TP Inférence IA sur système embarqué (Xilinx Zynq)

INP- ENSEIRB-MATMECA – Option SE –

Objectif

Dans la première session du TP, vous avez optimisé un réseau de neurone, avez créé sa description en HLS à travers HLS4ML et avez réalisé son implementation en code RTL (VHDL) sous forme d'IP. Il s'agit ici de tester ce réseau sur la carte NEXYS. Pour cela, il faut comprendre l'interface créée pour communiquer avec le réseau, utiliser le RTL en IP exploitable dans VIVADO. Il faudra ensuite créer le code nécessaire à la mise en œuvre de l'IP et son test fonctionnel.

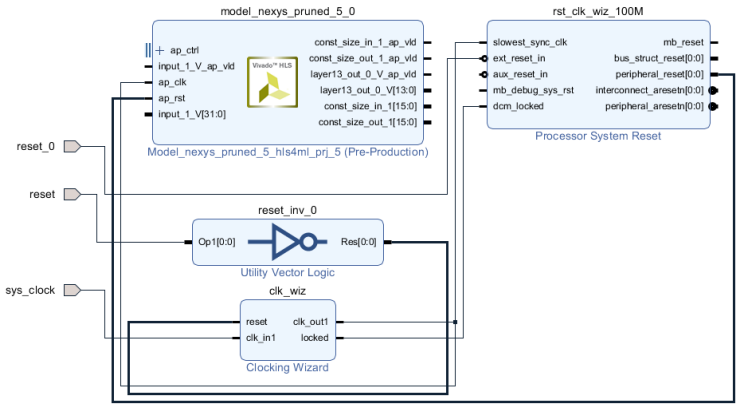
Simulation de l'IP CH3

L'objectif est de simuler l'IP model_nexys_pruned pour pouvoir l'exploiter par la suite.

1- VIVADO HLS
Suite à la session 1 du TP, vous avez soit créer une IP. Vous pouvez également récupérer l'IP depuis le répertoire iCloud : . Placer le dans la librairie des IP de VIVADO.

2- VIVADO (Block Design)
Maintenant que vous avez l'IP correspondant à votre reseau, créer un projet VIVADO nommé Simulation_MonIA. Choisir la carte Nexys4 DDR (xc7a100tcsg324-1) lors de la création du projet.
On minimise les fonctions externes par des signaux de reset et d'horloge pour simuler l'IP pour rendre le design valide comme le schéma ci-dessous.

Créer un banc de test de simulation avec l'IP model_nexys_pruned. Créer une horloge et des stimuli pour l'ensemble des endroits. Expliquer le fonctionnement de l'IP, la latence de calcul et la validation du calcul.



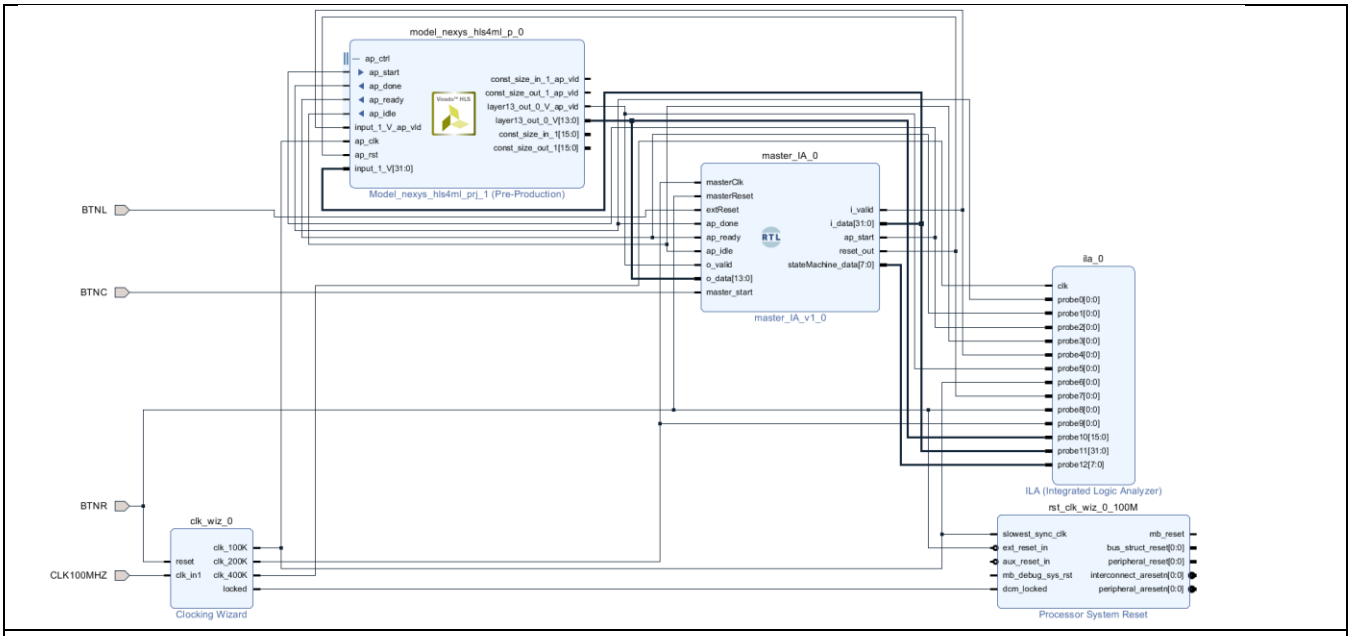
Architecture du design de test de l'IP

Deploiement de l'IP model_nexys_pruned

Créer un nouveau projet dans lequel nous allons créer un séquenceur pour piloter l'IP model_nexys_pruned. Nous allons utiliser un bouton « BTNR », « BTNC » et « BTNL » pour contrôler chaque calcul effectuer par le réseau. Créer un module « master_IA.vhd » qui pilotera l'interface de l'IP. On fera fonctionner l'IP à 100MHz. Pour vous aider, vous avez le fichier de contrainte de la carte et un modèle pour la « master_IA.vhd » dans le répertoire de la session du TP.

