Auteur : Spinelli Isaia et

Lankeu Ngassam Cédric

Prof : [Etienne](https://cyberlearn.hes-so.ch/user/view.php?id=104149&course=1) Messerli

Ing : Sébastien Masle

Date : 28.11.2019

Salle : A09

Classe : CSN

Acquisition de position

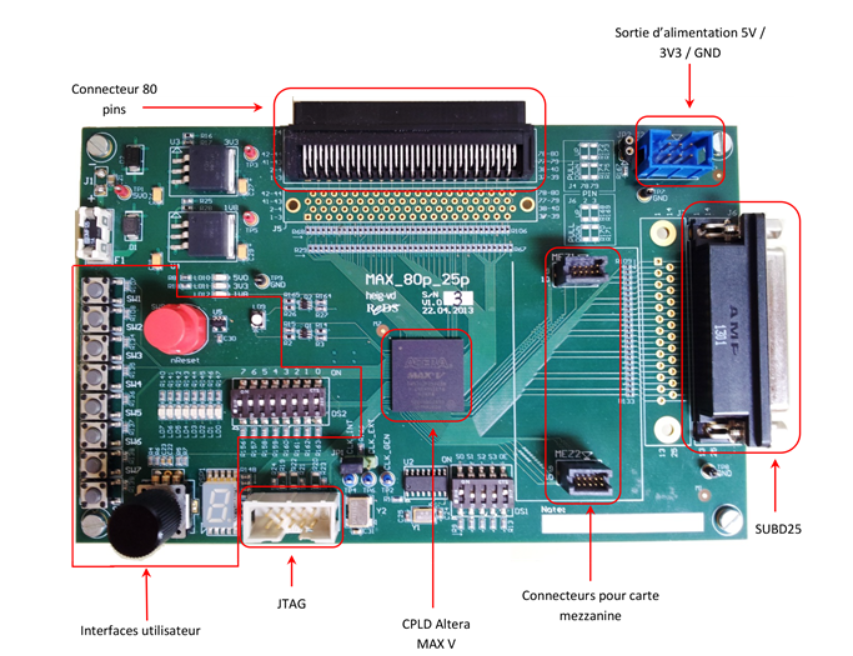
[CONCEPTION DE SYSTÈMES NUMÉRIQUES (CSN)](https://cyberlearn.hes-so.ch/course/view.php?id=14116#section-1) 

Table des matières

[Objectifs - 2 -](#_Toc25929717)

[Présentation du système - 2 -](#_Toc25929718)

[Spécification du système - 3 -](#_Toc25929719)

[Analyse du fonctionnement - 4 -](#_Toc25929720)

[Schéma bloc - 5 -](#_Toc25929721)

[Machine d’état - 6 -](#_Toc25929722)

[Compteur et décodeur d’état futur - 8 -](#_Toc25929723)

[Bascule RS - 8 -](#_Toc25929724)

[Regroupement des blocs - 9 -](#_Toc25929725)

[Synthèse ou quantité logique - 9 -](#_Toc25929726)

[Vérification du fonctionnement - 9 -](#_Toc25929727)

[Test du système - 10 -](#_Toc25929728)

[Conclusion - 11 -](#_Toc25929729)

[Difficultés rencontrées - 11 -](#_Toc25929730)

[Compétences acquises - 11 -](#_Toc25929731)

