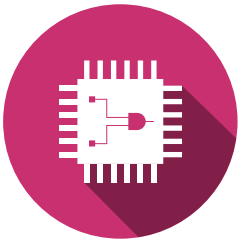


Convertisseur numérique-analogique

Table des matières

1 Introduction

Dans ce laboratoire, nous allons illustrer l'écriture du code VHDL de blocs combinatoires et de blocs séquentiels synchrones par la réalisation d'un générateur de fonctions numérique. Tous ces signaux sont codés en tant que nombres de 16 bits. Le générateur délivre :

- un signal en dents de scie (**sawtooth**)
- un signal carré (**square**)
- un signal triangulaire (**triangle**)
- un signal sinusoïdal (**sine**)

2 Lancement

Le circuit se trouve dans la librairie **digitalToAnalogConverter**. Le banc de test dans la librairie **digitalToAnalogConverter_test**.

*Rappel : le programme de modélisation doit être lancé à travers le fichier **digitalToAnalogConverter.bat**.*

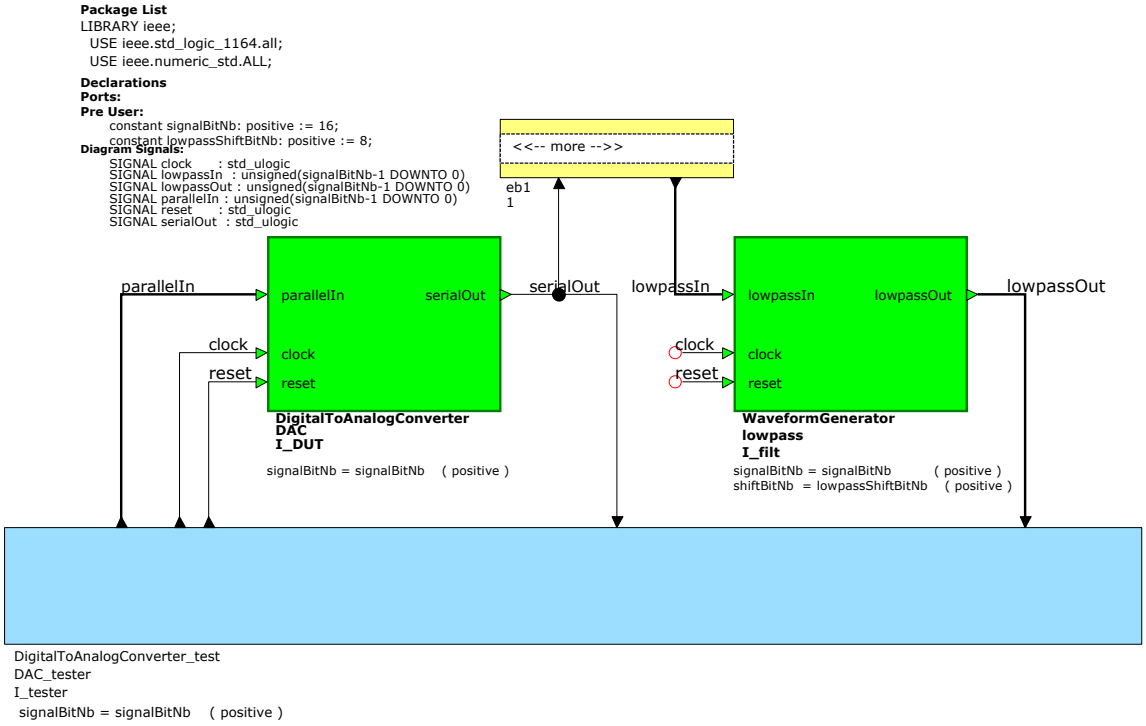


FIGURE 1 – Convertisseur numérique-analogique

3 Objectifs

3.1 Modulateur de premier ordre

Le modulateur SD reçoit un signal numérique et le convertit en signal codé sur un bit, avec la forme d'un train d'impulsions.

