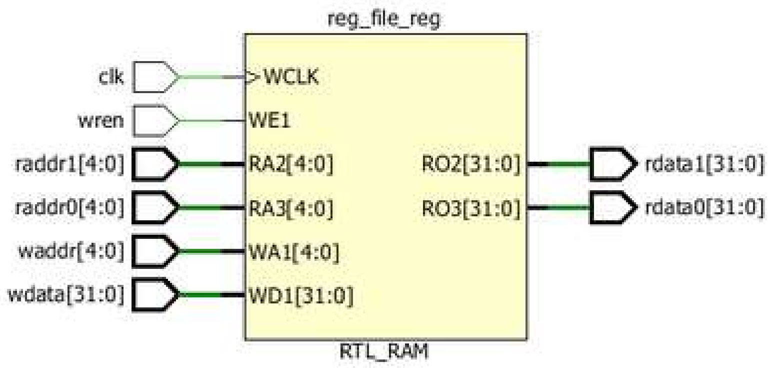
CSN-221 PROJECT:

DESIGN METHODOLOGY FOLLOWED:

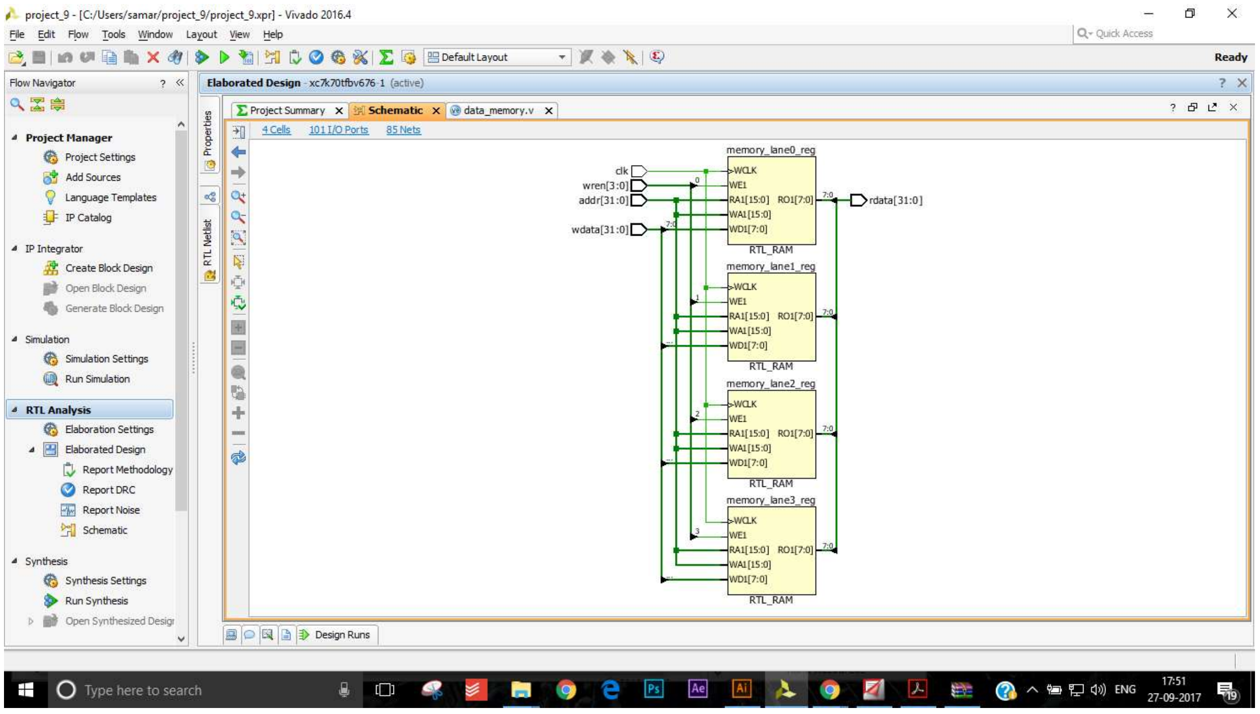
This is a 32 bit RISC processor which has been implemented through verilog. The ISA that is being used to facilitate the implementation of the processor is MIPS (Microprocessor without Interlocked Pipeline Stages). The inspiration behind using the MIPS methodology is due to the advantage it offers with regards to it's fixed instruction length. This might put a limit on the number of basic instructions that the processor can implement however it makes the processor much faster. Dealing with these subtle tradeoffs is paramount to the success of this project.

