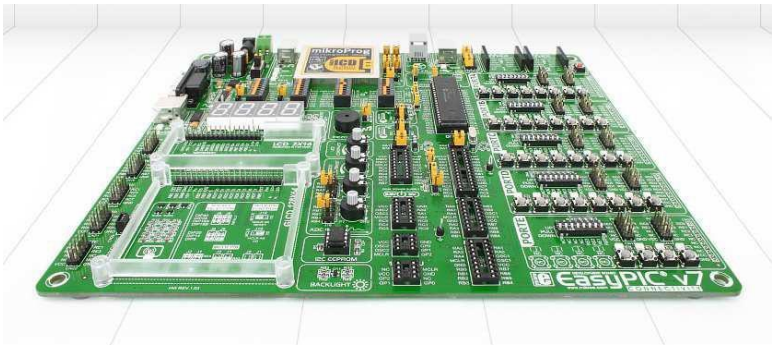
# TP 6 – Informatique embarquée

# PWM et Buzzer

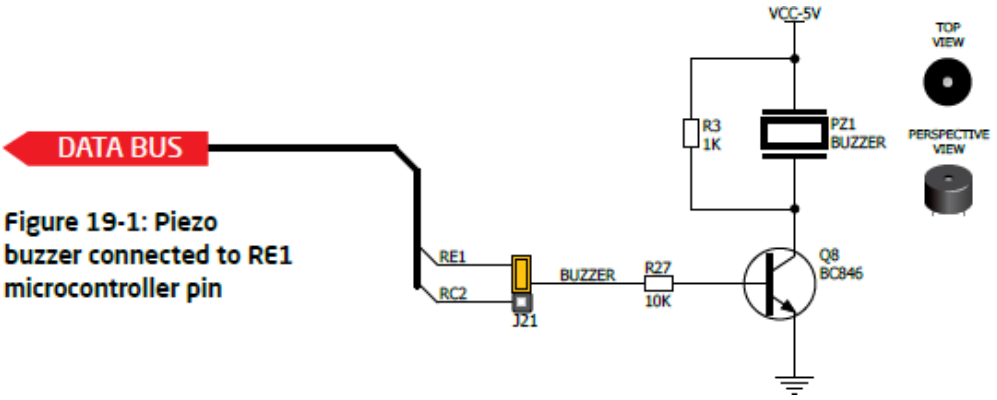



# 1. Objectifs

Produire des sons et mélodies simples grâce au buzzer connecte avec un générateur PWM sur la carte EasyPic7.

# 2. Le Buzzer

Comme le montre le schéma de câblage représenté ci-après, le Buzzer peut être connecté sur les broches RE1 ou RC2 du microcontrôleur :




Repérez sur la carte le jumper J21 et le placer de telle sorte que le buzzer soit relié au port RE1, comme indiqué sur le schéma. Positionnez également le jumper J17 sur la position Vcc.

Pour faire fonctionner le buzzer, vous utiliserez le module sound de la bibliothèque de MikroC, qui permet de génère des signaux PWM avec fréquence et durée variable. Seule 2 fonctions existent, une fonction Init et une fonction Play. Cette dernière permet seulement de renseigner la fréquence du son, mais pas son volume (rapport cyclique de la modulation PWM).

# 3. Génération de notes :

Dans cette partie, on vous propose de générer une série de notes afin d'accorder une guitare. L'accord d'une guitare peut être réalisé à l'aide d'un diapason qui émet un LA (440 Hz). Cette note correspond à la 5ème corde d'une guitare jouée à vide.

Pour accorder votre guitare, vous avez bien entendu besoin de générer les notes des 6 cordes (entre parenthèses après la note : l'octave) :

| | |
|---|---|
| 6ème corde ($RD_5$) : Mi (3) = 330 Hz | 5ème corde ($RD_4$) : La (3) = 440 Hz |
| 4ème corde ($RD_3$) : Ré (4) = 588 Hz | 3ème corde ($RD_2$) : Sol (4) = 830 Hz |
| 2ème corde ($RD_1$) : Si (4) = 988 Hz | 1ème corde ($RD_0$) : Mi (5) = 1320 Hz |

Pour passer d'une note à l'autre, vous utiliserez 6 boutons poussoirs du port D, de $RD_0$ à $RD_5$. Ne pas oublier de valider les pull-down des broches 0 à 5 du port D.

