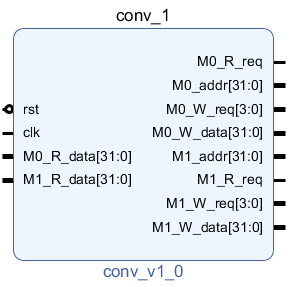
HOMEWORK 5

Student ID: P76087081 Name: Tran Thi Ai

**Design idea:**

(Please describe the function and features of the convolution circuit)



In this convolution IP, I used the finite-state machine (FSM) to execute the convolution function. The *start* signal is given from the last data of the *input.hex*. Therefore, the system first reads the last input until it reads 32'b1 to start and continues to the next state.

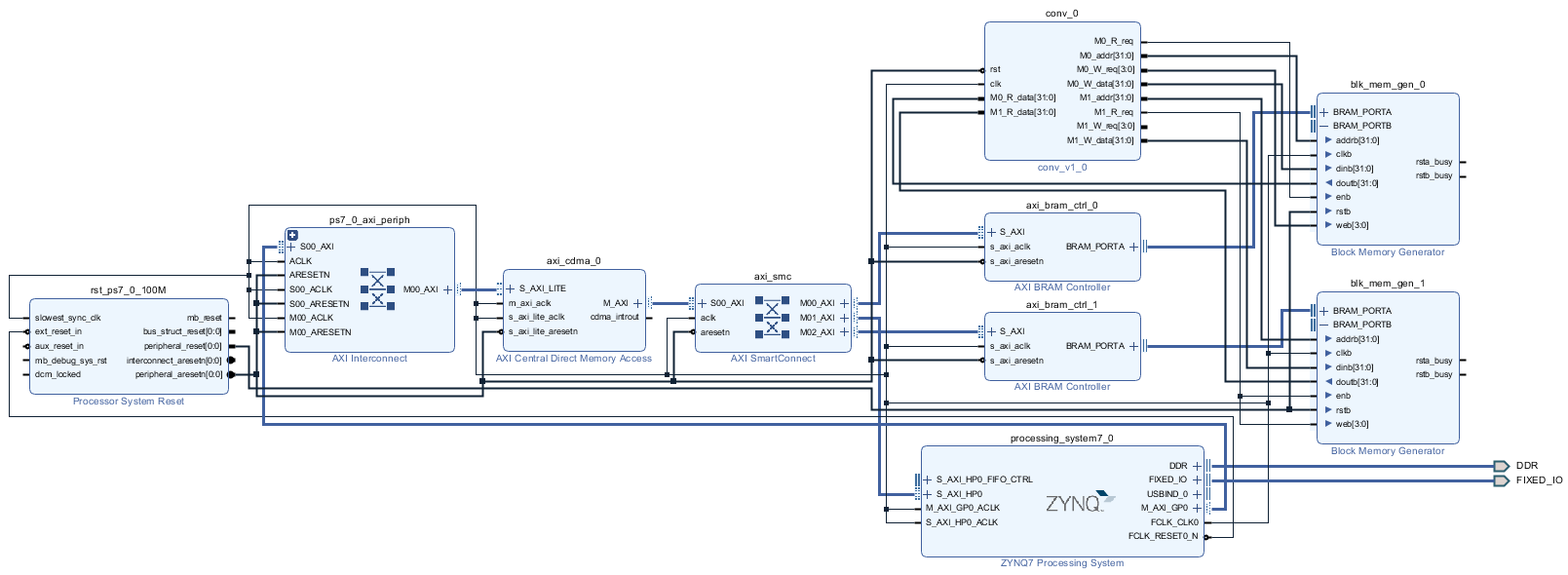
In the beginning, the system reads the input data by control the *M0\_addr* with *M0\_R\_req*. First of all, it reads 9 data of kernel (from weight0 to weight8) and 1 data of bias from the last 10 data of the input file. Then read each 9 pixels continuously before doing the convolution operation.

Convolution operation: product[i]=pixel[i]\*weight[i] (with i=0~8). Then, summary the 9 products with the bias to make the result.

After that, control *M1\_addr* and *M1\_w\_req* to write result from the beginning to the end.

**Block design Screenshot:**

(Please attach a screenshot and describe the block design function.)

