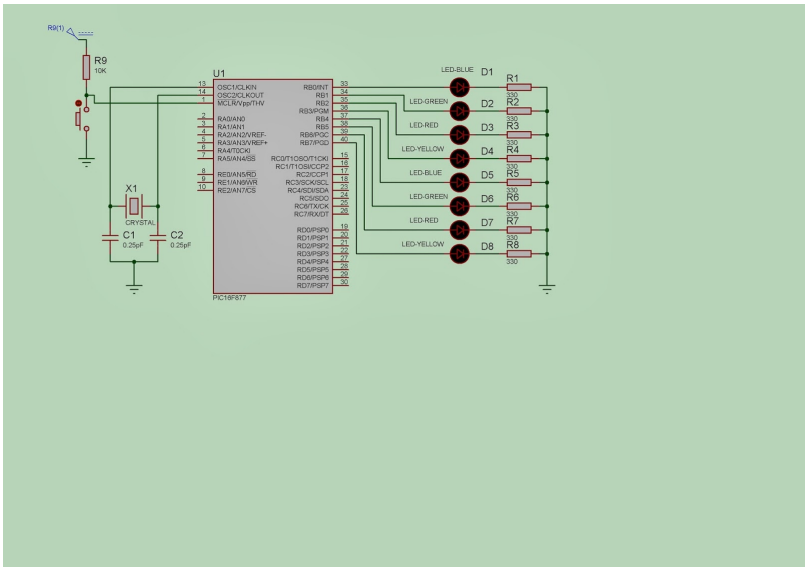
PIC – Aplicatii cu Led-uri folosind PIC16F877 partea.I

written by Adrian Micu

Desi pe tot internetul gasim aceleasi aplicatii cu led-uri (aprinderea unui led, led-ul care palpaie,etc….) si eu voi prezenta in continuare cateva aplicatii cu led-uri.

Cu ajutorul acestor aplicatii un **INCEPATOR** invata sa foloseasca microcontroller-ul, sa inteleaga cum se folosec porturile lui, cum ne ‘jucam” cu ele, etc…, de aceea cine stie deja lucrurile acestea il rog sa nu ma „injure” si sa aiba rabdare ca voi posta si proiecte mai complicate.

Pornind de la schema urmatoare:



dorim sa facem ca ledurile D1,D2,..D8 sa se aprinda pe rand (adica D1–D2–D3–D4–….–D8 api D8–D7–D6–….–D1).

Vom lucra in mediul „MikroC for PIC” produs de mikroelektronika, iar ca placa de dezvoltare folosesc un EasyPIC6 produs tot de mikroelektronika.

Scrieu mai jos programul apoi il voi explica:

```
void main()
{
```

```
    int i;
      ANSEL=0;                          //configuram pinii
microcontrollerului ca fiind digitali
ANSELH=0;

TRISB=0x00;                          //configuram portul B ca fiind
un port de iesire
 PORTB=0x00000001;           // ii dam pinului RB0 de la portul B
valoarea 1

 while(1)
{
        for(i=0;i<7;i->
          {
               PORTB=PORTB<<1   //  mutam la stanga cu o pozitie
acel 1 din valoarea initiala a lui PORTB
Delay_ms(1000);             //  pana ajunge la pinul RB7
}

for(i=6;i>=0;i-)
           {
              PORTB=PORTB>>1;
              Delay_ms(1000);
            }
}
}
```

TRISB : cu ajutorul acestui registru facem ca pinii de la portul B sa fiie pini de iesire sau de intrare. Daca ii facem de iesire atuncii microcontrollerul va scoate semnal din microntroller catre ceva iar daca este de intrare atunci microcontrollerul asteapta sa primeasca semnale de la ceva. In cazul nostru **TRISB=0x00** adica de iesire.

PORTB: acest registru ne ajuta sa dam la fiecare pin al portului ce semnal vrem noi sa iasa. In cazul nostru **PORTB=0x00000001**, pinul RB0=1,RB1=0,RB2=0,….,RB7=0 adica pe pinul RB0 va iesi **1 logic (5v)** si va cauza aprinderea **led-ului D1.**

PORTB=PORTB<<1 : semnul „**<<**” inseamna **shift la stanga cu o unitate**

(casuta). Spre exemplu avem cazul nostru ,unde initial PORTB=0x00000001 iar apopi noi il shift-am (mutam) pe 1 de la sfarsit cu o casuta catre stanga de sase (6) ori: pt. i =0 avem PORTB=0x00000010; pt i=1 avem PORTB=00000100; si asa mai departe pana cand PORTB=0x10000000. Binenteles ca intre aceste mutari se executa si instructiunea **Delay_ms(1000)** care inseamna : asteapta (intarzie continuarea executarii) o secunda. Cand PORTB=0x10000000 atunci programul va executa urmatoul **for** (**for(i=6;i>=0;i–)**) si va **shift-a (muta)** pe 1 spre dreapta cu o unitate (casuta) pana cand **PORTB=0x00000001**, si apoi se reia de la inceput (de la primul **for**. Aceste doua **for-uri** fiind incluse intr-o intructiune **WHILE(1)** se vor executa la infinit.

Acestea fiind spuse in continuare voi pune un filmulet in care este explicat si exemplificat practic

Filmulet prezentare aplicatie + simulare in Proteus: