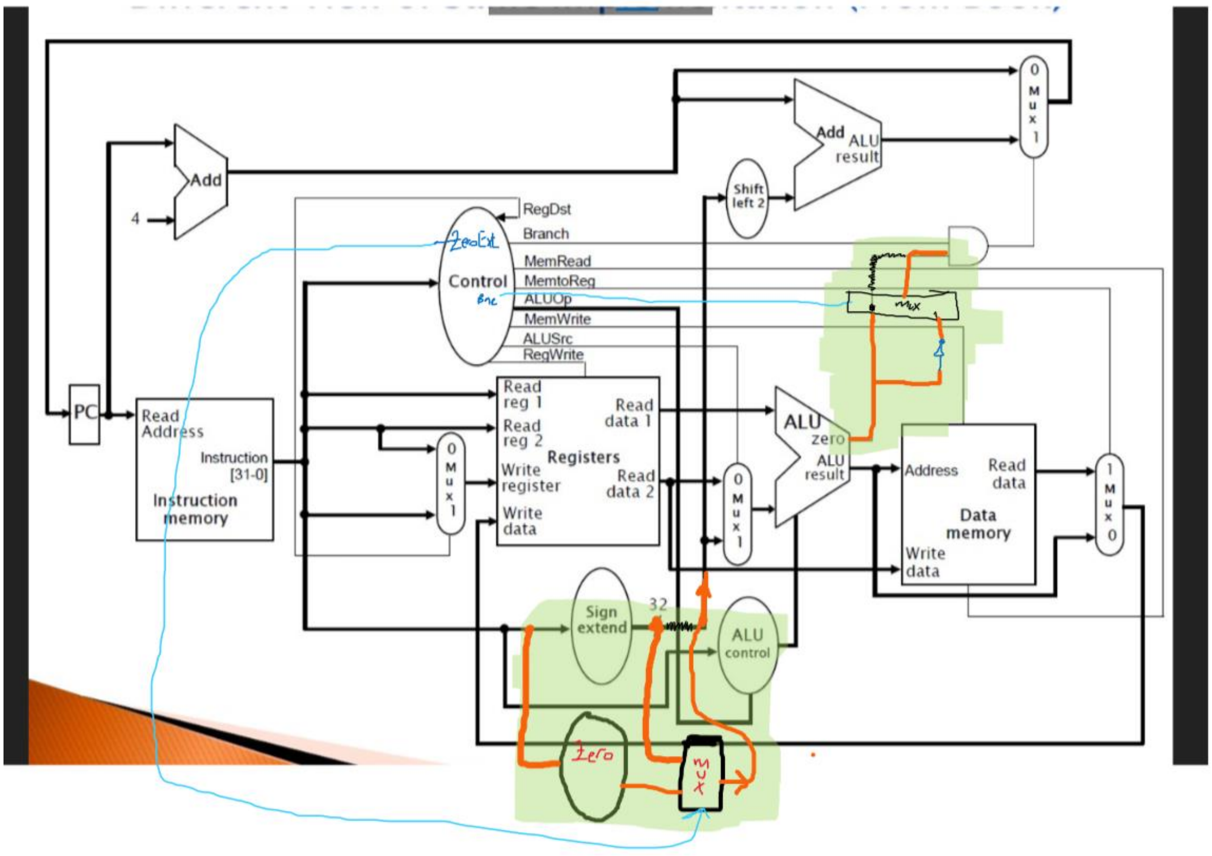
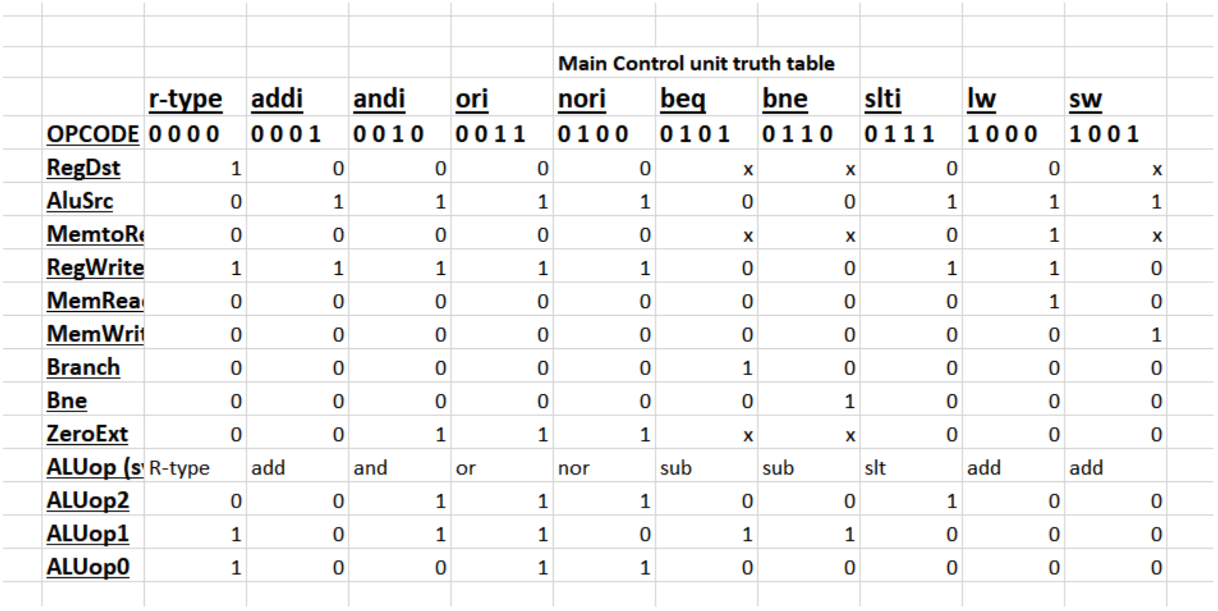
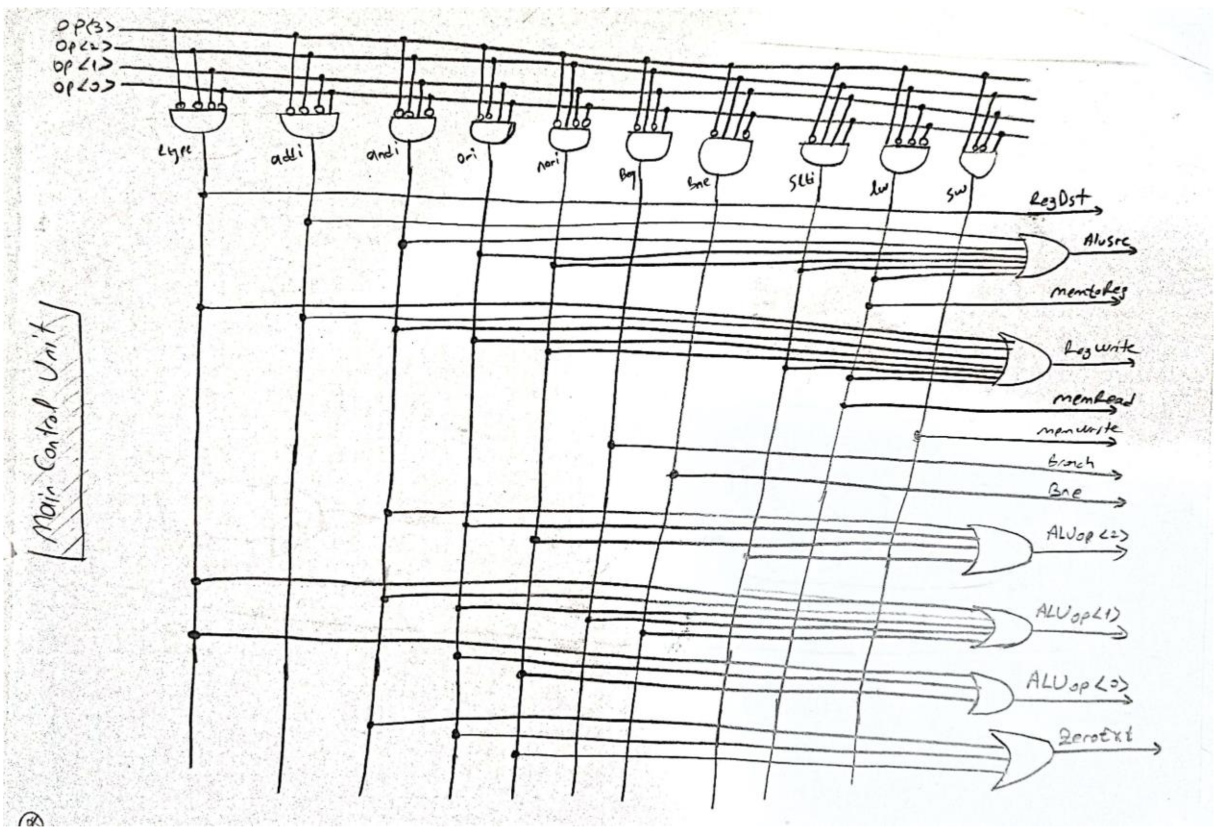
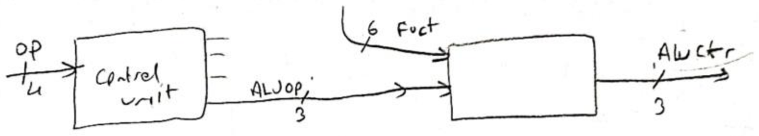




