University of Tehran

Electrical and Computer Engineering Department

ECE (8101) 342

Object Oriented Modeling of Electronic Circuits – Spring 1404

Homework 4: Embedded Processor Bracketing and ML Acceleration
Due Date: Khordad 19, 1404

Processor Bracketing
C++ ML Modeling
Using a BFM Accelerator
DSE Basics

In this homework you will use the SystemC environment to model a processor by bracketing a ML inference engine by the necessary ports of a processor, and then moving some of the operations to a given hardware accelerator.
A layer in an MLP effectively performs a matrix multiplication, where each output is the dot product of the input matrix with a corresponding weight matrix. A more detailed numerical explanation is provided in the attached example.

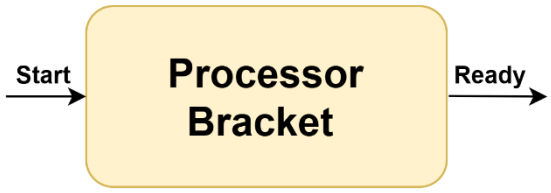
For this exercise, consider an MLP with the topology 3–8–10, meaning:
• 3 inputs in the first layer,
• 8 neurons in the hidden layer,
• and 10 output neurons.

In this homework, we consider the matrix multiplication computation for the first layer. This layer performs a matrix multiplication between:
• the 3 input values (serving as the input vector for this layer),
• and a 3×8 weight matrix, where each column corresponds to the weights associated with one hidden neuron.

Description: A Layer of an MLP.

1. Write a C++ description of the MLP first layer with 24 8-bit weights, 3 8-bit data inputs, 8 neurons, and 8 18-bit outputs. The data inputs and weights are initially available in their corresponding files that must be moved into the corresponding memories in the C++ layer description. There is also an output file that, when the MLP layer is complete, the output will be written into it. The SystemC embedded processor has a start input that begins the process of layer calculation. The SystemC processor bracket has a ready output that is issued when a layer operation is complete.

a. Write the MLP Layer C++ program and run it in the SystemC environment assuming a processor model with just the start and ready signals. This is an untimed model of your algorithm implementation.



b. Run this program in a RISC-V environment and calculate the approximate number of clocks for completing this layer calculation.

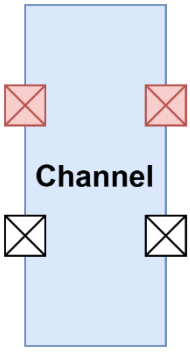
2. You now have a SystemC model for an embedded processor that is performing an MLP layer computation and is back-annotated with the number of clocks it takes to perform the calculation.

Description: A Matrix Multiplier Accelerator

3. Write Bus Functional Model (BFM) of a matrix multiplier multiplication of $[A]_{1\times3} \times [B]_{3\times2} = [O]_{1\times2}$. The multiplier block has local memories for its inputs and its output. In the next sections you will see that these data blocks will be provided by a channel transfer from an embedded system.
4. Provide interfaces for the accelerator such that when the startMMA signal is asserted, it gets the two blocks of data from its input channels and begins the multiplication process. Once the computation is complete, the accelerator asserts the intrMMA signal. This signal will later be used as an interrupt request to notify the processor that the result is ready, and the output block can be retrieved.

Description: Channels

5. Write an interface and a corresponding channel for a processing element to transfer a data segment consisting of nine 8-bit values to its destination.



6. In the above channel, include an interface for a blocking get for a target to receive a segment of data from the channel.
7. To transfer data to the accelerator, the processor issues two consecutive put segments, and assert the startMMA signal. The accelerator performs two consecutive get segments to begin its operation. When the accelerator completes its task, it issues an interrupt signal and

performs a put segment for the processor to get the output. The processor that sees the interrupt will perform a blocking get from the accelerator.

8. Considering a 32-bit data bus and a burst length of 8, estimate the number of clock cycles required to transfer the data through the channel.

Description: Putting the System Together

9. Modify the processor bracketing to be able to use the channel you developed above for putting the inputs for the accelerator and getting the output segment from it. Also include an interrupt input in this bracket.
10. Rewrite your MLP layer C++ program to take advantage of the accelerator. Your code now has parts for preparing data for the accelerator, receiving data from it, and looping enough time to complete the layer computation. The portioning of the matrix must be performed on the processor side. Specifically, the original matrix multiplication of size 3×8 should be divided into multiple 3×2 multiplications, each sent to the accelerator sequentially.
    To transfer each portion to the accelerator, the processor issues two consecutive put segments, and asserts the startMMA signal. The accelerator sees the asserted startMMA and performs two consecutive get segments to begin its operation.
    After each portion is processed by the accelerator, it interrupts the processor by asserting intrMMA and performs a put segment for the processor to get the output. The processor that sees the interrupt will perform a blocking get from the accelerator.
11. Estimate the overall number of clocks needed for this operation,