

**PROJET : CONVERTISSEUR ANALOGIQUE-  
NUMÉRIQUE PAR APPROXIMATION SUCCESSIVE  
(PARTIE LOGIQUE)**

KASSIN Kelly Yann Lionnel

**Encadré par** Monsieur François JACOB

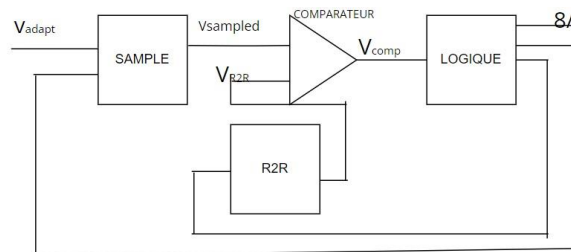
ELSE 3A FISE

2024/2025

## Introduction

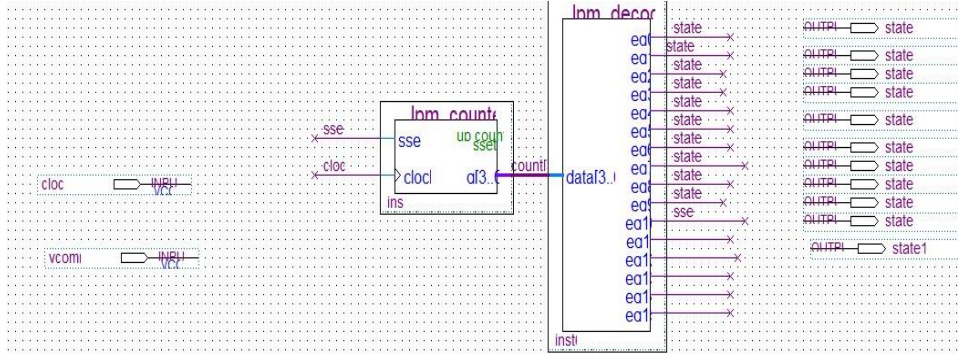
Dans le convertisseur Analogique Numérique par Approximation Successive, la partie logique représente le circuit de commande. Elle contrôle le **SAMPLE** qui échantillonne le signal à convertir et le **R2R** qui fournit une tension qui est comparée à la tension échantillonnée par le **SAMPLE**.

### Schéma du circuit



#### 1. Réglage de la logique sur l'horloge :

La logique sera réglée sur l'horloge et la conversion se fera sur 8 bits. On aura besoin de 10 coups d'horloge, 8 pour l'obtention de chacun des 8 bits, 1 pour activer la SAMPLE, et le dernier pour lui signaler la fin de la conversion de l'échantillon. Pour cela il est nécessaire de faire un compteur qui compte 10 coups d'horloge ne commençant pas par 0, car ce dernier est indépendant de l'horloge. Pour cela il faut créer sur **Quartus II** un compteur muni d'un set à 1. On utilise un Decodeur appelé **lpm\_decode** pour repérer les différents états possibles sur 4 bits. Cela permet de régler le set à l'état 10 et créer un compteur de 1 à 10.



#### 2. Methode de gestion des bits :

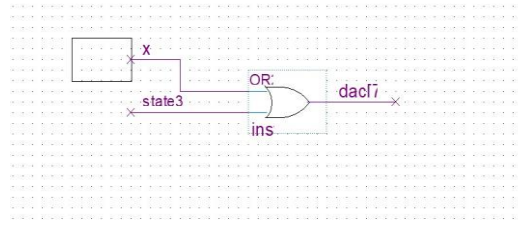
On note les bits  $dac[7]$  (MSB),  $dac[6]$  ...  $dac[0]$  (LSB). La logique est censée :

- 1- Mettre chaque bit à 1
- 2- Le remplacer par **Vcomp**
- 3- Mémoriser le bit

Le **Sample** est représenté par **state1** qui informe sur la fin de la conversion de l'échantillon.