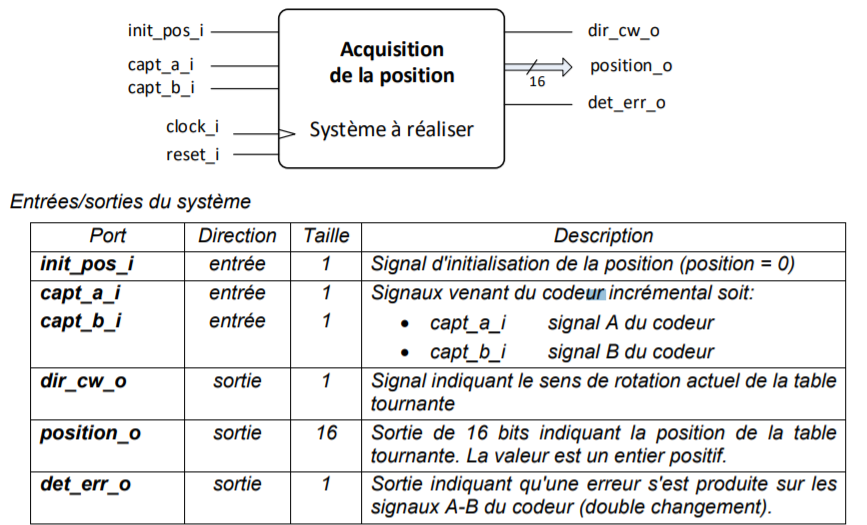
[Résultats obtenus - 11 -](#_Toc25929732)

## Objectifs

Nous souhaitons réaliser un système permettant de réaliser l’acquisition de la position d’un disque tournant. Nous allons utiliser la plateforme Servo-USB qui comprend un codeur incrémental entrainé par le disque qui nous permet de mesurer la position de celui-ci.

## Présentation du système

Le système reçoit les informations du codeur incrémental ainsi qu’un signal d’initialisation de la position. Voici le symbole du système à réaliser :





## Spécification du système

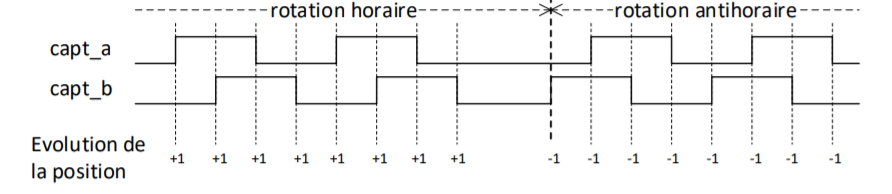
Le système reçoit aussi un signal d’initialisation permettant une remise à zéro de la position. Description du fonctionnement de la position :

* Lorsque le signal init\_pos\_i est actif, la position est initialisée à 0. D'autre part le signal d'erreur det\_err\_o est désactivé.
* Sinon: La position doit être incrémentée ou décrémentée à chaque changement d’état d’un des capteurs du codeur, soit :
  + Si le codeur tourne dans le sens horaire, la position est incrémentée à chaque changement d'état d’un des capteurs (capt\_a\_i ou capt\_b\_i)
  + Si le codeur tourne dans le sens anti-horaire, la position est décrémentée à chaque changement d'état d’un des capteurs (capt\_a\_i ou capt\_b\_i)

Lors d’un changement simultané des signaux capt\_a\_i et capt\_b\_i durant la même période d’horloge, le signal det\_err\_o doit être activé.

* En effet dans ce cas, il est impossible de déterminer dans quel sens le codeur a tourné !

Voici un exemple de fonctionnement :



Spécifications de la table tournante :

Le codeur de la table est de type HEDS-5310. Celui-ci a une résolution de 500 traits. Cela permet d'obtenir 2000 incréments par tour du codeur en comptabilisant tous les changements des signaux A et B (multiplication par 4). Un tour de la table tournante correspond à 5 tours du codeur, donc à un total de 10'000 incréments.

## Analyse du fonctionnement

Nous avons commencé par analyser le fonctionnement du système d’acquisition de position et définir une décomposition.

Nous avons rapidement compris qu’il fallait une machine d’état pour gérer les deux différentes séquences des capteurs ainsi qu’un compteur pour 16 bits afin de suivre la position du disque tournant.

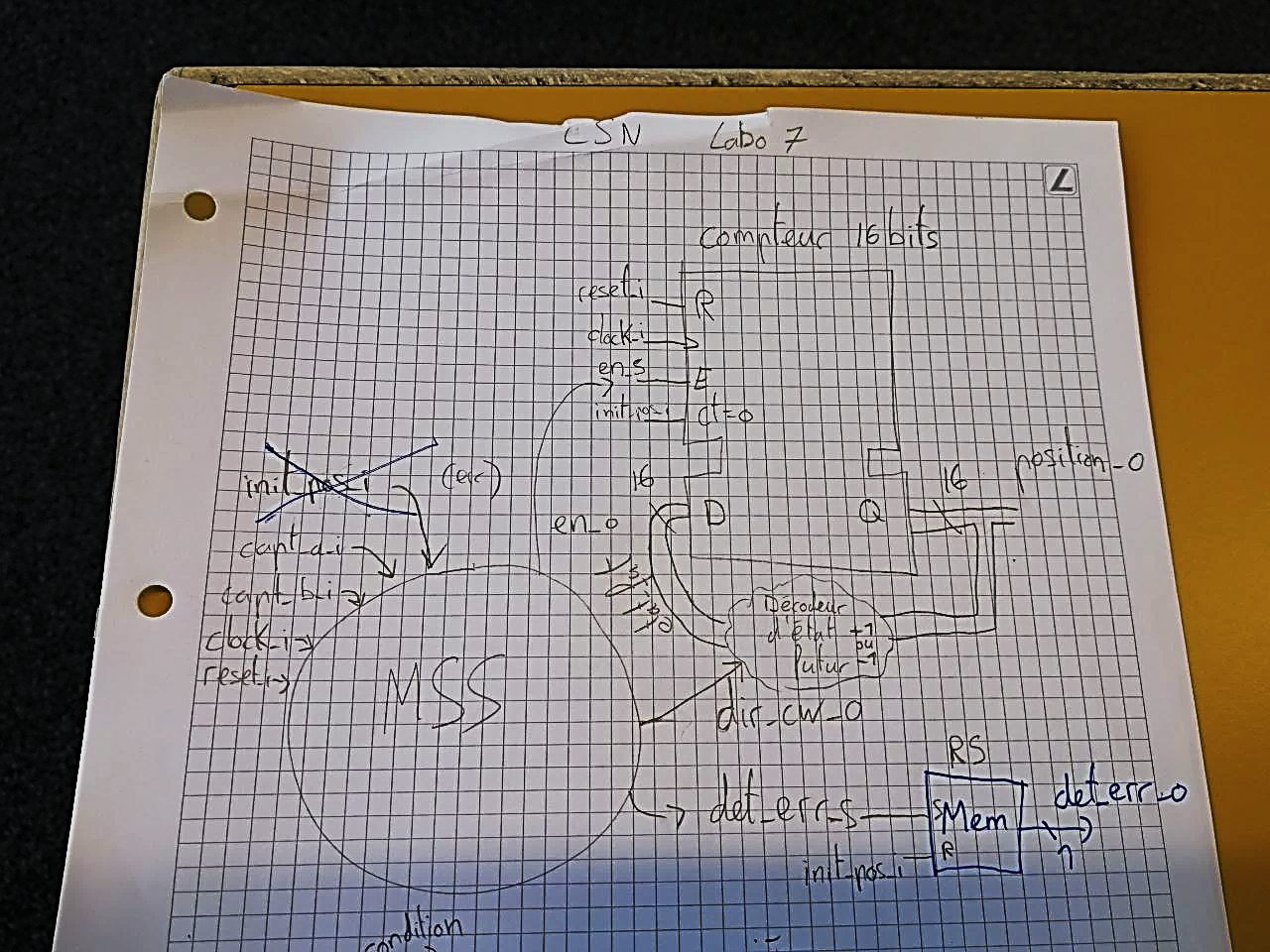
Plusieurs sorties sont demandées pour ce système comme détaillé précédemment. La sorite « **position\_o** » sur 16 bits est la sortie du compteur qui indiquera la position de la table tournante.

La machine d’état s’occupera de fournir les sorties « **dir\_cw\_o** » afin d’indiquer la direction de rotation de la plaque. Cette sortie devra être utiliser par le compteur afin de savoir si la position doit être incrémenté ou décrémente.

Finalement la sortie « **det\_err\_o** » doit être fourni afin d’indiquer qu'une erreur s'est produite sur les signaux A-B du codeur (double changement).

## Schéma bloc

Afin d’avoir une bonne vue globale du système nous avons décidé de faire un schéma en indiquant les différents points importants pour ce système :



En peut voir en bas à gauche notre machine d’état (MSS) avec comme entrées le reset, clock, capteur\_a et capteur\_b ainsi que comme sorties un enable, dir\_cw\_o (direction de rotation) et le det\_err.

En haut à droite notre compteur qui prend en entrée le reset, le clock, le enable fourni par la MSS, le signal init\_pos\_i afin d’initialiser la position (pos = 0) et D qui est la position future en fonction du signal « dir\_cw\_o » aussi fourni par la MSS. Puis comme sortis, la position actuelle de la plaque.

Nous avons décidé de gérer l’erreur avec une bascule RS (en bas à droite) qui nous permettra de garder l’information qu’une erreur s’est produite tout en continuant à faire fonctionner le système. De plus, comme demander, le signal « init\_pos\_i » doit permettre de reset l’erreur à 0. Il n’est pas indiqué sur notre schéma, mais le reset sera bien évidemment braché sur la bascule RS afin de remettre aussi à 0 une éventuelle erreur.

Comme mentionné plus tôt, la sortie « dir\_cw\_o » permettra d’indiquer au compteur dans quel sens tourne la plaque et donc de savoir s’il faut incrémenter ou décrémenter la position actuelle.

Finalement, il est important d’indiquer au compteur à quel moment la position de la plaque change afin d’effectuer une opération sur la position présente. Pour ceci, nous fournissant un signal « enable » au compteur qui sera actif lors d’un changement d’état d’un des capteurs afin qu’il puisse mettre à jour la position de la plaque.

### Machine d’état

#### Conception

Afin de concevoir cette machine d’état nous nous sommes aidés du principe de la table des états. Afin de la concevoir, il a d’abord fallu définir nos différents états :

1. Etat de départ lors d’un allumage du système ou d’un reset **(Start)**
2. Etat d’une erreur lors d’un double changement sur les capteurs **(Err)**
3. Quarts états pour chaque position du codeur incrémental :
   1. Activation du enable dans le sens horaire **(Exy+)**
   2. Attente d’un changement de position dans le sens horaire **(Exy+w)**
   3. Activation du enable dans le sens anti-horaire **(Exy-)**
   4. Attente d’un changement de position dans le sens anti-horaire **(Exy-w)**

***X = état du capteur a***

***Y = état du capteur b***

En tout, cela nous fait 1 + 1 + (4 x 4) états (**16**). Il est important de faire deux états distincts, lors de la détection d’un changement d’état et lors de l’attente d’un changent de position. Cela nous permet d’activer le signal « enable » un seul coup de clock pour que le compteur effectue qu’une opération, et non pas plusieurs, pour un seul changement de position.

Voici notre table des états :

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Etats** | **Entrées f(capt\_a\_i, capt\_b\_i)** | | | | **Sorties** | | |
| **00** | **01** | **11** | **10** | **en\_o** | **dir\_cw\_o** | **det\_err\_s** |
| **Start** | E00+w | E01+w | E11+w | E10+w | 0 | 0 | 0 |
| **E00+** | E00+w | E00+w | E00+w | E00+w | 1 | 0 | 0 |
| **E00+w** | **(E00+w)** | E01- | Err | E10+ | 0 | 0 | 0 |
| **E10+** | E10+w | E10+w | E10+w | E10+w | 1 | 0 | 0 |
| **E10+w** | E00- | Err | E11+ | **(E10+w)** | 0 | 0 | 0 |
| **E11+** | E11+w | E11+w | E11+w | E11+w | 1 | 0 | 0 |
| **E11+w** | Err | E01+ | **(E11+w)** | E10- | 0 | 0 | 0 |
| **E01+** | E01+w | E01+w | E01+w | E01+w | 1 | 0 | 0 |
| **E01+w** | E00+ | **(E01+w)** | E11- | Err | 0 | 0 | 0 |
| **E00-** | E00-w | E00-w | E00-w | E00-w | 1 | 1 | 0 |
| **E00-w** | **(E00-w)** | E01- | Err | E10+ | 0 | 1 | 0 |
| **E01-** | E01-w | E01-w | E01-w | E01-w | 1 | 1 | 0 |
| **E01-w** | E00+ | **(E01-w)** | E11- | Err | 0 | 1 | 0 |
| **E11-** | E11-w | E11-w | E11-w | E11-w | 1 | 1 | 0 |
| **E11-w** | Err | E01+ | **(E11-w)** | E10- | 0 | 1 | 0 |
| **E10-** | E10-w | E10-w | E10-w | E10-w | 1 | 1 | 0 |
| **E10-w** | E00- | Err | E11+ | **(E10-w)** | 0 | 1 | 0 |
| **Err** | E00+w | E01+w | E11+w | E10+w | 0 | 0 | 1 |

