Single-Cycle Processor

PROJECT REPORT

Prepared by:

1. Ahmed Mahmoud Abdul-Fattah [CSE].
2. Mahmoud Muhammad Hashem [CSE].
3. Musa Mahmoud Salah [CSE].

Control unit Truth table

Instruction	*Op Code*	*ALU-F*	*ALU SRC*	*ALUO/P*	*extend*	*imed*	*RW*	*MemToReg*	*Reg Dst*	*Mem Read*	*Mem Write*	*R0*	*SSET*	*TE*	*J*	*JAL*	*BEQ*	*BNE*	*B*
AND	*0*	*AND*	*0*	*1*	*x*	*x*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
OR	*0*	*OR*	*0*	*1*	*x*	*x*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
XOR	*0*	*XOR*	*0*	*1*	*x*	*x*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
EQV	*0*	*EQV*	*0*	*1*	*x*	*x*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
Add	*1*	*ADD*	*0*	*1*	*x*	*x*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
Sub	*1*	*Sub*	*0*	*1*	*x*	*x*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
SLt	*1*	*Slt*	*0*	1	*x*	*x*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
SEQ	*1*	*SEQ*	*0*	1	*x*	*x*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
AndI	*4*	*AND*	*1*	*1*	*1*	*0*	*1*	*0*	*x*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
ORI	*5*	*OR*	*1*	*1*	*0*	*0*	*1*	*0*	*x*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
XORI	*6*	*XOR*	*1*	*1*	*0*	*0*	*1*	*0*	*x*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
EQVI	*7*	*EQV*	*1*	*1*	*0*	*0*	*1*	*0*	*x*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
AddI	*8*	*ADD*	*1*	*1*	*2*	*0*	*1*	*0*	*x*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
SLTI	*9*	*Sub*	*1*	*1*	*2*	*0*	*1*	*0*	*x*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
SEQI	*10*	*Sub*	*1*	*1*	*2*	*0*	*1*	*0*	*x*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
SLL	*11*	*SLL*	*1*	*1*	*0*	*0*	*1*	*0*	*x*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
SRL	*12*	*SRL*	*1*	*1*	*0*	*0*	*1*	*0*	*x*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
ROR	*13*	*ROR*	*1*	*1*	*0*	*0*	*1*	*0*	*x*	*0*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
BEQ	*14*	*Sub*	*0*	*0*	*x*	*0*	*0*	*x*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*0*	*0*
BNE	*15*	*Sub*	*0*	*0*	*x*	*0*	*0*	*x*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*0*
LW	*16*	*Add*	*1*	*1*	*2*	*0*	*1*	*1*	*x*	*1*	*0*	*0*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
SW	17	*Add*	*1*	1	2	*0*	0	x	1	0	1	0	0	*x*	0	0	0	0	0
BEQZ	*20*	*Sub*	*0*	*0*	*x*	*1*	*0*	*x*	*1*	*0*	*0*	*1*	*0*	*0*	*0*	0	*1*	*0*	*1*
BNEZ	*21*	*Sub*	*0*	*0*	*x*	*1*	*0*	*x*	*1*	*0*	*0*	*1*	*0*	*0*	*0*	0	*0*	*1*	*1*
BLTZ	*22*	*Sub*	*0*	*2*	*x*	*1*	*0*	*x*	*1*	*0*	*0*	*1*	*0*	*0*	*0*	0	*0*	*0*	*1*
BGEZ	*23*	*Sub*	*0*	*2*	*x*	*1*	*0*	*x*	*1*	*0*	*0*	*1*	*0*	*1*	*0*	0	*0*	*0*	*1*
BGTZ	*24*	*Sub*	*0*	*3*	*x*	*1*	*0*	*x*	*1*	*0*	*0*	*1*	*0*	*0*	*0*	0	*0*	*0*	*1*
BLEZ	*25*	*Sub*	*0*	*3*	*x*	*1*	*0*	*x*	*1*	*0*	*0*	*1*	*0*	*1*	*0*	0	*0*	*0*	*1*
JR	*26*	*Add*	*0*	*1*	*x*	*1*	*0*	*x*	1	*0*	*0*	*1*	0	*0*	*0*	0	*0*	*0*	*1*
JALR	*27*	*Add*	*0*	*1*	*x*	*1*	*1*	*x*	*1*	*0*	*0*	*1*	*0*	*0*	*0*	*1*	*0*	*0*	*1*
SET	*28*	*Add*	*1*	*1*	*2*	*1*	*1*	*0*	*x*	*0*	*0*	*1*	*0*	*x*	*0*	*0*	*0*	*0*	*0*
SSET	*29*	*Add*	*0*	*1*	*0*	*1*	*1*	*0*	*1*	*0*	*0*	*2*	1	*x*	*0*	*0*	*0*	*0*	*0*
J	*30*	*x*	*x*	*x*	*x*	*2*	*0*	*x*	*x*	*0*	*0*	*x*	*x*	*x*	*1*	*0*	*0*	*0*	*0*
JAL	*31*	*x*	*x*	*x*	*x*	*2*	*1*	*x*	*x*	*0*	*0*	*x*	*x*	*x*	*1*	*1*	*0*	*0*	*0*

ALU Contol Unit Truth Table for R-type

OpCode	*2-bit f*	*index*	*ALUoperation*	A	B	C	D	E
0	*0*	*0*	*AND*	0	0	0	0	0
0	*1*	*1*	*OR*	0	0	0	0	1
0	*2*	*2*	*XOR*	0	0	0	1	0
0	*3*	*3*	*EQV*	0	0	1	1	1
1	*0*	*4*	*ADD*	0	1	0	0	0
1	*1*	*5*	*Sub*	0	1	0	0	1
1	*2*	*6*	*Slt*	0	1	1	0	1
1	*3*	*7*	*SEQ*	1	0	0	0	1

