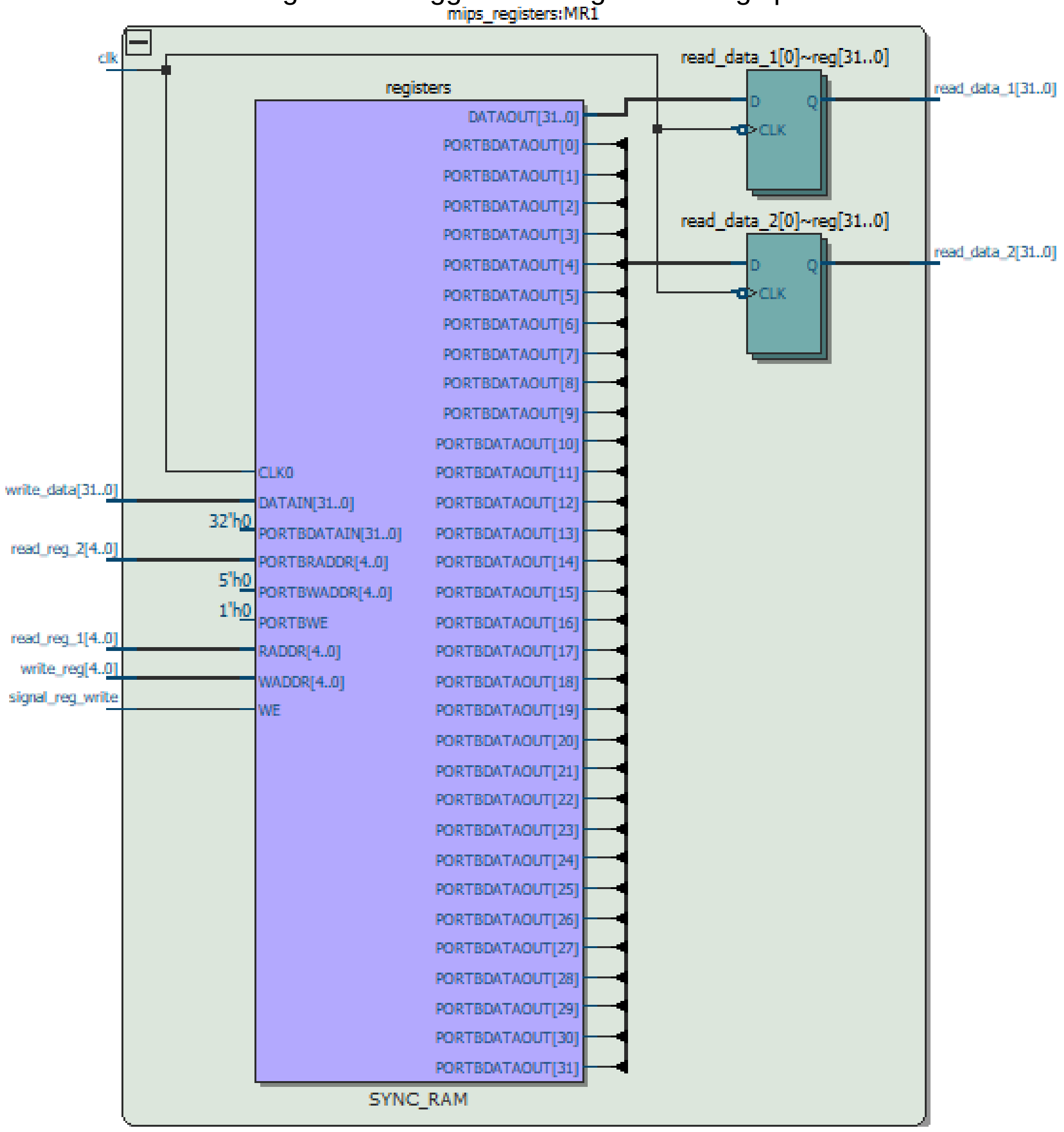
# COMPUTER ORGANİZATON
# MIPS_32 DOCUMENTATION

The descriptions of modules I use and their implementation with schematics.There is schematic designs on module descriptions under.Also I changed the shifter module with logical model.

