CSE331 COMPUTER ORGANIZATION HW4 REPORT

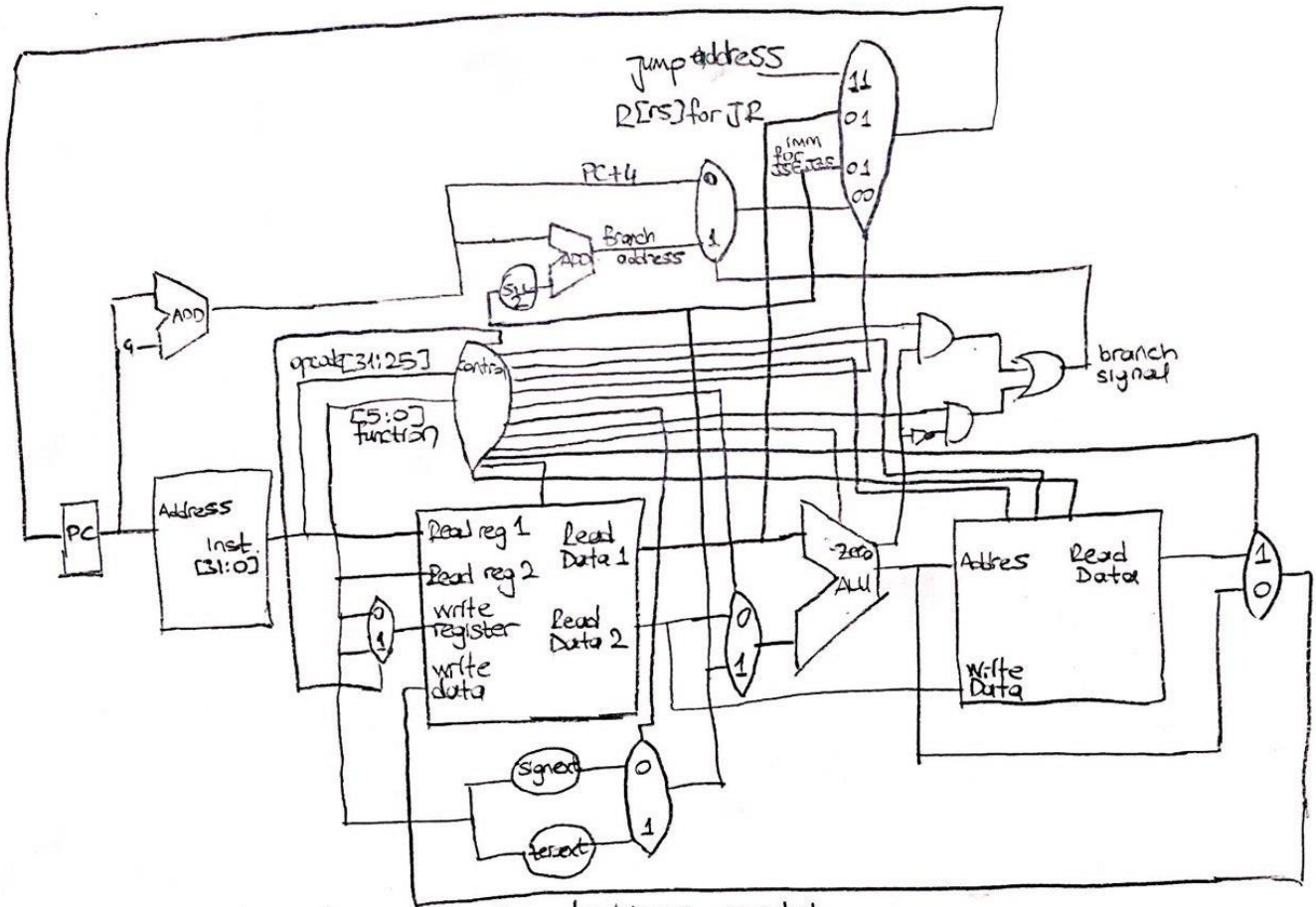
- I designed all my modules %100 correctly but I can't simulate my design because I can't figured it out how program counter does not work.

```
FUNCTION = 101010 SLT -> R -> R[rd] = 1 İF(R[rs] < R[rt]) ELSE 0
FUNCTION = 101001 SLTU -> R -> R[rd] = 1 İF(R[rs] < R[rt]) ELSE 0
FUNCTION = 100000 ADD -> R -> R[rd] = R[rs] + R[rt]
FUNCTION = 100010 SUB -> R -> R[rd] = R[rs] - R[rt]
FUNCTION = 100100 AND -> R -> R[rd] = R[rs] AND R[rt]
FUNCTION = 100101 OR -> R -> R[rd] = R[rs] OR R[rt]
FUNCTION = 100110 XOR -> R -> R[rd] = R[rs] XOR R[rt]
FUNCTION = 000011 SRA -> R -> R[rd] = R[rt] >> shamt
FUNCTION = 000010 SRL -> R -> R[rd] = R[rt] >> shamt
FUNCTION = 000000 SLL -> R -> R[rd] = R[rt] << shamt
FUNCTION = 001000 JR -> R -> PC = R[rs]

OPCODE = 000100 BEQ -> I -> İF(R[rs] == R[rt]) PC + 4 + {14{imm[15]},imm,2'b0}*4
OPCODE = 000101 BNE -> I -> İF(R[rs] != R[rt]) PC + 4 + {14{imm[15]},imm,2'b0}*4
OPCODE = 001000 ADDİ -> I -> R[rt] = R[rs] + signextendimm
