TP3 PIC32MX
Générateur de signal piloté via menu

CADRE DE LA MANIPULATION

Cette manipulation illustre la gestion d'un menu simplifié pour un générateur de signal ainsi que la réalisation du générateur en utilisant le DAC sur le bus SPI du kit PIC32MX. L'affichage du menu est effectué sur l'afficheur LCD et la navigation est réalisée via le codeur incrémental, réalisant les incréments, décréments, action OK et ESC. Il doit en outre être possible de mémoriser le réglage obtenu dans la mémoire programme du PIC32MX via un bouton supplémentaire.

STRUCTURE DE MÉMORISATION DES PARAMÈTRES

Pour la manipulation des données du générateur, on utilisera la structure avec les champs suivants (fournie par le fichier DefMenuGen.h) :

E_FormesSignal	Forme	Code de la forme du signal
int16_t	Frequence	Fréquence en Hz (20 à 2000)
int16_t	Amplitude	Amplitude en mV crête (0 à + 10000)
int16_t	Offset	Offset en mV (-5000 à +5000)
uint32_t	Magic	Valeur permettant de vérifier la sauvegarde.

COMPORTEMENT DU PROGRAMME

Au démarrage on affichera : (pendant 3 s.)

```
Tp3 GenSig 17-18
Nom1
Nom2
```

Les valeurs des paramètres sont récupérées de la mémoire programme pour autant que l'on retrouve la valeur du champ "Magic".

Après cela, il faut afficher le menu sur les 4 lignes :

```
*Forme =  vvvvvvv
 Freq [Hz] =  ffff
 Ampl [mV] = aaaaa
 Offset [mV] = ooooo
```

Avec les rotations à droite et à gauche, il faut déplacer l'astérisque indiquant la ligne active. Avec la rotation à droite il faut descendre l'astérisque et à l'inverse, avec la rotation à gauche il faut monter.
L'élément sélectionné est celui qui est précédé d'une "*". Lorsque l'on presse sur la touche "OK", on active la possibilité de modifier la valeur, à ce moment-là il faut remplacer l'astérisque ("*") par un point d'interrogation ("?").
En plus, après une durée d'inactivité de 5 secondes, il faut éteindre le rétro-éclairage.

SÉLECTION FORME

Le menu comporte 4 formes :

1. Sinus
2. Triangle
3. DentDeScie
4. Carre

Lorsque l'on a activé la sélection de la forme, les rotations + et – du PEC12 font s'afficher successivement les noms des formes. Lors de la pression sur la touche OK la forme affichée est sélectionnée et on retourne à l'affichage principal. Si au lieu de OK on utilise ESC, alors on retourne à l'affichage principal et c'est l'ancienne forme qui est conservée et affichée.

SÉLECTION FRÉQUENCE

Lorsque l'on a activé la sélection de la fréquence, les rotations + et – du PEC12 font incrémenter ou décrémenter la valeur de la fréquence par **pas de 20 Hz**. Lors de la pression sur la touche OK, la fréquence affichée est sélectionnée et on retourne à l'affichage principal. Si au lieu de OK on utilise ESC, alors on retourne à l'affichage principal et c'est l'ancienne fréquence qui est conservée et affichée.

La plage de fréquence doit être limitée entre 20 et 2'000 Hz. Lorsqu'on dépasse 2'000, on reboucle à 20, et inversement si on essaie d'être inférieur à 20 on reboucle à 2'000.

SÉLECTION AMPLITUDE

Note : On parle ici d'amplitude crête.
Lorsque l'on a activé la sélection de l'amplitude, les rotations + et – du PEC12 font incrémenter ou décrémenter la valeur de l'amplitude par **pas de 100 mV**. Lors de la pression sur la touche OK, l'amplitude est affichée et sélectionnée, ensuite on retourne à l'affichage principal. Si au lieu de OK on utilise ESC, alors on retourne à l'affichage principal et c'est l'ancienne amplitude qui est conservée et affichée.

