Computer Architecture & Organization

Programming Project on Simulator for APEX

Prathyusha Vudatha

[svudath1@binghamton.edu](mailto:svudath1@binghamton.edu)

B00708850

**Project Description**:

APEX(**A**nother **P**ipeline **EX**ample) simulator simulates the working of APEX processor. It take set of instructions as input and outputs the result as per APEX processor’s instruction set. It implements the 5-stage pipelined execution which occur cycle-by-cycle.

Stages of APEX processor:

1. **Fetch (IF)**: This stage fetches the instruction one-by-one.
2. **Decode**: Decode stage decodes the instruction fetched at the previous stage and reads the value of architectural registers form the Register File.
3. **Execute (EX)**: The execution stage performs the execution according to the decoded type.
4. **MUL1** : The MUL1 stage performs the execution according to the decoded type.
5. **MUL2 :** The MUL2 stage performs the execution according to the MUL1 type.
6. **Memory (MEM)**: Instruction performs read and write to the memory in this stage according to the requirement.
7. **Write Back (WB)**: The value generated in the Execution and Write Back stage is written back to the architectural register to the register file through a write port.

**Development Environment**:

* Language used: C++

**Flow of Program Execution:**

The main() function calls the functions for the stages of the APEX in the reverse order, i.e.:

* WriteBack()
* Memory()
* Execute()
* MUL\_1()
* MUL\_2()
* DecodeRF()
* InstructionFetch()

These functions are within a FOR loop which iterates until all the function stages get completed or ‘HALT’ instruction is encountered.

**Description of Variables used**:

* **Int data\_array[10000]**: The memory array of type integer ranging from 0 to 9999.
* **Int pc\_value**: Stores the value of Program Counter and increments for each instruction fetched.
* string instruction\_string; : Stores instruction string.
* string source\_registers\_add : String stores source register
* int source\_registers\_value : Integer index of source register
* int source2\_registers\_value: Interger index of source register 2
* string source2\_registers\_add: String stores source register
* string dest\_register\_add: string values of destination register
* int dest\_register\_value: integer index values of destination register
* string opcode: string value of the opcode
* string literal\_string: string value of the literal
* int literal\_value: integer value of the literal
* int pc\_value : pc\_value of the instuction
* int mem\_buff\_reg : Temporary register that stores the result in ALU.

**Description of functions used**:

* **Void initialize()**: A function to initialize the state of APEX at the beginning of execution.
* **Void Fetch()**: The Instruction Fetch (IF) stage which fetches the instructions to the APEX.
* **Void Decode()**: The Decode/Register File stage which decodes the fetched instruction and reads the value of registers from register file.
* **Void Execute()**: Execute stage which performs the execution such as Add, Sub, Mul, etc.
* **Void Memory()**: Memory stage where memory is accessed.
* **Void WriteBack()**: The Write Back (WB) stage in which values are written to the registers in the register file.
* **String convert(string)**: This function takes a number as a string and converts to an equivalent integer value.
* **Void display()**: The function used to display the cycles executed, contents of first 100 memory location and the architectural registers from the register file.