OPCODE = 001001 ADDİU -> I -> R[rt] = R[rs] + signextendimm
OPCODE = 001010 SLTİ -> I -> R[rt] = 1 İF(R[rs] < signextendimm) ELSE 0
OPCODE = 001011 SLTİU -> I -> R[rd] = 1 İF(R[rs] < R[rt]) ELSE 0
OPCODE = 001100 ANDİ -> I -> R[rt] = R[rs] AND zeroextendimm
OPCODE = 001101 ORİ -> I -> R[rt] = R[rs] OR zeroextendimm
OPCODE = 001110 XORİ -> I -> R[rt] = R[rs] XOR zeroextendimm
OPCODE = 001111 LUİ -> I -> R[rt] = {imm, 16'b0}
OPCODE = 100000 LB -> I -> R[rt] = {24'b0, M[R[rs] + zeroextendimm](7:0)}
OPCODE = 100001 LH -> I -> R[rt] = {16'b0, M[R[rs] + zeroextendimm](15:0)}
OPCODE = 100011 LW -> I -> R[rt] = M[R[rs] + signextendimm]
OPCODE = 101000 SB -> I -> M[R[rs] + signextendimm](7:0) = R[rt](7:0)
OPCODE = 101001 SH -> I -> M[R[rs] + signextendimm](15:0) = R[rt](15:0)
OPCODE = 101011 SW -> I -> M[R[rs] + signextendimm] = R[rt]
OPCODE = 010000 JS -> I -> PC = signextendimm
OPCODE = 010001 JZ -> I -> PC = zeroextendimm

OPCODE = 000010 J -> J -> PC = {PC[31:28],address,2'b0}
OPCODE = 000011 JAL -> J -> R[31] = PC+4; PC = {PC[31:28],address,2'b0}
```

