Travail Pratique #4 et #5

INF3500 : Conception et réalisation de systèmes numériques

Rapport remis par :

Anastasiya Basanets (1933929)

Mariam Sarwat (1928777)

Groupe : B1

TP4

Critère	Points
Partie A : conception du module	8
Partie B : banc d'essai et simulation	2
Partie C : implémentation	6
Partie D : analyse et Discussion	2
Rapport : présentation et qualité du français	2
Total	20

École Polytechnique de Montréal

Date de remise (16-04-2019)

TP4

Partie A : Implémentation de l'algorithme SHA-256

Tout d'abord, nous avons dû écrire notre pseudocode en VHDL. Pour ce faire, nous nous sommes inspiré des exemples vus en cours et nous avons réalisé une machine à états. La machine à état contient les états suivants :

État INIT : Dans cet état, on initialise les valeurs de départ dans H et W.

État 1 : Dans cet état, on vérifie si le enable est a '1'. Si oui, on initialise les variables (a, b, c etc.) dans nos signaux.

État 2 : On effectue la boucle interne if (if(counter < 16)) en initialisant W.

État 3 : On effectue les calculs de T1 et T2.

État 4 : On effectue la boucle principale de h = g jusqu'à a = T1 + T2 et on vérifie la condition while (0 à 63).

État 5 : On met à jour le H.

État 6 : On met à jour les H_prev.

État 7 : On concatène H dans la sortie.

