CSE331 ASSIGNMENT 4 REPORT

Ahmet Ergani

161044011

Summarization:

I have implemented a new controller unit, slightly modified the register unit, added data memory, instruction memort and nextPC unit and implemented a whole new datapath using this new modules and the previous modules

Explanation for modules:

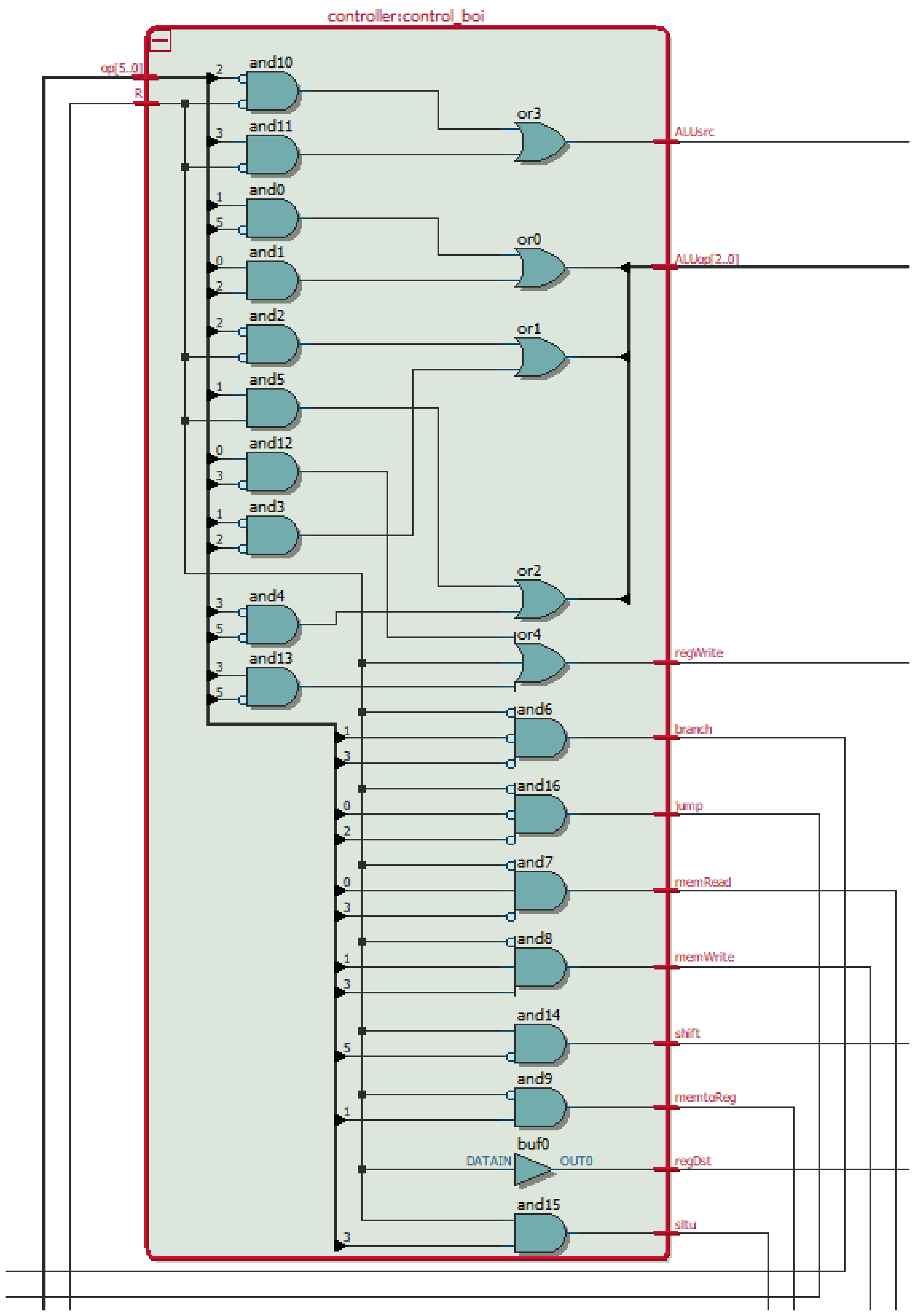
Controller unit:

It has 2 input (opcode/funct code, R) and 13 outputs

R is 1 if instruction is R-type, 0 otherwise

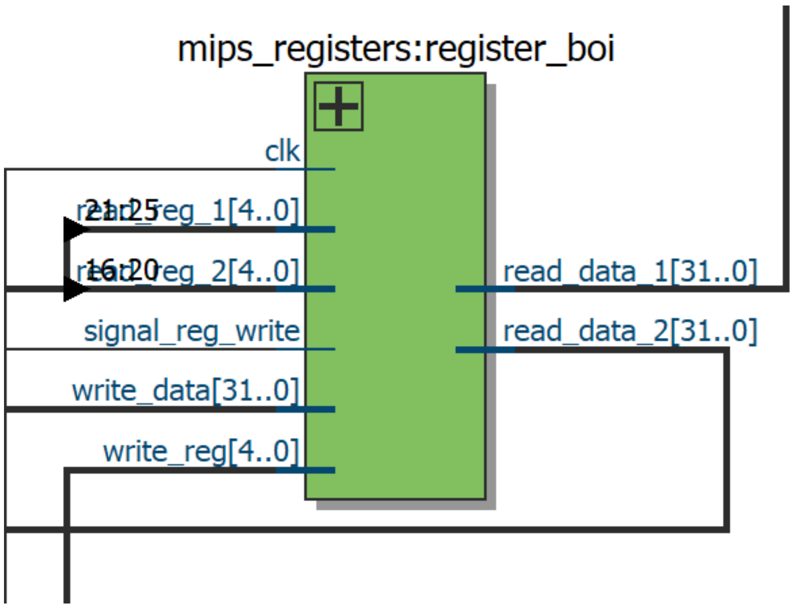
Calculated results of the outputs are:

ALUop[0]	:	op[5]'.op[1] + op[2].op[0]
ALUop[1]	:	R'.op[2] + op[2]'.op[1]'
ALUop[2]	:	op[5]'.op[3]' + R.op[1]
Branch	:	R'.op[3]'.op[1]'
memRead	:	R'.op[3]'.op[0]
memWrite	:	R'.op[3].op[1]
memtoReg	:	R'.op[1]
ALUsrc	:	R'.op[3] + R'.op[2]'
regWrite	:	R + op[3]'.op[0] + op[5]'.op[3]
regDst	:	R
shift	:	R.op[5]'
sltu	:	R.op[3]
jump	:	R'.op[2]'.op[0]'


controller:control_boi
op[5..0]
R
and10
and11
and0
and1
and2
and5
and12
and3
and4
and13
or3
or0
or1
or2
or4
and6
and16
and7
and8
and14
and9
buf0
DATAIN
OUT0
and15
ALUsrc
ALUop[2..0]
regWrite
branch
jump
memRead
memWrite
shift
memtoReg
regDst
sltu

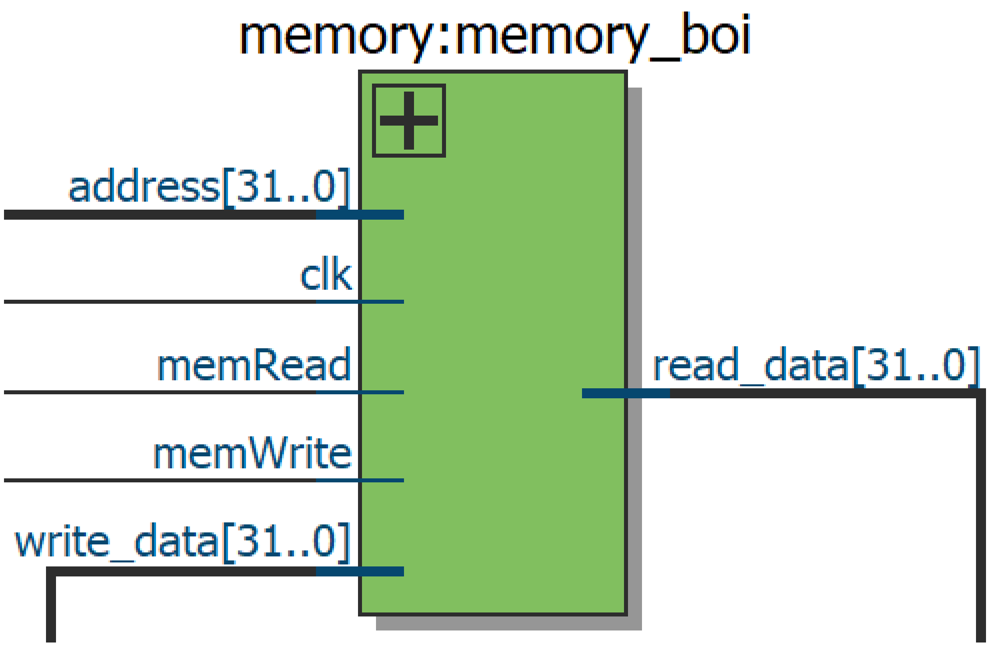
Register unit:

Only different part from the previous project is writing to file does not happen every negedge but happens when either read_reg is changed



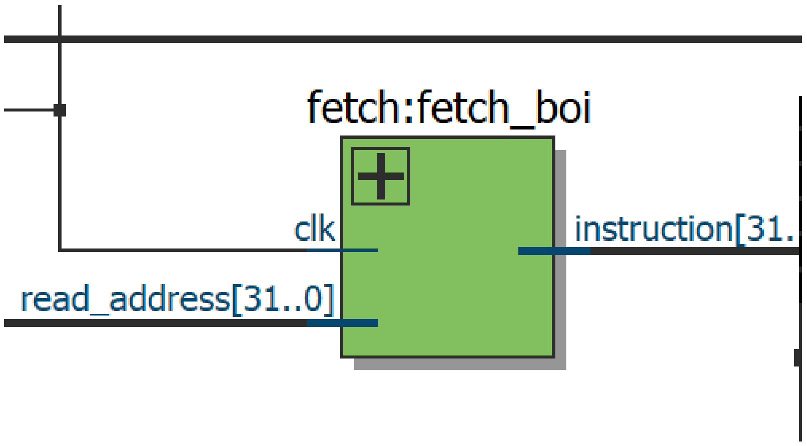
Data Memory unit:

Writes to file at posedge if the mem_write signal is 1. Reads from file when the address has changed and mem_read signal is 1



Instruction Memory unit:

Reads the necessary instruction from the file when the address is changed



nextPC unit:

There are 4 cases:

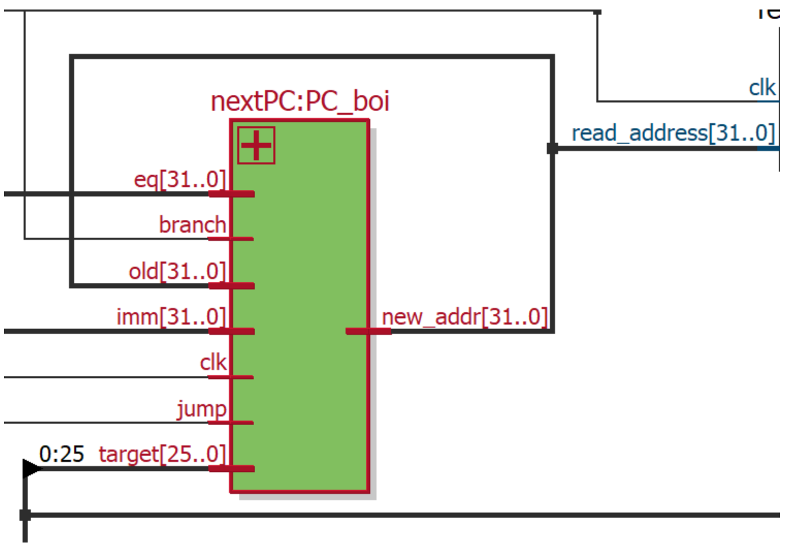
Case1: Initial state. Old address is xxxxxxxx…xxxxxxx. New addres should be 0

