RISC-V RV32I RTL DESIGN USING VERILOG HDL

FINAL PROJECT REPORT

UNDER GUIDELINES OF

MAVEN SILICON

BY

YK. MANASA REDDY

21BML0102

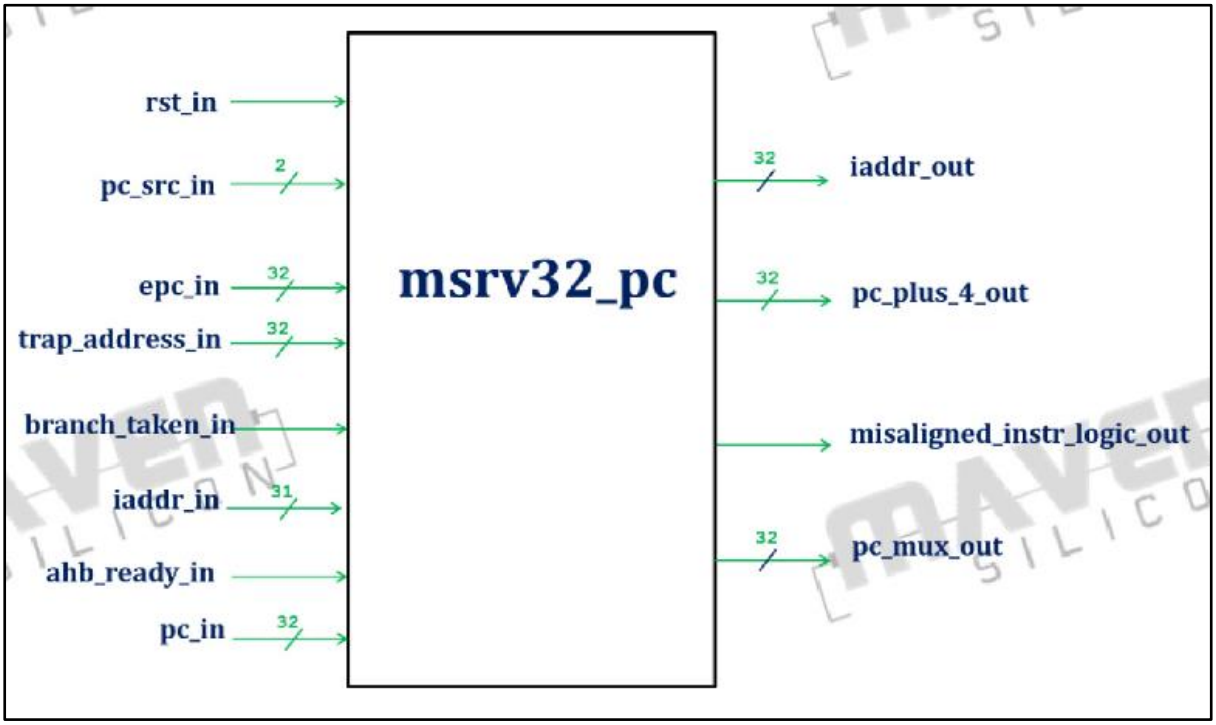
VIT-VELLORE

PROJECT OVERVIEW:

This project aims to design a 32-bit RISC-V processor with the RV32I instruction set architecture using Verilog hardware description language (HDL). The processor will be implemented at the Register-Transfer Level (RTL) and will include key components such as the instruction fetch, decode, execution units, and register file. The final design will provide a solid foundation for understanding and implementing RISC-V processors in custom hardware.

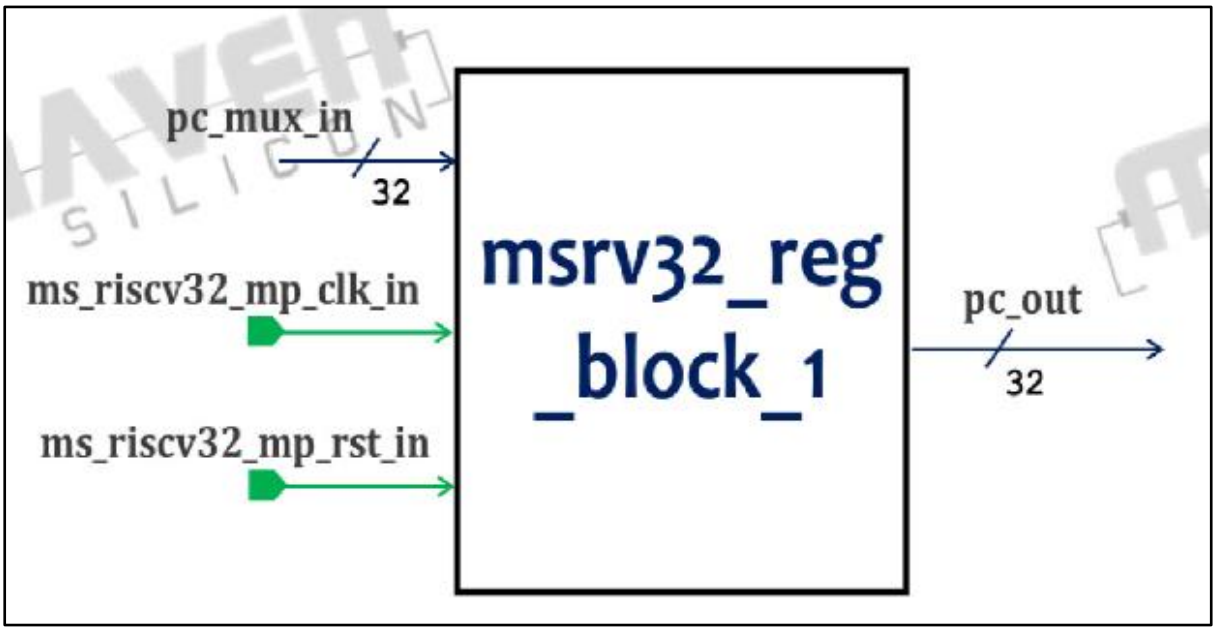
Sub Modules Block Diagram and Its Functionality:

1) **PC MUX:**



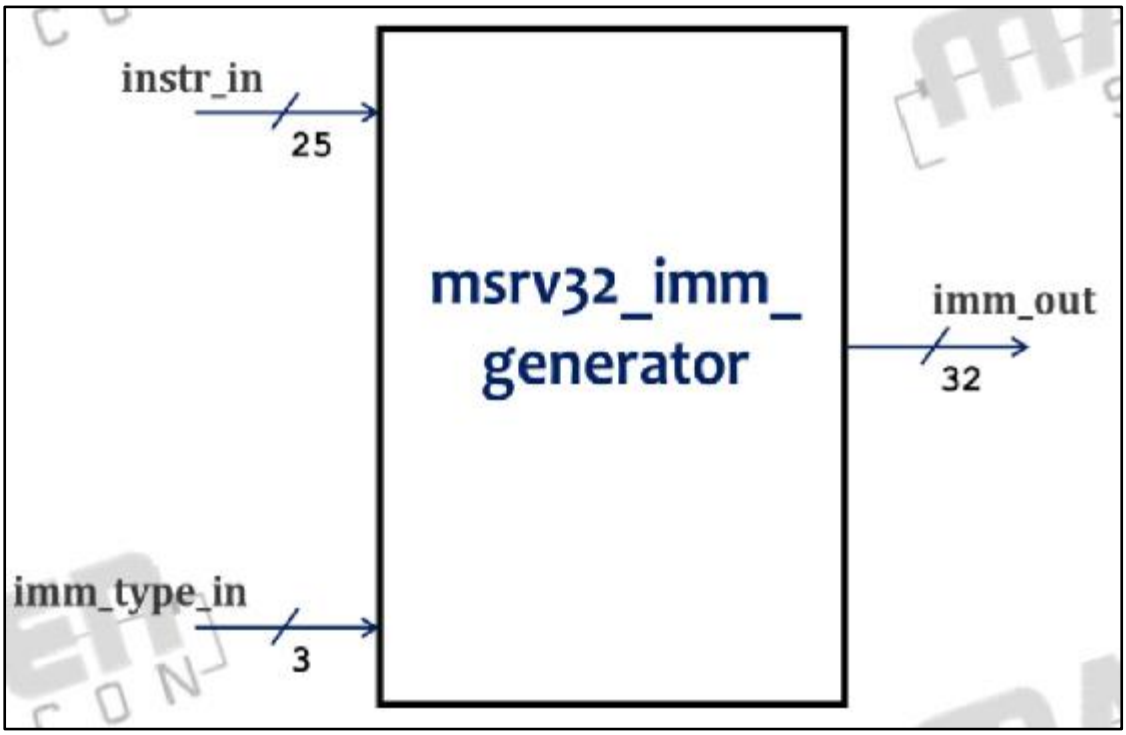
PC MUX is used to hold the address of next instruction to be executed by selecting between the sequential instruction address (PC+4) and a new address during control flow changes like jumps or branches.

2) REG BLOCK-1:



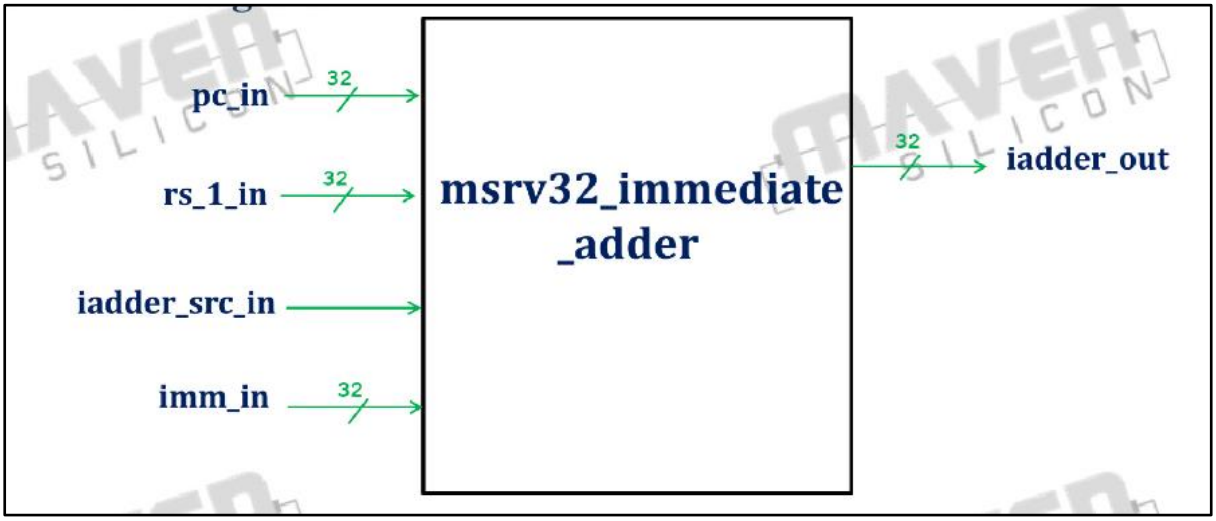
It registers the pc_mux_in and produces the output at the posedge of clk if there is no reset.