Pour chaque note, vous écrirez une procédure séparée selon le modèle suivant :

```
void Mi3() {
  Sound_Play(330, 1000); // Fréquence = 330Hz, durée = 1000 ms
```

```
}
```

Dans le programme principal, vous ferez tourner une boucle infinie while(1) à l'intérieure de laquelle vous testerez quel bouton poussoir est actif (faire un test if). Selon le bouton, vous lancerez la procédure requise, par exemple :

```
if (PORTD&0x20) // RD5 joue le Mi3();
while (PORTD&0x20) ; // Attente tant que le bouton est appuyé
```

Ne pas oublier d'initialiser correctement les ports (registres ANSELx, TRISx etc) ainsi que le buzzer : Sound_Init(&PORTE, 1);

Après avoir testé le programme, réessayer en commentant les lignes «while (PORT… ».

## 4. Composition de mélodies :

Maintenant que vous savez produire des notes, vous allez pouvoir jouer 2 mélodies, composées l'une et l'autre de 3 sons différents. L'idée ici est de basculer d'une mélodie à l'autre en appuyant sur un bouton poussoir. Pour l'instant, vous utiliserez la fonction Button mais dans l'exercice suivant, vous passerez par une interruption (INT0).

- Commençons par la première mélodie :

  Elle est composée de 3 tonalités à 659, 698 et 784 Hz. Créez 3 routines permettant de générer ces 3 sons pour une durée de 250 ms, vous les appellerez respectivement Tone1, Tone2 et Tone3. Recopiez ensuite la procédure suivante :

```
void Melody1() {          // Joue la mélodie "Yellow house"
        Tone1(); Tone2(); Tone3(); Tone3();
        Tone1(); Tone2(); Tone3(); Tone3();
        Tone1(); Tone2(); Tone3();
        Tone1(); Tone2(); Tone3(); Tone3();
        Tone1(); Tone2(); Tone3();
        Tone3(); Tone3(); Tone2(); Tone2(); Tone1();
}
```

- Seconde mélodie

  Elle est composée de 3 tonalités à 880, 1046 et 1318 Hz. Créez 3 routines permettant de générer ces 3 sons pour une durée de 50 ms, vous les appellerez respectivement ToneA, ToneC et ToneE. Recopiez ensuite la procédure suivante :

```
void Melody2() {
        unsigned short i;
        for (i = 9; i > 0; i--) {
                ToneA(); ToneC(); ToneE();
        }
}
```

Dans le programme principal, vous ferez tourner une boucle infinie while(1) à l'intérieure de laquelle vous testerez l'état du bouton poussoir $RD_0$. S'il est activé, la mélodie doit changer (utilisez un paramètre qui vaudra 1 ou 2 selon la mélodie à jouer, n'oublier pas de le déclarer – format char – et de l'initialiser !).

Si tout se passe bien, vous aurez remarqué que la mélodie ne bascule que si vous maintenez suffisamment longtemps votre doigt sur le bouton poussoir (en particulier s'agissant de la mélodie n°1). On comprend bien que Button ne génère pas une interruption, sinon une brève impulsion sur le bouton poussoir aurait dû la déclencher.

## 5. Composition de mélodies avec interruption :

L'utilisation du bouton poussoir sur le port $D_0$ n'est pas satisfaisante. Reprenez l'exercice précédent mais utiliser l'interruption INT0 (broche 0 du port B) pour basculer d'une mélodie à l'autre. Déclarer un variable flag de type bit qui vaudra 0 ou 1 selon la mélodie (ce n'est pas une obligation, l'idée est de tester un autre type de variable).

Que se passe-t-il maintenant lorsque vous appuyez brièvement sur le bouton poussoir? En quoi cette façon de procéder (i.e via une interruption) est-elle différente de la précédente?