Case2: Normal. New address should be old address + 1

Case3: Branch: If eq is 0, new address should be old address + immediate + 1

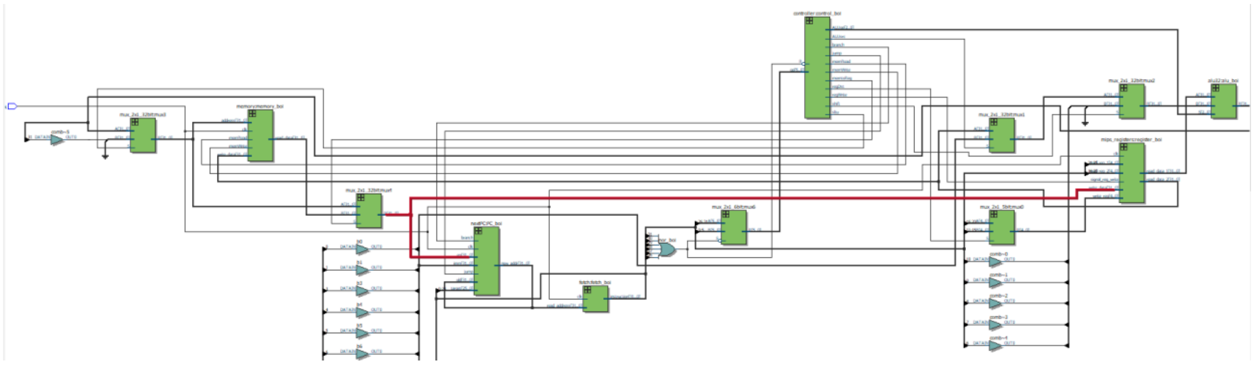
If eq is not 0, Case2

Case4: Jump. New address should be old address + target + 1