3) IMMEDIATE GENERATOR:



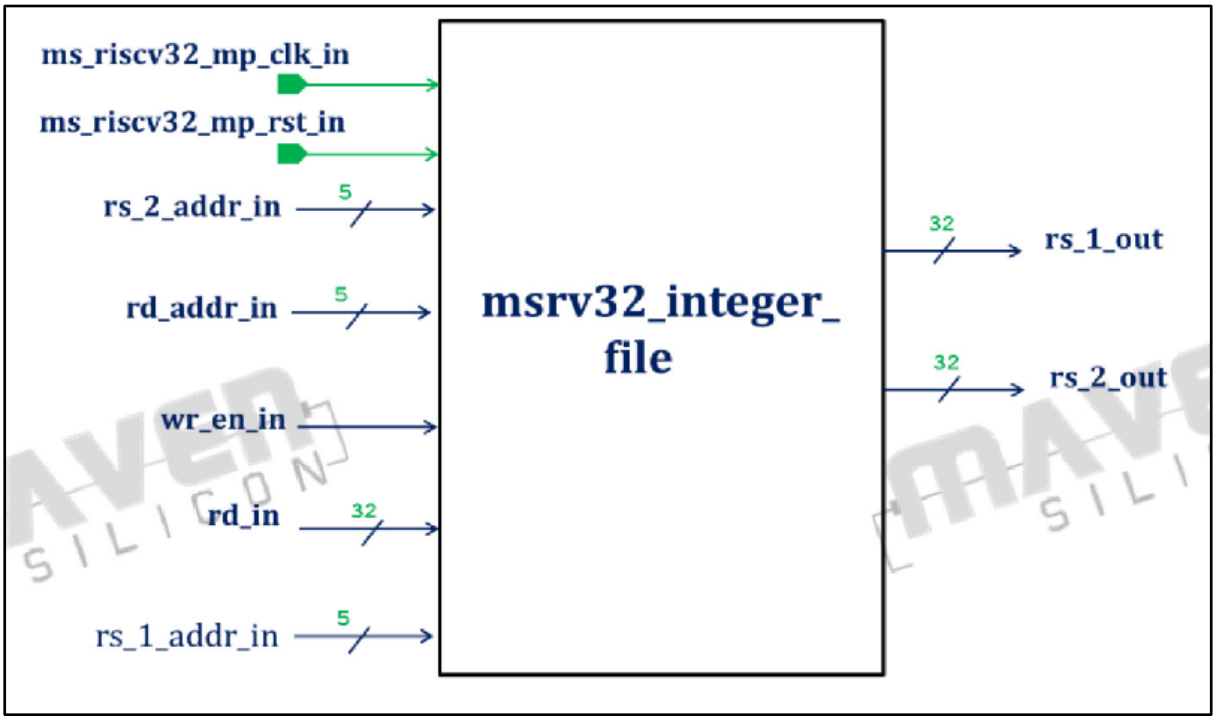
The Immediate Generator rearranges the immediate bits contained in the instruction. It takes the immediate field from an instruction and extends or sign-extends it to the appropriate bit width to provide the correct value for operations like arithmetic, logical, or data transfers.

4) IMMEDIATE ADDER:



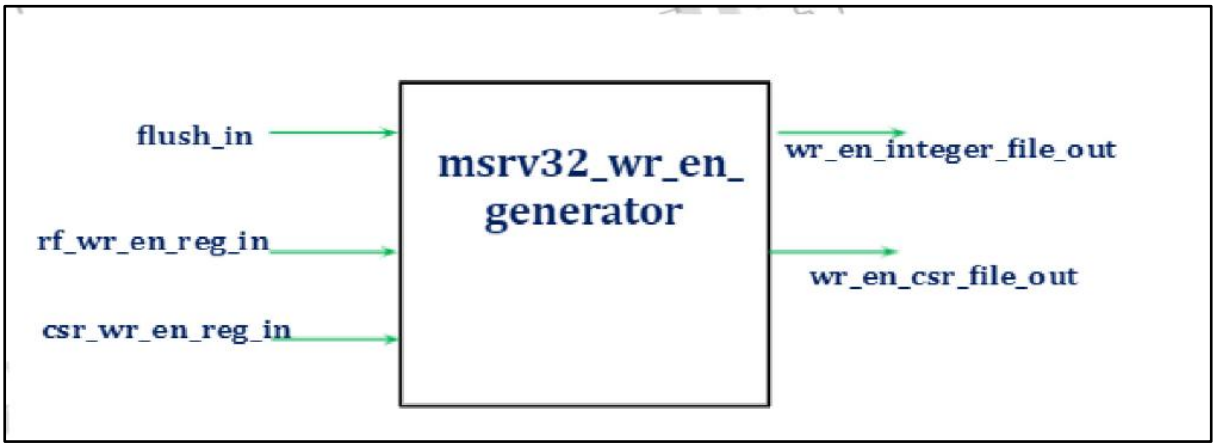
The Immediate Adder is responsible for adding the immediate value to a register value or another operand. It performs this operation as part of instruction execution, facilitating arithmetic and logical operations that involve immediate values in the RISC-V processor.

5) INTEGER FILE:



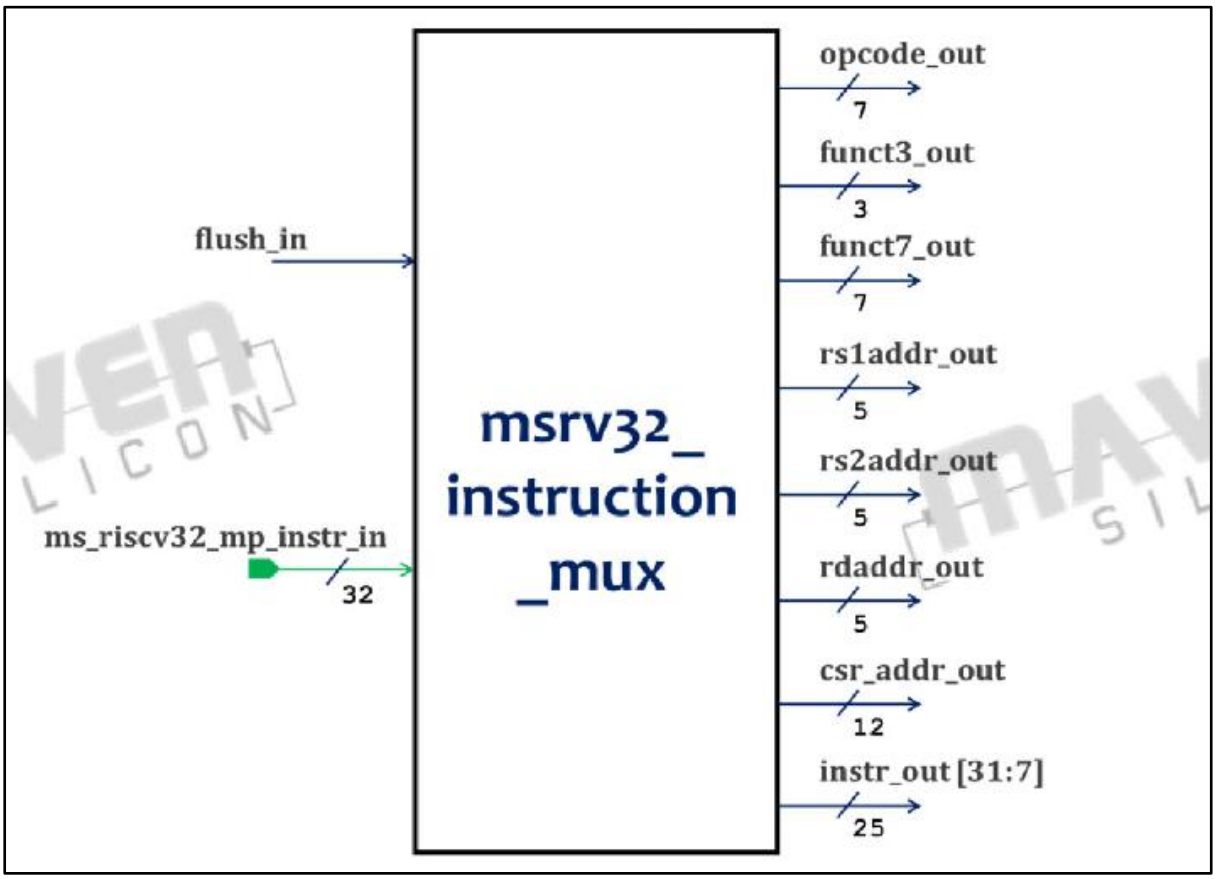
The Integer File serves as a set of registers for temporarily storing data. It allows for the reading and writing of register values during instruction execution, enabling data manipulation and transfer operations within the processor.

6) WRITE ENABLE GENERATOR:



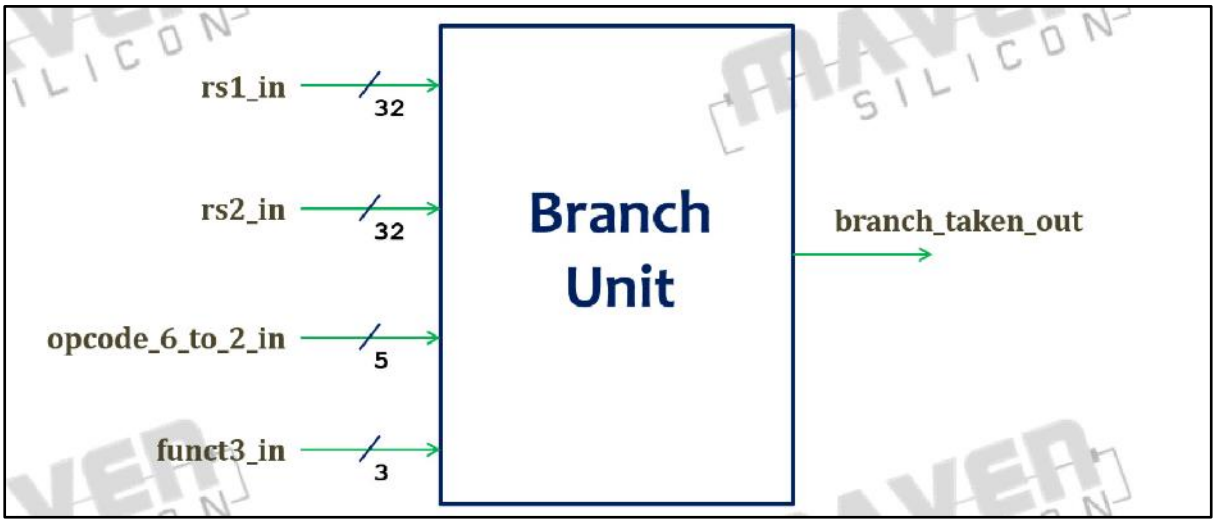
The Write Enable Generator decides if a particular register should be updated with new data when an instruction is executed. It checks the instruction and related conditions to control whether data should be written into the register file, managing how the processor handles information.

