Hardware assignment 3 report

Our goal was to implement matrix multiplication on basys3 board. The overall design needed to have a MAC , a ROM block and a RAM block .For this we proceeded as follows:

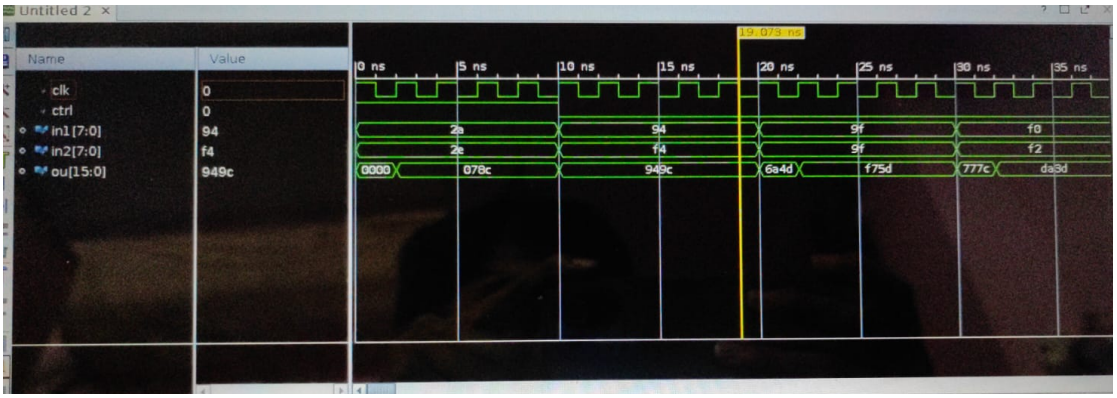
1. MAC (Multiplier Accumulator block) : The top level file mac_unit.vhd utilised the files bitmult.vhd, register.vhd and adder.vhd which respectively perform 8 *8 bit multiplication, stores product and 16 bit addition. MAC finally generates a 16 bit output which is then sent to RAM for accumulation.

2. ROM block- We store our input matrices in two rom blocks - rom_block_1.xci and rom_block_2.xci. The matrices are stored in column wise manner. Our fsm unit provides the required addresses from which the elements need to be fetched from ROM. It then stores the elements at that address in 8-bit registers (one register stores elements from matrix 1 and second register stores elements from matrix 2). They are sent to MAC for computation.

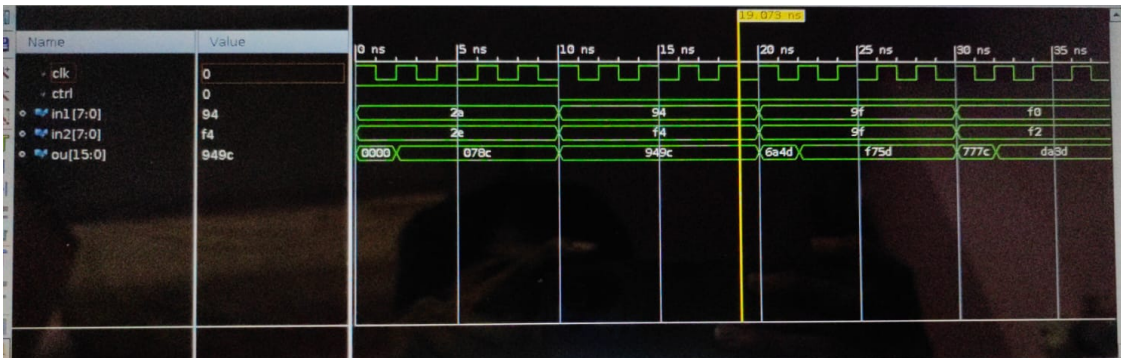
3. RAM block- Once an element of our output matrix has been computed we store it in RAM via using a 16-bit register. The elements are stored in a row wise fashion in RAM.

4. FSM unit - FSM helps drive our design. The three states are rom_state, ram-state and mac_state. When in rom_state , FSM provides the address to fetch the elements stored in our 2 roms and moves on to mac_state. While in mac_state we perform computation. After every 128 multiplications we move on the ram_state to store the final sum of a row and column otherwise we move on to the rom_state until(the element of the output matrix hasn't been fully computed). When in ram_state we store the element of the output matrix and then move on to the rom_state to start computing the next element. After we have stored all 16384 elements of the matrix, we are done.

SIMULATION WAVEFORMS



16-bit adder



Mac

BLOCK DIAGRAM