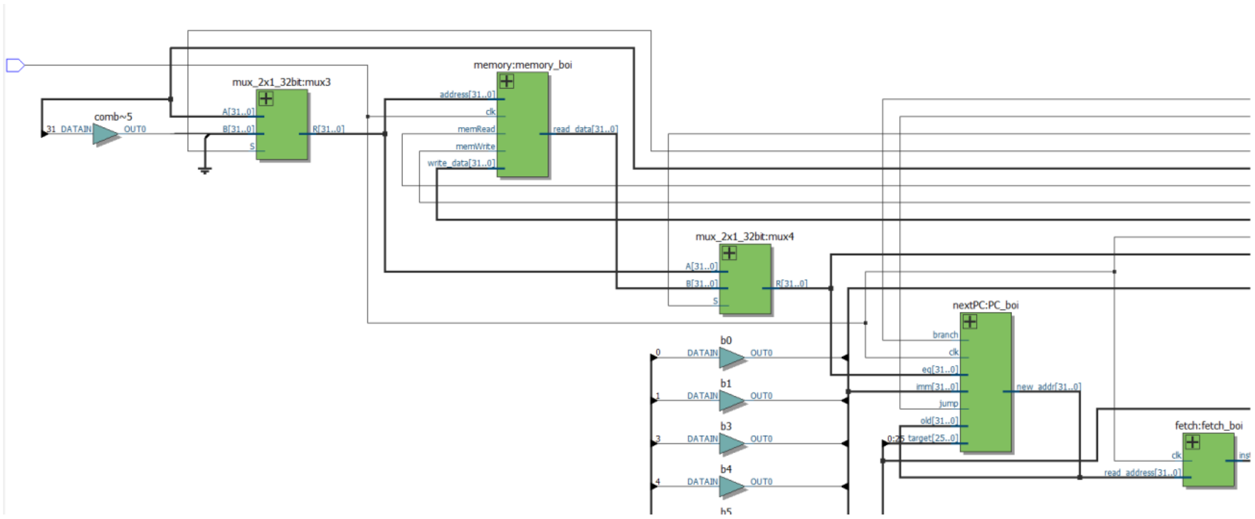


Resulting Datapath

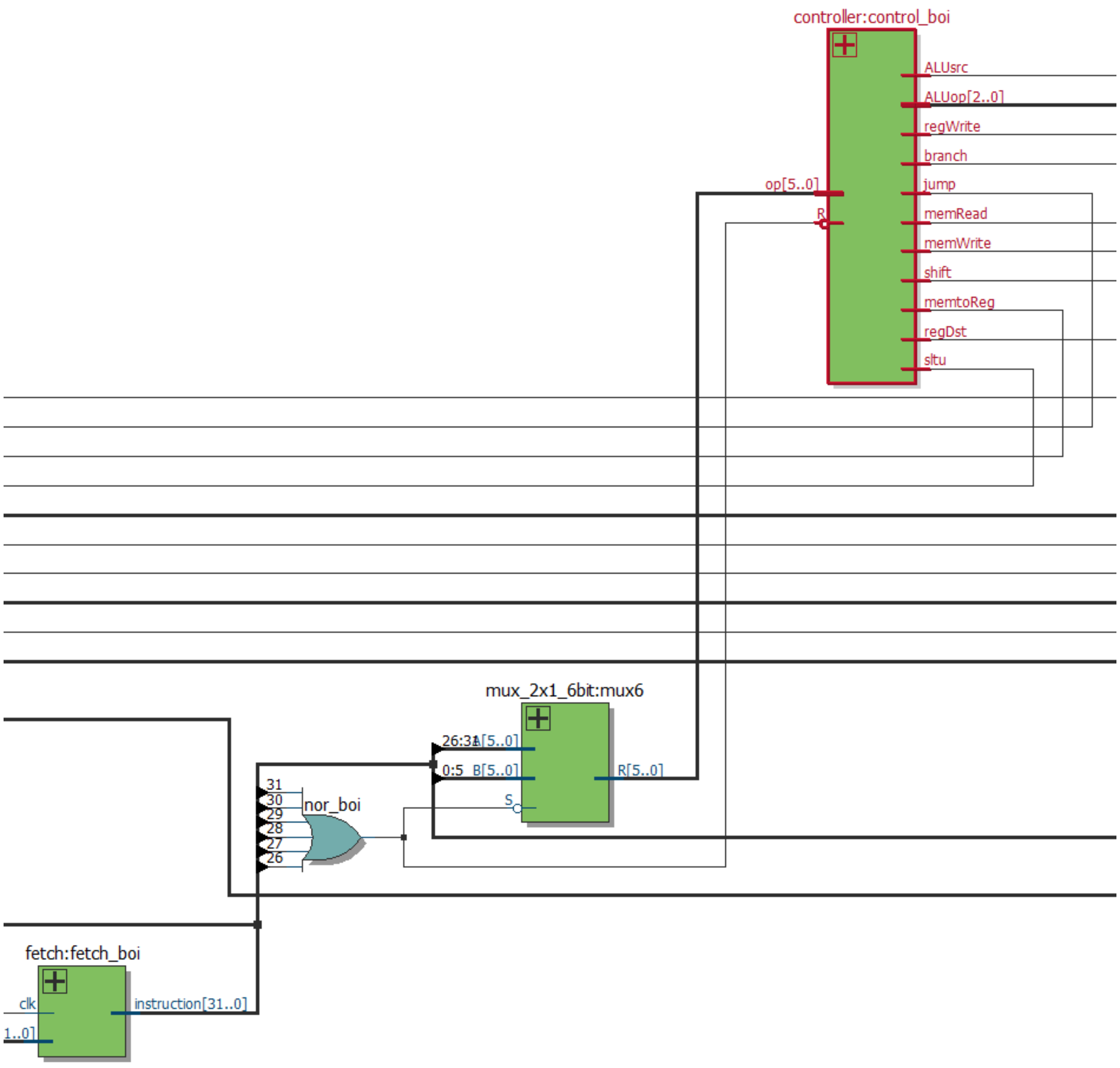
GENERAL VIEW



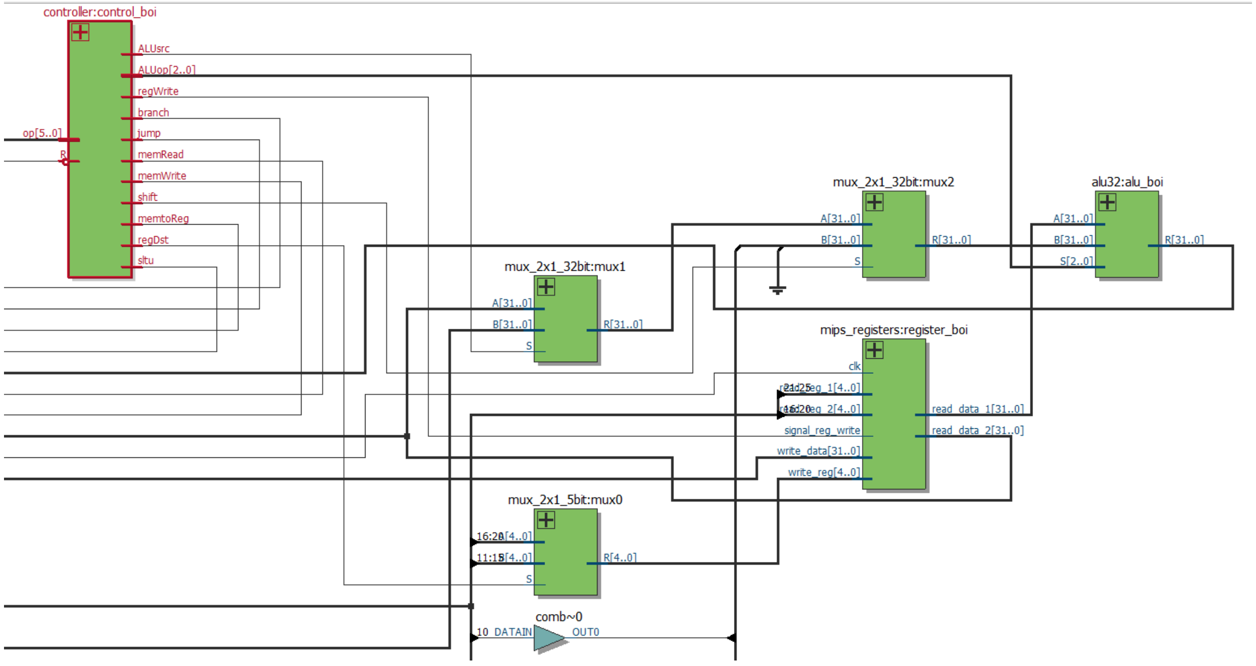
LEFT PART



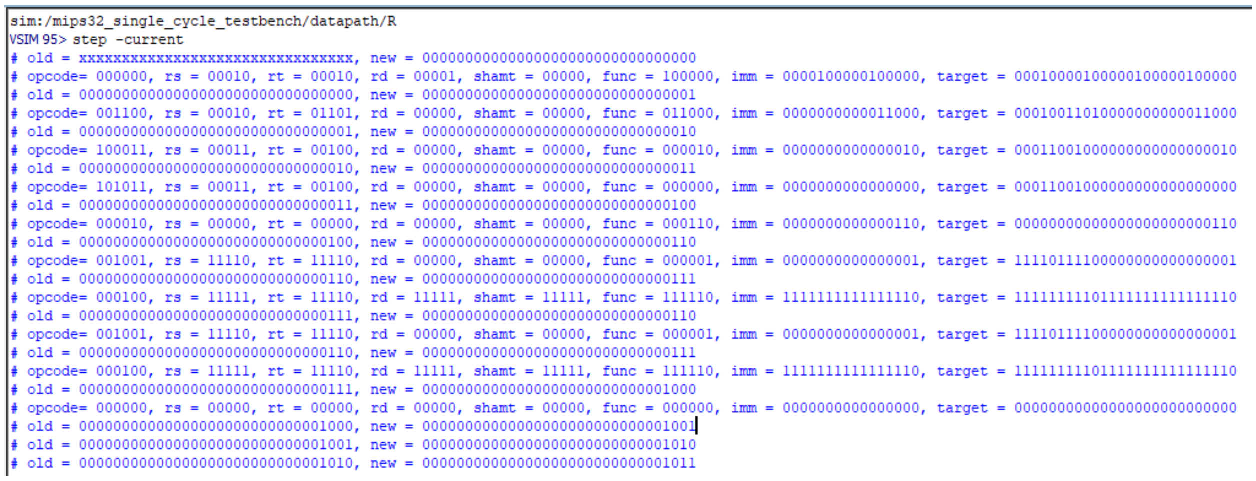
MID PART


controller:control_boi
ALUsrc
ALUop[2..0]
regWrite
branch
op[5..0]
jump
R
memRead
memWrite
shift
memtoReg
regDst
sltu
mux_2x1_6bit:mux6
26:31
A[5..0]
0:5
B[5..0]
R[5..0]
S
31
30
29
28
27
26
nor_boi
fetch:fetch_boi
clk
1..0]
instruction[31..0]

