Assignment 4, CS223 (Hardware Lab)

Dept of Comp. Sc. & Engg., Indian Institute of Technology Guwahati

Weightage=25% Marks, Demo on 19th Apr 2017, Left over demos on 21st Apr 2017

Computing Integer Matrix Multiplication using FPGA

Matrix multiplication naive approach take $O(N^3)$ time but it have high degree of parallelism.

Pseudocode

```
unsigned short A[16][16], B[16]16], C[16][16]
CreateTransposeOfB(); // It convert column access of B to row access
//Do matrix multiplication
for(i=0;i<N;i++){      // Take ith row of A
      for(j=0;j<N;j++){   // Take jth Col of B
          //For each C[i][j] computation can be done in parallel
          S=0;   // Take row of B-Transpose
          for(k=0;k<N;k++) // A[i] is same in this loop
          // Dot Product of Vector A[i] and B[j]
              S=S+A[i][k]*B[j][k]; // Both A and B have row access as B is transposed
          C[i][j]=S;

     }
}
```

All C_{ij} computation can be done in parallel. Also computing one Cij can be done in parallel by using log n steps.

In FPGA, we can take benefits of this parallelism to speed up the matrix multiplication. Matrix multiplication uses dot product heavily, so dot product operation on two vectors need to be done efficiently. As in dot product, we use this kind S=S+X[i]*Y[i] of operation, which can be easily pipelined and DSP(multiply and accurate unit) Slice of FPGA can be efficiently used.

To do:

Step 1: Transfer matrix A, matrix B to FPGA
Step 2: Compute the result in FPGA
Step 3: Transfer back the result C to PC

or

Step 1: Transfer matrix A
Step 2: At the time of transfer of B, Compute the result in FPGA

- Transfer some elements of B, compute the result for that elements in pipeline fashion

Step 3: Transfer the result C back to PC

Some Hint

- Suppose you create 16 module, each module will generate result for 256/16=16 Cij
- Store every row of A in two module: Store R0 of A in module M0, Store R1 of A in module M1, ….
- Every module takes a assigned Row of A and take 16 Rows/Columns of B, and calculate 16 Cij
- All 16 modules will/can be run in parallel.
- Digilent Atlys Board: There are 116 numbers of 18K bit of block RAM.
  - Every module of your code can instantiate two block RAM.
- Different array infer different block RAM, If you are declaring an array of significant size in a module/entity, ISE tool will infer block ram for that array.
- Every clock cycle in compute: Read one elements of B, broadcast that element to all modules. Every module will calculate the required calculation for that elements of B.
  - This can be from USB ports, get one element of B from ports and compute based on that

Pseudocode:

```
for(i=0;i<16;i++){
                Read ith Row and  send to ith Module
}
for(col=0;col<16;col++){
    for(i=0;i<16;i++)
                Read B[i][col] and sendTOAllModules
}

//All module work in parallel
Module(){
        moduleID=r;
        Receive Row of A
        elem=0; j=0;col=0;S=0;
        for(elem=0;elem<256;elem++){
                Recv B[i];  S = S + B[i]*A[j]
                j++;
                if(j%8===0) {
                  C[r][col]=S;
                  S=0; j=0; col++;
        }
      }
}
```