					Main Control unit truth table					
	r-type	**addi**	**andi**	**ori**	**nori**	**beq**	**bne**	**slti**	**lw**	**sw**
OPCODE	**0 0 0 0**	**0 0 0 1**	**0 0 1 0**	**0 0 1 1**	**0 1 0 0**	**0 1 0 1**	**0 1 1 0**	**0 1 1 1**	**1 0 0 0**	**1 0 0 1**
RegDst	1	0	0	0	0	x	x	0	0	x
AluSrc	0	1	1	1	1	0	0	1	1	1
MemtoR	0	0	0	0	0	x	x	0	1	x
RegWrite	1	1	1	1	1	0	0	1	1	0
MemRea	0	0	0	0	0	0	0	0	1	0
MemWri	0	0	0	0	0	0	0	0	0	1
Branch	0	0	0	0	0	1	0	0	0	0
Bne	0	0	0	0	0	0	1	0	0	0
ZeroExt	0	0	1	1	1	x	x	0	0	0
ALUop (s	R-type	add	and	or	nor	sub	sub	slt	add	add
ALUop2	0	0	1	1	1	0	0	1	0	0
ALUop1	1	0	1	1	0	1	1	0	0	0
ALUop0	1	0	0	1	1	0	0	0	0	0





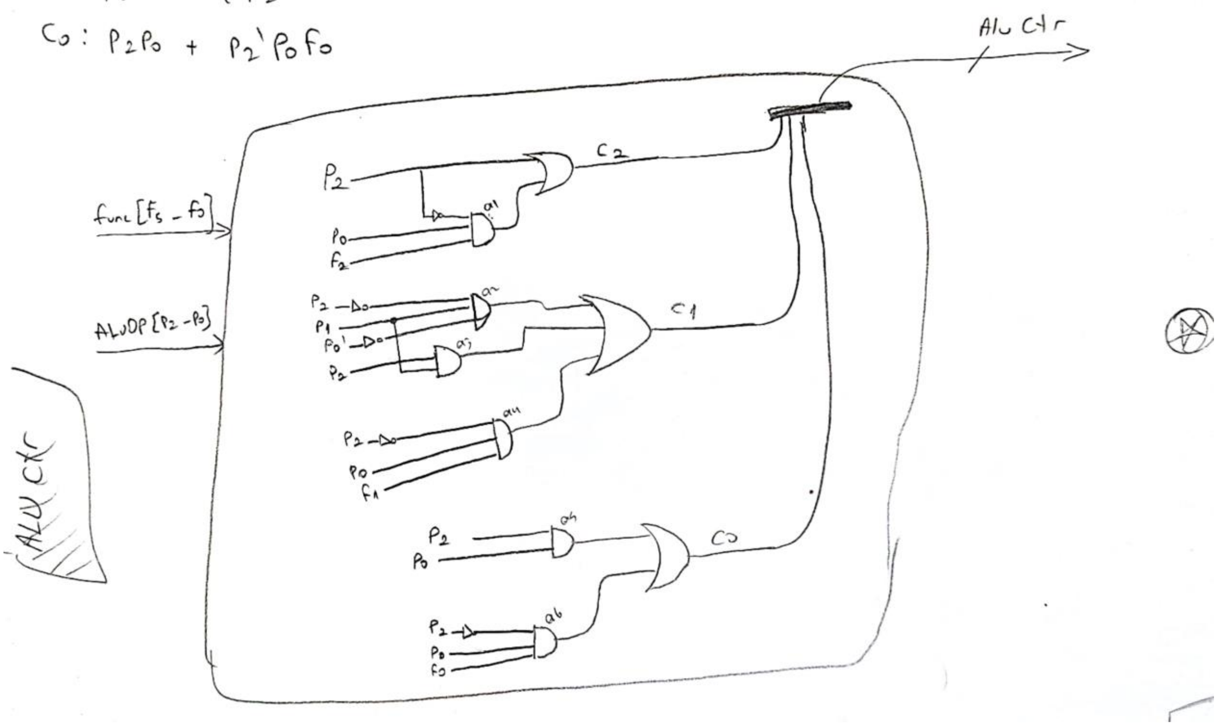
Inst. Opcode	ALU op $P_2 P_1 P_0$	Function $F_5 F_4 F_3 F_2 F_1 F_0$	Desired ALU action	ALUctr $C_2 C_1 C_0$
sw	000	XXXXXX	add	000
lw	000	XXXXXX	add	000
slti	100	XXXXXX	slt	100
Bne	010	XXX XXX	sub	010
Beq	010	XXX XXX	sub	010
nori	101	XXXXXX	nor	101 → 001 (xor)
Ori	111	XXX XXX	or	111 →
andi	110	XXXXXX	and	110
addi	000	XXXXXX	add	000
R-type	011	011000	add	000
R-type	011	011001	xor	001
R-type	011	011010	sub	010
R-type	011	011 100	slt	100
R-type	011	011 101	nor	101
R-type	011	011 110	and	110
R-type	011	011 111	or	111