La plage d'amplitude doit être limitée entre 0 et 10'000 [mV]. Introduire un mécanisme de rebouclement, similaire à la fréquence

SÉLECTION OFFSET

Lorsque l'on a activé la sélection de l'offset, les rotations + et – du PEC12 font incrémenter ou décrémenter la valeur de l'offset par **pas de 100 mV**. Lors de la pression sur la touche OK, l'offset est affiché et sélectionné, ensuite on retourne à l'affichage principal. Si au lieu de OK on utilise ESC, alors on retourne à l'affichage principal et c'est l'ancien offset qui est conservé et affiché.

La plage d'offset doit être limitée entre -5'000 et +5'000 [mV]. Introduire un mécanisme de butée.

MENU DE SAUVEGARDE

La sauvegarde se fait en appuyant sur le bouton de droite (S9).
L'affichage devient alors :

```
Sauvegarde ?
(appui long)
```

Un appui long (S9) effectue la sauvegarde. Toute autre action l'annule.
En fonction du cas, un message "Sauvegarde OK !" ou " Sauvegarde ANNULEE !" s'affiche 2 s, puis retour au menu principal.

PRINCIPE DE LA GÉNÉRATION DU SIGNAL

Utilisation de **100 échantillons** par période du signal. Il est demandé d'utiliser un tableau précalculé d'échantillons contenant les valeurs à envoyer au DAC.

La fonction ***void GENSIG_Execute(void)*** est prévue pour effectuer l'envoi d'un échantillon au DAC à chaque appel. Elle est appelée dans la réponse à l'interruption du timer 3.

La variation de fréquence s'obtient en modifiant la période d'échantillonnage (timer 3). La fonction ***void GENSIG_UpdatePeriode(S_ParamGen *pParam)*** est responsable de modifier la période du timer 3 en fonction de la fréquence du signal.

Lors de la modification de la forme, amplitude ou offset du signal, la fonction ***void GENSIG_UpdateSignal(S_ParamGen *pParam)*** est appelée. Cette fonction recalcule le tableau de 100 échantillons.

GESTION DES MENUS

Les textes des menus peuvent être déclarés comme suit :

```
const char MenuFormes[4][21] =
     { "  Sinus", "  Triangle", "  DentDeScie", "  Carre" };
```

Pour afficher par exemple le 2ème élément du menu :

```
printf_lcd ("%s", MenuFormes[1]);
```

ORGANISATION DU PROJET

Pour faciliter la réalisation du projet vous trouverez dans le répertoire ...\Maitres-Eleves\SLO\Modules\SL229_MINF\TP\TP3_MenuGen\Fichiers_TP3, différents fichiers qu'il faut ajouter au projet que vous allez créer avec Harmony :

- App.h et App.c Application avec quelques compléments
- DefMenuGen.h Définitions pour le menu et complément
- Generateur.h et Generateur.c Fonctions du générateur **à implémenter**
- GesPec12.h et GesPec12.c Fonctions du PEC12 : **ScanPec12() à implémenter**
- MenuGen.h et MenuGen.c Fonctions du menu **à implémenter**

- Mc32Debounce.h et .c Librairie pour anti-rebond
- Mc32gestSpiDac.h et .c Librairie de gestion du DAC externe
- Mc32NVMUtil.h et .c Librairie de gestion de donnée en mémoire NVM
- Mc32SpiUtil.h et .c Librairie SPI (utilisée par Mc32gestSpiDac)

ELÉMENTS À INTRODUIRE DANS LE MHC

- BSP du kit
- Drivers timer (static avec interruption)
  - Timer 1, période 1 ms, priorité 3
  - Timer 3, période initiale 100 us, priorité 7

COMPLÉMENT DU FICHIER SYSTEM_INTERRUPT.C

Dans la réponse à l'interruption du timer 1 (période 1 ms) :

- A chaque cycle, appel de la fonction **ScanPec12()** et inversion LED_1 pour contrôle.
- Après 3000 cycles (3 s), appel de fonction **APP_UpdateState()** tous les 10 cycles.

