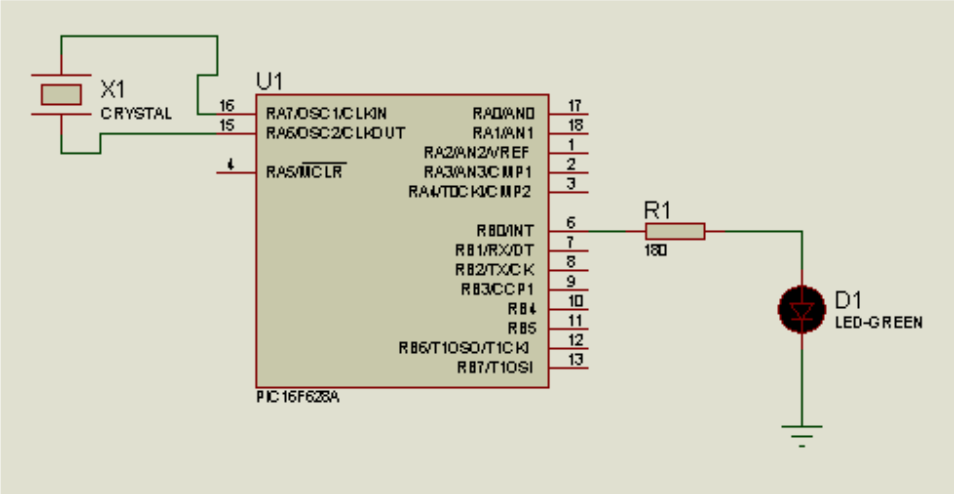
Curso: Engª da Computação Disciplina: Microcontroladores
Docente: Marcos Benevides
Resolução 3.a Lista de exercícios
Linguagem C – PORTS como Saída

1°) Desenvolva um programa em Linguagem C para piscar um led conectado a o pino B0 do PIC 16F628A em intervalos de tempo de 1 seg.

HARDWARE:



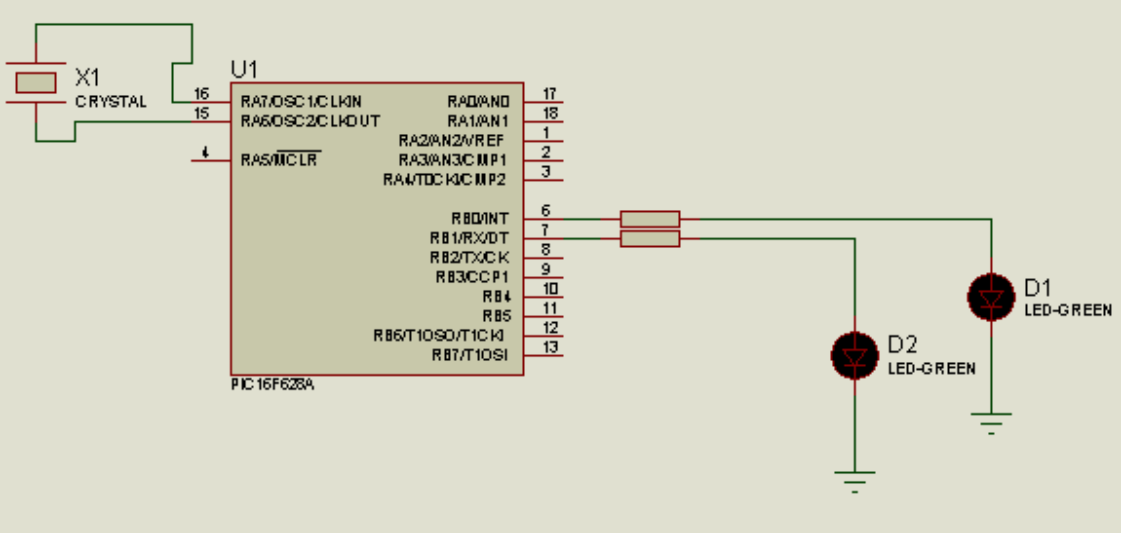
SOFTWARE:

```
#include <16f628a.h>
#use delay(crystal = 4MHz)
#fuses xt, nowdt, nolvp, brownout, protect
#byte PORTB = 0x06
#bit  LED1 = PORTB.0
main()
{
      set_Tris_b(0b11111110);
      while(1)
        {
          LED1 = 1;
          delay_ms(500);
          LED1 = 0;
          delay_ms(500);
        }

}
```

2°) Desenvolva um programa em Linguagem C para que dois LEDS conectados ao PIC 16F628A pisquem alternadamente. Cada um deve ficar aceso 100ms.