ALU Table controller

$C_2 : P_2 + P_2' P_0 f_2$

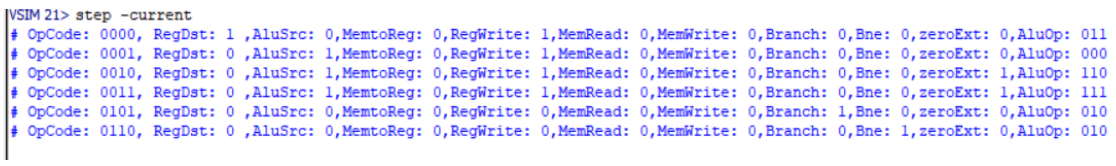
$C_1 : P_2' P_1 P_0' + P_2 P_1 + P_2' P_0 . f_1$

$C_0 : P_2 P_0 + P_2' P_0 f_0$



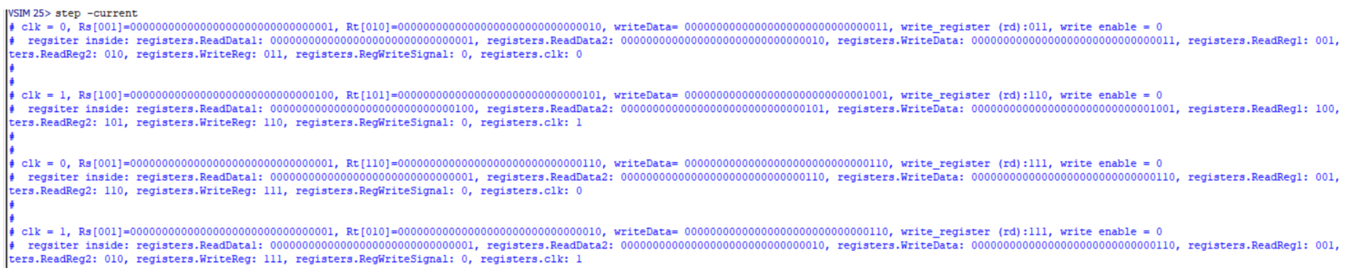
In my project all modules are working but I have problems in PC and clock so that my minimips works while only one instruction.

