

# 3 Stage Pipeline

Group 9:

CS14B023 Rahul Kejriwal

CS13B005 Bharat Sai Botta

CS16S033 Debanjan Ghatak

## 1. Work Split:

Team Member	Work Share
Rahul Kejriwal	<ol style="list-style-type: none"><li>1. Designed architecture and module interfaces.</li><li>2. Building module for 'Register File' unit.</li><li>3. Building module for 'Decode Instruction and Fetch Operands' unit.</li><li>4. Building corresponding testbenches.</li><li>5. Helping Ghatak for 'Execute and Store Back' unit.</li></ol>
Bharat Sai Botta	<ol style="list-style-type: none"><li>1. Building module for 'Fetch Instruction'.</li><li>2. Building corresponding testbenches.</li></ol>
Debanjan Ghatak	<ol style="list-style-type: none"><li>1. Building module for 'Execute and Store Back' unit.</li><li>2. Building corresponding testbenches.</li></ol>

## 2. Assumptions:

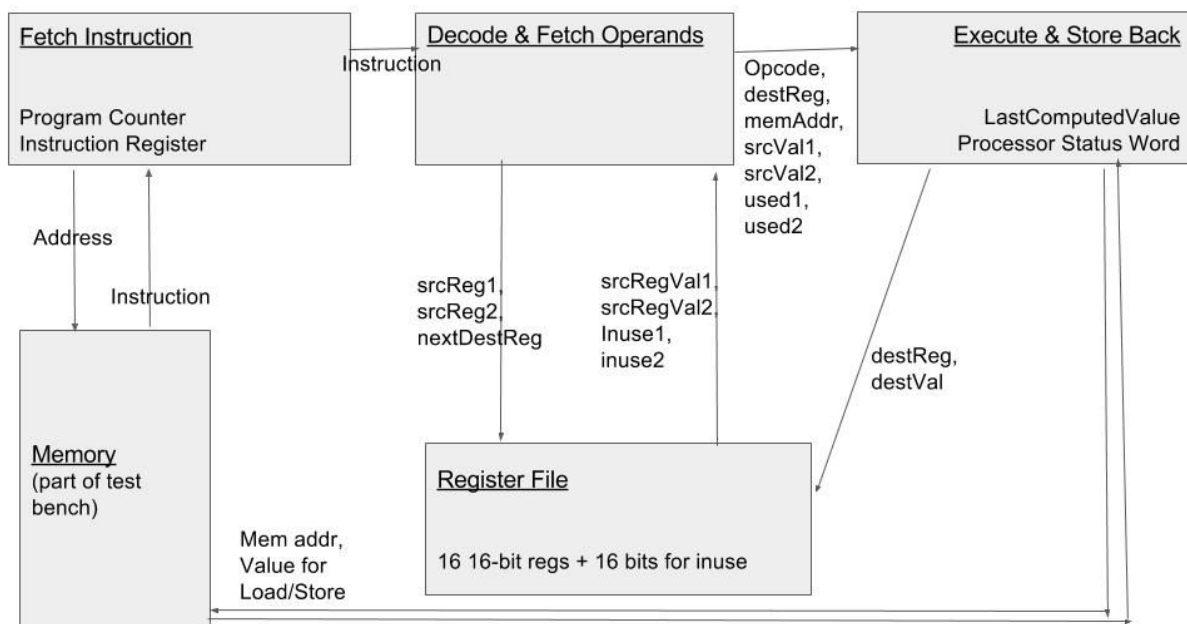
- a. We assume that the 'Fetch instruction' unit and the 'Execute and Store Back' (for LOAD/STORE instructions) don't use the same memory address in the same cycle. [As PS says one read/write port per memory address]
- b. All units finish their entire work within one clock cycle. [What this means, is that say 'Decode Instruction and Fetch Operands' takes 5ns and 'Execute and Store Back' takes 10 ns then the cycle width/duration is greater than

10ns. The cycle duration is large enough for each unit to finish their respective work for their current instruction within that cycle itself.]

- c. We are not using any delays (#n) in the verilog code. We do not worry about balancing the time taken by each pipeline stage to improve throughput.
- d. Register values can be changed during a cycle (not only on posedges). But register status bits change only on posedges.

### 3. System Architecture:

- a. Here, is the system architecture schematic showing the interfaces between the modules:



b. Description of Modules (and variables used in the schematic):

Module	Core Components	Input	Output
<b>Fetch Instruction Unit</b>	<ol style="list-style-type: none"> <li>1. Program Counter</li> <li>2. Instruction Register</li> </ol>	<ol style="list-style-type: none"> <li>1. Instruction (to be executed)</li> </ol>	<ol style="list-style-type: none"> <li>1. Instruction Address (to be read)</li> <li>2. Instruction (transfer to next unit)</li> </ol>
<b>Decode &amp; Fetch Operand Unit</b>		<ol style="list-style-type: none"> <li>1. Instruction (to be decoded)</li> <li>2. Value of 2 source registers, srcRegVal1 and srcRegVal2</li> <li>3. Status of 2 source registers, i.e., whether are being computed by the Execute Unit currently, inuse1 and inuse2</li> </ol>	<ol style="list-style-type: none"> <li>1. Source register address, srcReg1 and srcReg2 (to be read)</li> <li>2. Destination register address, nextDestReg (to set inuse bit)</li> <li>3. Opcode of instruction, opcode</li> <li>4. Destination register address, destReg for next instruction</li> <li>5. Source register values, srcVal1 and srcVal2</li> <li>6. Address of memory location in case of LOAD/Store</li> <li>7. Inuse bits for source registers, used1 and used2</li> </ol>
<b>Register File</b>	<ol style="list-style-type: none"> <li>1. 16-bit registers x16</li> <li>2. Inuse bits x16</li> </ol>	<ol style="list-style-type: none"> <li>1. Source register address, srcReg1 and srcReg2 (to be read)</li> <li>2. Destination register address of next instruction (to set inuse bit)</li> <li>3. Destination register address of current instruction, destReg and value to be stored there, destVal</li> </ol>	<ol style="list-style-type: none"> <li>1. Value of 2 source registers, srcRegVal1 and srcRegVal2</li> <li>2. Status of 2 source registers, i.e., whether are being computed by the Execute Unit currently, inuse1 and inuse2</li> </ol>
<b>Execute &amp; Store Back Unit</b>	<ol style="list-style-type: none"> <li>1. Processor Status Word</li> <li>2. LastDestVal (stores the value computed in the last cycle)</li> </ol>	<ol style="list-style-type: none"> <li>1. Opcode of instruction, opcode</li> <li>2. Destination register address, destReg for next instruction</li> <li>3. Source register values, srcVal1 and srcVal2</li> <li>4. Address of memory location in case of LOAD/Store</li> <li>5. Inuse bits for source registers, used1 and used2</li> <li>6. Memory read value for LOAD instruction</li> </ol>	<ol style="list-style-type: none"> <li>1. Memory write value for STORE instruction</li> <li>2. Memory address for LOAD/STORE instruction</li> <li>3. Destination register address, destReg</li> <li>4. Destination register value, destVal</li> </ol>