*dir\_cw\_o à 0 = sens de rotation horaire.*

*Remarque :*

Dans l’état Start et Err, nous ne connaissance momentanément plus le sens de rotation. De ce fait, l’état qui suit un de ces états est directement un état wait (attend un prochain changement) afin de ne pas faire d’opération inconnue sur la position. Dans notre cas, nous avons choisis de se mettre dans les états ou la rotation est sens horaire mais cela n’a pas importante.

Afin d’avoir une meilleure vision du fonctionnement ains que pour facilement se repérer dans le VHDL, nous avons tout de même fait un graphe d’état :

#### Description VHDL

#### Vérification du fonctionnement

### Compteur et décodeur d’état futur

Afin d’avoir un suivi de la position, il est indispensable d’avoir un compteur avec un décodeur d’état futur. Il y a donc un registre de 16 bits afin de mémoriser ou initialiser la valeur de la position ainsi qu’un décodeur d’état futur afin de savoir s’il faut incrémenter ou décrémenter la position actuelle.

#### Conception

Voici la table des fonctions synchrone du compteur :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Init\_pos\_i** | **En\_i** | **dir\_cw\_i** | **Cpt\_pres** | **Cpt\_fut** | **Fonctions** |
| 1 | - | - | - | = 0 | Initialisation de la position à 0 |
| 0 | 0 | - | . | = Cpt\_pres | Maintien de la position |
| 0 | 1 | 0 | - | = Cpt\_pres+1 | Incrémentation de la position |
| 0 | 1 | 1 | - | = Cpt\_pres-1 | Décrémentation de la position |

On peut voir que notre compteur est relativement simple. Il y a une fonction prioritaire d’initialisation (init\_pos\_i) commander par l’utilisateur. De plus, il y a une fonction d’activation (en\_i) fourni par la machine d’état dans le cas où il y a un changement de position. Finalement, si l’enable est actif, en fonction de l’entrée « dir\_cw\_i » aussi fourni par la MSS, une opération d’incrémentation ou de décrémentation en fonction du sens de rotation.

#### Description VHDL

#### Vérification du fonctionnement

### Bascule RS

Le dernier bloc est une simple bascule RS permettant d’enregistrer l’arrivé d’une erreur.

#### Conception

Le signal d’erreur levé par la MSS est directement connecté à l’entrée S de la bascule afin de la mémoriser.

Étant donné que l’erreur doit pouvoir être désactivé avec le signal « init\_pos\_i » celui-ci sera placé sur l’entrée R de la bascule.

Finalement, le signal reset\_i est bien évidemment branché sur le reset de la bascule.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Init\_pos\_i** | **Det\_err** | **Q\_pres** | **Q\_fut** | **Fonction** |
| 0 | 0 | - | Q\_pres | Mémorisation |
| 0 | 1 | - | 1 | Mise à 1 de l’erreur |
| 1 | 0 | - | 0 | Mise à 0 de l’erreur |

#### Description VHDL

#### Vérification du fonctionnement

## Regroupement des blocs

### Synthèse ou quantité logique

### Vérification du fonctionnement

## Test du système

## Conclusion

### Difficultés rencontrées

### Compétences acquises

### Résultats obtenus

*Date : 12.12.19*

*Nom des étudiants : Lankeu Ngassam Cédric et Spinelli Isaïa*