HARDWARE:

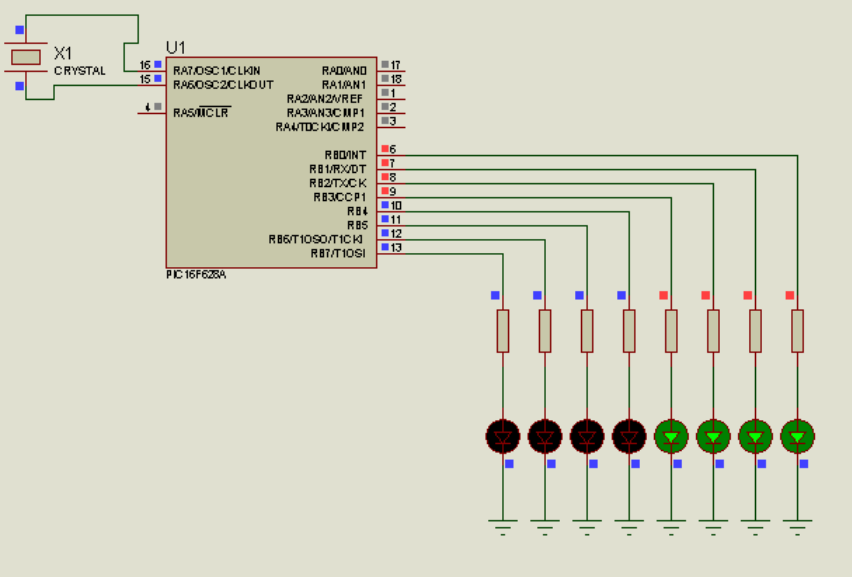


SOFTWARE:

```
#include <16f628a.h>
#use delay(crystal = 4Mhz)//definição da frequência do cristal
                          //para as rotinas de tempo
#fuses xt, nowdt, NOPROTECT, nobrownout, put //fusebits
#byte PORTB = 0x06
#bit  LED1    = PORTB.0
#bit  LED2    = PORTB.1
main()
{
  set_tris_b(0b11111100);//0xFC
  while(1)
    {
      LED1 = 1;
      LED2 = 0;
      delay_ms(100);
      LED1 = 0;
      LED2 = 1;
      delay_ms(100);
    }
}
```

3°) De acordo com o hardware mostrado abaixo, desenvolva um programa para que 4 leds pisquem alternadamente com os outros 4 leds restante em intervalos de tempo de 500ms.

HARDWARE:

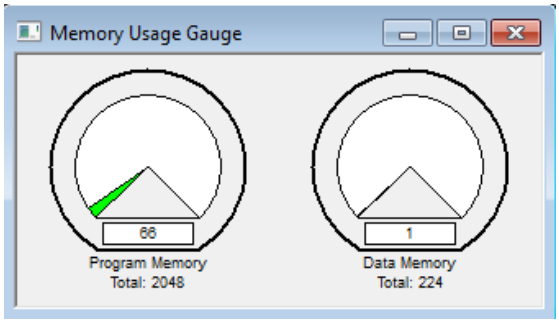


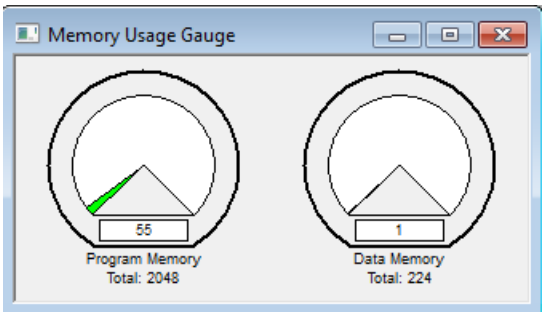
SOFTWARE:

```
#include <16f628a.h>
#use delay(crystal = 4Mhz)
#fuses xt, nowdt, NOPROTECT, nobrownout, put //fusebits
#byte PORTB = 0x06
#bit  LED1   = PORTB.0
#bit  LED2   = PORTB.1
#bit  LED3   = PORTB.2
#bit  LED4   = PORTB.3
#bit  LED5   = PORTB.4
#bit  LED6   = PORTB.5
#bit  LED7   = PORTB.6
#bit  LED8   = PORTB.7
main()
{
  set_tris_b(0b00000000);//0x00 em hexa
  while(1)
    {
      LED1=1;LED2=1;LED3=1;LED4=1;
      LED5=0;LED6=0;LED7=0;LED8=0;
      delay_ms(500);
      LED1=0;LED2=0;LED3=0;LED4=0;
      LED5=1;LED6=1;LED7=1;LED8=1;
      delay_ms(500);
    }
}
```

