ENSA Marrakech **2024-2025**

TP4
Processeur simple
GE41

La figure 1 illustre un processeur simple qui contient un certain nombre de registres à neuf bits, un multiplexeur, une unité d'additionneur/soustracteur et une unité de contrôle (machine à états finis). L'entrée des données de 9 bits au système se fait via l'entrée DIN. Ces données peuvent être chargées via un grand multiplexeur de différents registres. La taille des entrées et la taille de la sortie du multiplexeur est de 9 bits. Le multiplexeur permet également de transférer des données d'un registre à l'autre. La sortie du multiplexeur peut être appelée bus car elle permet justement de transférer des données d'un emplacement à un autre.

Pour effectuer l'addition ou la soustraction de nombres signés, le multiplexeur est utilisé pour placer d'abord un nombre de neuf bits sur le bus et en chargeant ce nombre dans le registre A. Par après, un deuxième nombre de neuf bits est placé sur le bus. L'unité d'addition/soustraction effectue l'opération requise, et le résultat est chargé dans le registre G. Les données dans G peuvent ensuite être transférées vers l'un des autres registres selon les spécifications.

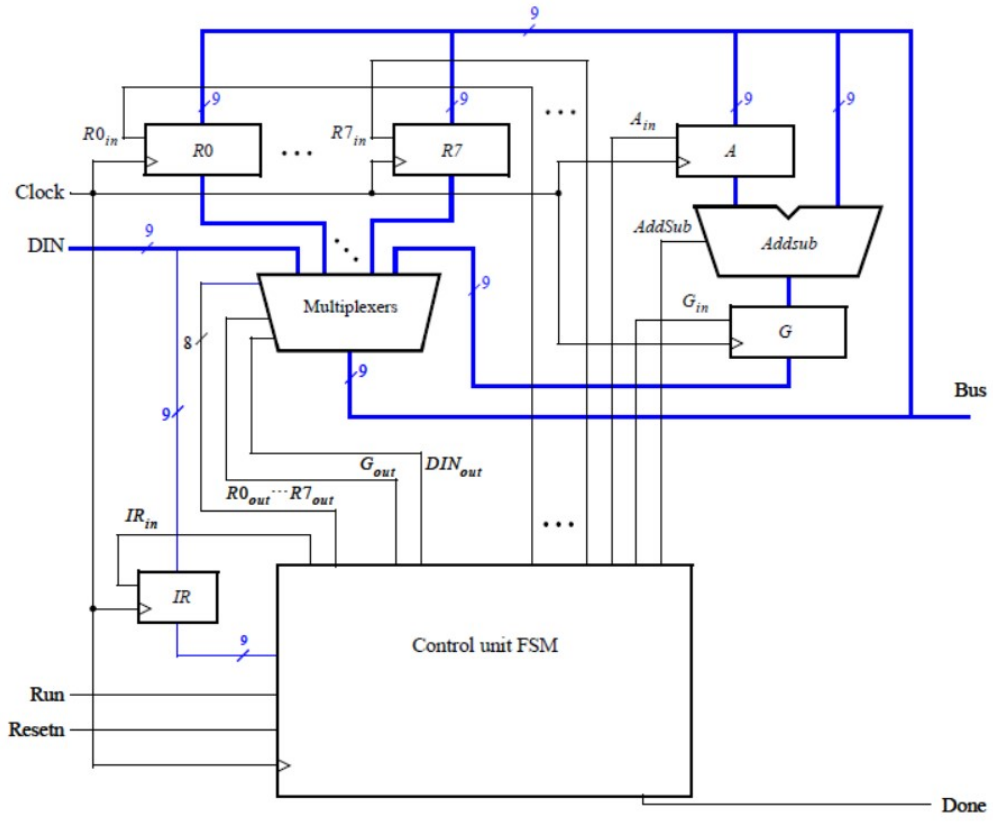

Fig. 1 : Processeur simple

Le système peut effectuer différentes opérations dans chaque cycle d'horloge, étant régi par l'unité de contrôle. Cette unité détermine quand des données particulières sont placées sur le bus et quel registre doit être chargé avec ces données. Par exemple, si l'unité de contrôle active les signaux R0out et Ain, alors le multiplexeur placera le contenu du registre R0 sur le bus et ces données seront chargées au prochain front d'horloge actif dans le registre A.