Dans le block Diagram, le signal reset issus de master_IA est actif niveau Haut. Dans le cas ou VIVADO le donne actif niveau bas dans son symbol, vous pouvez le modifier via le menu « Block Pin properties » de la broche Reset. Il suffira de changer CONFIG> POLARITY à ACTIVE_HIGH.

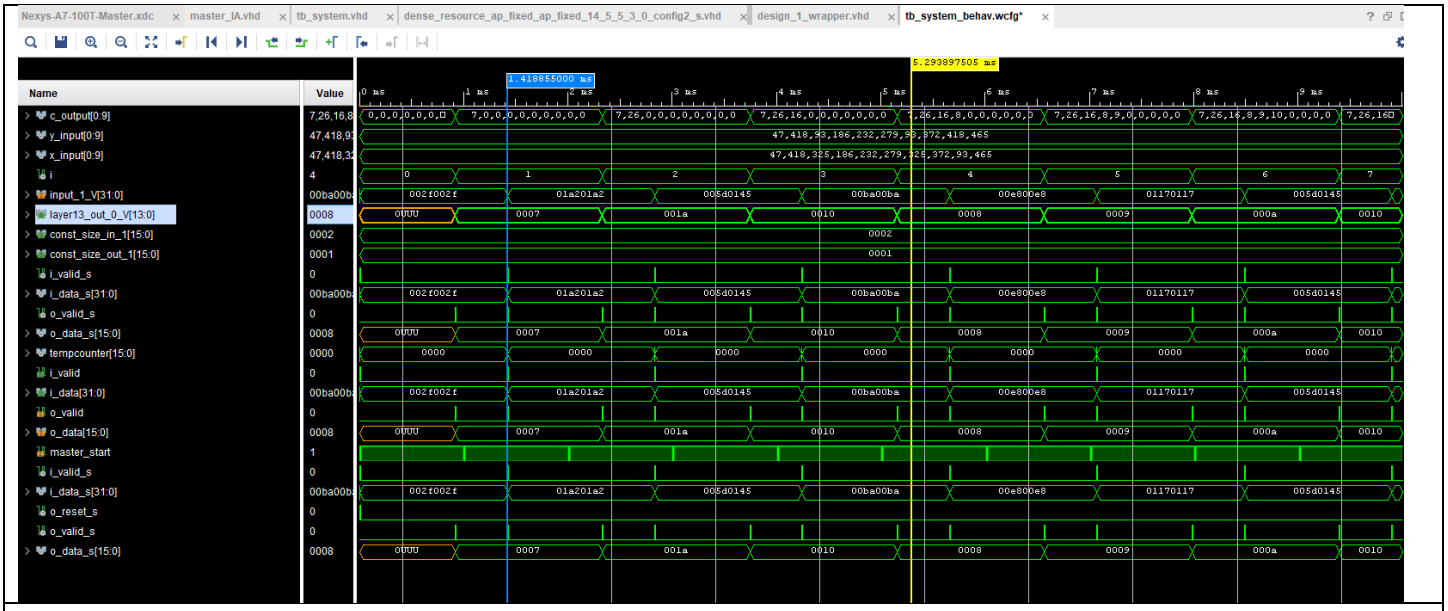
Deployer une IP ILA pour visualiser l'ensemble des signaux qui vous paraît pertinent.

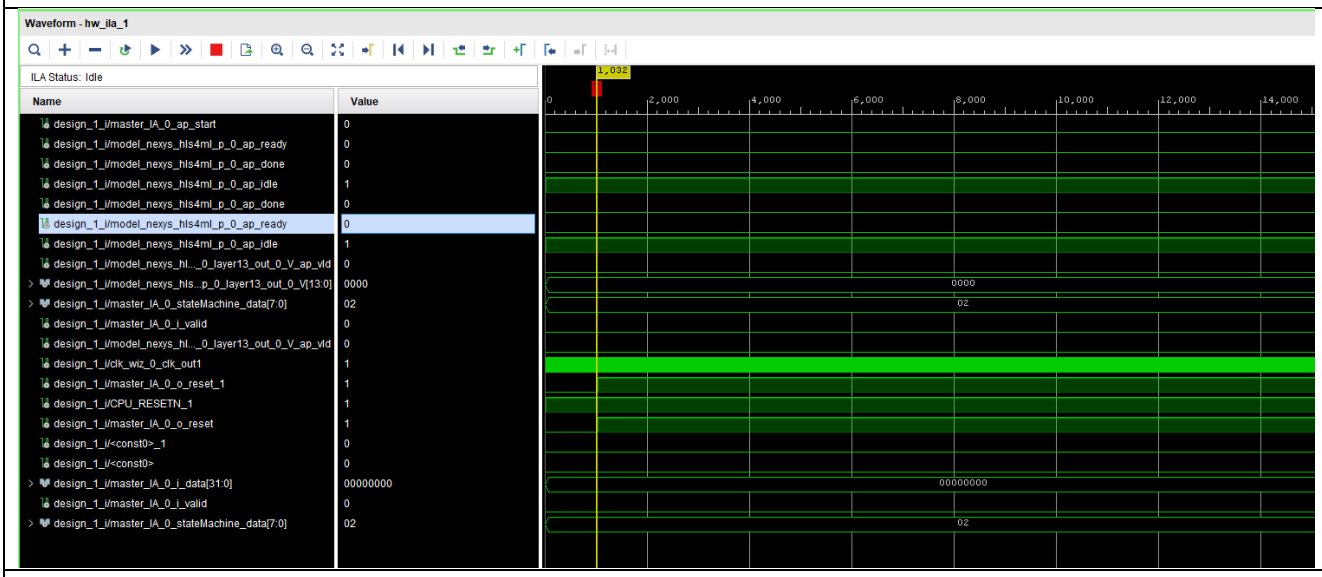


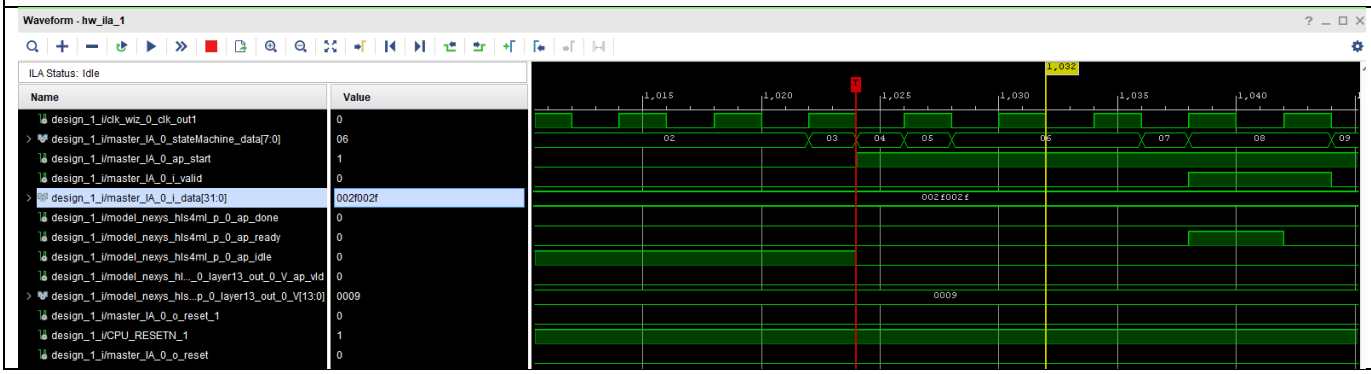
1. Simulation du design

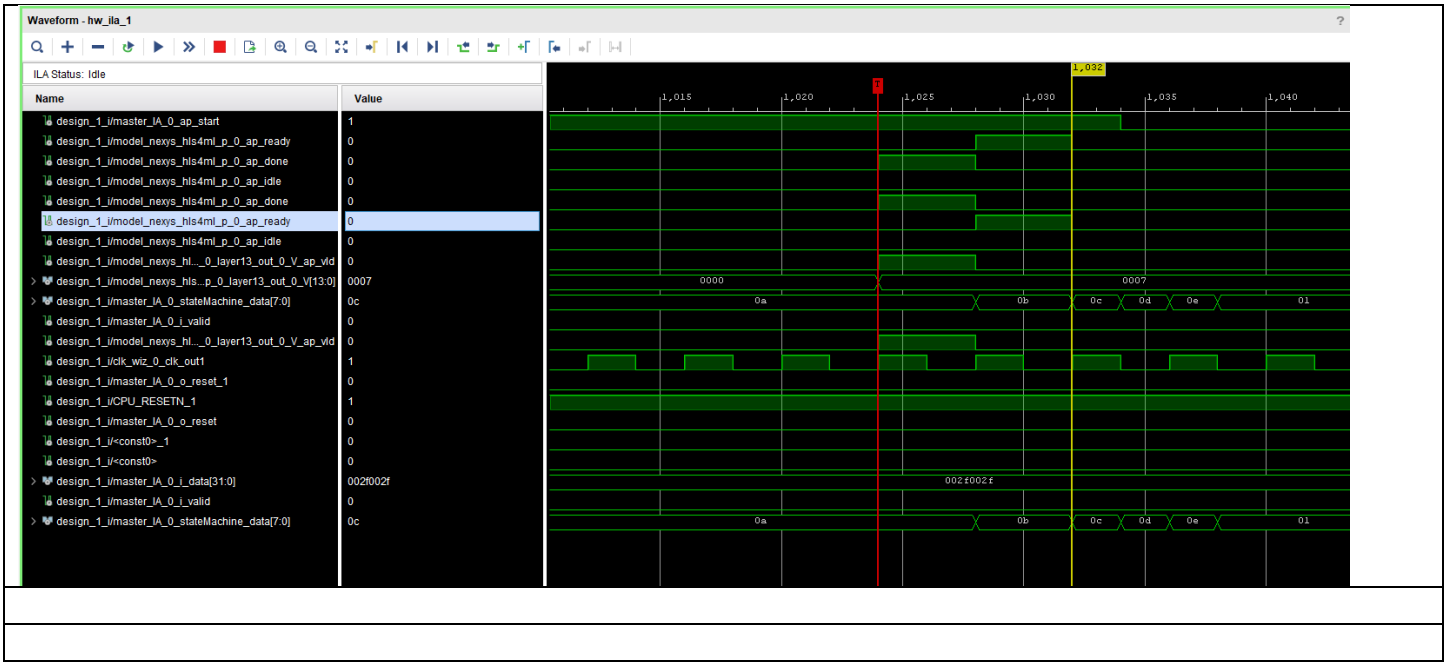
Vous devez enregistrer une liste de valeurs X et Y en nombres à virgules fixes pour valider votre design.

Créer un testbench VHDL pour simuler le système et vérifier le bon fonctionnement du design. Une fois que vous avez validé le fonctionnement, ajouter un Analyseur logique (ILA) permettant de visualiser les signaux à l'intérieur du FPGA. Enregistrer vos différents signaux lors des simulations pour votre rapport.


Nexys-A7-100T-Master.xdc
master_IA.vhd
tb_system.vhd
dense_resource_ap_fixed_ap_fixed_14_5_5_3_0_config2_s.vhd
design_1_wrapper.vhd
tb_system_behav.wcfg*
Name
Value
c_output[0:9]
y_input[0:9]
x_input[0:9]
i
input_1_V[31:0]
layer13_out_0_V[13:0]
const_size_in_1[15:0]
const_size_out_1[15:0]
i_valid_s
i_data_s[31:0]
o_valid_s
o_data_s[15:0]
tempcounter[15:0]
i_valid
i_data[31:0]
o_valid
o_data[15:0]
master_start
i_valid_s
i_data_s[31:0]
o_reset_s
o_valid_s
o_data_s[15:0]
1,418855000 ns
5,293897505 ns


Waveform - hw_ila_1
ILA Status: Idle
Name
Value
design_1_i/master_IA_0_ap_start
design_1_i/model_nexys_hls4ml_p_0_ap_ready
design_1_i/model_nexys_hls4ml_p_0_ap_done
design_1_i/model_nexys_hls4ml_p_0_ap_idle
design_1_i/model_nexys_hls4ml_p_0_ap_done
design_1_i/model_nexys_hls4ml_p_0_ap_ready
design_1_i/model_nexys_hls4ml_p_0_ap_idle
design_1_i/model_nexys_hl..._0_layer13_out_0_V_ap_vld
design_1_i/model_nexys_hls...p_0_layer13_out_0_V[13:0]
design_1_i/master_IA_0_stateMachine_data[7:0]
design_1_i/master_IA_0_i_valid
design_1_i/model_nexys_hl..._0_layer13_out_0_V_ap_vld
design_1_i/clk_wiz_0_clk_out1
design_1_i/master_IA_0_o_reset_1
design_1_i/CPU_RESETN_1
design_1_i/master_IA_0_o_reset
design_1_i/<const0>_1
design_1_i/<const0>
design_1_i/master_IA_0_i_data[31:0]
design_1_i/master_IA_0_i_valid
design_1_i/master_IA_0_stateMachine_data[7:0]


Waveform - hw_ila_1
ILA Status: Idle
Name
Value
design_1_i/clk_wiz_0_clk_out1
design_1_i/master_IA_0_stateMachine_data[7:0]
design_1_i/master_IA_0_ap_start
design_1_i/master_IA_0_i_valid
design_1_i/master_IA_0_i_data[31:0]
design_1_i/model_nexys_hls4ml_p_0_ap_done
design_1_i/model_nexys_hls4ml_p_0_ap_ready
design_1_i/model_nexys_hls4ml_p_0_ap_idle
design_1_i/model_nexys_hl..._0_layer13_out_0_V_ap_vld
design_1_i/model_nexys_hls...p_0_layer13_out_0_V[13:0]
design_1_i/master_IA_0_o_reset_1
design_1_i/CPU_RESETN_1
design_1_i/master_IA_0_o_reset



2. Implémentation du design et test sur la carte NEXYS.

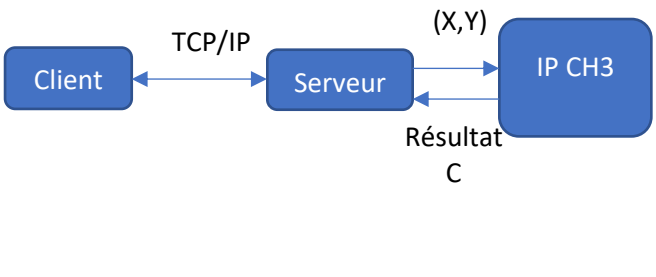
Réaliser la synthèse, l'implémentation et la création du « bitstream » sous VIVADO. Cela peut prendre plusieurs dizaines de minutes. Ne perdez pas de temps, profitez en pour rédiger votre rapport, continuer à optimiser le réseau sous spyder, à comprendre les pragma de HLS.

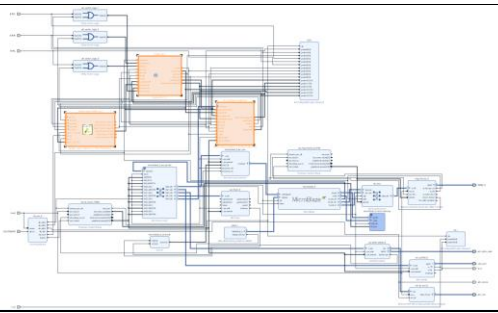
3. Test sur la carte

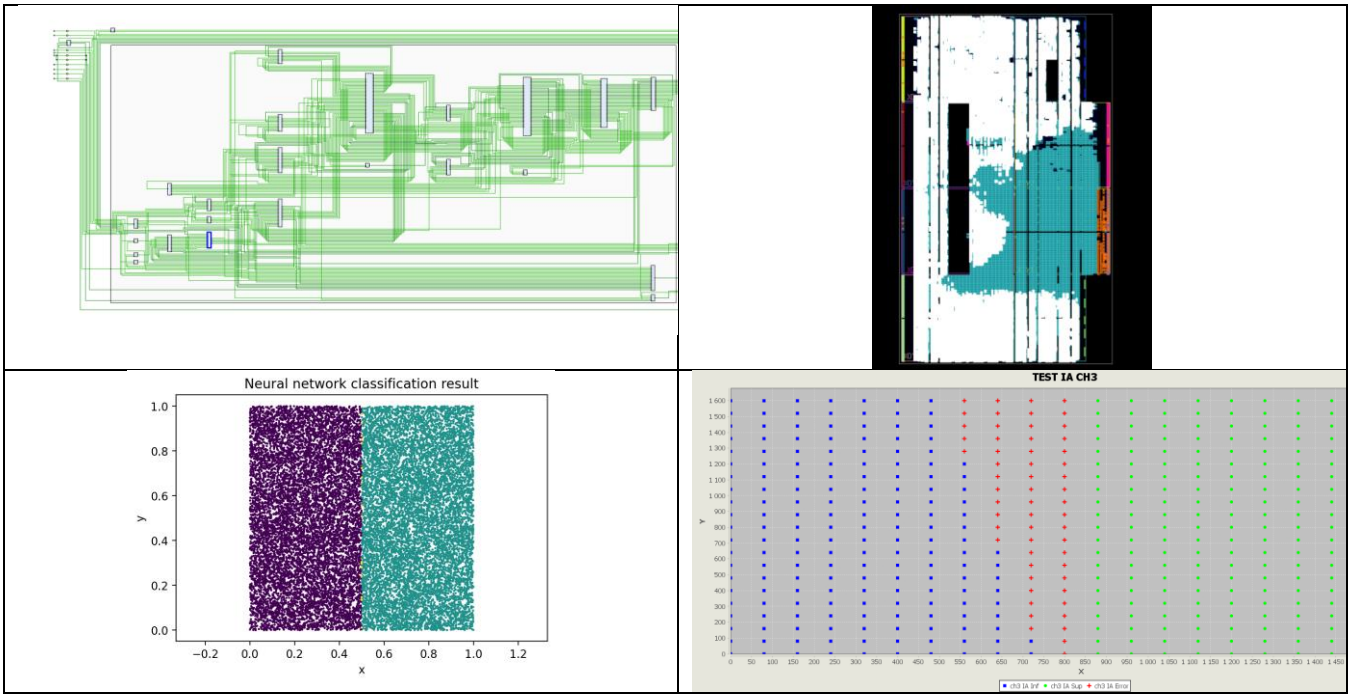
Avec le hardware manager, télécharger le binaire sur la carte. Utiliser l'ILA comme un oscilloscope numérique. Il faut sélectionner les signaux qui serviront de trigger. Valider le bon fonctionnement de votre design en utilisant les 3 boutons poussoir de la carte NEXYS. Sauvegarder les différents chronogrammes validant le bon fonctionnement de votre design.

Système complet : Démonstration

Pour que l'inférence soit complète, il faut ajouter une interface qui permet d'obtenir les données (X,Y) et pour récupérer le résultat. Cela permet d'effectuer un test sur plusieurs dizaines d'entrée et d'analyser les résultats. Pour cela, un microcontrôleur « microblaze » est ajouté pour une liaison TCP/IP en client/serveur. L'écriture et la lecture dans des registres permettront de piloter le système et d'acquérir les résultats.

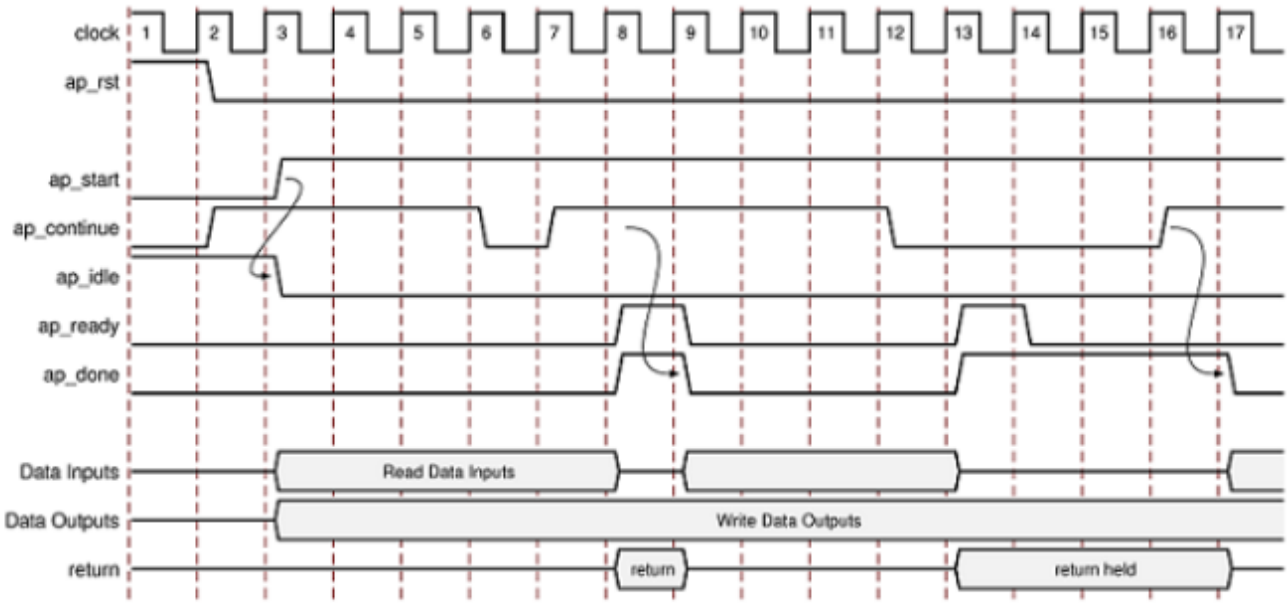






ANNEXES

Figure: Behavior of ap_ctrl_chain Interface



The timing diagram displays the following behavior after reset occurs:

Figure: RTL Port Timing with Default Synthesis

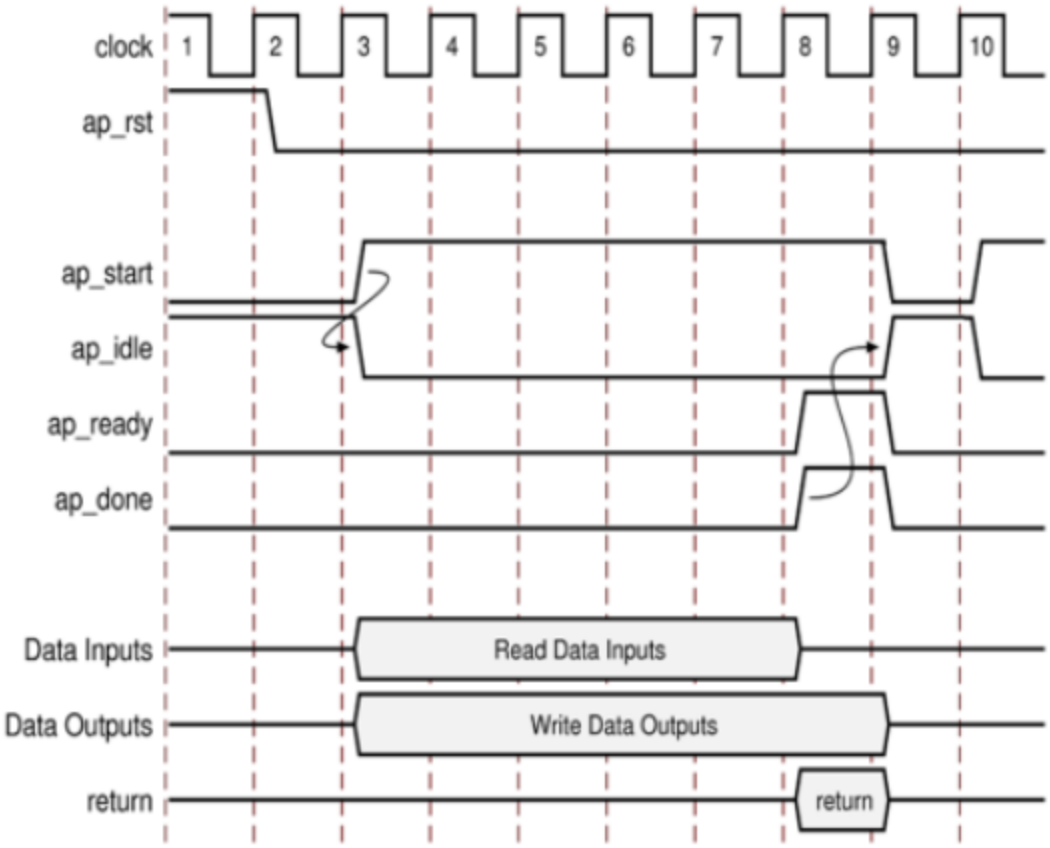
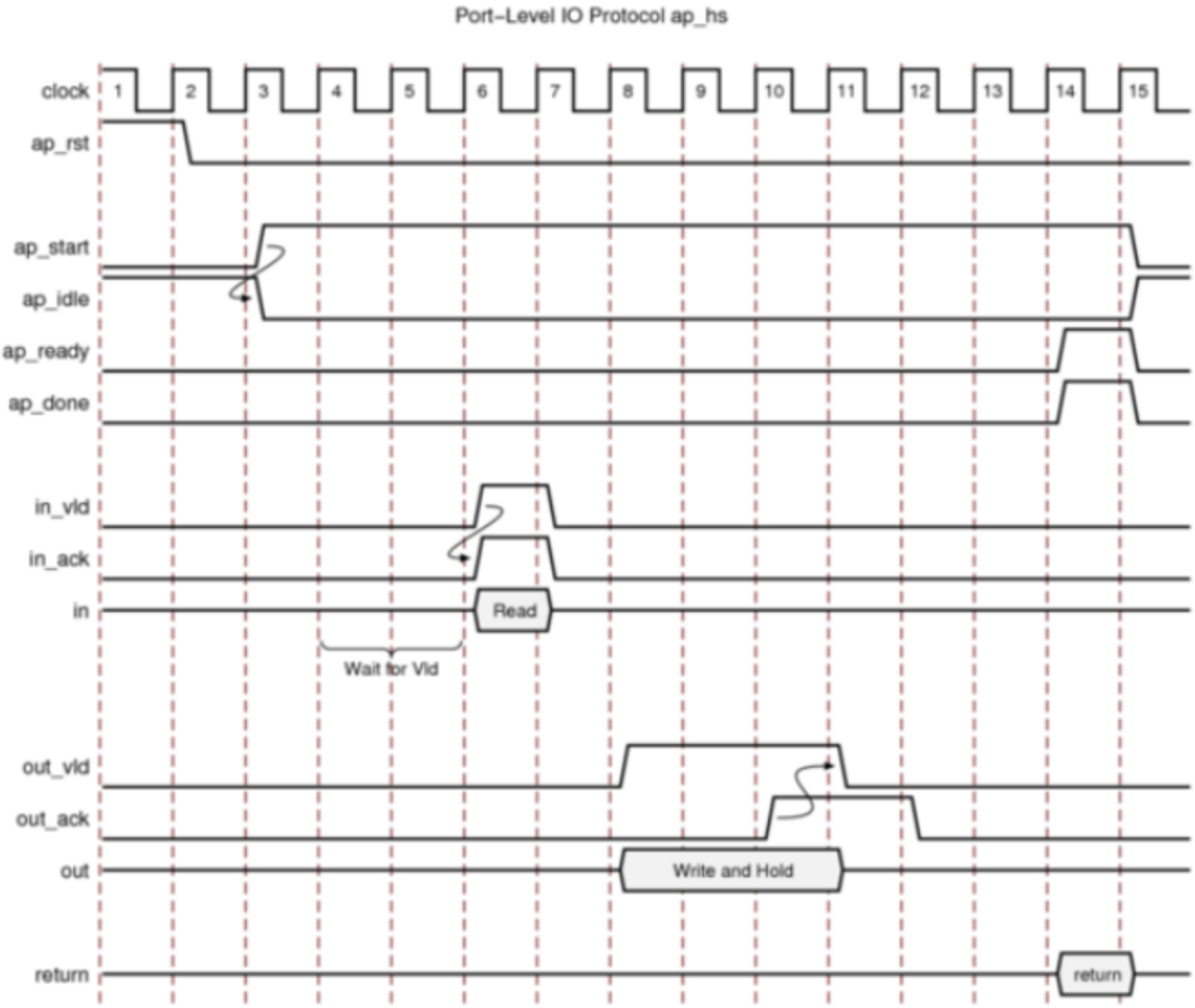