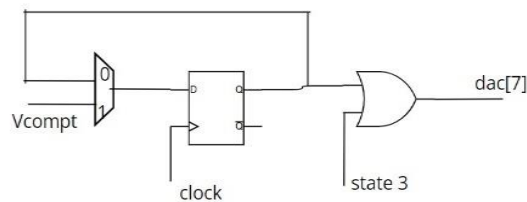
- **Mettre le bit à 1 :**

Pour mettre chaque bit à 1, il faut utiliser la porte OR à 2 entrées avec une entrée étant l'état où le bit est traité. On met cet état à 1.

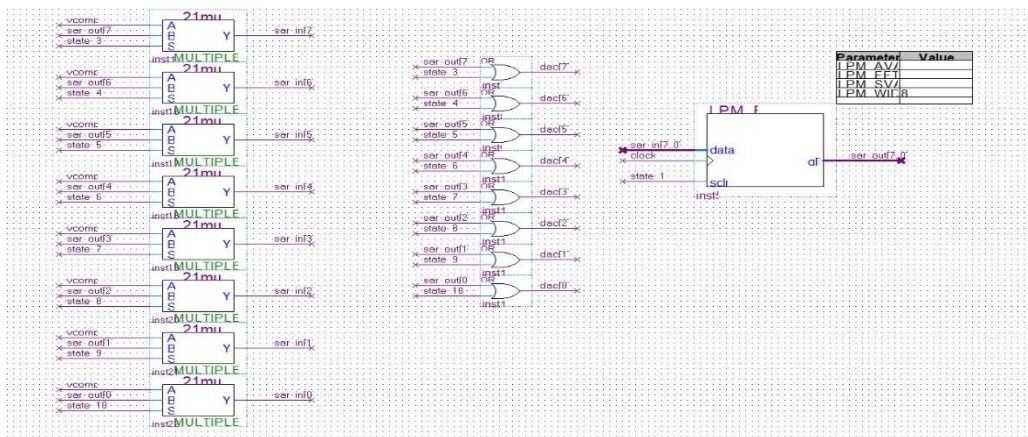


## • Remplacement du bit et mémorisation

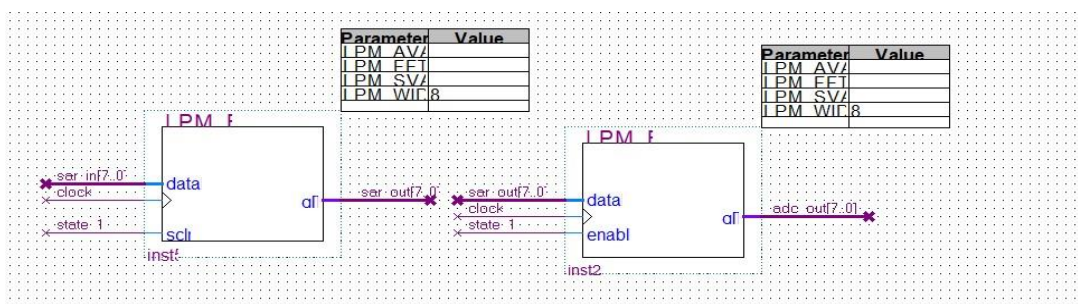
Pour remplacer le bit par **Vcomp** et le mémoriser on utilise la méthode du **data-gating**. On prend le multiplexeur noté **21MUX** et le registre noté **Imp\_FF** tout en suivant le schéma ci-dessous pour chaque bit. On met **Imp\_FF** à la place de la **Flip-Flop D** :