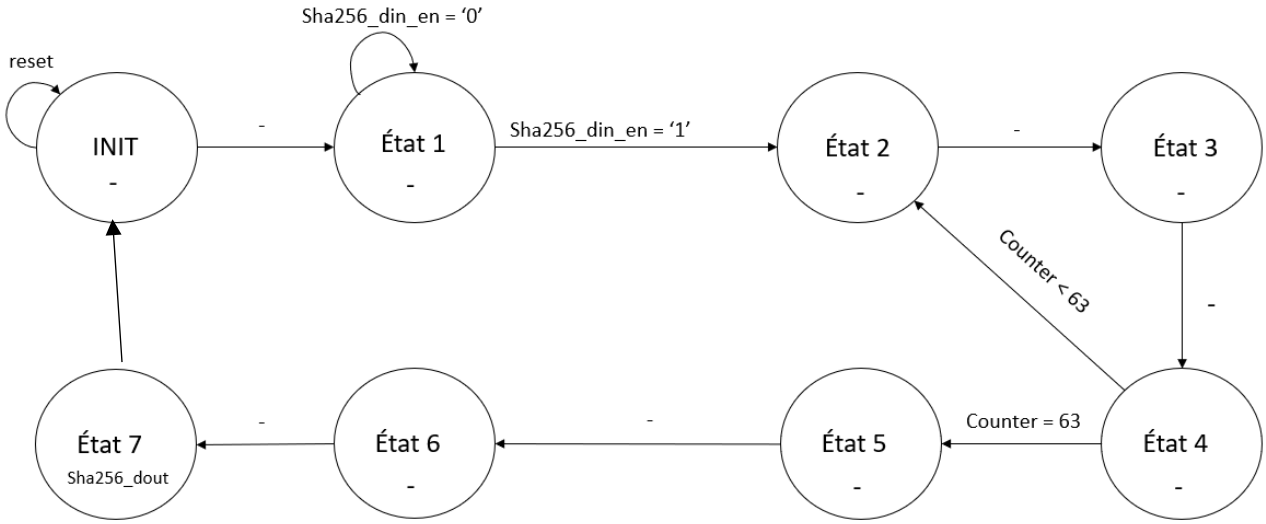


Figure 1 : Machine d'états

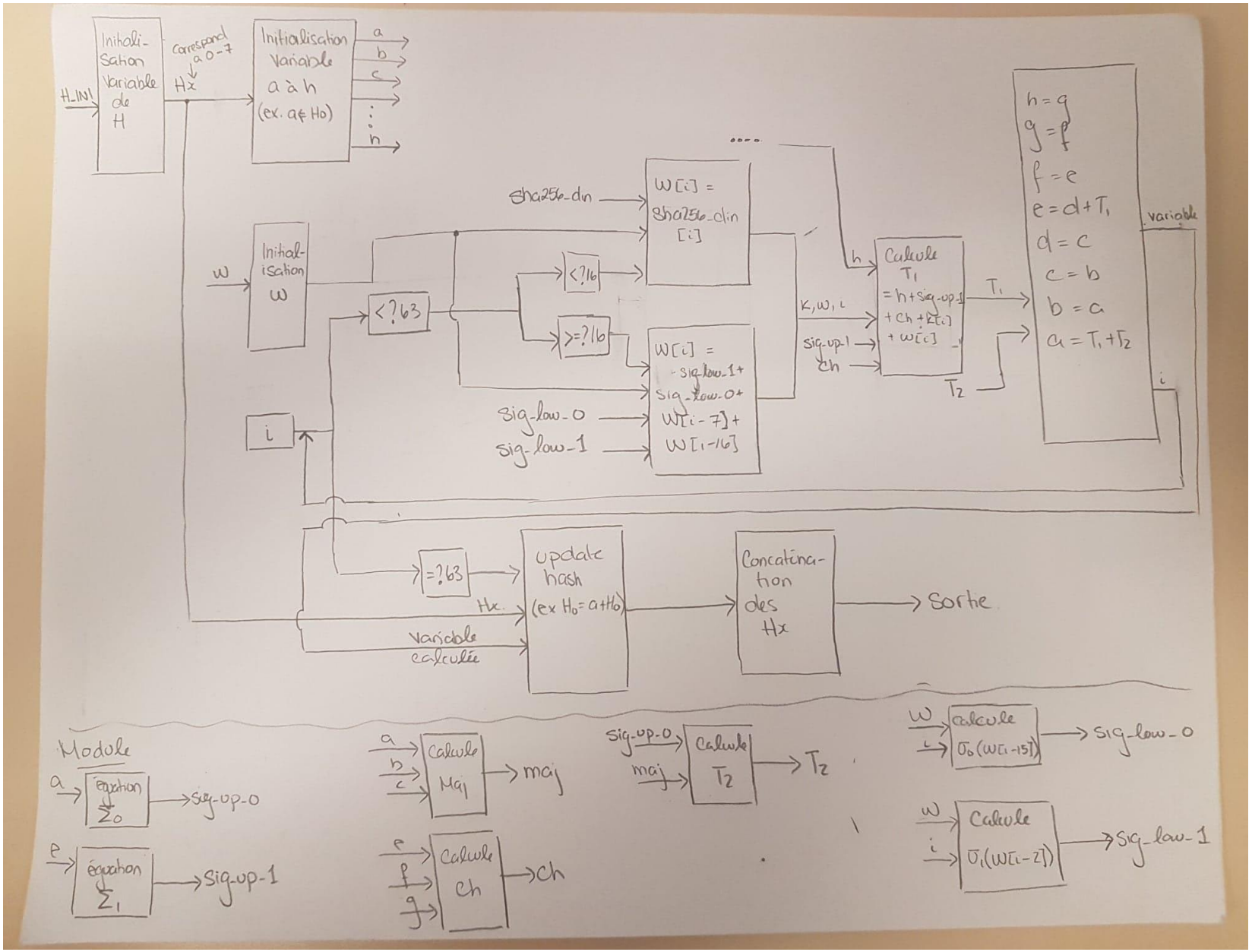
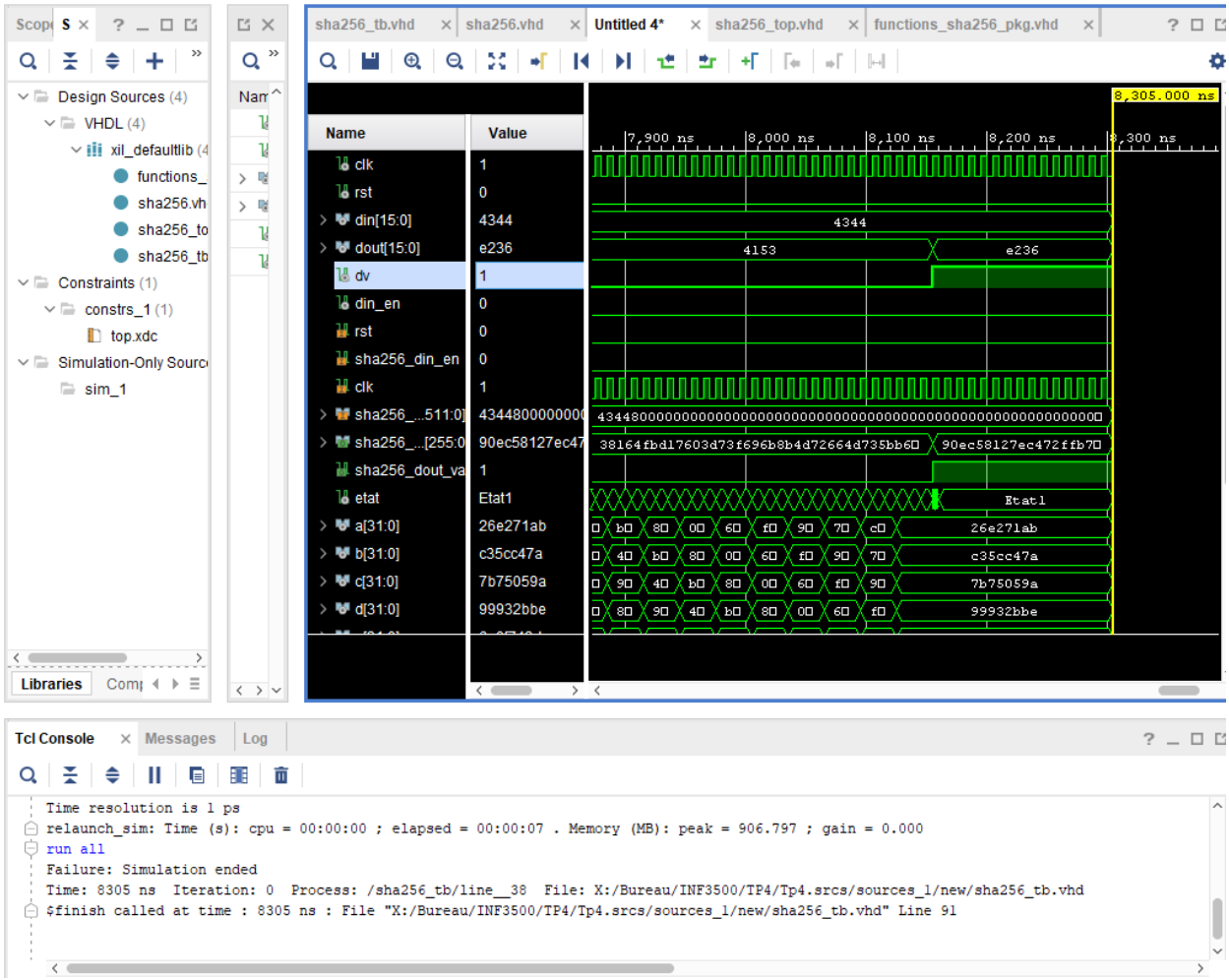


