TP N₀ 02 Utilisation des boutons.

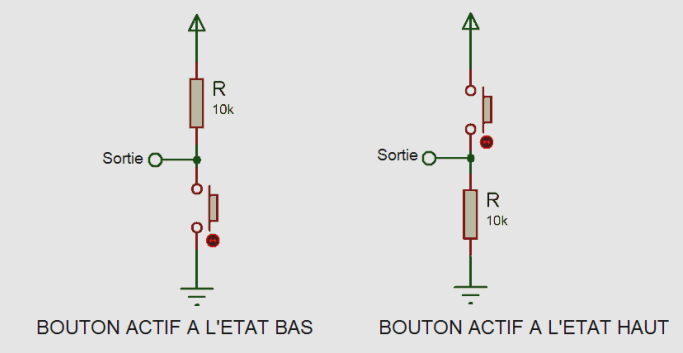
1. Objectif

L'objectif de ce deuxième TP d'apprendre comment rendre un microcontrôleur interagi avec le milieu extérieur par l'utilisation des boutons et les afficheurs 7 segments.

2. Utilisation de boutons

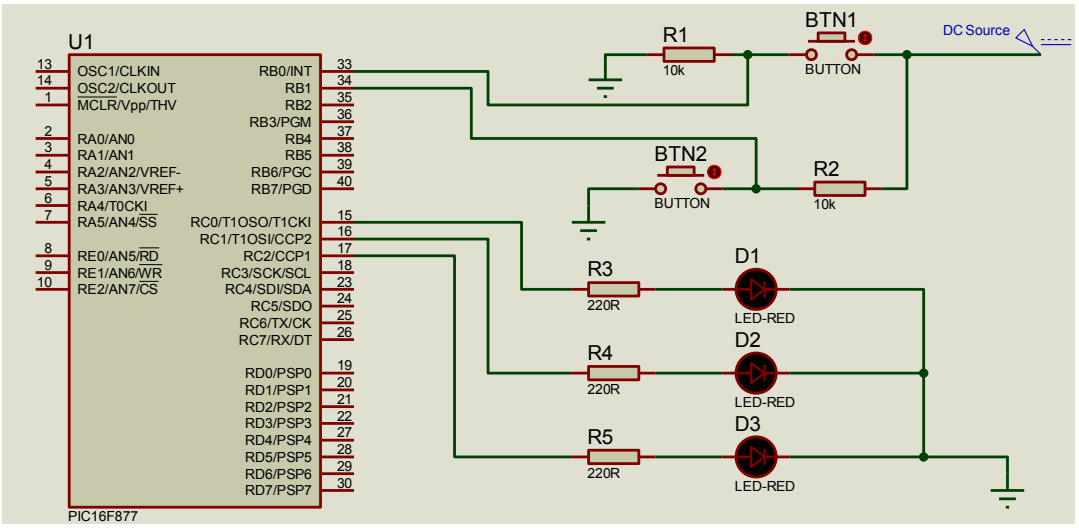
La mise en œuvre des boutons est l'un des alternatives les plus populaires dans les interfaces avec les utilisateurs. Les boutons sont simples à utiliser, et ils ont un coût, de mise en œuvre, économique. Le bouton-poussoir peut être normalement ouvert ou normalement fermé.

Installation des boutons peut être effectuée de deux manières, actif à l'état haut ou actif à l'état bas, la figure suivante montre comment configurer les deux possibilités :



La décision d'utiliser l'activation à l'état haut ou bas, dépend du développeur qui doit analyser la manière la plus simple au moment de la conception.

Pour analyser et étudier l'utilisation des boutons, on crée un nouveau projet Proteus avec les composants : PIC 16F877A, BUTTON, RES, et LED-RED. Le circuit correspondant dans ISIS est le suivant :



Le programme respectif du microcontrôleur est le suivant :

```
void main() {

 trisb = 0xFF  ;
 trisc = 0;
 while (1)
 {
  portc.f0  = portb.f0 ;
  portc.f1  = portb.f1 ;
  //portc = portb      ;
delay_ms(100) ;
 }
```

3. Travail demandé :

- **3.1.** Réaliser le circuit de test ci-dessous sous ISIS, Taper le code sous mikroC PRO, générer le ficher ***.hex*** et lancer la simulation.
- **3.2.** Noter le fonctionnement des LEDs en fonctions des Boutons BTN1 et BTN2.
- **3.3.** Pour commander la LED D3 ajouter la commande suivante au code Précédent :

```
portc.f2 =  portb.f0 && (!portc.f1)  ;
```

- **3.4.** Noter le fonctionnement de la LED D3 en fonctions des Boutons BTN1 et BTN2.
- **3.5.** Chercher le ***Help*** (l'aide) de mikroC PRO les autres operateurs logiques « ***Logic*** » et faire des essais sur la LED D3 en fonction des Boutons BTN1 et BTN2.

Bibliothèque Button

Le compilateur MikroC PRO dispose d'une bibliothèque ***Button***, qui peut être trouvé dans la palette ou l'onglet des bibliothèques. Cette bibliothèque contient la seule fonction :

Button(*port*, *pin*, *time*, *active_state*);

Où *port* est le port où le bouton est connecté
pin est le bit du port où le bouton est connecté,
time est le temps d'attente en millisecondes
active_state est l'état logique pour lequel on souhaite l'activation du bouton. La fonction retourne 0 si le bouton n'est pas actif et 255 si elle est active.

Pour analyser et étudier l'utilisation de cette bibliothèque, on utilise le circuit précédent, Mais le programme du microcontrôleur est le suivant :

```
void main() {
trisb = 0xFF  ;   trisc = 0;
while (1)
{
  if( Button(&PORTB, 0, 100, 1) ) //Test de d'état du bouton sur RB7,activé à l'état haut
  PORTC.F0=1; // Allumer le voyant si le bouton est actif.
  else  PORTC.F0=0; //Eteindre la LED s'il est relaché.

  if( Button(&PORTB, 1, 100, 0) ) //Test de d'état du bouton sur RB7,activé à l'état bas
  PORTC.F1=1; // Allumer le voyant si le bouton est actif.
  else  PORTC.F1=0; //Eteindre la LED s'il est relaché.

delay_ms(100) ;
}
}
```

- **3.6.** Noter le fonctionnement des LEDs en fonctions des Boutons BTN1 et BTN2, et le comparer avec le premier code.
- **3.7** Ecrire un programme MicroC qui permet de faire clignoter ou éteindre des LEDs sur le Port B (RB0… RB7), en fonction de l'état d'un interrupteur place sur RC0.
- **3.8.** Ecrire un programme qui permet de lire l'état de deux boutons sur RC0 et RC1 puis allume les LEDs sur le Port B (RB0, RB1, RB2, RB3) suivant la logique suivante :
  - Si RC0 = 0 et RC1 = 0 alors Eteindre toute les LEDs.
  - Si (RC0 AND RC1) = 1 alors Allumer toute les LEDs.
  - Si (RC0 XOR RC1) = 1 alors Clignoter les LEDs : RB0 = on , RB1 = off, RB2 = on, RB3= off.
    Délais de 1 seconde
    RB0 = off, RB1 = on, RB2 = off, RB3= on.