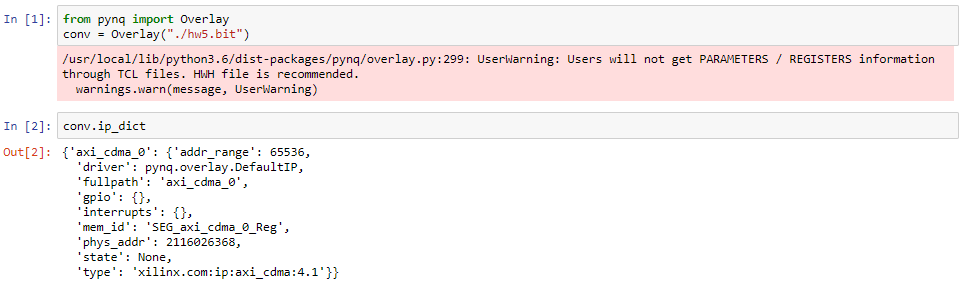


* **ZYNQ7 Processing System** is an ARM processor that can be designed with its internal software.
* **Processor System Reset** customize the design to fit with its application by setting certain parameters to enable/ disable the features.
* **AXI Interconnect** connect AXI master and slave devices from ZYNQ to CDMA.
* **Block Memory Generator** can generate block memory. It is similar to the general SRAM. It can adjust various parameters, such as memory size, single port or dual port, etc. It is a bit like memory compiler. In this case, the design consists of 2 BRAM: blk\_mem\_gen\_0 is used to contain the input data while blk\_mem\_gen\_1 is used to save the output.
* **AXI Bram Controller** is an IP provided by Xilinx Vivado. It mainly transfers the block memory read/write signal to AXI format, so that block memory can communicate with AXI bus. In the design, axi\_bram\_ctrl\_0 and axi\_bram\_ctrl\_1 connected to blk\_mem\_gen\_0 and blk\_mem\_gen\_0 respectively.
* **AXI Central Direct Memory Access (CDMA)** can move data between two memory addresses. Like a normal DMA, CDMA has both Master and Slave pins on the system. In this design, set ZYNQ as master and CDMA as slave, transmit control data through AXI BUS, tell CDMA where to send it, how much data to send, etc. Use CDMA as master, ZYNQ and block memory as slave to transfer data. Block memory must be connected to AXI Bram Controller to transfer data to AXI BUS.
* **Conv** is a customized IP. The IP achieves the operation designed for 2D convolution and converts the 28x28 input pixel to 26x26 output pixel. The filter size is 3x3 and the stride is 1.

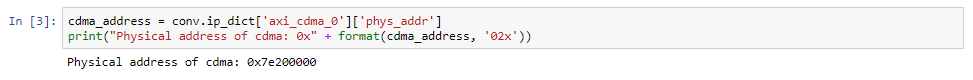
**Jupyter python code:**

(Please describe the function and execution flow of the jupyter python code.)

* Burn the *hw4.bit* file into the board.



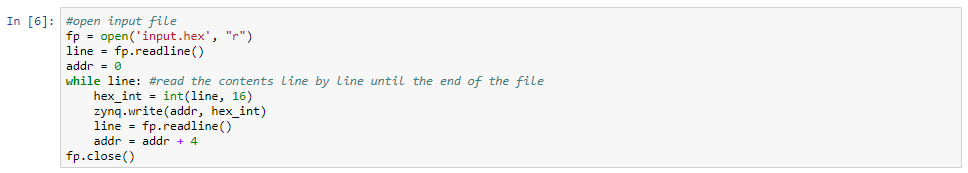
* Set CDMA operation address: cdma\_address = 0x7e200000



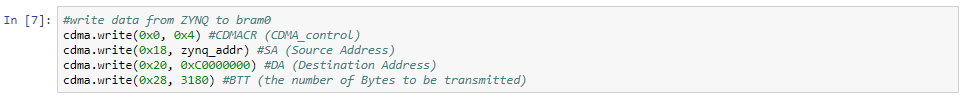
* Order the operating address of ZYNQ: zynq\_address = 0X30000000. Use MMIO operation cmda and zynq.



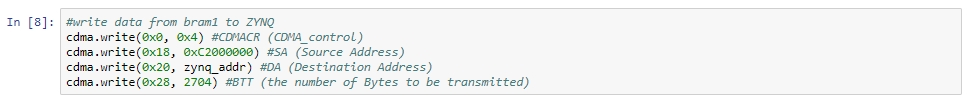
* Open input file and read the contents line by line until the end of the file



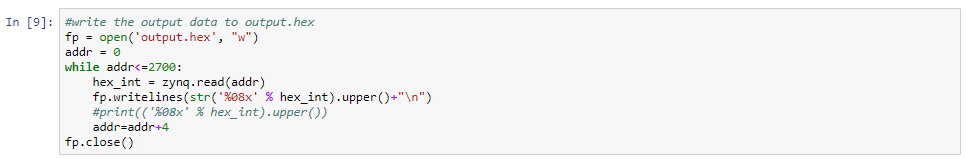
* Write data from ZYNQ to bram0
  + Write CDMACR (CDMA\_control) into 0x04 to start CDMA
  + Write SA (Source address) to the address ZYNQ just written to the data
  + Write DA (Destination address) to the address of axi\_bram (address = 0xC0000000)
  + Write BTT to the number of bytes to pass, not less than 3180, because at least (28\*28+10+1) words of data is passed. CDMA will start transmitting after the CDMA\_BTT is written.



* Write data from bram1 to ZYNQ
  + Write CDMACR (CDMA\_control) into 0x04 to start CDMA
  + Write SA (Source address) to address in *axi\_bram*
  + Write DA (Destination address) to the address of ZYNQ
  + Write BTT to the number of bytes to pass, not less than 2704, because at least (26\*26) words of data is passed. And CDMA will start transmitting after the CDMA\_BTT is written.



* Write the output data to *output.hex*



* Checking the contents of the output and comparing with the golden



**Lesson learn**

(Please write down the experience of completing this assignment, what you learned, and the points of difficulty.)

Through this homework, I learned how to use the *AXI Central Direct Memory Access* to transfer data between the *Block Memory* and the *ZYNQ CPU*. I also learned how to simulation the circuit in Vivado. The architecture of the block design in Vivado is almost the same as the last homework, just replacing the *mul16* IP with the *conv* IP and add one more *BRAM*, so there is not much problem in this part.

The difficulty point I encountered was that it is hard to build the *conv* IP. Fortunately, I was finding the way myself through the last IC design contest to making the convolution IP.