## MODULES

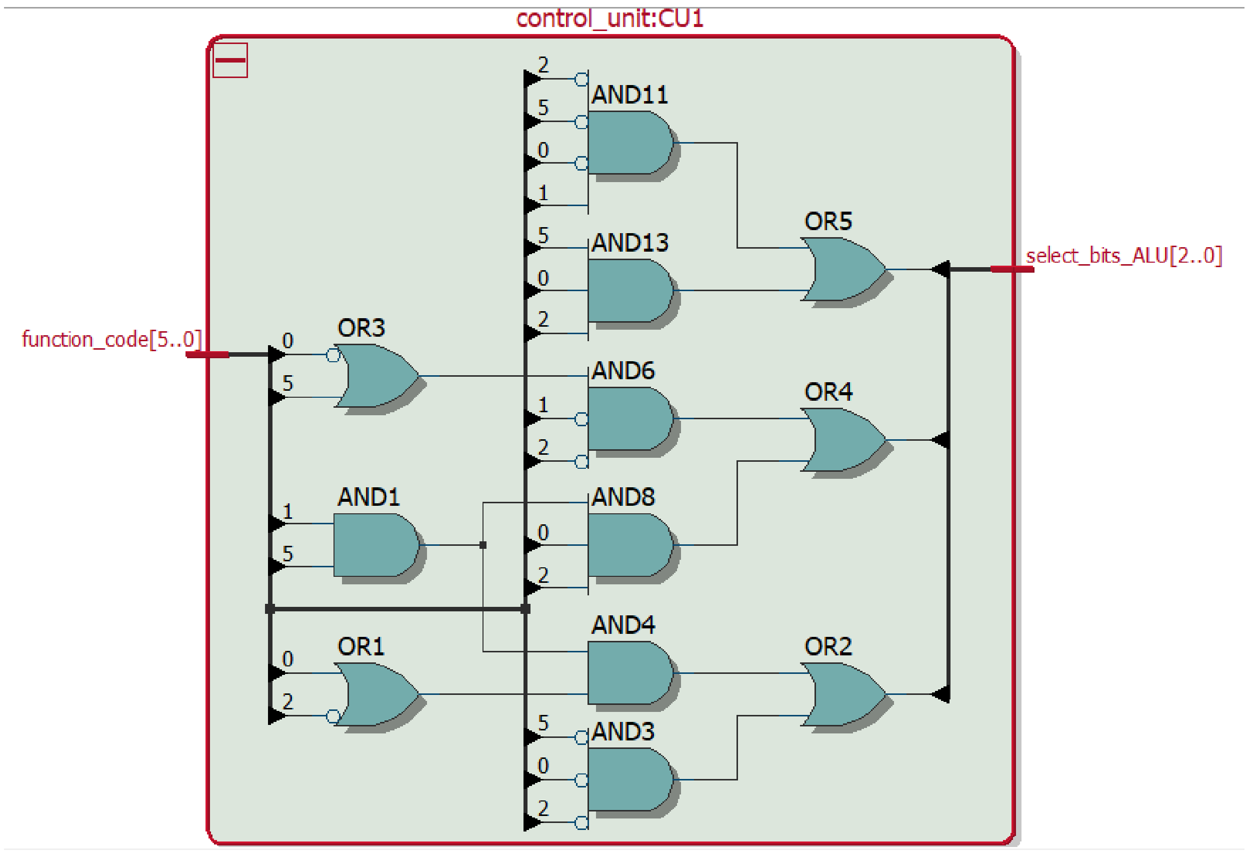
**mips_registers**: Make the inputs of alu32($rs , $rt) module ready for mips32 to use.And after perfoming writes the result($rd) in register.There is a write enable signal and by this you can choose between writing to a register or not.But this module fills the $rs and $rt register.There is not a signal to enable/disable reading.The clk triggers reading and writing operations.