Dans la réponse à l'interruption du timer 3 (cycle variable) :

- A chaque cycle, appel de la fonction **GENSIG_Execute()**. Ajouter LED0_W = 1 avant l'appel et LED0_W = 0 après l'appel pour contrôle.

COMPLÉMENT DES FICHIERS APP.C ET APP.H

Pour simplifier, les fichiers app.h et app.c sont fournis. Il faut mettre à jour l'affichage initial.

TEST INITIAL

Ces ajouts étant effectués, il faut vérifier si le système fonctionne. Vous devez obtenir:

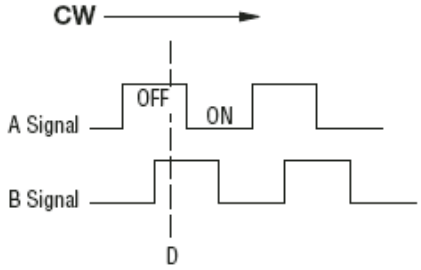
- Impulsions sur LED0, période 100 us.
- Un signal dent-de-scie descendant sur la sortie du DAC canal 0
- Un signal inversé à chaque ms sur LED1
- Un signal inversé à chaque 10 ms sur LED2

COMPLÉMENT DU FICHIER GESPEC12.C

Il faut rendre opérationnelle la fonction ScanPec12 en se basant sur les explications suivantes :

Les signaux A et B du codeur incrémental correspondent à un code gray sur 2 bits :

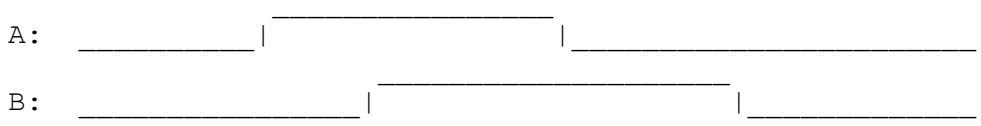
Rotation à droite
(Sens horaire = Clock Wise)



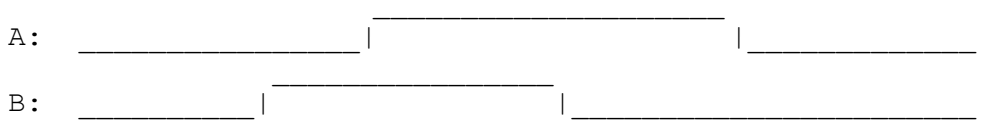
Les valeurs AB successives sont par exemple : 00, 01, 11, 10.

Ce dispositif est similaire à un codeur incrémental utilisé pour mesurer la position d'un axe moteur et donnant deux signaux déphasés mécaniquement de 90°.

Dans le sens horaire (CW) :

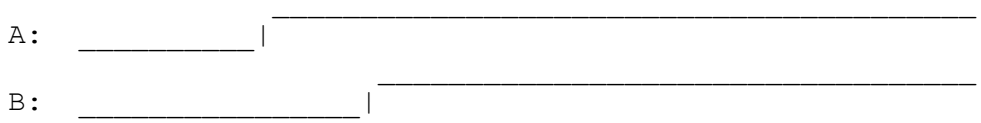


Dans le sens anti-horaire (CCW) :



Si nous n'effectuons qu'un cran, les signaux seront de la forme:

Dans le sens horaire CW:



Le temps entre la montée du signal B et celle du signal A dépend de la vitesse de rotation. Des essais ont montré que ce temps peut descendre aussi bas que 5 ms (à confirmer par une mesure), lorsque l'utilisateur tourne rapidement le bouton.

Il est par conséquent important que le temps d'analyse anti-rebond soit plus faible que les 10 ms usuelles. Un bon compromis paraît être 1 ms.
En plus des signaux A et B, ce codeur incrémental PEC12 propose également un bouton poussoir qu'on actionne simplement en pressant sur l'axe du bouton.

