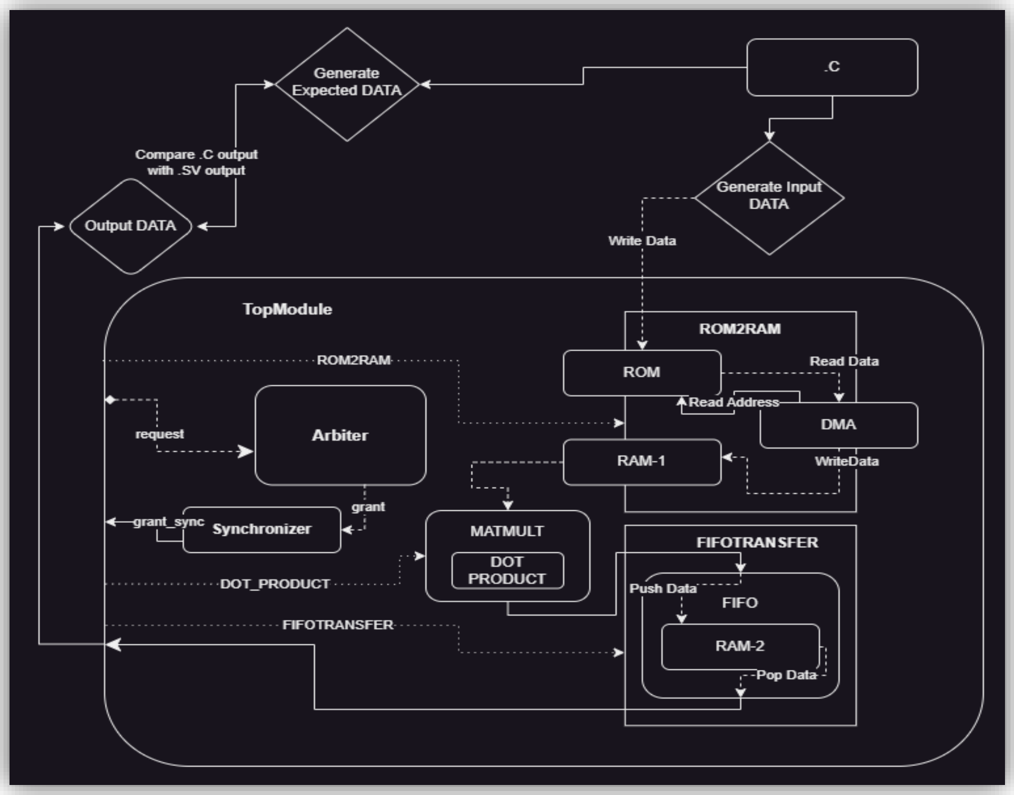
Hussein Abdulreda – Digital Hardware Description and Verification WS 2023/2024



1) <u>An abstract about the topic of the project</u>

   a. The design that I am suggesting here for the project does multiple process stages, where its main core functionality is to do **<u>Matrix Multiplication between NxN and NxN matrices</u>** that are generated using a .bin file (the .c code can be found under test file) which will be fed to the system in a later point of time specifically during the ROM initialization.

   b. The system has a multi-stage pipeline where the first interface contact starts from the TOP-Module which control the stimulus to create requests from the arbiter, which in its turn grant the required signals. These signals that are provided works on a clock of 200MHz as the rest of the system works on a clock of 100MHz, yet later the grant signals will be passed through synchronizer wrappers of each of them so they can be evaluated by the other domain.

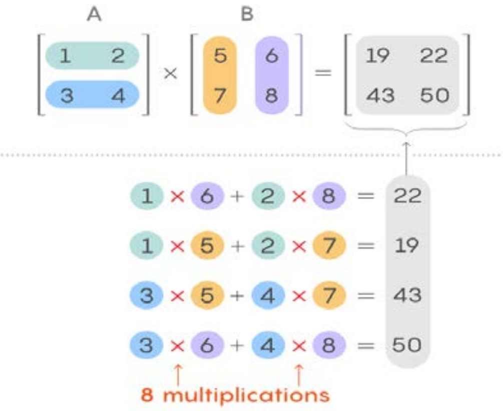
   c. The 4 signal states are **{IDLE, ROM2RAM, DOT_PRODUCT, FIFO_TRANSFER**}; These state machines start a process as follows:

      i. The **IDLE** mode is bluntly clear the machine doesn’t do anything and it waits for the next signal to be requested if it is found.

      ii. **ROM2RAM** job is to initialize the ROM memory from an init.mem file where this file has been already created from the generate environment as shown in the figure. During this state the data that is provided in serial will be transferred through a DMA to RAM-1 which will be returned to the ROM2RAM (top-module) in parallel manner as 2D array of integers(32Bit) (note: no significant bit is found here).

Please check the README.md for more clarifications about the project

iii. **DOT_PRODUCT** is stimulated after **ROM2RAM** validator comes through which states that the **ROM2RAM** process has been successfully done its job, as now the next phase can be processed through. Therefore, the **DOT_PRODUCT** state will receive 2 of NxN matrices which in its turn will multiply these matrices to retrieve back an output matrix where it will be passed on to the next state. However, it is worth to mention that the mat_mult module will have a generate block of dot_product module to generate multiple combinations that are compute in parallel wise.

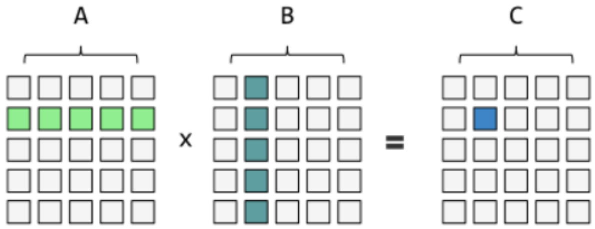


iv. **FIFO_TRANSFER** as well waits a validator signal from **DOT_PRODUCT**, as whenever this signal is stimulated then the data will be fed in FIFOTRANSFER (top-module) parallel wise in a 2D matrix, yet this module will push the data one by one to a FIFO of RAM-2 module which in its turn will be stream out the data from the other side by pop-ing the elements one by one to be fed back to then interface TOP-MODULE, this action will be stimulated as parallel2serial phase in the process.

d. The output data will come out as serial which will be saved in a file called *"output" this* file will be compared with the expected values that are also generated from test-environment phase.

2) The methods and design patterns used

a. Control pattern (**Moore FSM-Arbiter**)
b. Compute pattern (**Matrix multiplication [dot-product]**)
c. Storage pattern (**ROM, Simple dual port RAM, Sync-FIFO**)
d. Movement pattern (**DMA, Pipeline**)
e. Adapter pattern (**CDC, Parallel2Serial**)



3) The test strategies used

a. Testbenches for each module (code coverage)
b. Test environment in .C (under test file) which will compare the matrix multiplication done by the test environment and the output file from the DUT.

4) The programming languages and tools used

a. System Verilog
b. Vivado
c. VSCode (Icarus Verilog) + Formatter
d. Makefiles
e. Bash files

5) Any further observations

a. The project can be extended to have two different matrices NxM with MxN
b. CDC can be applied on FIFO as well as input would be on a clock and the output would on another clock.

Please check the README.md for more clarifications about the project