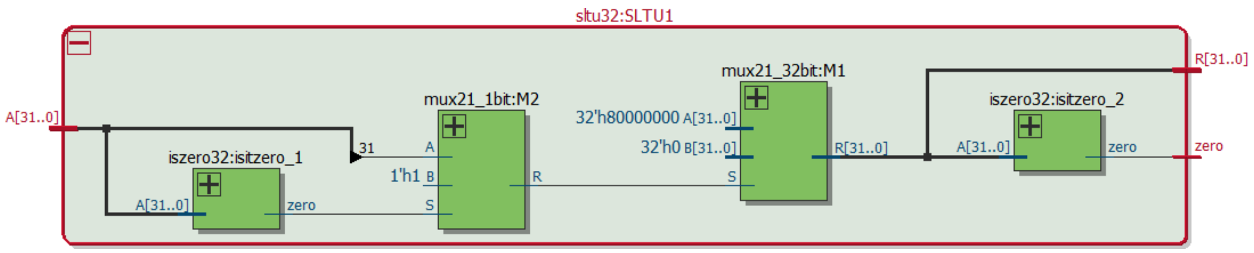
**control_unit**: Abstractly it takes one 6 bit input and output one 3 bit :zero bit (1 if the result is equal to 0) and 32 bit result. 6 bit input is the function code of instruction(instruction[5:0]). According to these logical expressions;

- s2 = f1f5(f0+f2')+f0'f2'f5'
- s1 = f1'f2'(f0'+f5)+f5f2f1f0
- s0 = f5'f2'f1f0'+f5f2f0

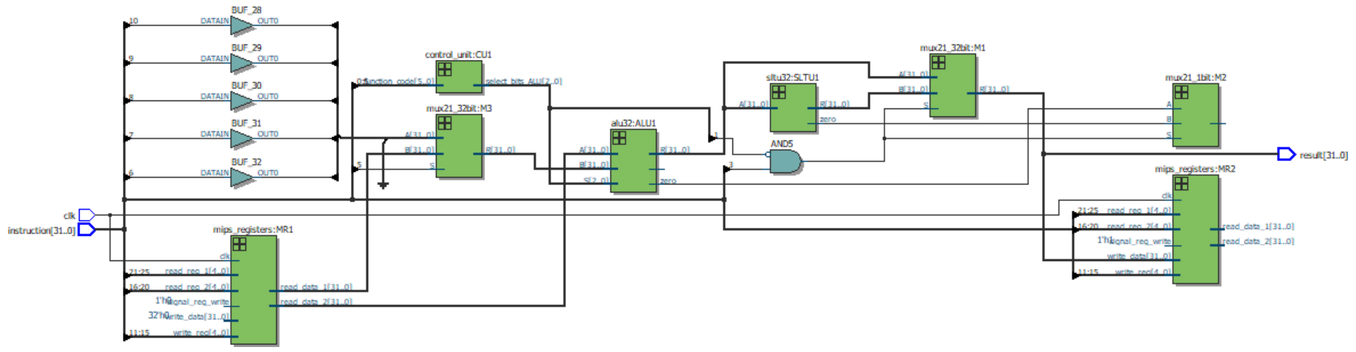
Calculates the selection bits of alu32 module.



**sltu32**: This modules triggers alu32 to perform sub32 module. Then takes it result and for first check if the result came from alu is 0 or not. If zero then the output of module is 0 cause it means that the $rs and $rt registers contents holds the same 32 bit data. If not zero checks the most significant(sign bit) to decide if $rt's content is higher than $rs's content or not. Abstractly it takes one 32 bit input and output two: zero bit (1 if the result is equal to 0) and 32 bit result.