RIGHT PART

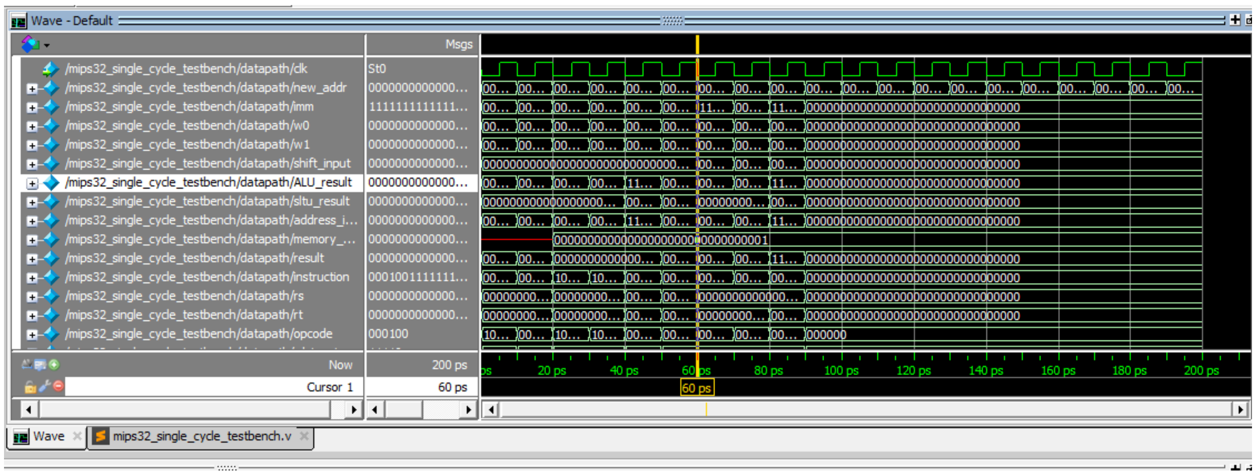


Testing

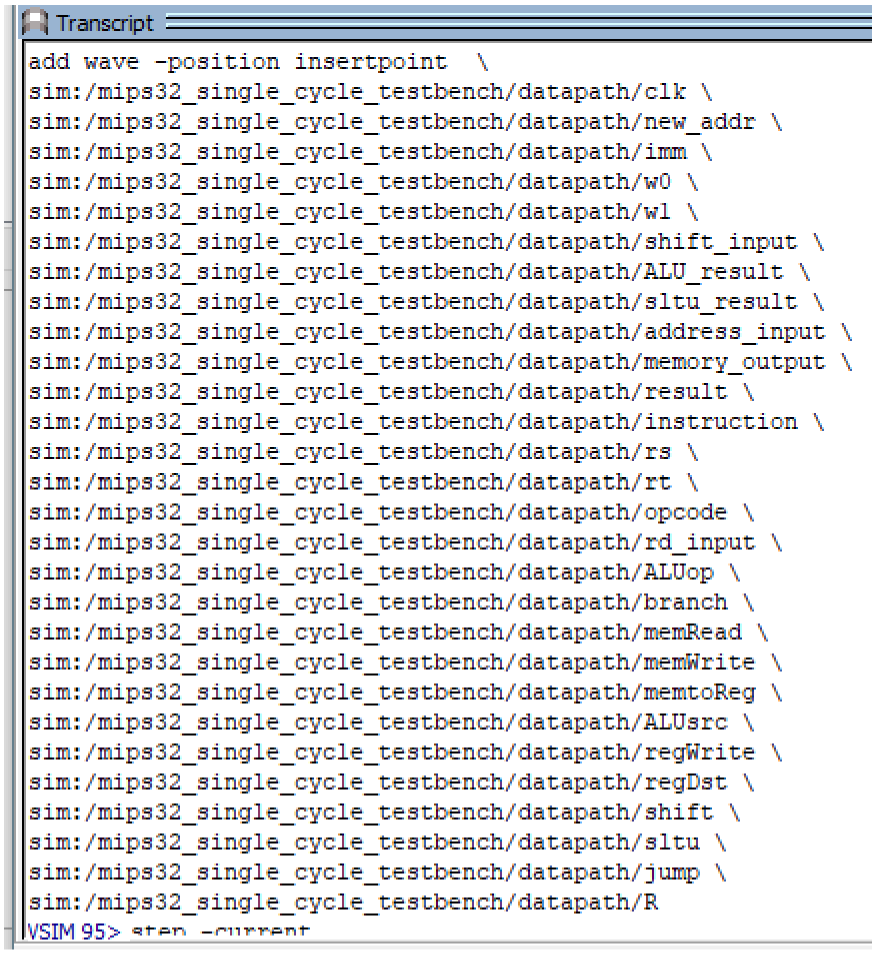


```
sim:/mips32_single_cycle_testbench/datapath/R
VSIM 95> step -current
# old = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, new = 00000000000000000000000000000000
# opcode= 000000, rs = 00010, rt = 00010, rd = 00001, shamt = 00000, func = 100000, imm = 0000100000100000, target = 00010000100000100000100000
# old = 00000000000000000000000000000000, new = 00000000000000000000000000000001
# opcode= 001100, rs = 00010, rt = 01101, rd = 00000, shamt = 00000, func = 011000, imm = 0000000000011000, target = 00010011010000000000011000
# old = 00000000000000000000000000000001, new = 00000000000000000000000000000010
# opcode= 100011, rs = 00011, rt = 00100, rd = 00000, shamt = 00000, func = 000010, imm = 0000000000000010, target = 00011001000000000000000010
# old = 00000000000000000000000000000010, new = 00000000000000000000000000000011
# opcode= 101011, rs = 00011, rt = 00100, rd = 00000, shamt = 00000, func = 000000, imm = 0000000000000000, target = 00011001000000000000000000
# old = 00000000000000000000000000000011, new = 00000000000000000000000000000100
# opcode= 000010, rs = 00000, rt = 00000, rd = 00000, shamt = 00000, func = 000110, imm = 0000000000000110, target = 00000000000000000000000110
# old = 00000000000000000000000000000100, new = 00000000000000000000000000000110
# opcode= 001001, rs = 11110, rt = 11110, rd = 00000, shamt = 00000, func = 000001, imm = 0000000000000001, target = 11110111100000000000000001
# old = 00000000000000000000000000000110, new = 00000000000000000000000000000111
# opcode= 000100, rs = 11111, rt = 11110, rd = 11111, shamt = 11111, func = 111110, imm = 1111111111111110, target = 11111111101111111111111110
# old = 00000000000000000000000000000111, new = 00000000000000000000000000000110
# opcode= 001001, rs = 11110, rt = 11110, rd = 00000, shamt = 00000, func = 000001, imm = 0000000000000001, target = 11110111100000000000000001
# old = 00000000000000000000000000000110, new = 00000000000000000000000000000111
# opcode= 000100, rs = 11111, rt = 11110, rd = 11111, shamt = 11111, func = 111110, imm = 1111111111111110, target = 11111111101111111111111110
# old = 00000000000000000000000000000111, new = 00000000000000000000000000001000
# opcode= 000000, rs = 00000, rt = 00000, rd = 00000, shamt = 00000, func = 000000, imm = 0000000000000000, target = 00000000000000000000000000
# old = 00000000000000000000000000001000, new = 00000000000000000000000000001001
# old = 00000000000000000000000000001001, new = 00000000000000000000000000001010
# old = 00000000000000000000000000001010, new = 00000000000000000000000000001011
```