## Circuit

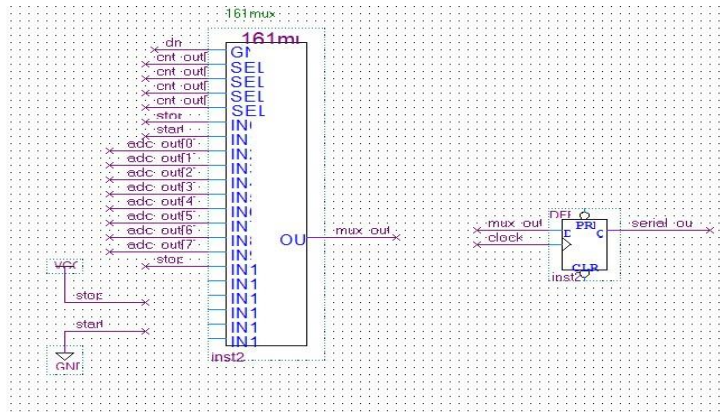


Les bits sont mémorisés dans un premier registre. Mais au 10 eme coups d'horloge, les bits sont effacés pour commencer une autre opération de conversion. Pour conserver ces bits, il va falloir les envoyer vers un autre registre. Les deux registres doivent être configurés sur **LPM\_WIDTH= 8**, muni des ports **data [LPM\_WIDTH-1...0]**, **q [LPM\_WIDTH-1...0]**. Puis le premier aura un autre port **Sclear** réglé sur **state1** tandis que le second aura un **Enable** réglé sur **state1**.

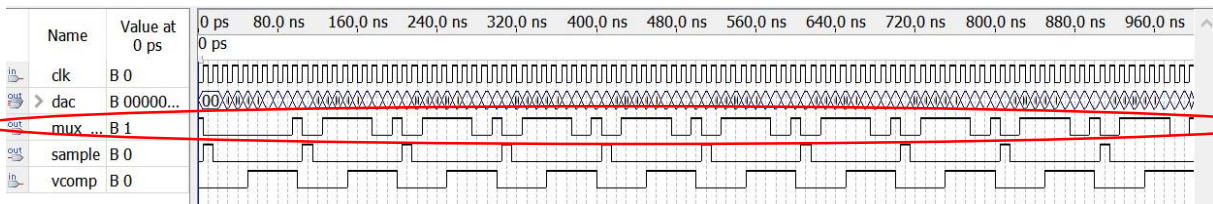


### 3. Transfert des bits par le protocole RS232

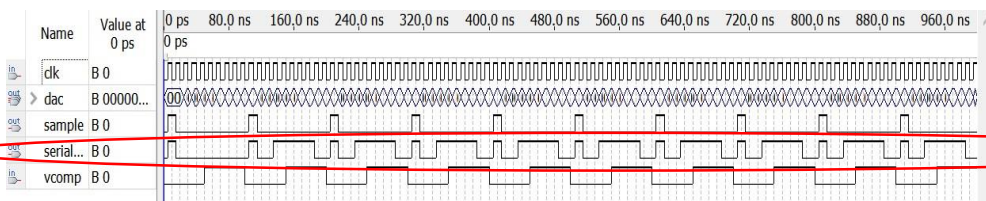
Le protocole est un moyen de transfert d'information d'un composant à un autre. Pour le protocole **RS232** on utilise le multiplexeur **161MUX** réglé sur **10 coups d'horloge**, où on compte 8 coups d'horloge pour le transfert de chacun des 8 bits constituant l'information, 1 pour le bit de **Start** pour débiter le transfert et le denier pour le bit de **stop** qui signale la fin du transfert. Le multiplexeur 161MUX offrent 16 bits. Il est réglé sur l'horloge par les ports **self0**, **self1**, **self2** et **self3**. Et nos bits de Start et de stop sont respectivement en IN1 et IN10, avec les 8 bits de constituant l'information entre IN1 et IN10.



Signal « mux\_out » en sortie du multiplexeur



Signal « serial\_out » correspondant au signal « mux\_out » après le passage dans la bascule DFF



La bascule DFF permet d'éliminer le bruit du signal mux\_out.

### Conclusion :

La partie logique du convertisseur Analogique-Numérique est celle qui joue le rôle le plus important du circuit. En dehors du comparateur, elle contrôle les autres composants du circuit lors de conversion et change les bits jusqu'à avoir les bits idéaux constituant l'information, les mémorise et les transmet.