

TP 3

Circuits numériques et microcontrôleurs

Périphériques E/S & Bus

Préparation

En plus de la documentation du TP 2 servant de base pour ce TP, il est également demandé de consulter la documentation technique des convertisseurs CAN et CNA. Plus précisément, focalisez-vous sur les chronogrammes pour interfacer ces composants. Des explications complémentaires sont données dans ce document de TP, pour améliorer la compréhension.

Les convertisseurs Analogique <==> Numérique sont implantés sur la carte d'extension qui sera interfacée à la carte FPGA DE1 par le port d'extension GPIO_1.

Sur la base des documents de référence concernant la carte DE1, des convertisseurs CNA, CAN et des schémas de la carte d'extension donnés en annexe 1, il est demandé de répondre aux questions suivantes :

- 1) Identifier les convertisseurs CAN et CNA sur ces schémas et les signaux de contrôle associés.
- 2) Comment sont câblés les bus de données des composants CAN et CNA ?
- 3) Proposer une solution fonctionnelle pour accéder à ces deux composants dont l'un est en lecture seule et l'autre est en écriture seule.
- 4) Sur la base du texte explicatif de l'exercice de ce TP, décrire un automate permettant de lancer une conversion Analogique/Numérique puis de lire la valeur de celle-ci et l'afficher sur les LEDs vertes (LEDG).

Exercice de conception : contrôler des circuits CAN et CNA

On souhaite utiliser les composants CAN et CNA de la carte d'extension fille, qui sera branchée sur le port GPIO_1 de la carte DE1. On va d'abord s'interfacer à ces composants en développant une interface dite « bas-niveau », en réalisant une architecture matérielle décrite en VHDL. Le cahier des charges sera le suivant :

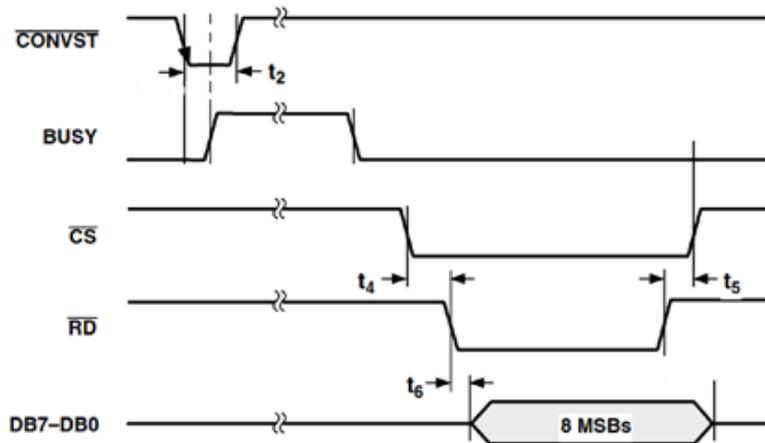
- Utiliser un automate unique permettant de piloter les convertisseurs CNA et le CAN. L'horloge de 50MHz (sortie de PLL) sera utilisée.
- Utiliser les interrupteurs SW comme valeur qui sera transmise au CNA pour la convertir en analogique.
- Utiliser les LEDs vertes pour afficher le résultat de la conversion du CAN de la tension analogique.
- On bouclera la sortie analogique du CNA vers le CAN. Ainsi, la valeur des interrupteurs SW vont subir une conversion analogique puis cette tension analogique sera convertie par un CAN pour ensuite afficher le résultat sur les LED vertes.

Pour faciliter la compréhension des chronogrammes des convertisseurs CAN et CNA, nous présentons une petite étude de ces composants et de leur interface en simplifiant le plus possible les signaux de contrôle. En effet, du fait que nous contrôlerons totalement ces composants par une description VHDL (donc un contrôle non standardisé), et compte tenu de certaines caractéristiques temporelles des signaux de contrôle, nous faisons les constats suivants :

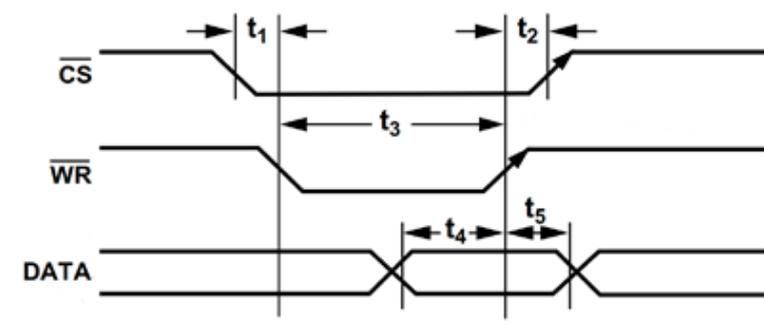
- Pour le convertisseur CNA :
 - les timings indiquent WR sera mis à 0 après avoir mis CS à 0. On remettra WR à 1 avant CS. Une période d'horloge à 50Mhz suffira.
 - Le signal LDA permet de valider la sortie de la donnée vers l'analogique. Ce signal peut être mis à 0 tout le temps pour une conversion en continu.
 - Tous les autres signaux du circuit CNA sont câblés physiquement sur la carte fille.
- Pour le convertisseur CAN :
 - les timings indiquent que nous pouvons contrôler le CS et le RD simultanément car les délais entre ces deux signaux peuvent être nuls

- o Le signal CONVST doit être maintenu à 1 par défaut. Son passage à 0 permet d'initier une conversion Analogique > Numérique. Il est important de mettre le signal CONVST à 0 pendant 2 ou 3 cycles d'horloge (50MHz) puis le repasser à 1. On s'assure ainsi que CONVST soit passé à 1 après que BUSY est passé au niveau 1.
- o Le signal BUSY passe à 1 pendant la conversion. Le passage à 0 du signal indique la fin de conversion. Le temps de conversion est de l'ordre de 4µs.
- o On doit respecter un minimum de 100ns entre la fin de la transition 0->1 du signal CS et le passage à 0 du signal CONVST (pour lancer une nouvelle conversion).