Dans une interface utilisateur classique, on a besoin de:

- une information d'incrémentation, généralement un bouton marqué "+"
- une information de décrémentation, généralement un bouton marqué "-"
- une information d'acceptation, généralement un bouton marqué "OK"
- une information d'annulation ou de remontée dans les menus, généralement un bouton marqué "ESC"

Le PEC12 nous offre l'information d'incrémentation (rotation dans le sens horaire) ainsi que l'information de décrémentation (rotation dans le sens anti-horaire).
Par contre, il ne nous offre qu'un switch. Il nous faut trouver une solution pour différencier l'opération OK de l'opération ESC.

Il suffit pour cela de différencier le temps de pression sur le bouton :

- Une pression de moins de 500 ms signifiera OK
- Une pression de plus de 500 ms signifiera ESC

DÉTAIL DU TRAITEMENT

Comme les signaux du PEC12 peuvent présenter des rebonds, il faut tout d'abord appliquer un traitement anti-rebonds classique. On effectue l'anti-rebond sur 3 cycles d'une milliseconde.
Une fois que l'on a les signaux A et B filtrés, il faut déterminer si il y eu un flanc montant ou descendant sur B, on détermine le sens de la manière suivante :

- Si B⇧ : Si A = 1 → rotation à droite (horaire = CW)
  Si A = 0 → rotation à gauche (anti-horaire = CCW)
- Si B⇩ : Si A = 1 → rotation à gauche (anti-horaire = CCW)
  Si A = 0 → rotation à droite (horaire = CW)

Pour obtenir l'information OK ou ESC, il faut appliquer l'anti-rebond sur le bouton.
Dès que, après filtrage, le bouton est On, il faut déclencher un compteur. Dès relâchement du bouton, il faut tester la valeur du compteur :

- Count < 500 → action OK
- Count >= 500 → action ESC

Remarque : il serait également possible d'établir automatiquement l'action ESC dès que la pression dure plus de 500 ms. Toutefois, une action uniquement au relâchement du bouton est plus naturelle.

Il faut encore détecter l'absence d'activité après 5 secondes afin éteindre le rétroéclairage de l'afficheur.

Le détail des fonctions à réaliser est fourni avec le canevas du fichier GesPec12.c.

UTILISATION DES FONCTIONS DU PEC12

Dans les fonctions de gestion du menu, il faut utiliser les fonctions fournies par le module GesPec12. Par exemple :

```
if ( Pec12IsPlus() )  {   // test si incrément
        Pec12ClearPlus();
        // traitement d'incrémentation
}
```

COMPLÉMENT DU FICHIER MENUGEN.C

Il s'agit de réaliser le contenu des fonctions suivantes :

- void MENU_Initialize(S_ParamGen *pParam)
- void MENU_Execute(S_ParamGen *pParam)

L'initialisation consiste à récupérer les paramètres sauvegardés (à réaliser par la suite), à initialiser toute les valeurs de gestion et à afficher le menu initial.

L'exécution consiste, en utilisant les fonctions du PEC12 + bouton, à répondre aux réglages de l'utilisateur et à appeler les fonctions de mise à jour du générateur.

☝ La fonction MENU_Execute est prévue pour un appel cyclique. Elle doit être basée sur une machine d'état et ne pas être bloquante.

COMPLÉMENT DU FICHIER GENERATEUR.C

Il s'agit de réaliser le contenu des fonctions suivantes :

- void GENSIG_Initialize(S_ParamGen *pParam)
- void GENSIG_UpdatePeriode(S_ParamGen *pParam)
- void GENSIG_UpdateSignal(S_ParamGen *pParam)

L'initialisation consiste à paramétrer le générateur selon les réglages initiaux de fréquence, forme, amplitude et offset.

La fonction de mise à jour de la période modifie la période du timer 3 en fonction de la fréquence.