The following document details the thought process behind approaching the various parts of the processor.

REGISTER FILE: In the Register File Module we have 6 inputs and 2 outputs. The inputs correspond to clock, write if enable, two 5-bit addresses which are to be read and one 5-bit address which is to be written. The input also consists of one 32-bit line from which data which is to be written comes. The module also consists of two 32-bit output lines on which data is Written. When wren is set to 1, the data is written into the register file at a location whose address is given by waddr.



DATA MEMORY :In the data memory module we have 4 inputs, namely clock, address, write data, write enable and one output - read data.32 bit address and data are fed into our unit.The number of memory units is 4, each of which has 65536 8 bit locations to store data.The write enable signal is fed into each of the 4 units, the value of which if set to one enables us to write data into our memory.The address of the location in memory where the data is written is given by a 16-bit number which is given by WA1[15:0].For the time when data has to be read from the memory, every unit supplies us 8 bits, adding up to the desired 32 bit number.The address of the location from where the number is to be read is specified by RA1[15:0].



INSTRUCTION MEMORY: Here in we only have one input which is the 32 bit address line which carries the address of the location in the instruction_memory from where we have to fetch our 32 bit instruction.The readmemh function reads hexadecimal numbers from .mips files and maps them into the instruction_memory.The instruction at the output is assigned a value which is present at one of the 256 available storage locations in the instruction memory.

CONTROL UNIT :

The control unit functions as the “Brain” of the processor where it interprets the instruction that is being provided to the processor and it takes the necessary action. It tells the computer's memory, arithmetic/logic unit and input and output devices on how to respond to a program's instructions.
In this design the 32 bit instruction is read and the necessary action is taken. The first step to be taken is that the instructions need to be demarcated. Depending on the format of the instructions (R,J,I) the instructions are demarcated and the opcode is obtained. Now the CU at this point in time in the project gives the following output lines

1) ALU MUX SELECT
2) ALU CONTROL
3) DATA WRITE ENABLE

Depending on the various bits that might be transferred by the lines of the control unit, the CU takes the necessary action and activates the ALU for logical operations or data memory for writing in the data.

ALU: A 4-bit control line and two 32-bit registers for two operands have been used as inputs to the ALU module. 11 operations have been defined namely, and, or, add, xor, nor, subtract, set on less than, shift left logical, shift right logical, signed add, signed subtract. The operation is selected based on the opcode input in control line using switch-case. We have also taken care of overflow conditions by using an overflow (underflow) flag. A zero flag has also been used to determine whether the result is zero or not.

OBJECTIVES OF THE PROJECT:

The project is still in its nascent stages with only the implementation of the the basic parts of the processor completed. We wish to implement a much more nuanced version of the processor. The following objectives are the ones that we wish, standout as the hallmarks of the project.

1) IMPLEMENTING THE PROCESSOR WITH A 5 STAGE PIPELINE:

Pipelining is a crucial factor that supports the operation of the processor and makes the operation of the processor much more faster by increasing the throughput of the processor. We wish to make a classic MIPS Processor supplemented by a 5 stage pipelined process. This design process requires some serious research in this direction in order to deal with the various problems that arise with the implementation of a pipelined processor namely the data hazards and control hazards. These need to be eradicated by the implementation of a forward stalling mechanism. This is one of the objectives that we wish to accomplish by the end of the project.

2) USE OF A DIFFERENT ISA OTHER THAN MIPS:

The objective with which we have approached this project is that we wish this project to serve as a litmus test for the superiority of MIPS ISA over other ISA that are prevalent. However MIPS has a relative disadvantage with regards to it’s fixed instruction length. So we wish to eradicate this relative disadvantage keeping in with the spirit with which the design of the MIPS ISA was approached.

These are the future objectives of this project.