- I ADDED 2 INSTRUCTIONS WHICH ARE JS AND JZ.



control signals top to bottom: regdst
beq
memread
memwrite
Jumptypes
alusrc
zero_sign_ext
bne
aluctr
memtoreg
regwrite
types

FUNCTION	OPCODE	REGDST	BEQ	BNE	MEMREAD	MEMWRİTE	MEMTOREG	ALUCTR_FUNCTION	ALUCTR_OPCODE	ALUSRC	REGWRITE	TYPES	ZERO_SIGN_EXT	CHECK_R_TYPE	JUMP_TYPES
101010	,000000	1	0	0	0	0	0	,111	X	0	1	X	X	1	,00
101001	,000000	1	0	0	0	0	0	,111	X	0	1	X	X	1	,00
100000	,000000	1	0	0	0	0	0	,010	X	0	1	X	X	1	,00
100010	,000000	1	0	0	0	0	0	,100	X	0	1	X	X	1	,00
100100	,000000	1	0	0	0	0	0	,000	X	0	1	X	X	1	,00
100101	,000000	1	0	0	0	0	0	,001	X	0	1	X	X	1	,00
100110	,000000	1	0	0	0	0	0	,011	X	0	1	X	X	1	,00
,000011	,000000	1	0	0	0	0	0	,101	X	1	1	X	X	1	,00
,000010	,000000	1	0	0	0	0	0	,101	X	1	1	X	X	1	,00
,000000	,000000	1	0	0	0	0	0	,110	X	1	1	X	X	1	,00
,001000	,000000	1	0	0	0	0	0	X	X	X	0	X	X	1	,10
X	,000100	X	1	0	0	0	0	X	,100	0	0	X	X	0	,00
X	,000101	X	0	1	0	0	0	X	,100	0	0	X	X	0	,00
X	,001000	0	0	0	0	0	0	X	,010	1	1	X	1	0	,00
X	,001001	0	0	0	0	0	0	X	,010	1	1	X	1	0	,00
X	,001010	1	0	0	0	0	0	X	,111	1	1	X	1	0	,00
X	,001011	1	0	0	0	0	0	X	,111	0	1	X	X	0	,00
X	,001100	0	0	0	0	0	0	X	,000	1	1	X	0	0	,00
X	,001101	0	0	0	0	0	0	X	,001	1	1	X	0	0	,00
X	,001110	0	0	0	0	0	0	X	,011	1	1	X	0	0	,00
X	,001111	0	0	0	1	0	1	X	,010	0	1	,10	X	0	,00
X	,100000	0	0	0	1	0	1	X	,010	1	1	,00	0	0	,00
X	,100001	0	0	0	1	0	1	X	,010	1	1	,01	0	0	,00
X	,100011	0	0	0	1	0	1	X	,010	1	1	,11	1	0	,00
X	,101000	0	0	0	0	1	1	X	,010	1	0	,00	1	0	,00
X	,101001	0	0	0	0	1	1	X	,010	1	0	,01	1	0	,00
X	,101011	0	0	0	0	1	1	X	,010	1	0	,11	1	0	,00
X	,010000	X	0	0	0	0	X	X	X	X	0	X	1	0	,01
X	,010001	X	0	0	0	0	X	X	X	X	0	X	0	0	,01
X	,000010	X	0	0	0	0	X	X	X	X	0	X	X	0	,11
X	,000011	X	0	0	0	0	X	X	X	X	0	X	X	0	,11