La fonction de mise à jour du signal consiste à recalculer les valeurs de la table d'échantillons en fonction de la forme, de l'amplitude et de l'offset fournis.

```
// T.P. 2016 100 echantillons
#define MAX_ECH 100


// Execution du générateur
// Fonction appelée dans Int timer3 (cycle variable variable)

// Version provisoire pour test du DAC à modifier
void  GENSIG_Execute(void)
{
   static uint16_t EchNb = 0;
   const uint16_t Step = 65535 / MAX_ECH;

   SPI_WriteToDac(0, Step * EchNb );      // sur canal 0
   EchNb++;
   EchNb = EchNb % MAX_ECH;
}
```

Cette fonction est à adapter en vue d'utiliser la table d'échantillons précalculée (sous cette forme, elle génère une dent de scie pour test).

SAUVEGARDE ET RESTAURATION DES PARAMÈTRES

Un complément sera fourni pour la description des actions de ces fonctions et le mécanisme d'écriture/lecture de la mémoire programme avec les plib harmony.

TRAVAIL ET ÉTABLISSEMENT DU RAPPORT

Le travail s'effectue par groupe de deux en partageant la réalisation, ce qui permet d'acquérir des compétences en collaboration dans la réalisation d'un programme.

A rendre au plus tard à la fin de la 2ème séance :

- Organigramme (flowchart) ou GNS (Nassi-Schneidermann) de la fonction ScanPec12. A compléter par des explications.
- Diagramme décrivant le mécanisme de navigation dans le menu.

Le rapport final doit contenir au minimum les éléments suivants :

- Mesure des signaux A et B du PEC12, en rotation lente et rapide, gauche et droite, avec observation des éventuels rebonds.
- Explications sur l'obtention des valeurs des 100 échantillons pour chaque signal, ainsi que de la modification de la fréquence, de l'amplitude et de l'offset du signal
- Des copies d'écran d'oscilloscope des signaux obtenus à 20 Hz et 500 Hz, avec les différentes formes (amplitude au max. et offset à 0).
- Des copies d'écran d'oscilloscope montrant différents réglages d'amplitude et d'offset sur un signal triangle. Valeurs en relation avec la fiche de contrôle.
- Une copie d'écran d'oscilloscope montrant la durée du traitement d'un échantillon (utilisation d'une sortie activée au début de la réponse et désactivée à la fin de la réponse à l'interruption). En observant simultanément les signaux CS_DAC, SCK et MOSI du bus SPI avec un signal à 800 Hz.
- Détermination de la fréquence maximum et mesure de la durée du traitement de l'échantillon proche de la fréquence limite.
- Une copie d'écran d'oscilloscope d'une seule période temporelle du signal : sinus, fréquence 1 kHz, amplitude 7,5 V, offset 0 mV. Echelles optimales.
- Une copie d'écran d'oscilloscope de la FFT du sinus précédent. Echelles optimales (pour une résolution fréquentielle optimale, attention au nombre de périodes du signal source).
- Calcul de la qualité du sinus (rapport total de distorsion, TDR) basé sur la mesure précédente.
  Pour simplifier, on ne tient compte que de la première raie de bruit :

  $$TDR = \frac{A_{harmoniques+bruit}}{A_1} = \frac{\sqrt{{A_{tot}}^2 - {A_1}^2}}{A_1} \approx \frac{A_{bruit,1}}{A_1}$$

  A_1 : amplitude RMS de la fondamentale
  A_{tot} : amplitude RMS totale du signal
  $A_{bruit,1}$: amplitude RMS de la 1ère raie du bruit

  Quelle est la fréquence principale de ce bruit est à quoi est-il dû ?
- Listing des programmes en C (app.c, system_interrupt.c, Generateur.c, MenuGen.c, fonction ScanPec12 de GesPec12.c). A fournir en annexe.
- Le fonctionnement doit être démontré en suivant la fiche de contrôle.

Votre travail sera évalué sur la base de :

- Qualité, facilité de réutilisation/modification et taux d'aboutissement du code.
- Fonctionnement et taux d'aboutissement.

DURÉE DE LA RÉALISATION

A réaliser en 6 séances. Le volume de travail est assez important. Il est nécessaire de partager la réalisation et de collaborer au mieux au sein du groupe.