Figure 2 : Diagramme de blocs

Partie B : Simulation

Comme le top étais déjà fourni, il suffisait seulement de partir la simulation.

Schéma 1 : Simulation de notre machine à états



On peut observer que les valeurs de sorties, correspondent bien aux valeurs d'entrées de notre banc d'essai. En fait, sur la photo présente, on peut observer que la simulation est terminée et n'affiche aucune erreur. Cela signifie que les valeurs trouvées en sorties correspondent entre le banc d'essai et notre machine d'état.

Partie C : Implémentation

Afin de faire l'implémentation des fonctions de compressions nous avons dû utiliser le fichier top fourni qui permet à relier les pins sur la carte FPGA avec notre code. En effet, l'implémentation du code a été exécuter convenablement sans erreur. De plus, nous avons réussi à générer le bitstream.

Discussion

Schéma 5 : Tableau des résultats

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF
✓ synth_1	constrs_1	synth_design Complete!								2751	2950
✓ impl_1	constrs_1	write_bitstream Complete!	-0.341	-51.872	0.140	0.000	0.000	0.150	0	2775	2950

Figure 3 : Nombre de LUT et Bascule utiliser

LUT	Bascules	Fréquence	Latence	Débit
2775	2950	100 MHz	8305 ns	$96{,}3 \times 10^{12}$

On observe que notre code a recourt à 2775 LUT et 2951 bascules. De plus, la fréquence et la période a été fixé dans le fichier de contraintes à 100 MHz.

De plus, nous avons pu calculer le débit en faisant le calcule suivant : (le nombre d'états x la fréquence) / la latence. En effet, on obtient un débit de $96{,}3 \times 10^{12}$ opération par seconde.

Finalement, nous avons atteint les objectifs de ce laboratoire, soit d'implémenter l'algorithme SHA-256. De plus, nous avons réussi à simuler et implémenter correctement notre circuit.