MY CONTROL SIGNALS SHOULD BE LIKE THAT

```
# instruction_function = 101010, instruction_opcode = 000000, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 111, ALUSRC = 0, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = 101001, instruction_opcode = 000000, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 111, ALUSRC = 0, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = 100000, instruction_opcode = 000000, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 010, ALUSRC = 0, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = 100010, instruction_opcode = 000000, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 100, ALUSRC = 0, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = 100100, instruction_opcode = 000000, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 000, ALUSRC = 0, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = 100101, instruction_opcode = 000000, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 001, ALUSRC = 0, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = 100110, instruction_opcode = 000000, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 011, ALUSRC = 0, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = 000011, instruction_opcode = 000000, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 101, ALUSRC = 1, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = 000010, instruction_opcode = 000000, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 101, ALUSRC = 1, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = 000000, instruction_opcode = 000000, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 110, ALUSRC = 1, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = 001000, instruction_opcode = 000000, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 111, ALUSRC = 1, REGWRITE = 0, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 10
# instruction_function = xxxxxx, instruction_opcode = 000100, REGDST = 1, BEQ = 1, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 100, ALUSRC = 0, REGWRITE = 0, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 000101, REGDST = 1, BEQ = 0, BNE = 1, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 100, ALUSRC = 0, REGWRITE = 0, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 001000, REGDST = 0, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 010, ALUSRC = 1, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 1, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 001001, REGDST = 0, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 010, ALUSRC = 1, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 1, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 001010, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 111, ALUSRC = 1, REGWRITE = 1, TYPES = 10, ZERO_SIGN = 1, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 001011, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 111, ALUSRC = 0, REGWRITE = 1, TYPES = 10, ZERO_SIGN = 1, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 001100, REGDST = 0, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 000, ALUSRC = 1, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 001101, REGDST = 0, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 001, ALUSRC = 1, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 001110, REGDST = 0, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 011, ALUSRC = 1, REGWRITE = 1, TYPES = 10, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 001111, REGDST = 0, BEQ = 0, BNE = 0, MEMREAD = 1, MEMWRITE = 0, MEMTOREG = 1, ALUCTR = 010, ALUSRC = 0, REGWRITE = 1, TYPES = 10, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 100000, REGDST = 0, BEQ = 0, BNE = 0, MEMREAD = 1, MEMWRITE = 0, MEMTOREG = 1, ALUCTR = 010, ALUSRC = 1, REGWRITE = 1, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 100001, REGDST = 0, BEQ = 0, BNE = 0, MEMREAD = 1, MEMWRITE = 0, MEMTOREG = 1, ALUCTR = 010, ALUSRC = 1, REGWRITE = 1, TYPES = 01, ZERO_SIGN = 0, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 100011, REGDST = 0, BEQ = 0, BNE = 0, MEMREAD = 1, MEMWRITE = 0, MEMTOREG = 1, ALUCTR = 010, ALUSRC = 1, REGWRITE = 1, TYPES = 11, ZERO_SIGN = 1, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 101000, REGDST = 0, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 1, MEMTOREG = 1, ALUCTR = 010, ALUSRC = 1, REGWRITE = 0, TYPES = 00, ZERO_SIGN = 1, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 101001, REGDST = 0, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 1, MEMTOREG = 1, ALUCTR = 010, ALUSRC = 1, REGWRITE = 0, TYPES = 01, ZERO_SIGN = 1, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 101011, REGDST = 0, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 1, MEMTOREG = 1, ALUCTR = 010, ALUSRC = 1, REGWRITE = 0, TYPES = 11, ZERO_SIGN = 1, JUMP_TYPES = 00
#
# instruction_function = xxxxxx, instruction_opcode = 010000, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 110, ALUSRC = 0, REGWRITE = 0, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 01
#
# instruction_function = xxxxxx, instruction_opcode = 010001, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 110, ALUSRC = 0, REGWRITE = 0, TYPES = 00, ZERO_SIGN = 0, JUMP_TYPES = 01
#
# instruction_function = xxxxxx, instruction_opcode = 000010, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 111, ALUSRC = 1, REGWRITE = 0, TYPES = 10, ZERO_SIGN = 1, JUMP_TYPES = 11
#
# instruction_function = xxxxxx, instruction_opcode = 000011, REGDST = 1, BEQ = 0, BNE = 0, MEMREAD = 0, MEMWRITE = 0, MEMTOREG = 0, ALUCTR = 111, ALUSRC = 0, REGWRITE = 0, TYPES = 10, ZERO_SIGN = 1, JUMP_TYPES = 11
```

AS WE SEEN ABOVE, ALL OF THE CONTROL SIGNALS SIMULATES CORRECTLY.

```
# pc = 00000000, read_reg1 = 01000, read_reg2 = 01001, writeReg = 10000
# Registers-> read_data1 = 00000000000000000000000000001111, read_data2 = 00000000000000000000000011110000, writeData = 00000000000000000000000011111111
# ALUOut = 00000000000000000000000011111111, zero = 0
# Data Memory-> address = 00000000000000000000000011111111, read_data = 00000000000000000000000000011111, writeData = 00000000000000000000000011110000
#
# pc = xxxxxxxx, read_reg1 = xxxxx, read_reg2 = xxxxx, writeReg = xxxxx
# Registers-> read_data1 = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, read_data2 = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, writeData = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# ALUOut = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, zero = x
# Data Memory-> address = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, read_data = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, writeData = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
#
# pc = xxxxxxxx, read_reg1 = xxxxx, read_reg2 = xxxxx, writeReg = xxxxx
# Registers-> read_data1 = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, read_data2 = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, writeData = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# ALUOut = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, zero = x
# Data Memory-> address = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, read_data = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, writeData = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
#
# pc = xxxxxxxx, read_reg1 = xxxxx, read_reg2 = xxxxx, writeReg = xxxxx
# Registers-> read_data1 = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, read_data2 = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, writeData = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# ALUOut = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, zero = x
# Data Memory-> address = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, read_data = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, writeData = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
#
```

AND MY MIPS32_TESTBENCH SIMULATION WHICH IS NOT WORKING :(