ScreenShots



```
Wave - Default
                                                        Msgs
/mips32_single_cycle_testbench/datapath/clk             St0
/mips32_single_cycle_testbench/datapath/new_addr        0000000000000...
/mips32_single_cycle_testbench/datapath/imm             1111111111111...
/mips32_single_cycle_testbench/datapath/w0              0000000000000...
/mips32_single_cycle_testbench/datapath/w1              0000000000000...
/mips32_single_cycle_testbench/datapath/shift_input     0000000000000...
/mips32_single_cycle_testbench/datapath/ALU_result      0000000000000...
/mips32_single_cycle_testbench/datapath/sltu_result     0000000000000...
/mips32_single_cycle_testbench/datapath/address_i...    0000000000000...
/mips32_single_cycle_testbench/datapath/memory_...      0000000000000...
/mips32_single_cycle_testbench/datapath/result          0000000000000...
/mips32_single_cycle_testbench/datapath/instruction     0001001111111...
/mips32_single_cycle_testbench/datapath/rs              0000000000000...
/mips32_single_cycle_testbench/datapath/rt              0000000000000...
/mips32_single_cycle_testbench/datapath/opcode          000100
                                                  Now   200 ps
                                             Cursor 1   60 ps
```



Transcript

```
add wave -position insertpoint  \
sim:/mips32_single_cycle_testbench/datapath/clk \
sim:/mips32_single_cycle_testbench/datapath/new_addr \
sim:/mips32_single_cycle_testbench/datapath/imm \
sim:/mips32_single_cycle_testbench/datapath/w0 \
sim:/mips32_single_cycle_testbench/datapath/w1 \
sim:/mips32_single_cycle_testbench/datapath/shift_input \
sim:/mips32_single_cycle_testbench/datapath/ALU_result \
sim:/mips32_single_cycle_testbench/datapath/sltu_result \
sim:/mips32_single_cycle_testbench/datapath/address_input \
sim:/mips32_single_cycle_testbench/datapath/memory_output \
sim:/mips32_single_cycle_testbench/datapath/result \
sim:/mips32_single_cycle_testbench/datapath/instruction \
sim:/mips32_single_cycle_testbench/datapath/rs \
sim:/mips32_single_cycle_testbench/datapath/rt \
sim:/mips32_single_cycle_testbench/datapath/opcode \
sim:/mips32_single_cycle_testbench/datapath/rd_input \
sim:/mips32_single_cycle_testbench/datapath/ALUop \
sim:/mips32_single_cycle_testbench/datapath/branch \
sim:/mips32_single_cycle_testbench/datapath/memRead \
sim:/mips32_single_cycle_testbench/datapath/memWrite \
sim:/mips32_single_cycle_testbench/datapath/memtoReg \
sim:/mips32_single_cycle_testbench/datapath/ALUsrc \
sim:/mips32_single_cycle_testbench/datapath/regWrite \
sim:/mips32_single_cycle_testbench/datapath/regDst \
sim:/mips32_single_cycle_testbench/datapath/shift \
sim:/mips32_single_cycle_testbench/datapath/sltu \
sim:/mips32_single_cycle_testbench/datapath/jump \
sim:/mips32_single_cycle_testbench/datapath/R
VSIM 95> step -current
```