7) INSTRUCTION MUX:



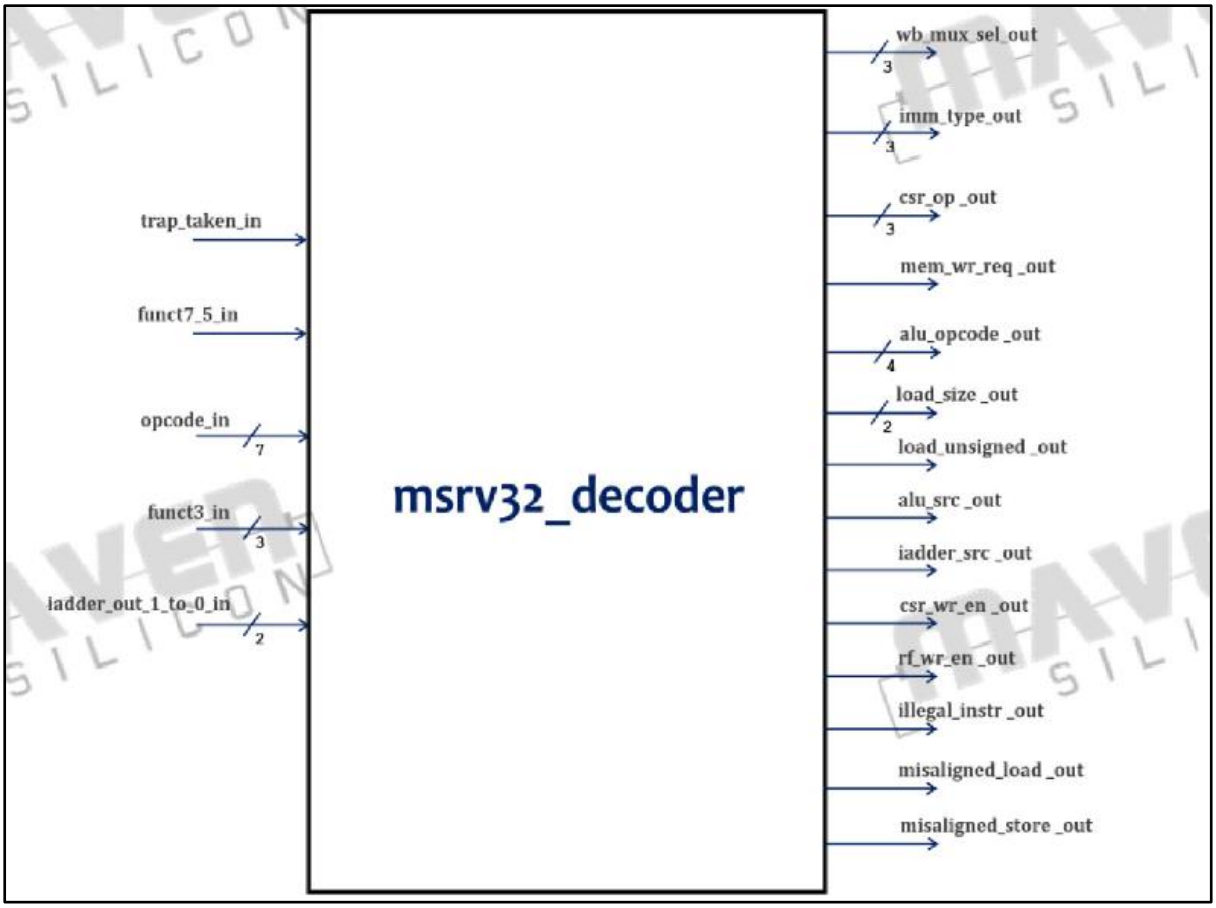
The Instruction MUX is responsible for selecting the appropriate instruction to be executed at a given time. It chooses between different instruction sources, such as the output of the instruction memory or control unit, to ensure that the correct instruction enters the processor's pipeline for execution.

8) BRANCH UNIT:



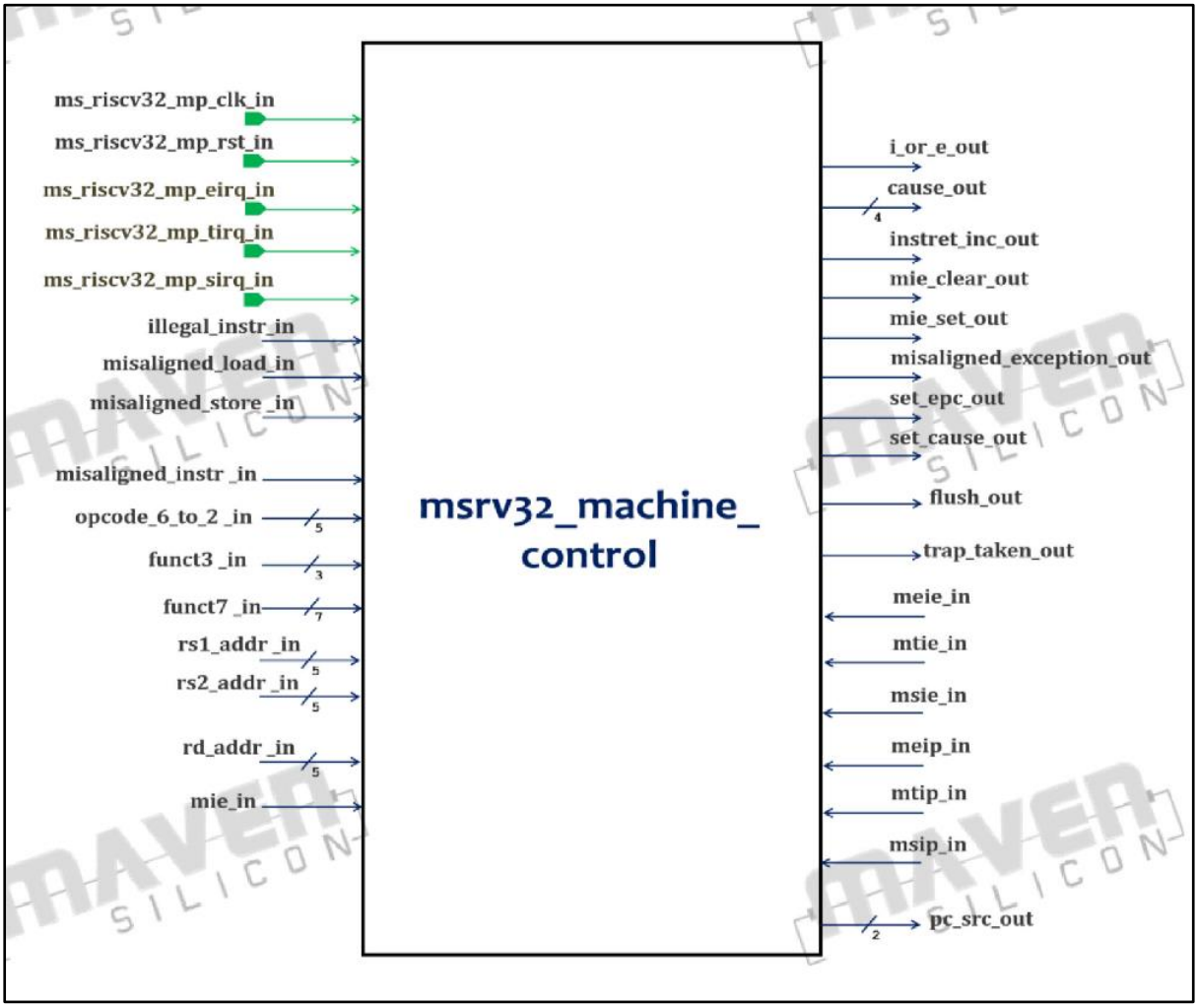
The Branch Unit is responsible for detecting and managing branch instructions. It determines whether a branch should be taken, calculates the target address, and controls the flow of instructions, ensuring the correct execution of conditional branches in the processor.

9) DECODER:



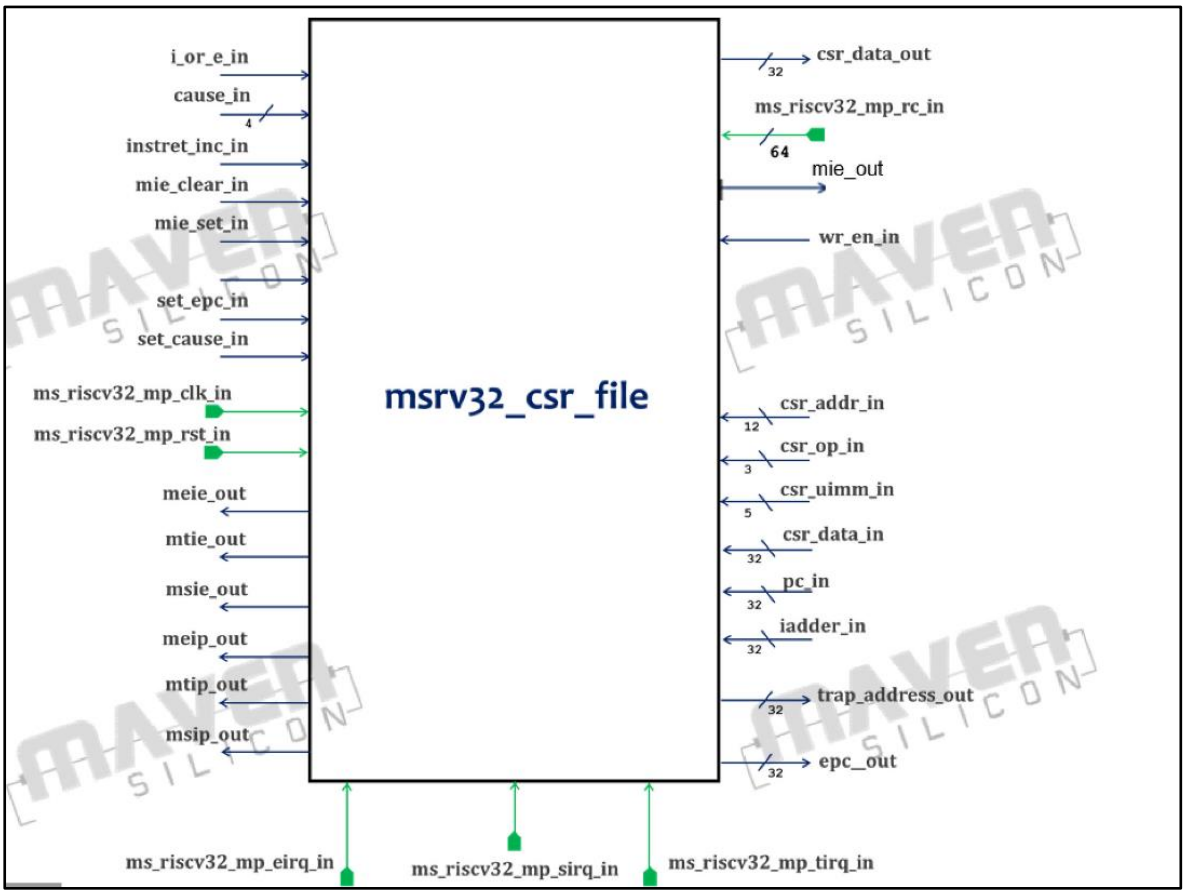
The Decoder decodes the instruction and generates the signals that control the memory, the load unit, the store unit, the ALU, the two register files (Integer and CSR), the Immediate Generator and the Write back Multiplexer.