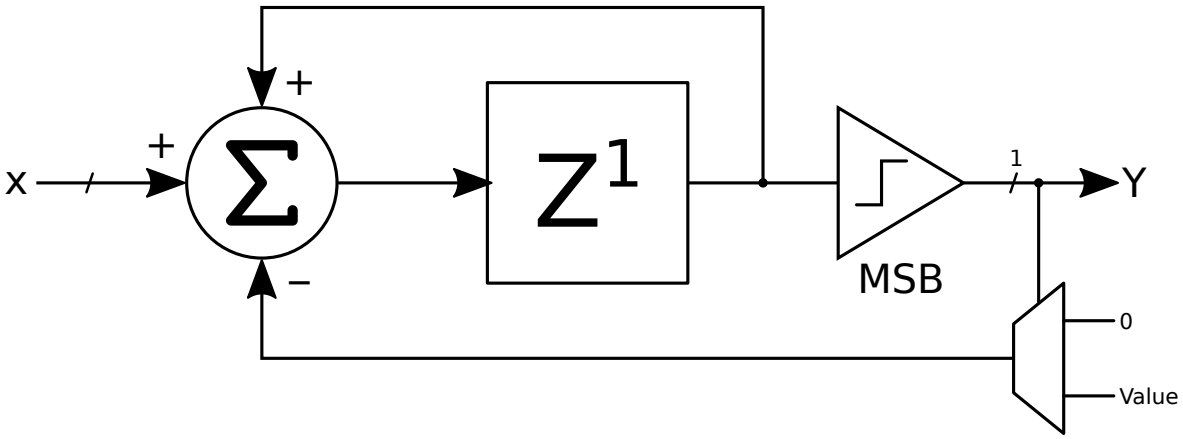
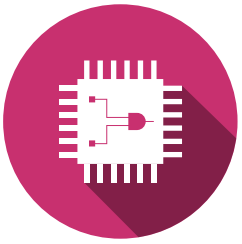


FIGURE 2 – Modulateur du premier ordre

Le modulateur SD de premier ordre peut se comprendre par l'analogie de la mesure du débit d'un



cours d'eau :

— Le cours d'eau est amené dans un bassin qu'il va remplir.

— Dès que le niveau du bassin dépasse une ligne de référence, un opérateur retire un seau d'eau du bassin.

— Le débit moyen du cours d'eau est égal à la contenance du seau fois le nombre de seaux écopés par unité de temps.

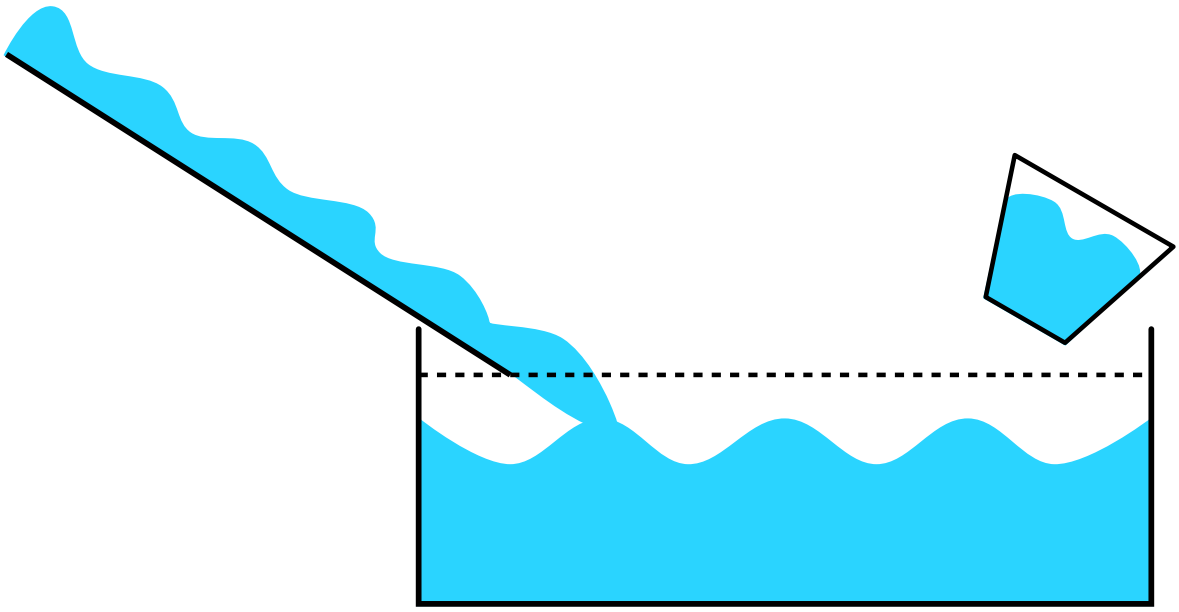


FIGURE 3 – Analogie par l'eau

Le modulateur se réalise comme suit :

— A chaque période d'horloge, la nouvelle valeur du signal est additionnée à un accumulateur.