Outra maneira alternativa de resolução é considerar a escrita de uma só vez no PORTB, onde o programa ficará mais enxuto e ainda economizará 11 words da memória de programa:

```
#include <16f628a.h>
#use delay(crystal = 4Mhz)
#fuses xt, nowdt, NOPROTECT, nobrownout, put //fusebits
#byte PORTB = 0x06
main()
{
  set_tris_b(0b00000000);
  PORTB = 0;
  while(1)
    {
      PORTB = 0b00001111;
      delay_ms(500);
      PORTB = 0b11110000;
      delay_ms(500);
    }
}
```

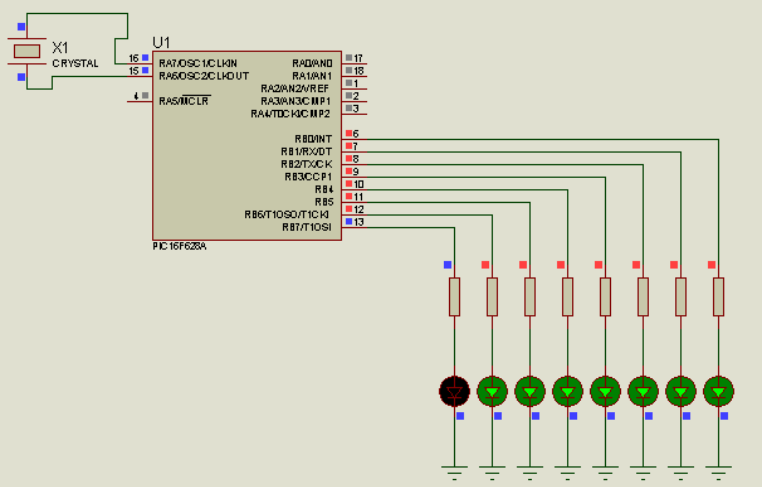




Obs: parece pouco em comparação com os recursos disponíveis de um computador por exemplo, mas para um microcontrolador isso pode fazer muita diferença!

4°) Utilizando o mesmo hardware da questão acima, desenvolva um programa em que os leds acendam em sequência e quando atingir o último retornem sendo apagados.

HARDWARE:

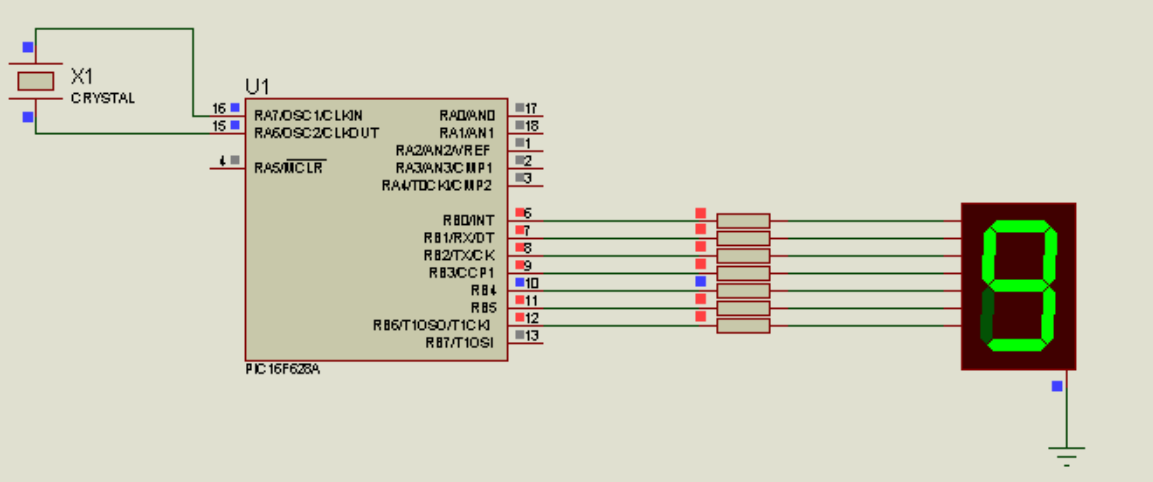


SOFTWARE:

```
#include <16f628a.h>
#use delay(crystal = 4Mhz)
#fuses xt, nowdt, NOPROTECT, nobrownout, put //fusebits
#byte PORTB = 0x06
#bit   LED1    = PORTB.0
#bit   LED2    = PORTB.1
#bit   LED3    = PORTB.2
#bit   LED4    = PORTB.3
#bit   LED5    = PORTB.4
#bit   LED6    = PORTB.5
#bit   LED7    = PORTB.6
#bit   LED8    = PORTB.7
main()
{
  set_tris_b(0x00);
  PORTB = 0x00; //PORTB inicialmente deve ser zerado
                //para que nenhum led inicie aceso
  while(1)
    {
//acende
      LED1=1;delay_ms(100);
      LED2=1;delay_ms(100);
      LED3=1;delay_ms(100);
      LED4=1;delay_ms(100);
      LED5=1;delay_ms(100);
      LED6=1;delay_ms(100);
      LED7=1;delay_ms(100);
      LED8=1;delay_ms(100);
//apaga
      LED8=0;delay_ms(100);
      LED7=0;delay_ms(100);
      LED6=0;delay_ms(100);
      LED5=0;delay_ms(100);
      LED4=0;delay_ms(100);
      LED3=0;delay_ms(100);
      LED2=0;delay_ms(100);
      LED1=0;delay_ms(100);
    }
}
```

5°) Conforme o hardware abaixo, desenvolva um programa para fazer a contagem de 0 a 9 em um display de 7 segmentos catodo-comum.

HARDWARE:



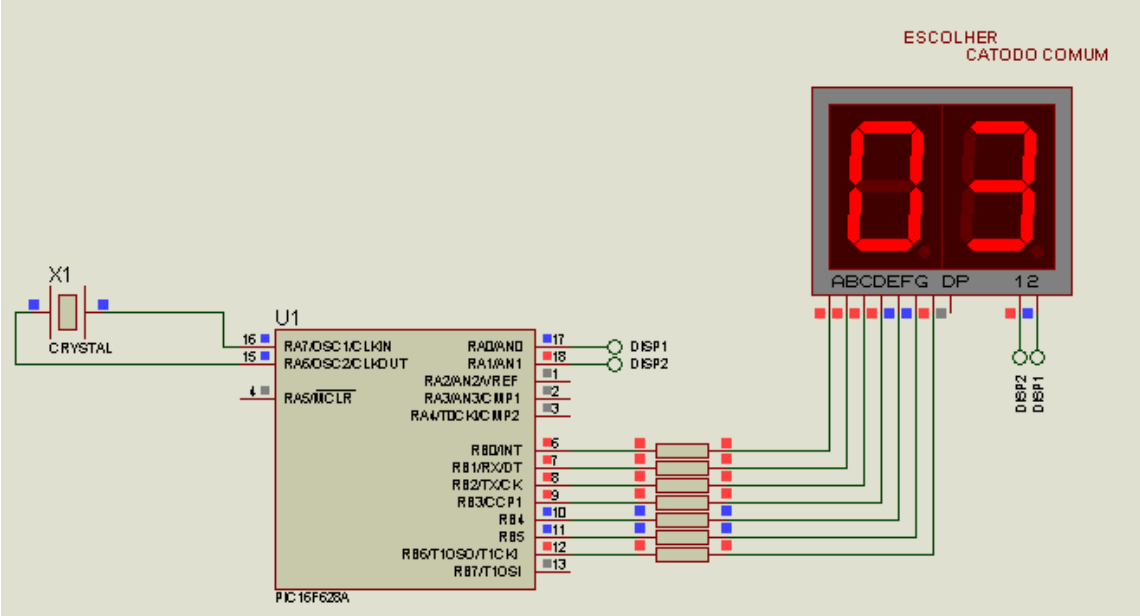
SOFTWARE:

```
#include <16f628a.h>
#use delay(crystal = 4Mhz)
#fuses xt, nowdt, NOPROTECT, nobrownout //fuses bits
#byte PORTB = 0x06
main()
{

  set_tris_b(0b10000000); //0xFE (em hexa)
  while(1)
   {
    //      0bgfedcba
    PORTB = 0b00111111;//0
    delay_ms(500);
    PORTB = 0b00000110;//1
    delay_ms(500);
    PORTB = 0b01011011;//2
    delay_ms(500);
    PORTB = 0b01001111;//3
    delay_ms(500);
    PORTB = 0b01100110;//4
    delay_ms(500);
    PORTB = 0b01101101;//5
    delay_ms(500);
    PORTB = 0b01111101;//6
    delay_ms(500);
    PORTB = 0b00000111;//7
    delay_ms(500);
    PORTB = 0b01111111;//8
    delay_ms(500);
    PORTB = 0b01101111;//9
    delay_ms(500);
   }
}
```