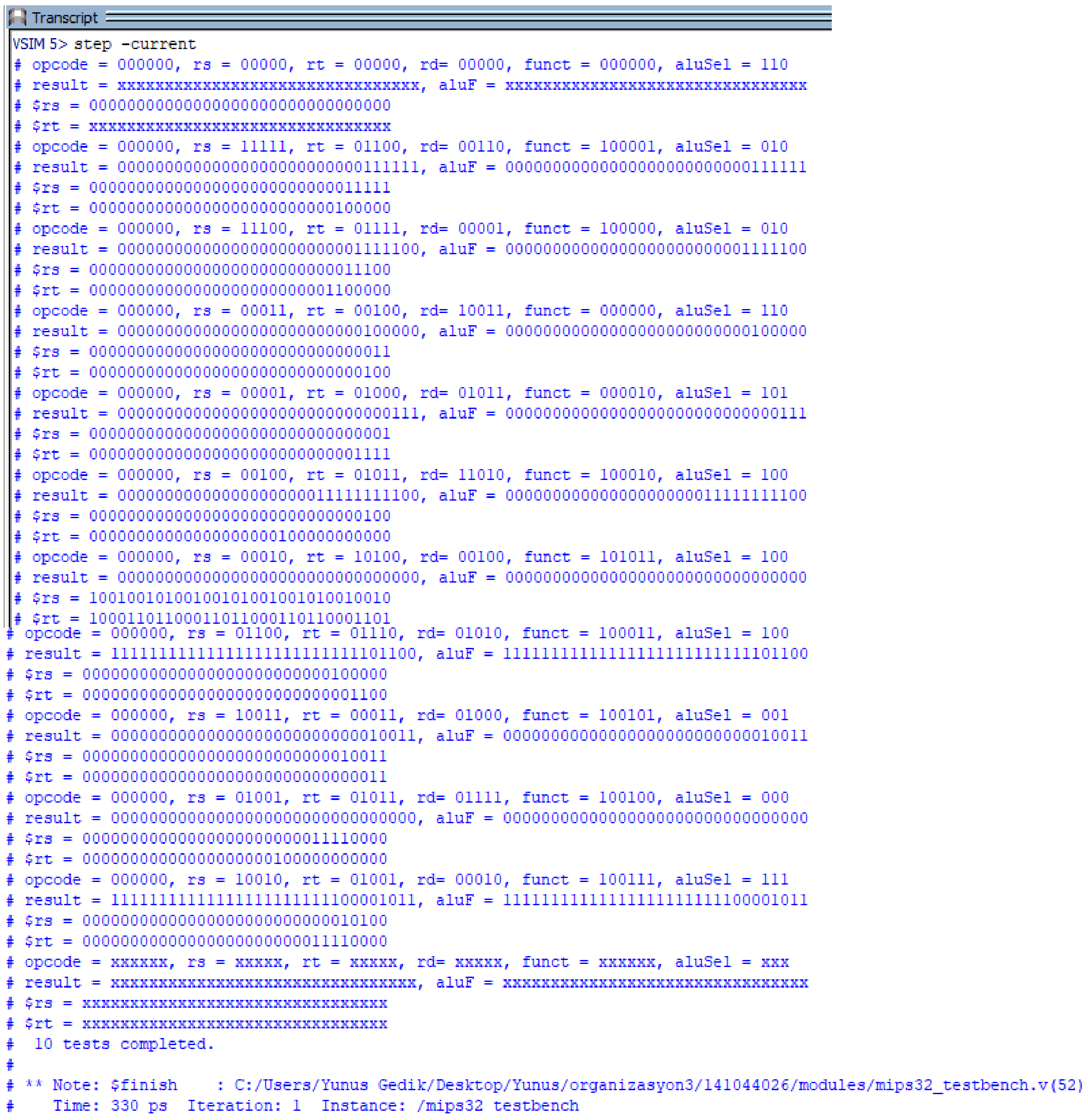
**mips32**:Takesthe32bitinstructionthenbyusingcontrolunitachievetheselectionbitsofalu32 to use.Then via mips_registers module gets the contents of required registers.After that extend the shift amount(shamt) to 32 bit.Checks if the instruction is a shift insturction or not.If it is makes the $rs registers content shift amount, if not ontuinues with $rs's content that mips_registers returned.Runs the alu32 module with contents.Calcultes overflow bit.(The template in moodle do not have a output for zero and overflow bits so I am calculating them but do not use anywhere.)For last checks if the instruction that module takes is sltu or not.If it is inputs the alu32s output for sltu32 module.If not continue with alu32 result and writes the result in register via mips_register module.