La figure ci-dessous illustre le chronogramme de contrôle des signaux du convertisseur Analogique/Numérique CAN :



La figure ci-dessous illustre le chronogramme de contrôle des signaux du convertisseur Numérique/Analogique CNA :



La figure de la page 5 (de ce sujet) illustre le diagramme temporel (timing) des signaux de contrôle pour contrôler les convertisseurs CAN et CNA.

1. Réaliser l'automate qui permet de contrôler le CNA et qui produit une tension analogique en fonction de la valeur des interrupteurs SW.
2. Ajoutez dans cet automate la partie permettant de piloter le convertisseur CAN et afficher la valeur de la tension convertie sur les LED vertes (LEDG). On rebouclera la tension du CNA sur l'entrée du CAN pour vérifier ainsi le bon fonctionnement de l'ensemble.

Exercice 2 : rendre ces périphériques visibles par le processeur NIOS-II

On souhaite maintenant interfacer ces convertisseurs CAN et CNA au bus microprocesseur NIOS-II. On utilisera l'automate précédent pour interfacer les convertisseurs au travers de 2 registres (un entrant et l'autre sortant). Pour cela, on va devoir intervenir dans l'architecture du processeur, au niveau de l'outil QSYS pour ajouter ces 2 registres.

3. Ajouter ces deux registres d'entrées/sorties au niveau du bus AVALON avec l'outil QSYS.
4. Identifier les adresses E/S des deux ports pour accéder aux CNA et CAN.
5. Ecrire le programme en C, sur le processeur NIOS-II, pour vérifier le fonctionnement de l'interfaçage. On s'interfacera également aux interrupteurs et aux LED vertes.

Exercice 3 : Utiliser le système comme générateur sinusoïdal

On souhaite maintenant générer un signal sinusoïdal sur la sortie du convertisseur CNA. Le cahier des charges pour cette manipulation est le suivant :

- La période de cette sinusoïde sera de 500 μ s.
 - On assurera 25 échantillons par période pour obtenir une bonne qualité du signal généré. On utilisera pour cela une LUT.
 - Pour garantir une qualité de la période de génération des échantillons, on utilisera une interruption d'un TIMER qu'il faudra configurer à la bonne période.
6. Modifier l'architecture du processeur, sur QSYS, pour ajouter le TIMER et prendre en compte les contraintes du cahier des charges. Générer l'architecture. Compiler ensuite l'architecture sur QUARTUS.
 7. Réaliser le programme permettant de générer le signal sinusoïdal dans le respect du cahier des charges. Visualiser le signal sur un oscilloscope.

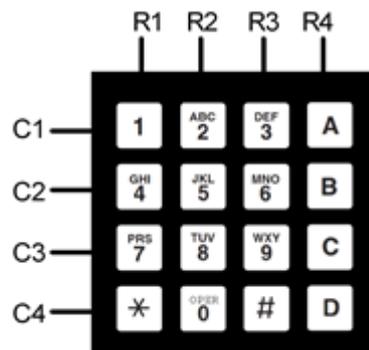
Exercice 4 : Interfacer un clavier au processeur NIOS-II

On souhaite interfacer le clavier de la carte fille. Ce travail nécessite 3 étapes :

- A- Interfacer les pins du clavier au niveau du FPGA en respectant le fait que le niveau logique 1 soit récessif et le niveau 0 soit dominant. Pour simplifier cela, il y a des résistances PULLUP déjà placées sur les pins du clavier. Ainsi, pour mettre un signal à 1 récessif, il suffit que le pin du FPGA associé soit en haute impédance.
- B- On doit interfacer les 8 bits du clavier au niveau du processeur NIOS. Pour cela, on utilisera un registre en lecture (clavin) et écriture (clavout) séparées. On pourra alors contrôler chaque pin du clavier avec le registre en écriture et lire son état grâce au registre en lecture.
- C- Il faudra écrire un programme en C pour réaliser l'automate de contrôle du clavier tel qu'il a été étudié en TD.

Concernant les points A et B, on répondra en intervenant sur chaque pin des signaux du clavier de la manière suivante : par exemple pour le bit 0 du registre clavier

```
clavin(0)    <= GPIO_1(20); -- lire la pin
GPIO_1(20) <= 'Z' when  clavout(i)='1' else
              '0';         -- 0 dominant et 1 récessif
```



- 8. Modifier l'architecture du système NIOS sur l'outil QSYS pour ajouter un port IO avec lecture et écriture séparées.
- 9. Modifier le VHDL pour assurer la liaison entre les 2 ports du processeur NIOS et l'interface au clavier de la carte fille.
- 10. Ecrire en C un programme de l'automate de détection d'une touche appuyée et une fonction qui permet de décoder cette touche par rapport à la signification de chaque touche (se référer aux exemples vus en TD).