— Le bit de poids fort du résultat est le signal de sortie du modulateur (impulsion ou pas).

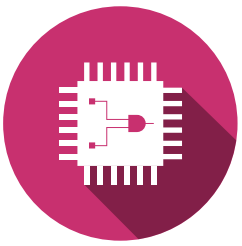
— Si ce bit de poids fort est à '1', une valeur de 2n est soustraite à l'accumulateur, n étant le nombre de bits du signal à convertir.

L'accumulateur interne a besoin de plus de bits que le signal d'entrée. Prévoir 4 bits supplémentaires et la simulation montrera le nombre effectivement nécessaire.

3.1.1 Todo

Ce bloc contient un paramètre générique**bitNb**. Ce paramètre est défini aussi bien pour l'entité que pour l'architecture à écrire.

1. Ecrire l'architecture VHDL du modulateur SD.
2. Compiler et simuler le bloc **DAC**$_{tb}$.*Trouverunmoyend'estimercettevaleurmoyennepourcomparer*
3. Trouver un moyen d'estimer cette valeur moyenne pour comparer le signal parallèle d'entrée et le train d'impulsions en sortie. *La restitution analogique du signal qui reste longtemps à '1' ou à '0' est délicate.*
4. Ajouter un gain de 1/2 et un décalage au signal d'entrée pour éviter ce problème en réduisant la gamme du signal. Refaire la simulation avec cette modification.



5. Examiner le déroulement temporel du contenu de l'accumulateur et déterminer le nombre de bits nécessaire.
6. Réduire le nombre de bits de l'accumulateur et relancer la simulation.

3.2 Modulateur de deuxième ordre

Le modulateur du deuxième ordre comporte deux accumulateurs.

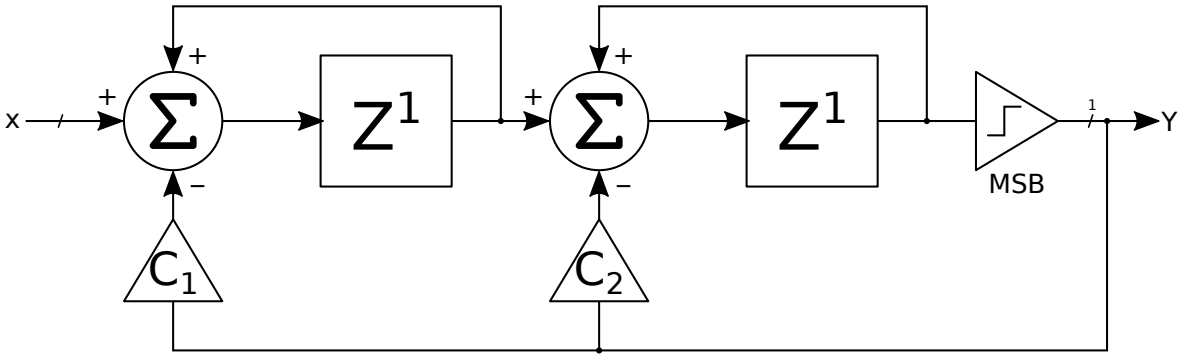


FIGURE 4 – Modulateur de deuxième ordre

Pour ce circuit, les nombres seront considérés comme signés. Lorsque la valeur du second accumulateur est positive :

— La sortie vaut '1'.
— Une valeur de **c1 = 2nBits - 1** est retranchée du premier accumulateur.
— Une valeur de **c2 = 2nBits + 3** est retranchée du second accumulateur.

Lorsque la valeur du second accumulateur est négative :

— La sortie vaut '0'.
— Une valeur de **c1 = 2nBits - 1** est ajoutée au premier accumulateur.
— Une valeur de **c2 = 2nBits + 3** est ajoutée au second accumulateur.

La valeur **nBits** est le nombre de bits du signal d'entrée du modulateur, **x**.

3.2.1 Todo

Prévoir pour chacun des accumulateurs 8 bits de plus que le signal d'entrée. Inverser le bit de poids fort du signal d'entrée et lui appliquer un gain de 7/8.

— Ecrire une deuxième architecture VHDL pour le modulateur SD.

— Rceompiler et simuler le bloc **DAC**$_{t}b.Vrifierlaformedusignaldesortie.Dterminerlenombreminimaldebi$