Controller test_bench:



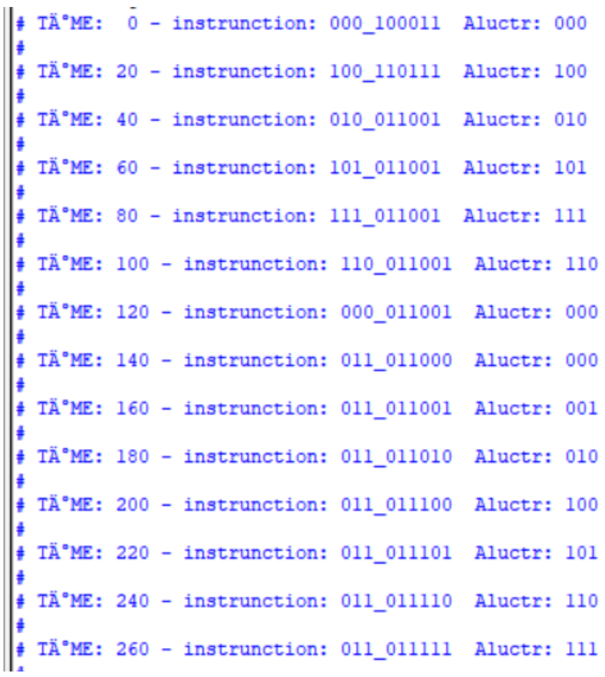
```
VSIM 21> step -current
# OpCode: 0000, RegDst: 1 ,AluSrc: 0,MemtoReg: 0,RegWrite: 1,MemRead: 0,MemWrite: 0,Branch: 0,Bne: 0,zeroExt: 0,AluOp: 011
# OpCode: 0001, RegDst: 0 ,AluSrc: 1,MemtoReg: 1,RegWrite: 1,MemRead: 0,MemWrite: 0,Branch: 0,Bne: 0,zeroExt: 0,AluOp: 000
# OpCode: 0010, RegDst: 0 ,AluSrc: 1,MemtoReg: 0,RegWrite: 1,MemRead: 0,MemWrite: 0,Branch: 0,Bne: 0,zeroExt: 1,AluOp: 110
# OpCode: 0011, RegDst: 0 ,AluSrc: 1,MemtoReg: 0,RegWrite: 1,MemRead: 0,MemWrite: 0,Branch: 0,Bne: 0,zeroExt: 1,AluOp: 111
# OpCode: 0101, RegDst: 0 ,AluSrc: 0,MemtoReg: 0,RegWrite: 0,MemRead: 0,MemWrite: 0,Branch: 1,Bne: 0,zeroExt: 0,AluOp: 010
# OpCode: 0110, RegDst: 0 ,AluSrc: 0,MemtoReg: 0,RegWrite: 0,MemRead: 0,MemWrite: 0,Branch: 0,Bne: 1,zeroExt: 0,AluOp: 010
```