10) MACHINE CONTROL:



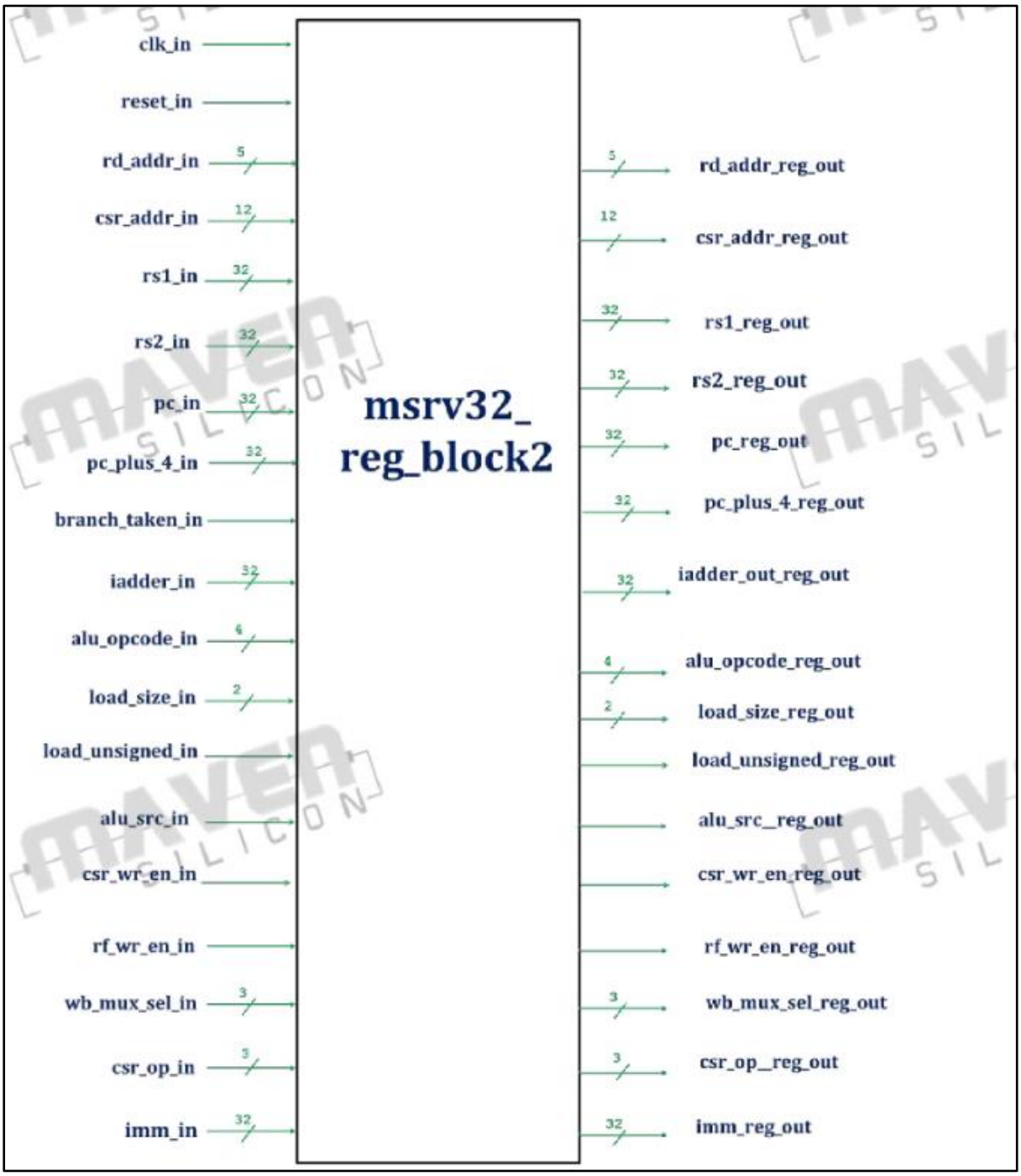
The Machine Control unit manages the overall control and coordination of the processor's operation. It generates control signals that synchronize different components, handle exceptions, and ensure the correct execution of instructions, serving as the central control hub for the processor's functionality.

11) CSR FILE:



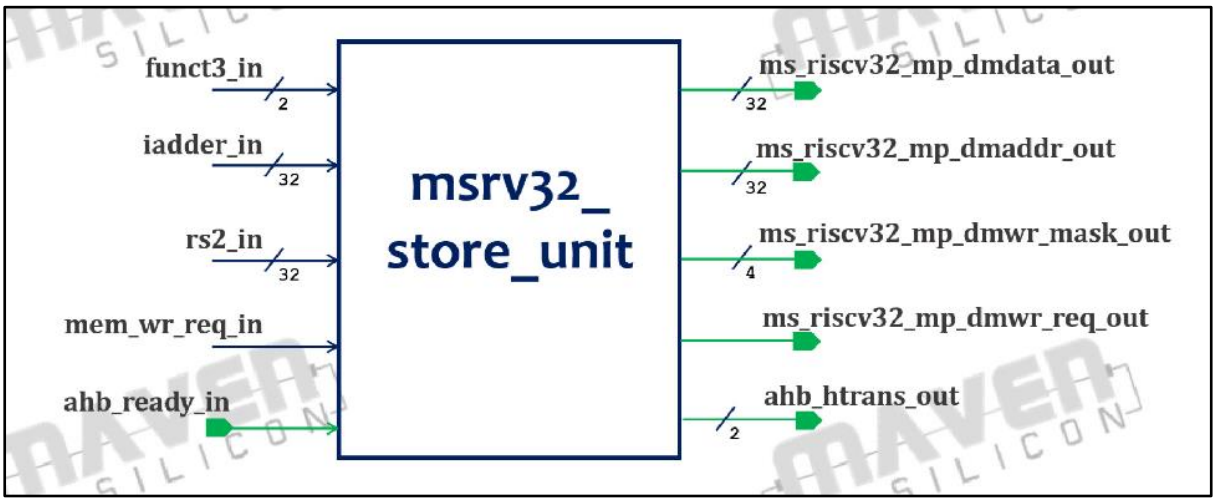
The CSR Register File has the control and status registers required for the implementation of M-mode. Read, Write, set, and clear operations can be applied to the registers.

12) REG BLOCK-2:



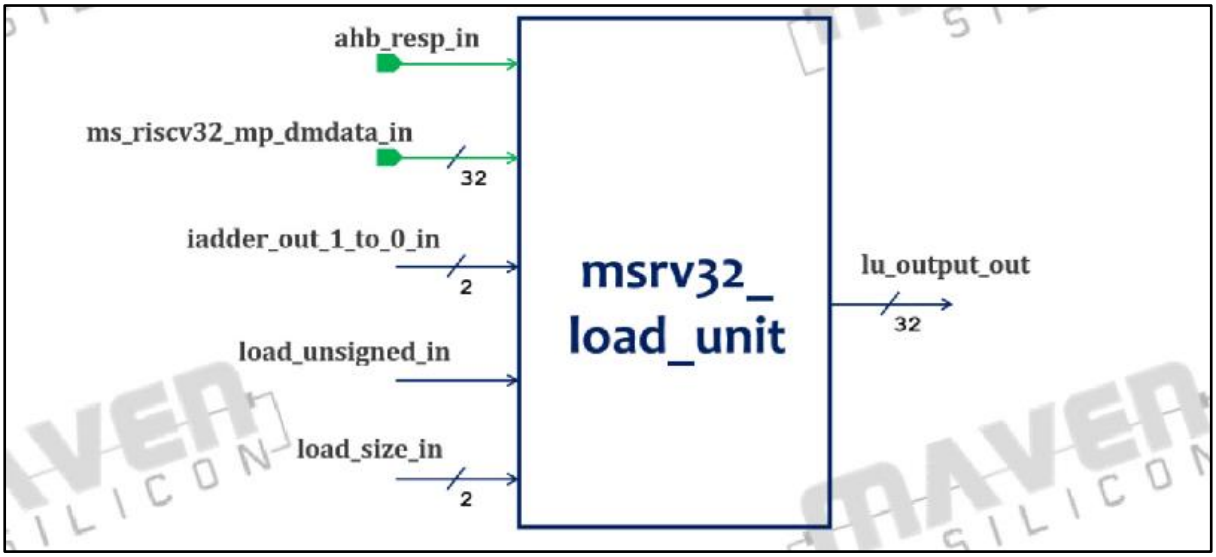
The Register Block-2 captures and stores incoming data on the posedge of the clock signal and produces the corresponding outputs, allowing the processor to maintain and update temporary data as part of its normal operation when no reset is active.

13) STORE UNIT:



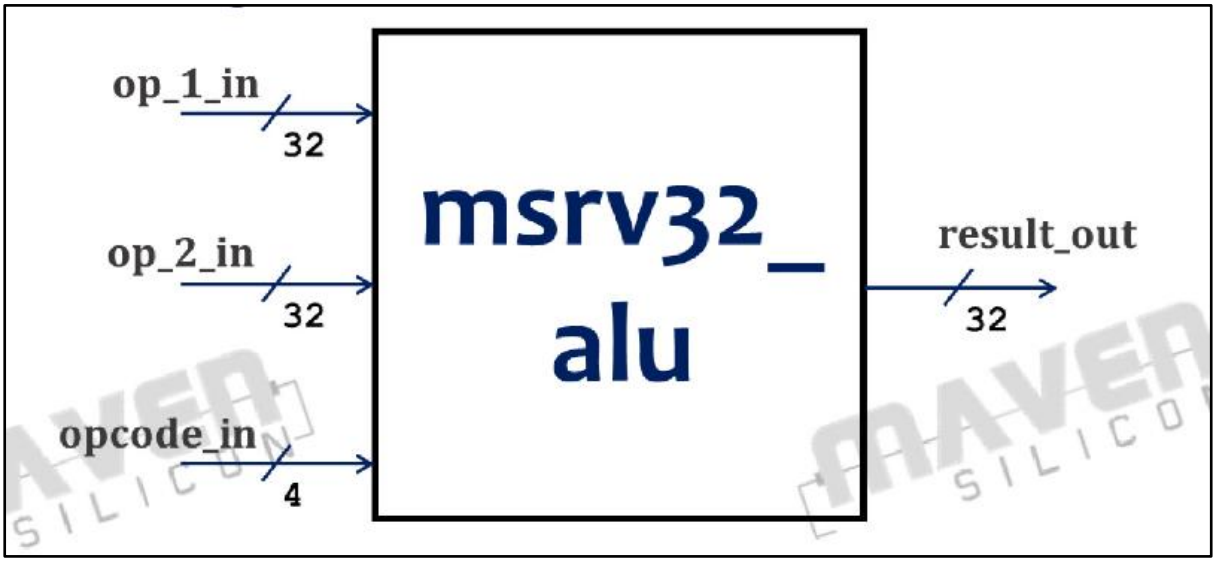
The Store Unit is responsible for handling instructions that store data from registers into memory. It calculates the memory address, controls data, and writes the data to the specified memory location.

