ECE-586

COMPUTER ARCHITECTURE

ROLES AND RESPONSIBILITIES

MIPS- LITE 5 STAGE PIPELINE SIMULATOR

TEAM 18:

KUMAR DURGA MANOHAR KARNA (912527531)

SAI ROHITH REDDY YERRAM (909940739)

NIKHITHA VADNALA (941449738)

We Kumar Durga Manohar Karna, Sai Rohith Reddy Yerram & Nikhitha Vadnala have worked with a great co-ordination to complete this MIPS Lite 5 stage Pipeline Simulator Project, coming to roles and responsibilities, we have divided our work equally among ourselves to complete the project on time without any errors and any miss conceptions.

Roles and Responsibilities:

Kumar Durga Manohar Karna: Worked on the proper implementation of the pipeline stages is crucial for maintaining the correct flow of operations, ensuring that data is processed efficiently, and avoiding hazards or conflicts that could disrupt the pipeline's functionality. It involves carefully designing and implementing each stage, considering factors such as data dependencies, control signals, and synchronization mechanisms to ensure smooth transitions between stages.

Calculating Stall Conditions:

- Developed logic to detect situations where the pipeline must be stalled. This typically involves data hazards, control hazards, and structural hazards. Also implementing mechanisms to handle stalls once identified, such as inserting no-operation (NOP) instructions or implementing forwarding and bypassing techniques.

Sai Rohith Reddy Yerram: Worked on instruction set implementation involves writing the actual implementation code for all the instructions defined in the MIPS-lite instruction set architecture. It requires coding the functionality of each instruction according to the specifications laid out in the ISA documentation. This includes correctly handling the various operands, addressing modes, and any special cases or exceptions defined for each instruction. Also has implanted the integrating part as the instructions into the functional simulator and pipeline simulator components.

Vadnala Nikhitha: Worked on generating the control signals which are then propagated to the downstream stages of the pipeline, where they control and orchestrate the execution of the instruction. Each stage of the pipeline performs specific tasks based on the received control signals, such as reading operands from registers, performing arithmetic or logical operations, accessing memory, or writing results back to registers. Running the implemented design with the trace files to test its correctness. And modifying the design based on the results to ensure that the simulator behaves as expected.