Figure: Behavior of ap_hs Interface



1. The block waits for `ap_start` to go High before it begins operation.
2. Output `ap_idle` goes Low immediately to indicate the design is no longer idle.
3. The `ap_start` signal must remain High until `ap_ready` goes High. Once `ap_ready` goes High:
    - If `ap_start` remains High the design will start the next transaction.
    - If `ap_start` is taken Low, the design will complete the current transaction and halt operation.
4. Data can be read on the input ports.
5. Data can be written to the output ports.

    Note: The input and output ports can also specify a port-level I/O protocol that is independent of the control protocol. For details, see Port-Level Protocols for Vivado IP Flow.

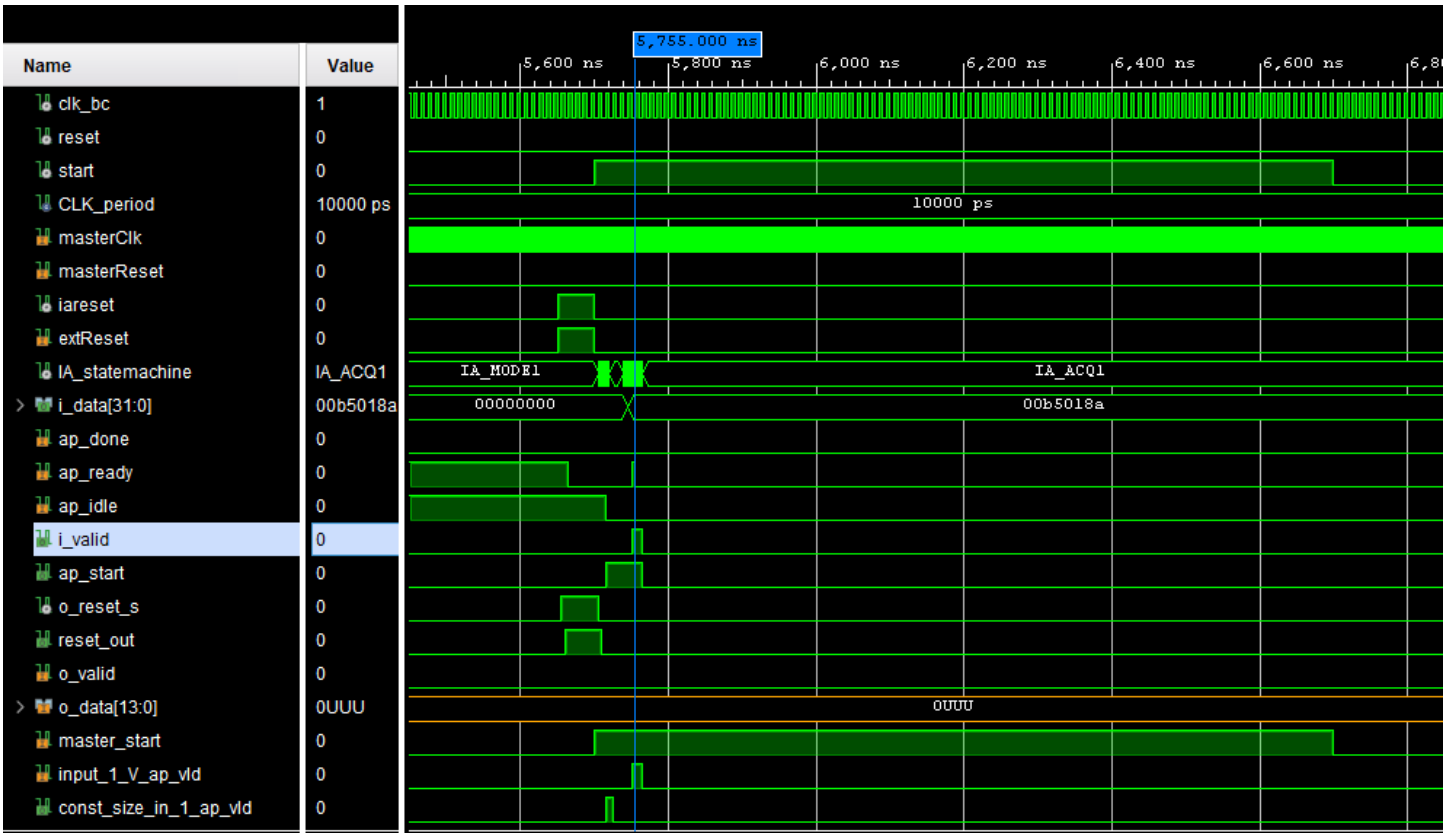
6. Output `ap_done` goes High when the block completes operation.

