CS6230
Project 2: Matrix Multiplication Unit Design

JADAPALLI ASHOK KUMAR -EE24M094 BONDUGULA PRANAV-EE24M088

Table of contents:

CS6230
Project 2: Matrix Multiplication Unit Design

JADAPALLI ASHOK KUMAR -EE24M094 BONDUGULA PRANAV-EE24M088

Project Description:

Build a '4x4' systolic array module using the MAC. Use the reference MAC model to implement Matrix Multiplication simulation using the cocoTb framework. After building the systolic array, perform Matrix Multiplication of two 4x4 matrices, containing random values, for both the data types (int8 and bfloat16).

Project Overview:

This project implements 4x4 systolic array for matrix multiplication using Bluespec system Verilog (BSV). The systolic array structure enables efficient parallel computations, making it well-suited for hardware acceleration.

The system supports two precision formats:

- Int8 operation for integer arithmetic
- BF16 operations for reduced-precision floating-point arithmetic.

Software Architecture:

The MAC unit composed of two main modules:

1. Processing Element(PE) Module: Each PE is a basic unit of the systolic array responsible for performing the MAC operation on incoming matrix elements and propagating intermediate values to adjacent PEs.
   - Interface
     The PE_Ifc exposes the following methods:
     - Load_A(Bit#(16) a): Load an element from matrix A into the PE
     - Load_B(Bit#(16) b): Load an element from matrix B into the PE
     - Load_C(Bit#(16) c): Load an accumulator value c into the PE
     - Load_s1_or_s2(Bit#(1) sel): Select the operation mode(int8 or bf16).
     - Get_Result(): Retrieve the result computed by the PE.
     - Pass_A(Bit#(16) a): Pass an element of A to the next PE.
     - Pass_B(Bit#(16) b): Pass an element of B to the next PE.
   - Internal Design:
     The PE Includes:
     - Registers(reg_A) and (reg_B) to store matrix elements.
     - A MAC unit(macunit) for multiply-accumulate operations.

     The PE uses the load_* methods to set up inputs and control values, while results can be retrieved using get_Result().

2. Systolic Array Matrix Multiplier Module
   - Purpose:The mkMmUnit module orchestrates the behaviour of the systolic array by instantiating a 4x4 array of PEs and providing methods to interact with the array.
   - Interface: The MmUnit_Ifc exposes the following methods:
     - ✓Load_A_row(Vector#(4, Bit#(16) aRow, UInt#(8) rowIndex): Load a row of matrix A into the specified row of PEs.
     - ✓Load_B_col(Vector#(4,Bit#(16) bCol, UInt#(8) colIndex): Load a column of matrix B into the specified column of PEs.
     - ✓Load_C(Vector#(16,Bit#(32)) cValues): Load accumulator values(C) into the PEs.

- ✓Load_s1_or_s2(Bit#(1) sel): Set the precision mode(int8 or bf16) for all PEs.
- ✓Get_results(): Retrieve the final results from all PEs in a flattened vector format.

➢ Internal Design:

- The module instantiates a 4x4 array of PEs:
  - Pes[row][col]: Represents a PE at row row and column col:
- Data propagation:
  - Matrix A rows are loaded horizontally into PEs.
  - Matrix B columns are loaded vertically.
  - Results are aggregated from all PEs into a vector.
- Computation:
  - Each PE computes the MAC operation: Result=AxB+C
- Output Retrieval:
  - Results are collected from all PEs using get_results

Conclusion:

This MM unit design achieves efficient parallel matrix multiplication with modular, reusable components.