14) LOAD UNIT:



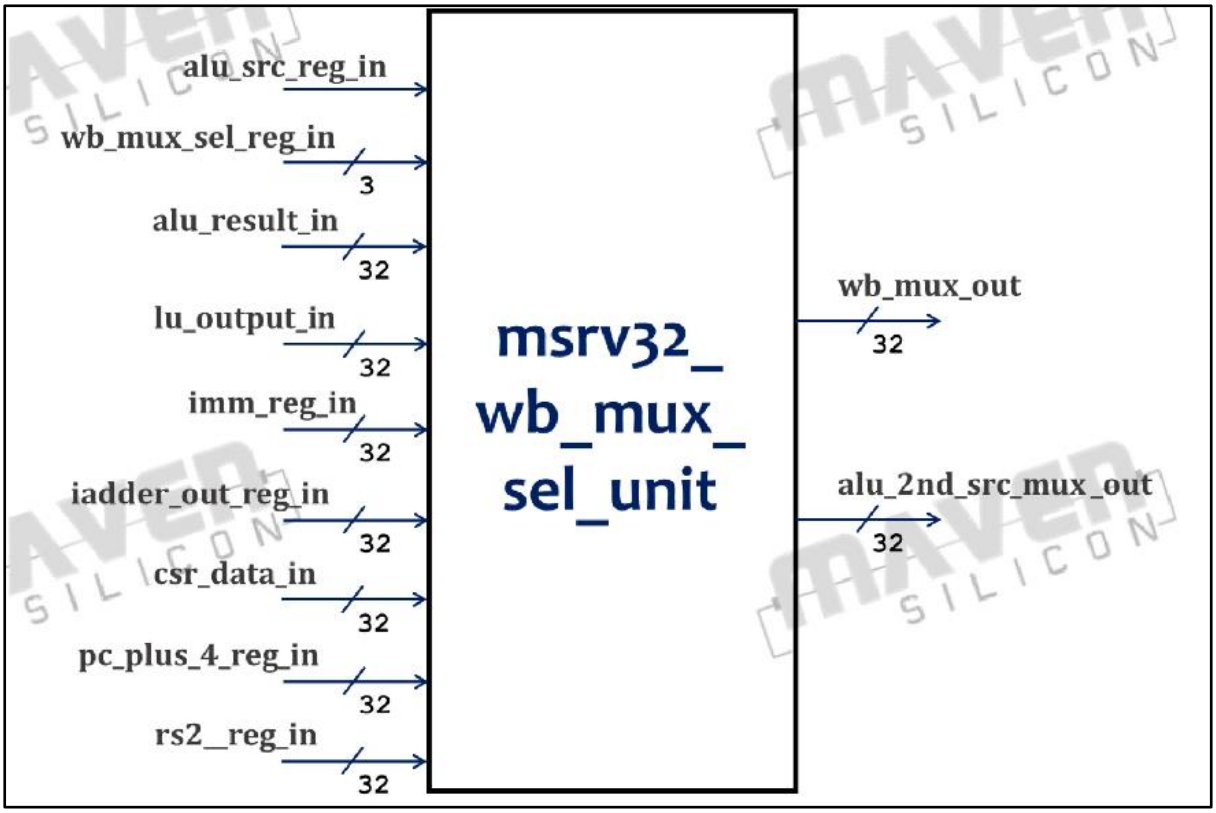
The load Unit reads the data_in input signal and forms a 32-bit value based on the load instruction type. It calculates the memory address, controls data reads from memory.

15) ALU:



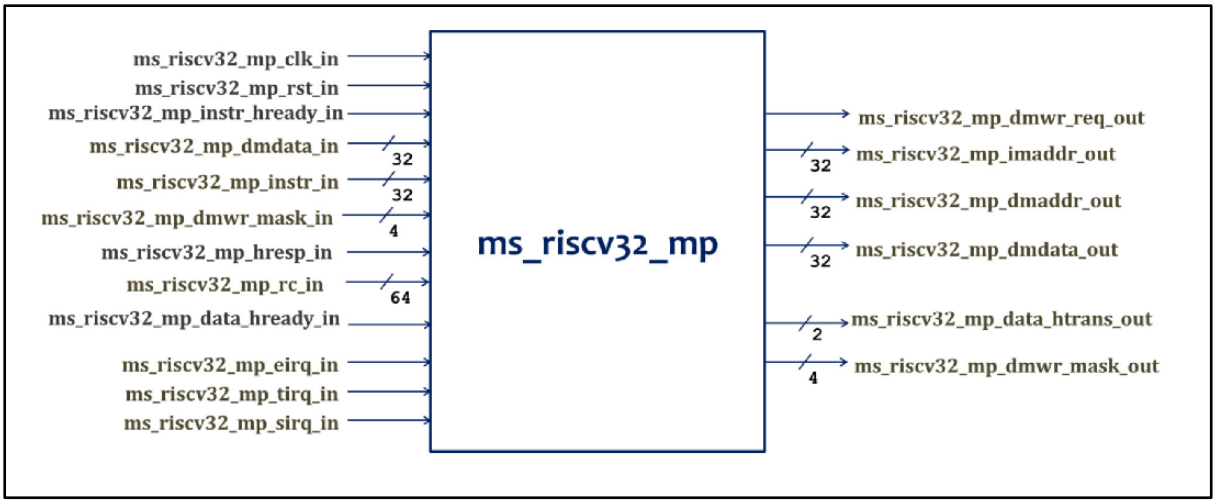
The Arithmetic Logic Unit (ALU) is responsible for performing various arithmetic and logical operations on data. It takes inputs from the processor's registers, processes them according to the instruction's opcode, and produces the output.

16) WB MUX SELECTION UNIT:



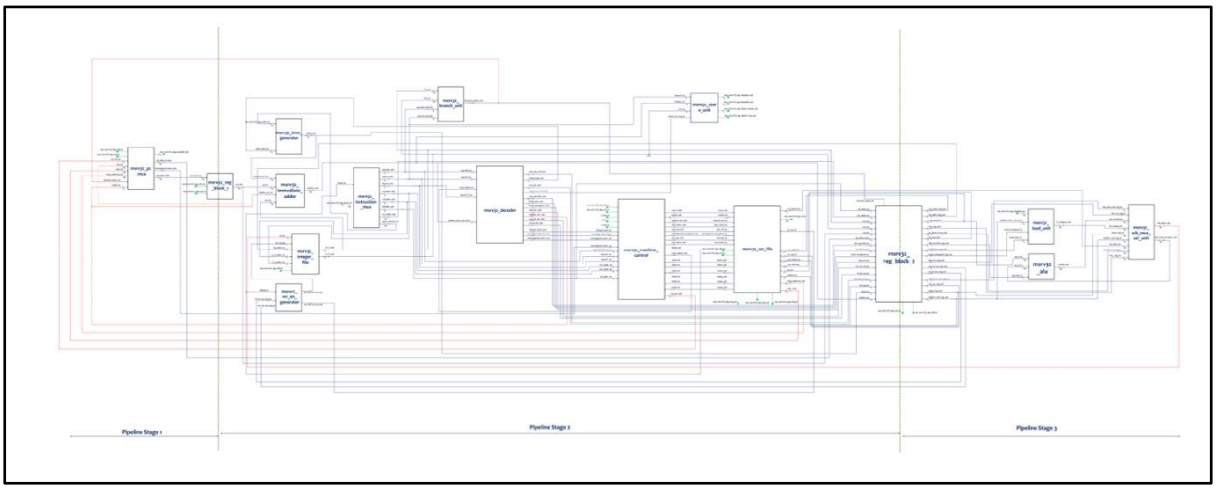
The Write-Back (WB) Selection MUX is responsible for choosing the data that should be written back to a register when multiple data sources are available. It selects the correct data to update the destination register, ensuring the proper flow of results from executed instructions to the processor's registers.

TOP MODULE:

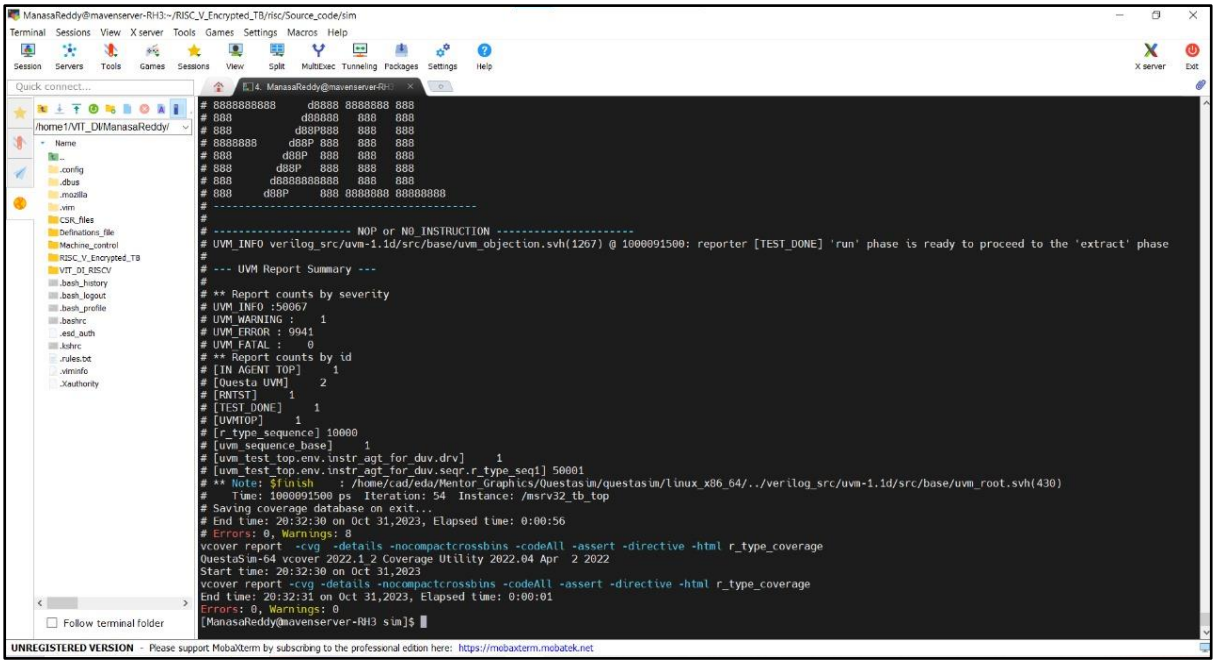


The TOP MODULE serves as the highest-level module that integrates and orchestrates all the essential components of the processor, including the instruction fetch, decode, execution units, memory access, and control units. It manages the flow of instructions and data throughout the processor, ensuring the coordinated operation of the entire RV32I processor design.

MICRO ARCHITECTURE OF RISC-V RV 32I:

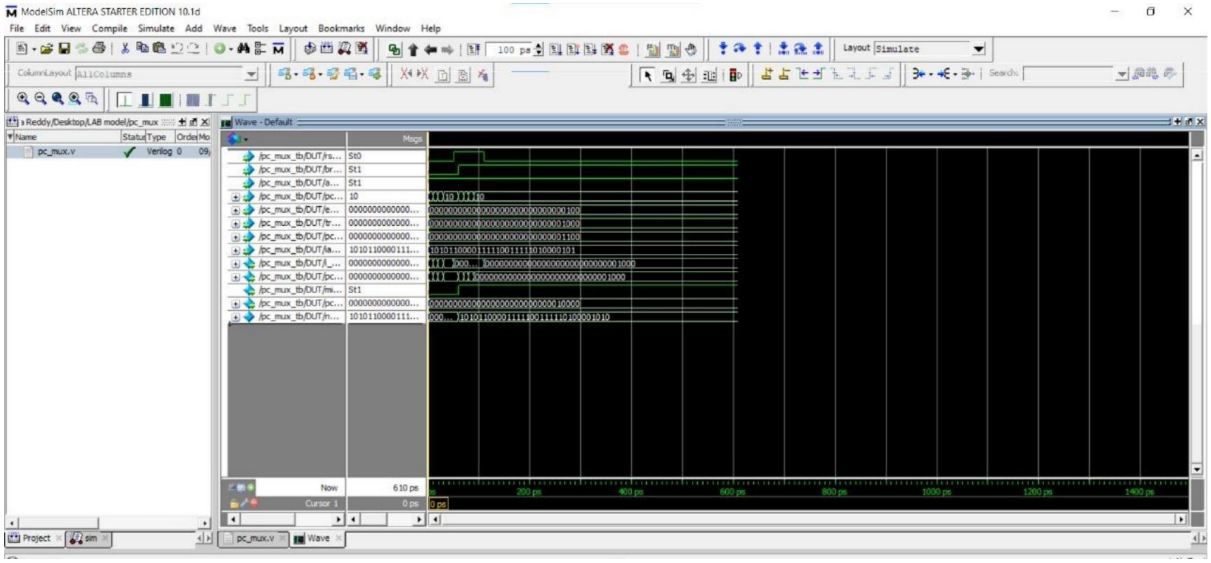


FINAL UVM VERIFICATION FOR RISC-V RV32I RTL

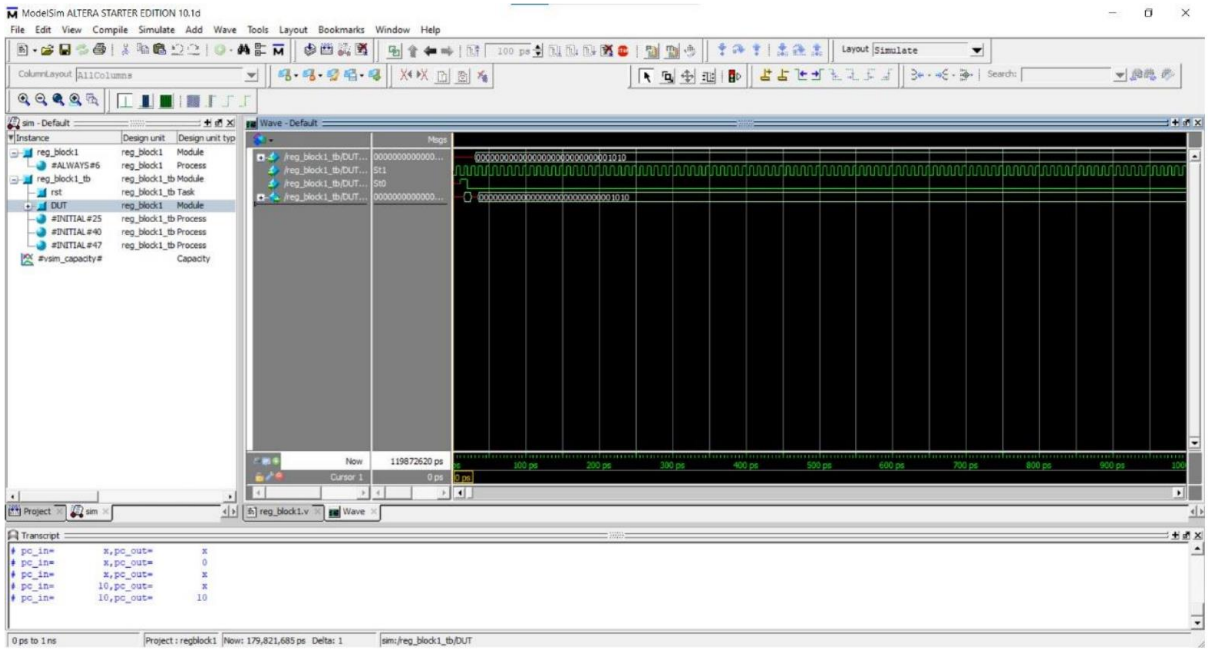


OUTPUT WAVEFORMS

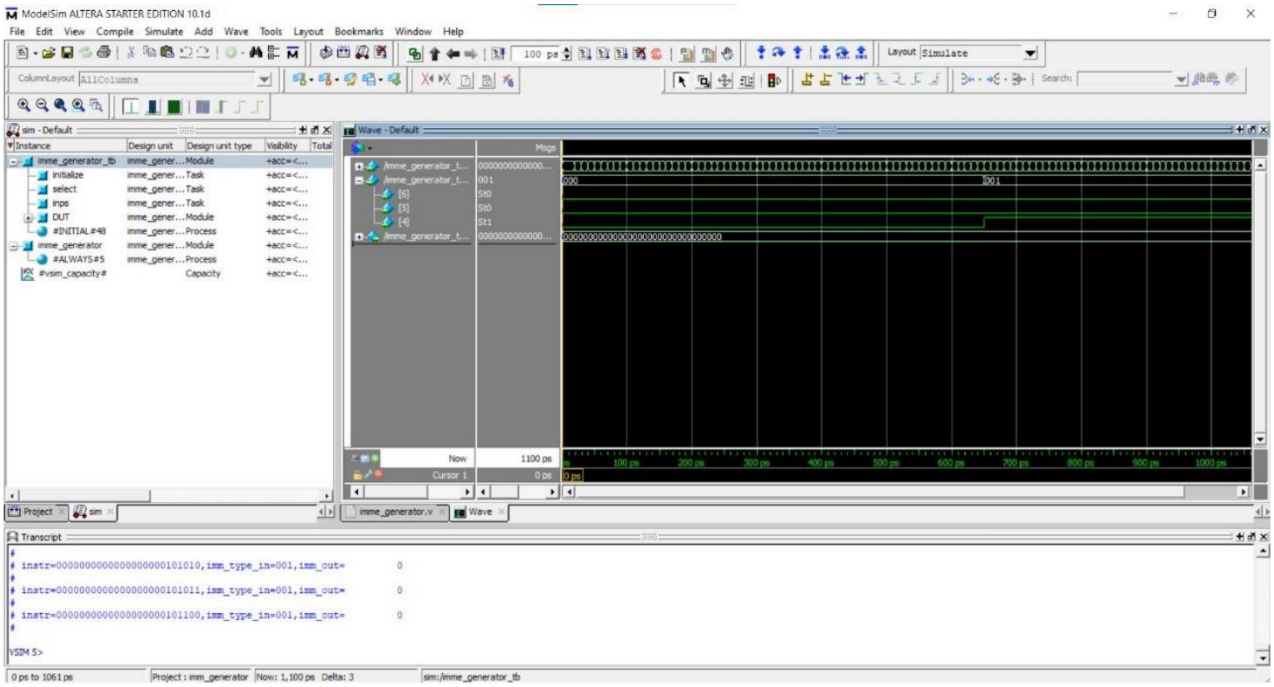
PC MUX:



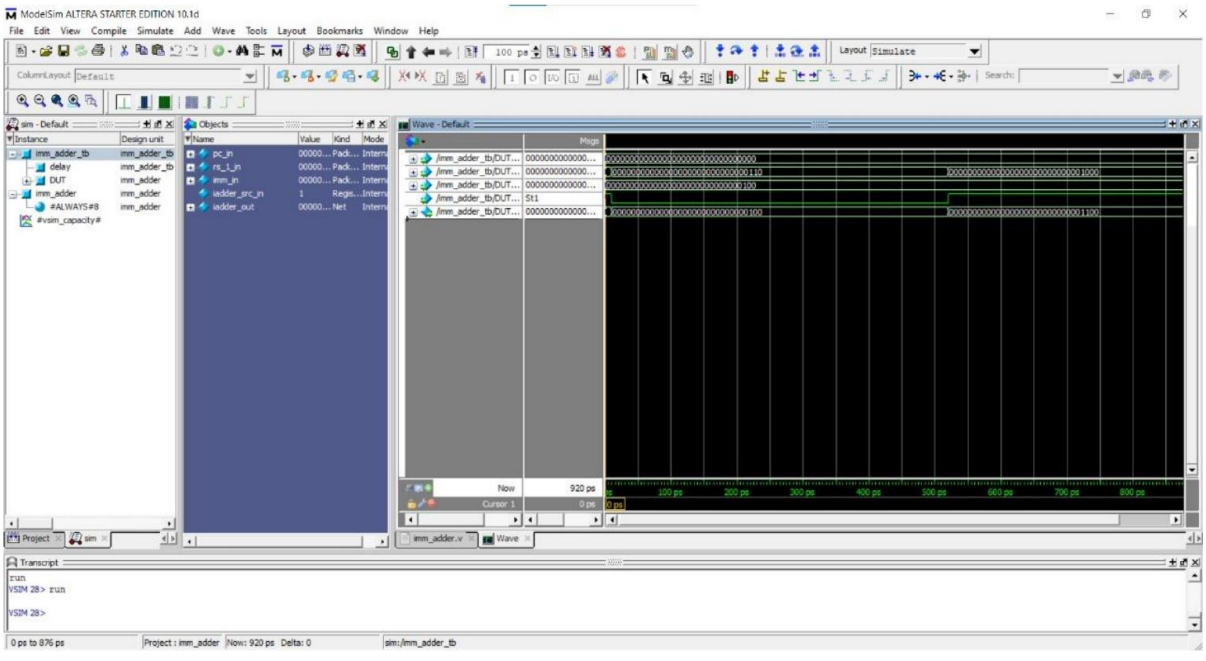
REG BLOCK-1:



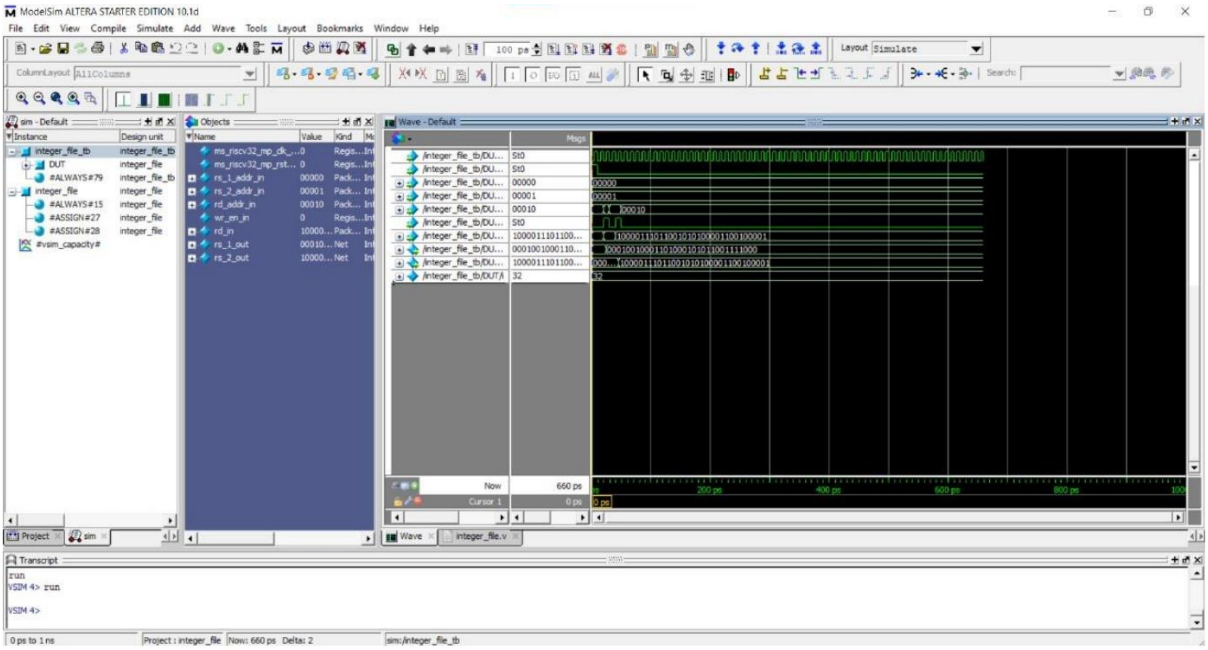
IMMEDIATE GENERATOR:



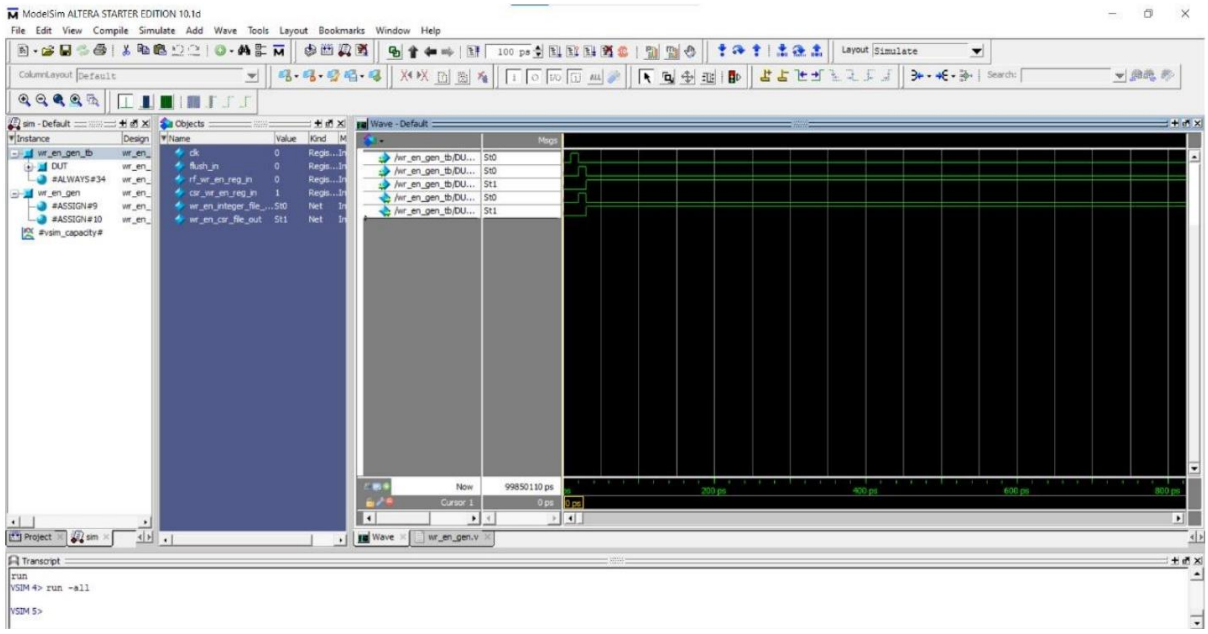
IMMEDIATE ADDER:



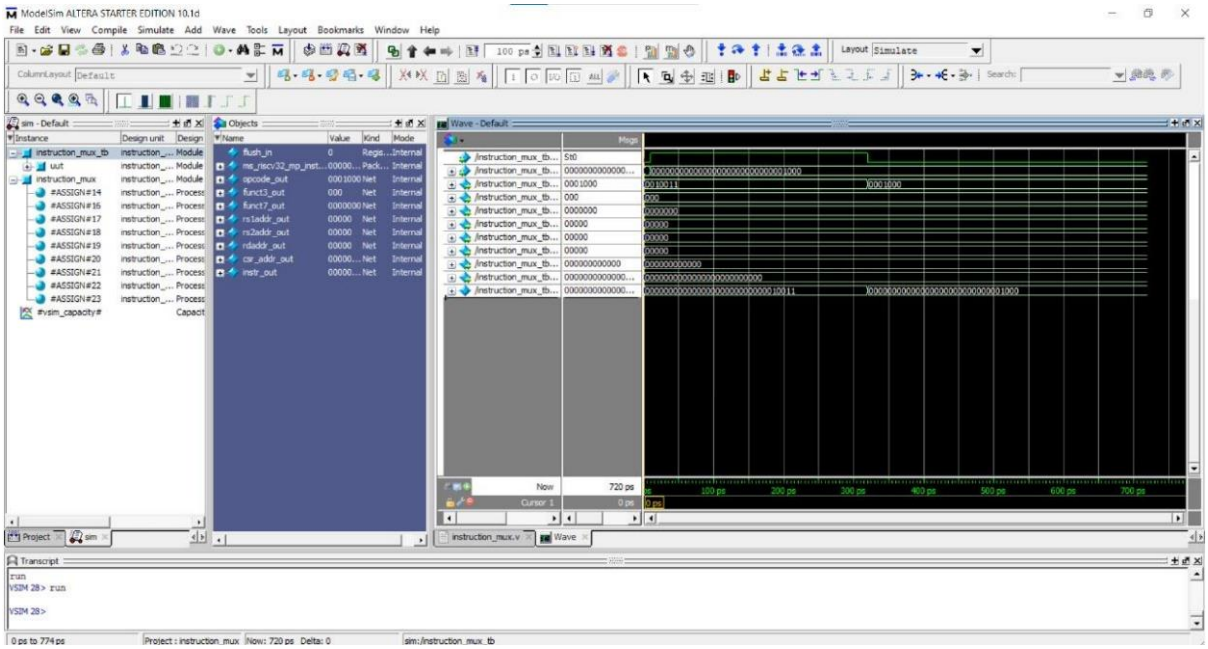
INTEGER FILE:



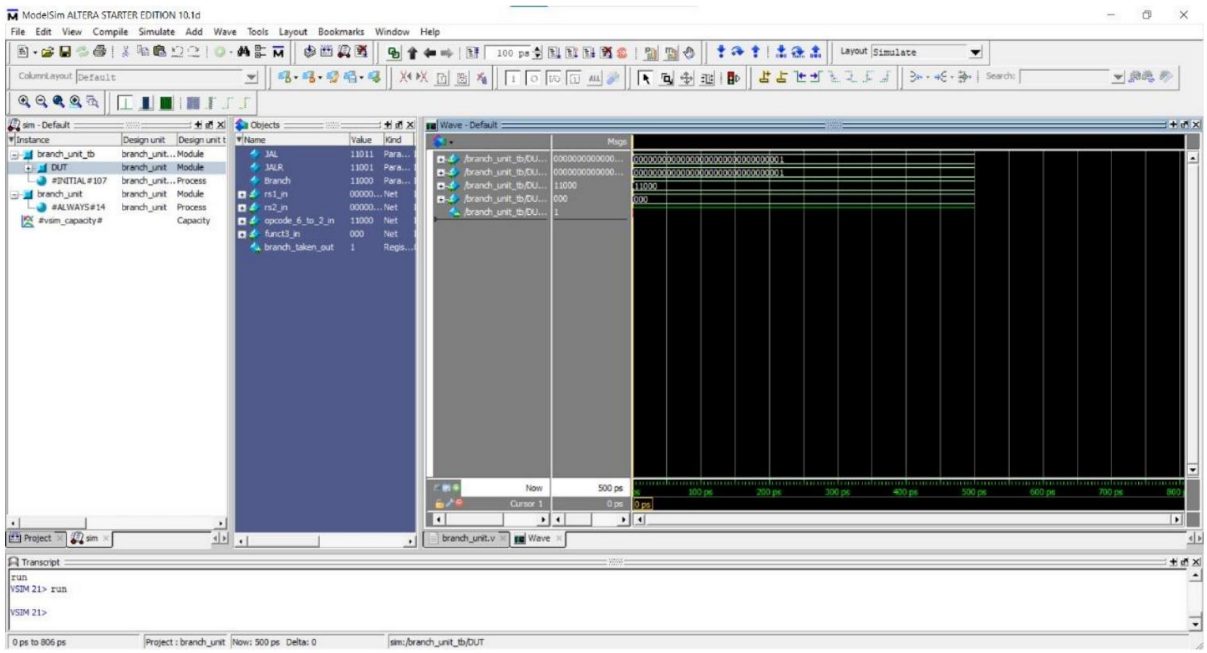
WRITE ENABLE GENERATOR;



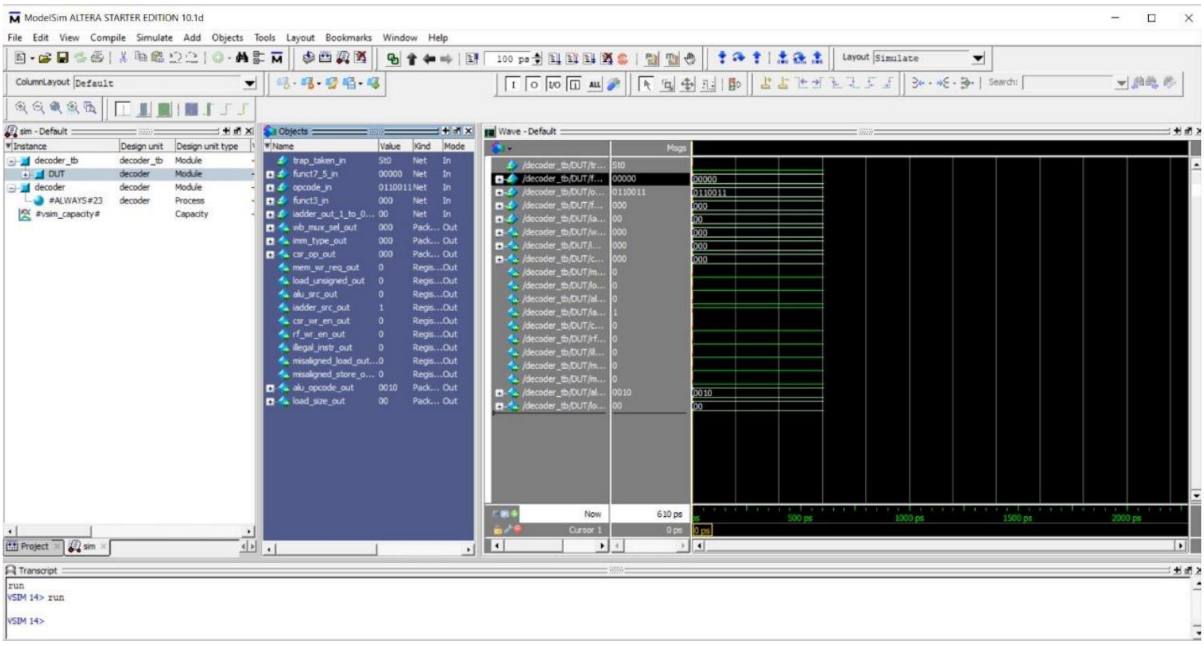
INSTRUCTION MUX:



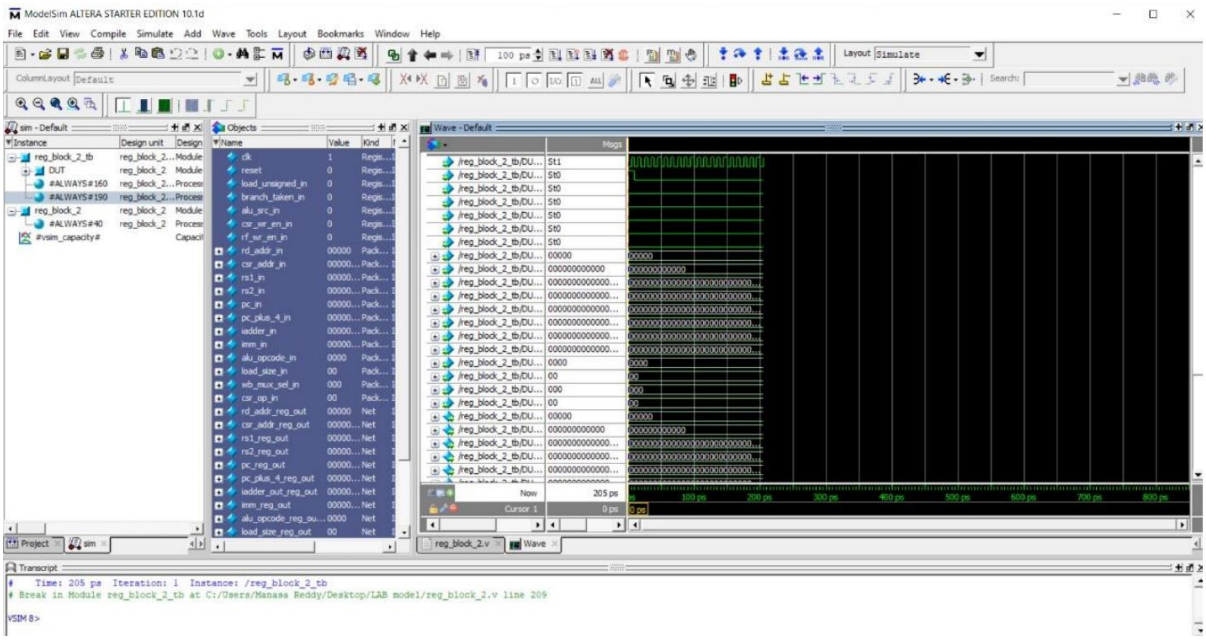
BRANCH UNIT:



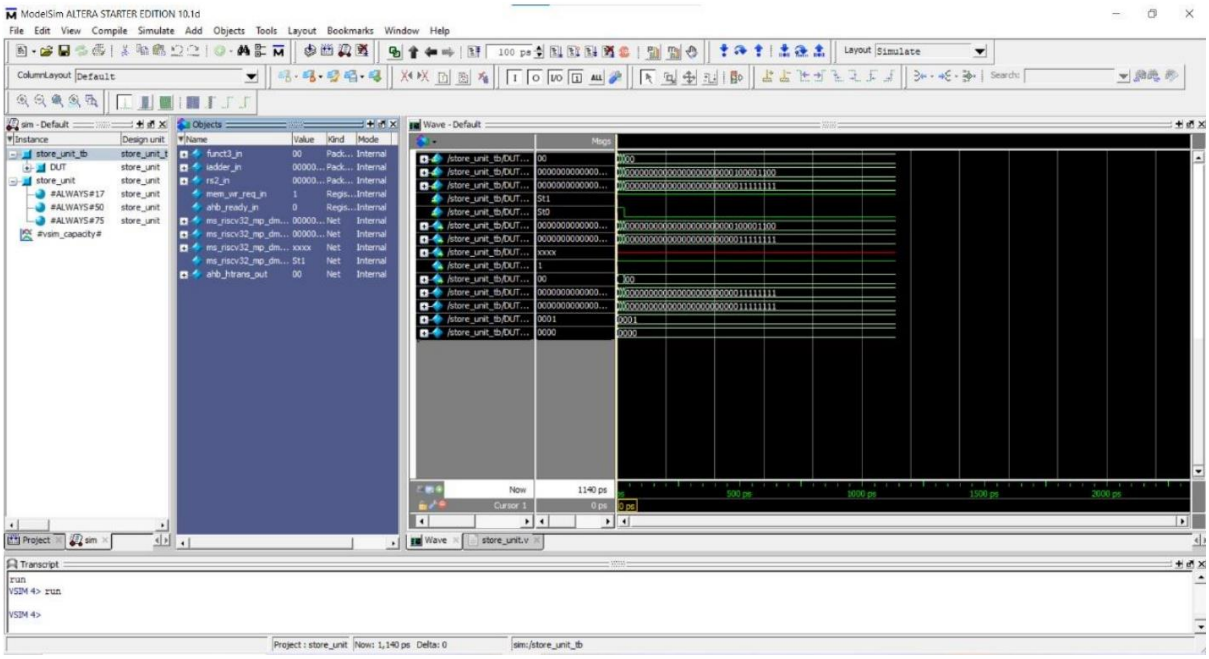
DECODER:



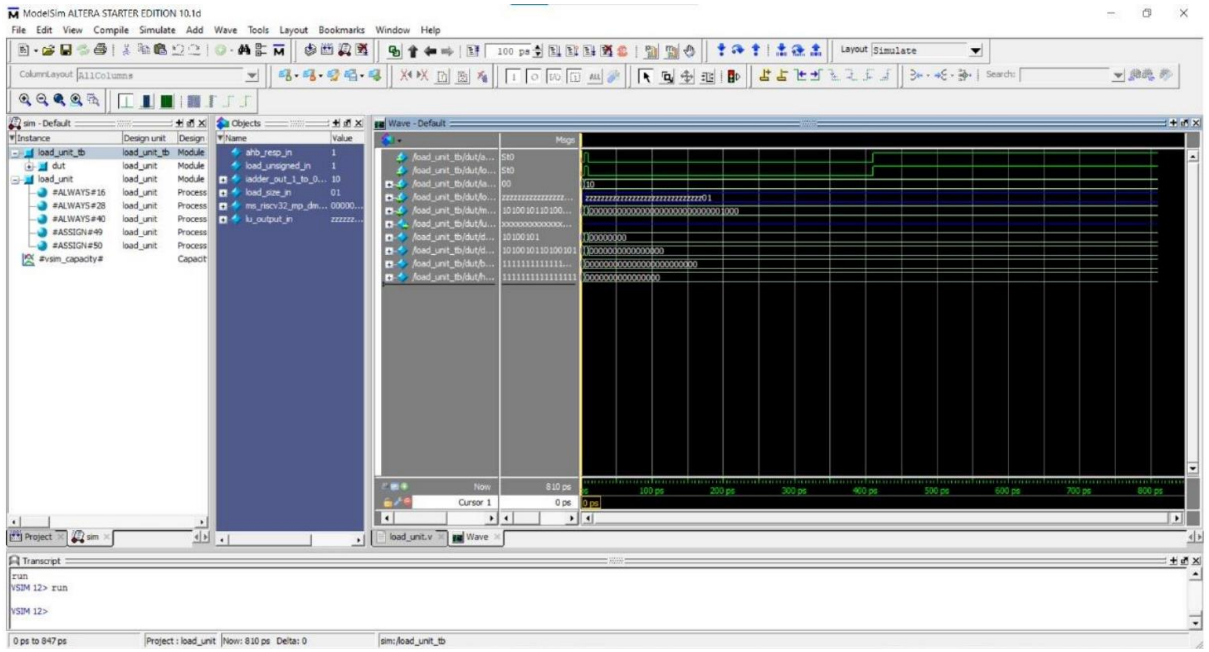
REG BLOCK-2:



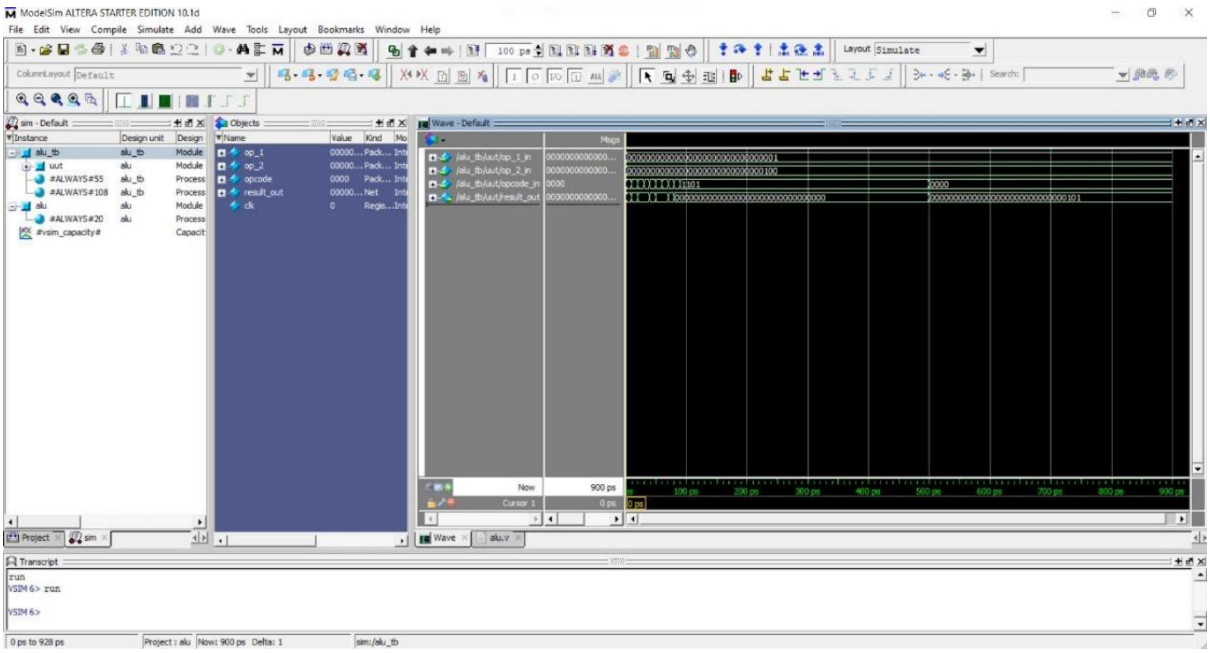
STORE UNIT:



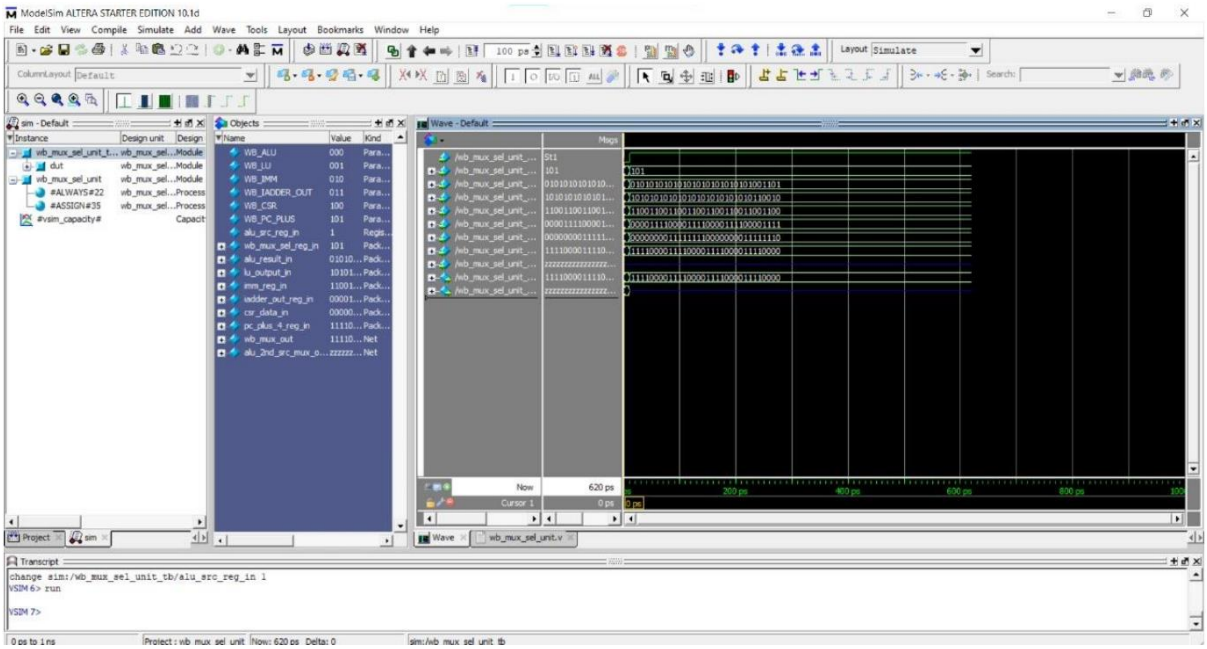
LOAD UNIT:



ALU:



WB MUX SELECTION UNIT:



THANK YOU