Register test_bench

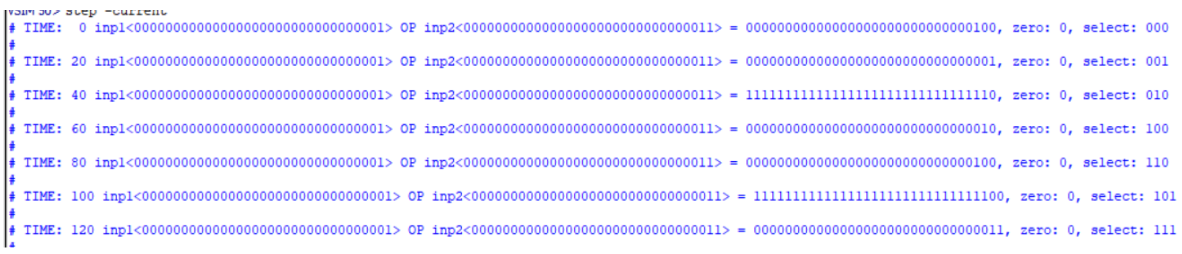


```
VSIM 25> step -current
# clk = 0, Rs[001]=00000000000000000000000000000001, Rt[010]=00000000000000000000000000000010, writeData= 00000000000000000000000000000011, write_register (rd):011, write enable = 0
#  regsiter inside: registers.ReadData1: 00000000000000000000000000000001, registers.ReadData2: 00000000000000000000000000000010, registers.WriteData: 00000000000000000000000000000011, registers.ReadReg1: 001,
ters.ReadReg2: 010, registers.WriteReg: 011, registers.RegWriteSignal: 0, registers.clk: 0
#
#
# clk = 1, Rs[100]=00000000000000000000000000000100, Rt[101]=00000000000000000000000000000101, writeData= 00000000000000000000000000001001, write_register (rd):110, write enable = 0
#  regsiter inside: registers.ReadData1: 00000000000000000000000000000100, registers.ReadData2: 00000000000000000000000000000101, registers.WriteData: 00000000000000000000000000001001, registers.ReadReg1: 100,
ters.ReadReg2: 101, registers.WriteReg: 110, registers.RegWriteSignal: 0, registers.clk: 1
#
#
# clk = 0, Rs[001]=00000000000000000000000000000001, Rt[110]=00000000000000000000000000000110, writeData= 00000000000000000000000000000110, write_register (rd):111, write enable = 0
#  regsiter inside: registers.ReadData1: 00000000000000000000000000000001, registers.ReadData2: 00000000000000000000000000000110, registers.WriteData: 00000000000000000000000000000110, registers.ReadReg1: 001,
ters.ReadReg2: 110, registers.WriteReg: 111, registers.RegWriteSignal: 0, registers.clk: 0
#
#
# clk = 1, Rs[001]=00000000000000000000000000000001, Rt[010]=00000000000000000000000000000010, writeData= 00000000000000000000000000000110, write_register (rd):111, write enable = 0
#  regsiter inside: registers.ReadData1: 00000000000000000000000000000001, registers.ReadData2: 00000000000000000000000000000010, registers.WriteData: 00000000000000000000000000000110, registers.ReadReg1: 001,
ters.ReadReg2: 010, registers.WriteReg: 111, registers.RegWriteSignal: 0, registers.clk: 1
```

Alu Controller test_bench:

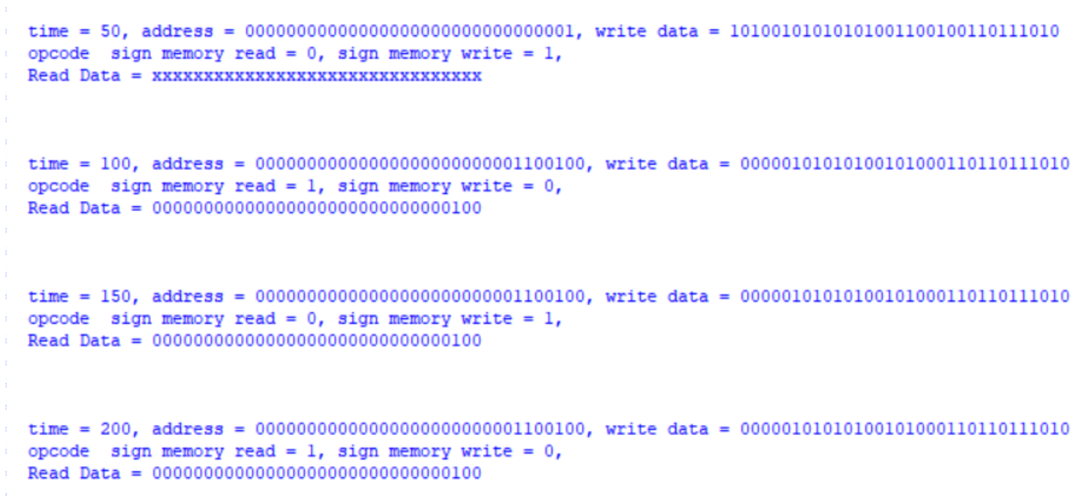

```
# TÄ°ME:  0 - instrunction: 000_100011  Aluctr: 000
#
# TÄ°ME: 20 - instrunction: 100_110111  Aluctr: 100
#
# TÄ°ME: 40 - instrunction: 010_011001  Aluctr: 010
#
# TÄ°ME: 60 - instrunction: 101_011001  Aluctr: 101
#
# TÄ°ME: 80 - instrunction: 111_011001  Aluctr: 111
#
# TÄ°ME: 100 - instrunction: 110_011001  Aluctr: 110
#
# TÄ°ME: 120 - instrunction: 000_011001  Aluctr: 000
#
# TÄ°ME: 140 - instrunction: 011_011000  Aluctr: 000
#
# TÄ°ME: 160 - instrunction: 011_011001  Aluctr: 001
#
# TÄ°ME: 180 - instrunction: 011_011010  Aluctr: 010
#
# TÄ°ME: 200 - instrunction: 011_011100  Aluctr: 100
#
# TÄ°ME: 220 - instrunction: 011_011101  Aluctr: 101
#
# TÄ°ME: 240 - instrunction: 011_011110  Aluctr: 110
#
# TÄ°ME: 260 - instrunction: 011_011111  Aluctr: 111
```