**mips32_testbench**: This module is the testbench of our project.Tests mips32 module with different 10 inputs.Nearly same as the testbench that Mr. Bayrakci show us in course.Takes inputs from file and display the result in modelsim and also save the results in a file.Module takes 10 inputs but prints 12 results in modelsim that I could not understand why,the first and the last prints of module is junk.So you can ignore them.

## MODELSIM RESULTS

I did not display the shift amount here cause the shift amount is saved in $rs register in $shift instructions.



```
VSIM 5> step -current
# opcode = 000000, rs = 00000, rt = 00000, rd= 00000, funct = 000000, aluSel = 110
# result = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, aluF = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# $rs = 00000000000000000000000000000000
# $rt = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# opcode = 000000, rs = 11111, rt = 01100, rd= 00110, funct = 100001, aluSel = 010
# result = 00000000000000000000000000111111, aluF = 00000000000000000000000000111111
# $rs = 00000000000000000000000000011111
# $rt = 00000000000000000000000000100000
# opcode = 000000, rs = 11100, rt = 01111, rd= 00001, funct = 100000, aluSel = 010
# result = 00000000000000000000000001111100, aluF = 00000000000000000000000001111100
# $rs = 00000000000000000000000000011100
# $rt = 00000000000000000000000001100000
# opcode = 000000, rs = 00011, rt = 00100, rd= 10011, funct = 000000, aluSel = 110
# result = 00000000000000000000000000100000, aluF = 00000000000000000000000000100000
# $rs = 00000000000000000000000000000011
# $rt = 00000000000000000000000000000100
# opcode = 000000, rs = 00001, rt = 01000, rd= 01011, funct = 000010, aluSel = 101
# result = 00000000000000000000000000000111, aluF = 00000000000000000000000000000111
# $rs = 00000000000000000000000000000001
# $rt = 00000000000000000000000000001111
# opcode = 000000, rs = 00100, rt = 01011, rd= 11010, funct = 100010, aluSel = 100
# result = 00000000000000000000001111111100, aluF = 00000000000000000000001111111100
# $rs = 00000000000000000000000000000100
# $rt = 00000000000000000000100000000000
# opcode = 000000, rs = 00010, rt = 10100, rd= 00100, funct = 101011, aluSel = 100
# result = 00000000000000000000000000000000, aluF = 00000000000000000000000000000000
# $rs = 10010010100100101001001010010010
# $rt = 10001101100011011000110110001101
# opcode = 000000, rs = 01100, rt = 01110, rd= 01010, funct = 100011, aluSel = 100
# result = 11111111111111111111111111101100, aluF = 11111111111111111111111111101100
# $rs = 00000000000000000000000000100000
# $rt = 00000000000000000000000000001100
# opcode = 000000, rs = 10011, rt = 00011, rd= 01000, funct = 100101, aluSel = 001
# result = 00000000000000000000000000010011, aluF = 00000000000000000000000000010011
# $rs = 00000000000000000000000000010011
# $rt = 00000000000000000000000000000011
# opcode = 000000, rs = 01001, rt = 01011, rd= 01111, funct = 100100, aluSel = 000
# result = 00000000000000000000000000000000, aluF = 00000000000000000000000000000000
# $rs = 00000000000000000000000011110000
# $rt = 00000000000000000000100000000000
# opcode = 000000, rs = 10010, rt = 01001, rd= 00010, funct = 100111, aluSel = 111
# result = 11111111111111111111111100001011, aluF = 11111111111111111111111100001011
# $rs = 00000000000000000000000000010100
# $rt = 00000000000000000000000011110000
# opcode = xxxxxx, rs = xxxxx, rt = xxxxx, rd= xxxxx, funct = xxxxxx, aluSel = xxx
# result = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, aluF = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# $rs = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# $rt = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
#  10 tests completed.
#
# ** Note: $finish    : C:/Users/Yunus Gedik/Desktop/Yunus/organizasyon3/141044026/modules/mips32_testbench.v(52)
#    Time: 330 ps  Iteration: 1  Instance: /mips32 testbench
```

141044026
Yunus Gedik