Note: If there is an ap_return port, the data on this port is valid when ap_done is High. Therefore, the ap_done signal also indicates when the data on output ap_return is valid.

7. The ap_ctrl_chain control protocol provides an active-High ap_continue signal that indicates when the downstream block that consumes the output data is ready for new data inputs. This allows the downstream block to provide back-pressure to prevent the flow of data.
    - If the ap_continue signal is High when ap_done is High, the design continues operating.
    - If the downstream block is not able to consume new data inputs, the ap_continue signal is Low. If the ap_continue signal is Low when ap_done is High, the design stops operating, the ap_done signal remains High waiting for ap_continue to go High.
8. When the design is ready to accept new inputs, the ap_ready signal is pulsed High for one clock cycle. The ap_ready port of a downstream block can directly drive the ap_continue port. Following is additional information about the ap_ready signal:
    - The ap_ready signal is inactive until the design starts operation.
    - In non-pipelined designs, the ap_ready signal is asserted at the same time as ap_done.
    - In pipelined designs, the ap_ready signal might go High at any cycle after ap_start is sampled High. This depends on how the design is pipelined.
    - If ap_start remains high after ap_ready goes high, the next transaction starts immediately.
    - When ap_start goes low right after ap_ready goes high, the design keeps executing until ap_done is high and then stops operation, unless ap_start goes high again in the meantime, which starts a new transaction.
9. The ap_idle signal indicates when the design is idle and not operating. Following is additional information about the ap_idle signal:
    - If the ap_start signal is Low when ap_ready is High, the design stops operation, and the ap_idle signal goes High one cycle after ap_done.
    - If the ap_start signal is High when ap_ready is High, the design continues to operate, and the ap_idle signal remains Low.

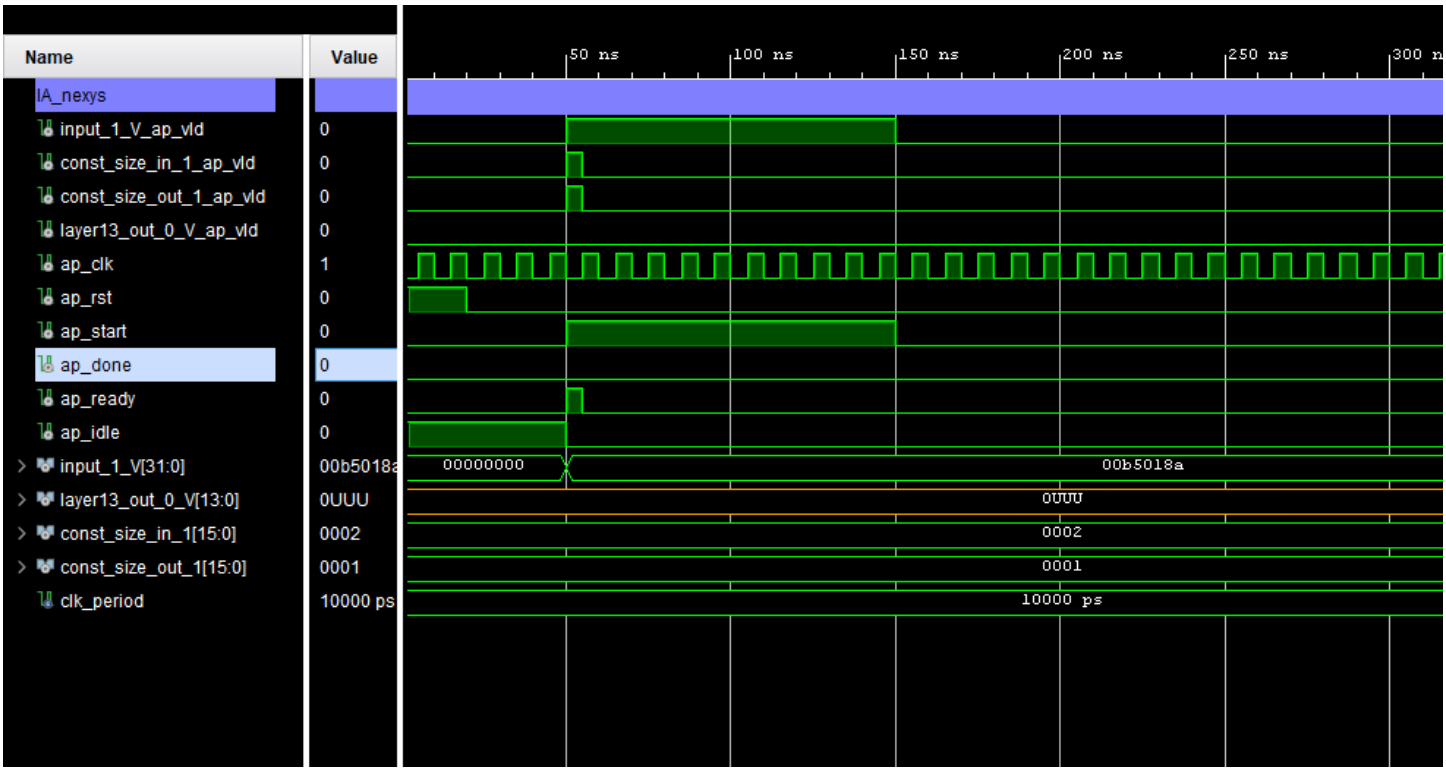
Resultat simulation avec master_AI:

Declenchement du calcul :



Resultat du calcul :

Simulation IP seul :



Resultat calcul IP seul :