Notre processeur simple est conçu peut exécuter certaines opérations sous forme d'instructions. Le tableau 1 répertorie les instructions que le processeur doit prendre en charge pour cet exercice. La colonne de gauche affiche le nom de l'instruction et ses opérandes. La signification de la syntaxe Rx ← [Ry] est que le contenu du registre Ry est chargé dans le registre Rx. L'instruction mv permet de copier des données d'un registre à un autre. Pour l'instruction mvi (i commet immédiat), l'expression Rx ←D indique que la constante de neuf bits D est chargée dans le registre Rx.

Operation	Function performed
mv Rx,Ry	$Rx \leftarrow [Ry]$
mvi $Rx,\#D$	$Rx \leftarrow D$
add Rx, Ry	$Rx \leftarrow [Rx] + [Ry]$
sub Rx, Ry	$Rx \leftarrow [Rx] - [Ry]$

Tableau 1 : Instructions exécutées dans le processeur.

Chaque instruction peut être codée en utilisant le format à neuf bits IIIXXXYYY, où III spécifie l'instruction, XXX donne le registre Rx et YYY donne le registre Ry. Bien que seuls deux bits soient nécessaires pour coder nos quatre instructions, nous utilisons trois bits pour permettre d'ajouter autres instructions si besoin. On va considérer que III = 000 représente l'instruction mv, 001 pour mvi, 010 pour add et 011 pour sub. Les instructions sont chargées à partir de l'entrée externe DIN et stockées dans le registre IR (voir figure 1). Pour l'instruction mvi, le champ YYY est ignoré et la donnée immédiate #D doit être fournie sur l'entrée DIN, dans le cycle d'horloge après le stockage du mot de l'instruction mvi dans IR.

	T_1	T_2	T_3
(mv): I_0	$RY_{out}, RX_{in},$ *Done*		
(mvi): I_1	$DIN_{out}, RX_{in},$ *Done*		
(add): I_2	RX_{out}, A_{in}	RY_{out}, G_{in}	$G_{out}, RX_{in},$ *Done*
(sub): I_3	RX_{out}, A_{in}	$RY_{out}, G_{in},$ *AddSub*	$G_{out}, RX_{in},$ *Done*

Tableau 2 : Signaux de contrôle activés pour chaque instruction pendant les temps T1, T2 et T3 .

Partie 1

Concevoir et implémenter le processeur spécifié à l'aide du code VHDL comme suit :

1. Créer un nouveau projet Quartus pour cet exercice.

2. Compléter le code VHDL du fichier processor.vhd fourni pour réaliser le circuit du processeur. Compiler le circuit.

3. Utiliser la simulation fonctionnelle pour vérifier que le code est correct.

Partie 2

1. créer un autre projet Quartus qui sera utilisé pour l'implémentation du circuit sur le FPGA. Ce projet doit consister en une entité de niveau supérieur (top.vhd) contenant les ports des entrées et de sorties appropriés de la carte DE1 SOC. L'entité « processor » constitue un composant (component) du « top ». Utiliser les switches SW_{8-0} pour l'entrée DIN du processeur et utiliser le switch SW_9 pour l'entrée Run. Utiliser également le bouton-poussoir KEY_0 pour aResetn et KEY_1 pour Clock. Connecter les fils du bus du processeur aux $LEDR_{8-0}$ et connecter le signal Done à la $LEDR_9$. Compiler le circuit et le télécharger dans FPGA sur DE1 SOC.

2. Tester la fonctionnalité du circuit en observant les LEDs. Étant donné que l'entrée d'horloge du processeur est contrôlée par KEY_1, il est possible de parcourir l'exécution des instructions et d'observer le comportement du circuit.