6°) Conforme o hardware abaixo, desenvolva um programa para fazer a contagem de 0 a 20 multiplexando no tempo dois displays de 7 segmentos. A contagem deverá ser feita em intervalos de tempo de 100ms.

HARDWARE:



SOFTWARE:

<u>Solução:</u>

Para escrevermos nos dois displays não podemos fazer isso ao mesmo tempo, pois ambos iriam mostrar o mesmo número. Para tal, devemos primeiro escrever em um dos displays o número correspondente e depois no outro. Essa escrita é feita várias vezes, para que seja possível ver o número, até se atingir o tempo de cada dígito e só assim então mudar de número. Segue abaixo a solução para os dois primeiros números: 00 e 01. Os demais números seguem o mesmo raciocínio. Vale ressaltar que essa não é a solução única (nunca é) e também não é a que desenvolve menor código. Mas é a solução possível com as ferramentas disponíveis até agora na nossa disciplina. Mais à frente, desenvolveremos uma solução mais compacta com a utilização de outros conhecimentos.

```
#include <16f628a.h>
#use delay(crystal = 4Mhz)
#fuses xt, nowdt, NOPROTECT, nobrownout
#byte PORTA = 0x05
#byte PORTB = 0x06
main()
{
    set_tris_a(0b11111100);
    set_tris_b(0b10000000);
    PORTA = 0;//zerando PORTS para que não haja
    PORTB = 0;//caracteres estranhos ao iniciar
              //programa
    while(1)
      {

          PORTA = 0b00000001;//00
          PORTB = 0b00111111;
          delay_ms(25);
          PORTA = 0b00000010;
          PORTB = 0b00111111;
          delay_ms(25);

          PORTA = 0b00000001;
          PORTB = 0b00111111;
          delay_ms(25);
          PORTA = 0b00000010;
          PORTB = 0b00111111;
          delay_ms(25);

          PORTA = 0b00000001;
          PORTB = 0b00111111;
          delay_ms(25);
          PORTA = 0b00000010;
          PORTB = 0b00111111;
          delay_ms(25);

          PORTA = 0b00000001;
          PORTB = 0b00111111;
          delay_ms(25);
          PORTA = 0b00000010;
          PORTB = 0b00111111;
          delay_ms(25);

          PORTA = 0b00000001;
          PORTB = 0b00111111;
          delay_ms(25);
          PORTA = 0b00000010;
          PORTB = 0b00111111;
          delay_ms(25);
```

```
PORTA = 0b00000001;
PORTB = 0b00111111;//0
delay_ms(25);
PORTA = 0b00000010;
PORTB = 0b00000110;//1
delay_ms(25);

PORTA = 0b00000001;
PORTB = 0b00111111;//0
delay_ms(25);
PORTA = 0b00000010;
PORTB = 0b00000110;//1
delay_ms(25);

PORTA = 0b00000001;
PORTB = 0b00111111;//0
delay_ms(25);
PORTA = 0b00000010;
PORTB = 0b00000110;//1
delay_ms(25);

PORTA = 0b00000001;
PORTB = 0b00111111;//0
delay_ms(25);
PORTA = 0b00000010;
PORTB = 0b00000110;//1
delay_ms(25);

PORTA = 0b00000001;
PORTB = 0b00111111;//0
delay_ms(25);
PORTA = 0b00000010;
PORTB = 0b00000110;//1
delay_ms(25);
```

Continua....