INFO2 : INFORMATIQUE EMBARQUEE
TD4 : Utilisation des ports d'E/S - affichage multiplexé

1 Présentation

La carte d'évaluation EasyPIC v7 comporte quatre afficheurs sept segments à cathode commune. Le schéma électrique relatif à cette partie de la carte est donné ci-dessous :

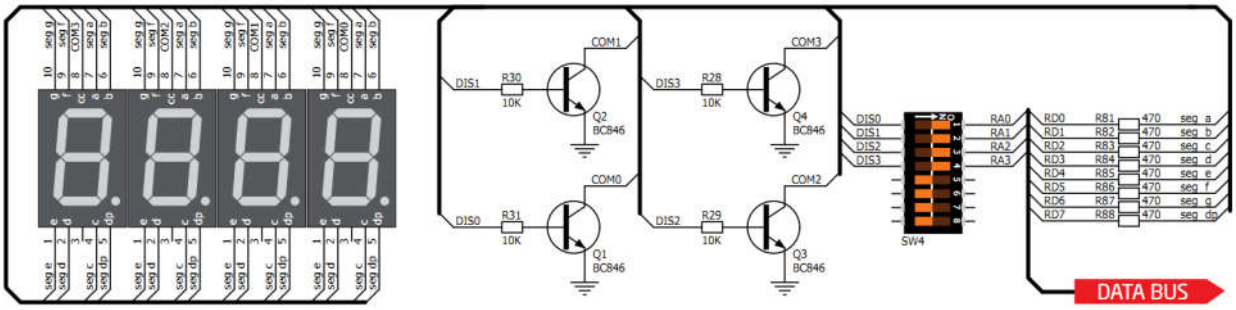


Figure 14-2: 4-digit 7-segment display schematic

EasyPIC v7

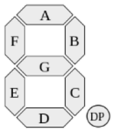
page 27

1. Si on utilisait une sortie du microcontrôleur par segment, combien en faudrait-il pour piloter les quatre afficheurs ? Est-ce réaliste ?

En fait, ces afficheurs sont à piloter par le microcontrôleur de la carte d'évaluation avec la technique dite de l'**affichage multiplexé**.

2. Avec cette technique, de combien de sorties du microcontrôleur a-t-on besoin seulement ? Un autre avantage est une consommation réduite. Quel est cela dit l'inconvénient de cette technique ?

3. Quels doivent être l'état des broches de RA3 à RA0 et de RD7 à RD0 pour que « 3 » soit affiché sur l'afficheur de poids faible ? On rappelle le nom classique des segments d'un afficheur dans la figure ci-dessous.



2 Affichage d'un chiffre sur un afficheur

4. Déclarer et initialiser un tableau **`tabValSeg`** de données contenant le niveau logique des segments à déposer sur le port D de façon à afficher le chiffre désiré (de 0 à 9). L'indice de l'élément du tableau correspond au chiffre à afficher. Par exemple, si on veut afficher un 3, **`tabValSeg[3]`** contient la valeur du port D que vous avez trouvée à la question précédente lorsque vous vouliez afficher un 3.

5. Définir une fonction **`affChiffre()`** affichant le chiffre **`pChiffre`** (un paramètre entre 0 et 9) sur l'afficheur de position **`pPosition`** (un nombre entre 0 (poids faible) et 3 (poids fort)). Le prototype est donné ci-dessous. On justifiera le choix des types des arguments de la fonction.

```
void affChiffre(unsigned char pChiffre, unsigned char pPosition);
```

3 Affichage d'un nombre sur les quatre afficheurs

On se propose d'afficher un nombre compris entre 0000 et 9999 sur les afficheurs sept segments.

3.1 Décomposition du nombre en unités, dizaines, centaines et milliers

6. Définir une fonction **`separe()`** dont le prototype est :

```
unsigned char separe(int pNombre, unsigned char pT[]);
// autre façon d'écrire possible :
unsigned char separe(int pNombre, unsigned char* pT);
```

Cette fonction reçoit un nombre et sépare les unités, dizaines, centaines et milliers et range ces données dans un tableau (l'indice 0 du tableau reçoit les unités).

La fonction retourne 0 si tout va bien, 1 s'il y a une erreur (nombre passé en argument négatif ou supérieur à 9999).

Exemple d'utilisation :

```
unsigned char tabChiffres[4];
int nb;
...
nb = 7543;
if(separe(nb, tabChiffres)==1)
{
        printf("\n Erreur : nombre incorrect");
}
else
 {
        // suite du programme si le nombre est correct
}
```

3.2 Affichage du nombre sur les afficheurs sept segments

7. Ecrire un programme qui :
   - Initialise les entrées/sorties nécessaires.
   - Puis, dans une boucle infinie :
     - Affiche les nombres de 0000 à 9999 avec un délai de 10 ms entre chaque incrémentation.

Ainsi, un afficheur est allumé 2.5 ms puis éteint 7.5 ms. Cette durée d'extinction est inférieure à la durée de persistance rétinienne, ce qui fait que l'œil ne perçoit pas l'extinction.

En outre, à condition de considérer que les instructions (sauf les temporisations) ont une durée d'exécution négligeable, cela permet de réaliser un compteur au 1/100ème de seconde (=> 100 incrémentations par seconde).

Dans les faits, le compteur retarde un peu justement parce qu'on n'a pas pris en compte la durée d'exécution des instructions. Nous verrons plus tard comment pallier à ce problème.

4 Cahier des charges d'un projet à réaliser en TP

8. Pour la raison évoquée un peu plus haut, ce chronomètre retarde. Il faut changer de modèle de programmation en utilisant un timer (le timer 0 par exemple) pour imposer un nouvel affichage tous les 10 ms exactement. Un modèle (n'utilisant pas le concept d'interruption) pourrait être celui-ci-dessous :

```
Initialisations diverses

Dans une boucle infinie :

        Si le flag du timer est levé :

                Le remettre à 0

                Réinitialiser TMR0

                Gérer l'affichage
```

9. Réaliser le projet dont le cahier des charges figure ci-dessous :

Réaliser sur l'afficheur un chronomètre au 1/100ème de seconde. Tant qu'on appuie sur RE0, le chronomètre compte. Quand on relâche RE0, le chronomètre se fige sur la valeur du temps écoulé. Un appui sur RE1 remet le chronomètre à 0.