ALU test_bench:


```
VSIM 50> step -current
# TIME:  0 inp1<00000000000000000000000000000001> OP inp2<00000000000000000000000000000011> = 00000000000000000000000000000100, zero: 0, select: 000
#
# TIME: 20 inp1<00000000000000000000000000000001> OP inp2<00000000000000000000000000000011> = 00000000000000000000000000000001, zero: 0, select: 001
#
# TIME: 40 inp1<00000000000000000000000000000001> OP inp2<00000000000000000000000000000011> = 11111111111111111111111111111110, zero: 0, select: 010
#
# TIME: 60 inp1<00000000000000000000000000000001> OP inp2<00000000000000000000000000000011> = 00000000000000000000000000000010, zero: 0, select: 100
#
# TIME: 80 inp1<00000000000000000000000000000001> OP inp2<00000000000000000000000000000011> = 00000000000000000000000000000100, zero: 0, select: 110
#
# TIME: 100 inp1<00000000000000000000000000000001> OP inp2<00000000000000000000000000000011> = 11111111111111111111111111111100, zero: 0, select: 101
#
# TIME: 120 inp1<00000000000000000000000000000001> OP inp2<00000000000000000000000000000011> = 00000000000000000000000000000011, zero: 0, select: 111
```

Also immediate types are work

Memory test_bench:

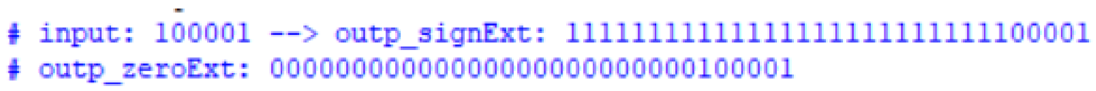

```
time = 50, address = 00000000000000000000000000000001, write data = 10100101010101001100100110111010
opcode  sign memory read = 0, sign memory write = 1,
Read Data = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx


time = 100, address = 00000000000000000000000001100100, write data = 00001010101001010001101101111010
opcode  sign memory read = 1, sign memory write = 0,
Read Data = 00000000000000000000000000000100


time = 150, address = 00000000000000000000000001100100, write data = 00001010101001010001101101111010
opcode  sign memory read = 0, sign memory write = 1,
Read Data = 00000000000000000000000000000100


time = 200, address = 00000000000000000000000001100100, write data = 00001010101001010001101101111010
opcode  sign memory read = 1, sign memory write = 0,
Read Data = 00000000000000000000000000000100
```

Sign-zero Extenders:



```
# input: 100001 --> outp_signExt: 11111111111111111111111111100001
# outp_zeroExt: 00000000000000000000000000100001
```

Not working parts:

PC, branch-bne (due to pc)