ALU Contol Unit Truth Table for R-type

OpCode	*ALUoperation*	A	B	C	D	E
4	*AND*	0	0	0	0	0
5	*OR*	0	0	0	0	1
6	*XOR*	0	0	0	1	0
7	*EQV*	0	0	0	1	1
8	*ADD*	0	1	0	0	0
9	*Slt*	0	1	1	0	1
10	*Seq*	1	0	0	0	1
11	*SLL*	0	0	1	0	0
12	*SRL*	0	0	1	0	1
13	*ROR*	0	0	1	1	0
14	*Sub*	0	1	0	0	1
15	*Sub*	0	1	0	0	1
16	*Add*	0	1	0	0	0
17	*Add*	0	1	0	0	0
20	*Sub*	0	1	0	0	1
21	*Sub*	0	1	0	0	1
22	*Sub*	0	1	0	0	1
23	*Sub*	0	1	0	0	1
24	*Sub*	0	1	0	0	1
25	*Sub*	0	1	0	0	1
26	*Add*	0	1	0	0	0
27	*Add*	0	1	0	0	0
28	*Add*	0	1	0	0	0
29	*Add*	0	1	0	0	0
30	*x*	x	x	x	x	x
31	*x*	x	x	x	x	x

Test Code

instruction	Binary Code	Hexadecimal Code
set r1, 0x55	11100 001 0101 0101	E155
sset r1, 0x12	11101 001 0001 0010	E912
sset r1, 0x10	11101 001 0001 0000	E910
sset r1, 0x20	11101 001 0010 0000	E920
sset r1, 0x00	11101 001 0000 0000	E900
set r6, 0x22	11100 110 0010 0010	E622
sset r6, 0x06	11101 110 0000 0110	EE06
sset r6, 0x02	11101 110 0000 0010	EE02
sset r6, 0x11	11101 110 0001 0001	EE11
eqv r3, r6, r1	00000 011 110 001 11	03C7
sub r0,r0,r0	00001 000 000 000 01	0801
lw r1, 0(r0)	10000 001 000 00000	8100
lw r2, 1(r0)	10000 010 000 00001	8201
lw r3, 2(r0)	10000 011 000 00010	8302
addi r4, r4, 10	01000 100 100 01010	448A
sub r4,r4,r4	00001 100 100 100 01	0C91
add r4,r2,r4	00001 100 010 100 0	0628
slt r5,r2,r3	00001 101 010 011 10	0D4E
beq r5,r0,2	01110 101 000 00010	7502
add r2,r1,r2	00001 010 001 010 00	0A28
beq r0,r0,-5	01110 000 000 11011	701B
sw r4, 0(r0)	10001 100 000 00000	8C00
jal func	11111 000 0000 0110	F806
ror r6, r6, 8	01101 110 110 01000	6EC8
lw r4, 1(r0)	10000 100 000 00001	8401
lw r5, 2(r0)	10000 101 000 00010	8502
add r6, r6, r5	00001 110 110 101 00	0ED4
beq r0,r0,-1	01110 000 000 11111	701F
func: sub r0,r0,r0	00001 000 000 000 01	0801
lw r1, 0(r0)	10000 001 000 00000	8100
lw r2, 0(r1)	10000 010 001 00000	8220

lw r3, 1(r1)	10000 011 001 00001	8321
and r4,r2,r3	00000 100 010 011 00	044C
or r5, r2, r3	00000 101 010 011 01	054D
sw r4, 1(r0)	10001 100 000 00001	8C01
sw r5, 2(r0)	10001 101 000 00010	8D02
jr r7	11010 111 0000 0000	D700

$$ALUSRC = (ANDI + ORI + XORI + EQVI + AddI + SLTI + SEQI + SLL + SRL + ROR + LW + SW + set)$$

$$ALU\text{-}Output_0 = \overline{(BEQ + BNE + BEQZ + BNEZ + BLTZ + BGEZ)} \Rightarrow \text{zero bit}$$

$$ALU\text{-}Output_1 = (BLTZ + BGEZ + BGTZ + BLEZ) \Rightarrow \text{the one bit}$$

$$extender_0 = ANDI \Rightarrow \text{zero bit}$$

$$extender_1 = (AddI + SLTI + SEQI + LW + SW + set) \Rightarrow \text{The one bit}$$

$$immediate_0 = (BEQZ + BNEZ + BLTZ + BGEZ + BGTZ + BLEZ + JR + JALR + SET + SSET) \Rightarrow \text{The zero bit}$$

$$immediate_1 = (J + JAL) \Rightarrow \text{The one bit}$$

$$RW = \overline{(BEQ + BNE + SW + BEQZ + BNEZ + BLTZ + BGEZ + BGTZ + BLEZ + JR + J)}$$

$$MemToReg = LW$$

$$RegDST = (BEQ + BNE + SW + BEQZ + BNEZ + BLTZ + BGEZ + BGTZ + BLEZ + JR + JALR + SSET)$$

$$MemRead = LW \quad , MemWrite = SW$$

$R_0 = (BEQZ + BNEZ + BLTZ + BGEZ + BGTZ + BLEZ + JR + JALR + SET)$ → The zero bit

$R_0 = SSET$ → The one bit

$SSET = SSET$, $TE = (BGEZ + BLEZ)$, $J = (J + JAL)$, $JAL = (JALR + JAL)$

$BEQ = (BEQ + BEQZ)$, $BNE = (BNE + BNEZ)$, $Jr = JR$

$B = (BEQZ + BNEZ + BLTZ + BGEZ + BGTZ + BLEZ + JR + JALR)$

ALU Signals NOT R Type

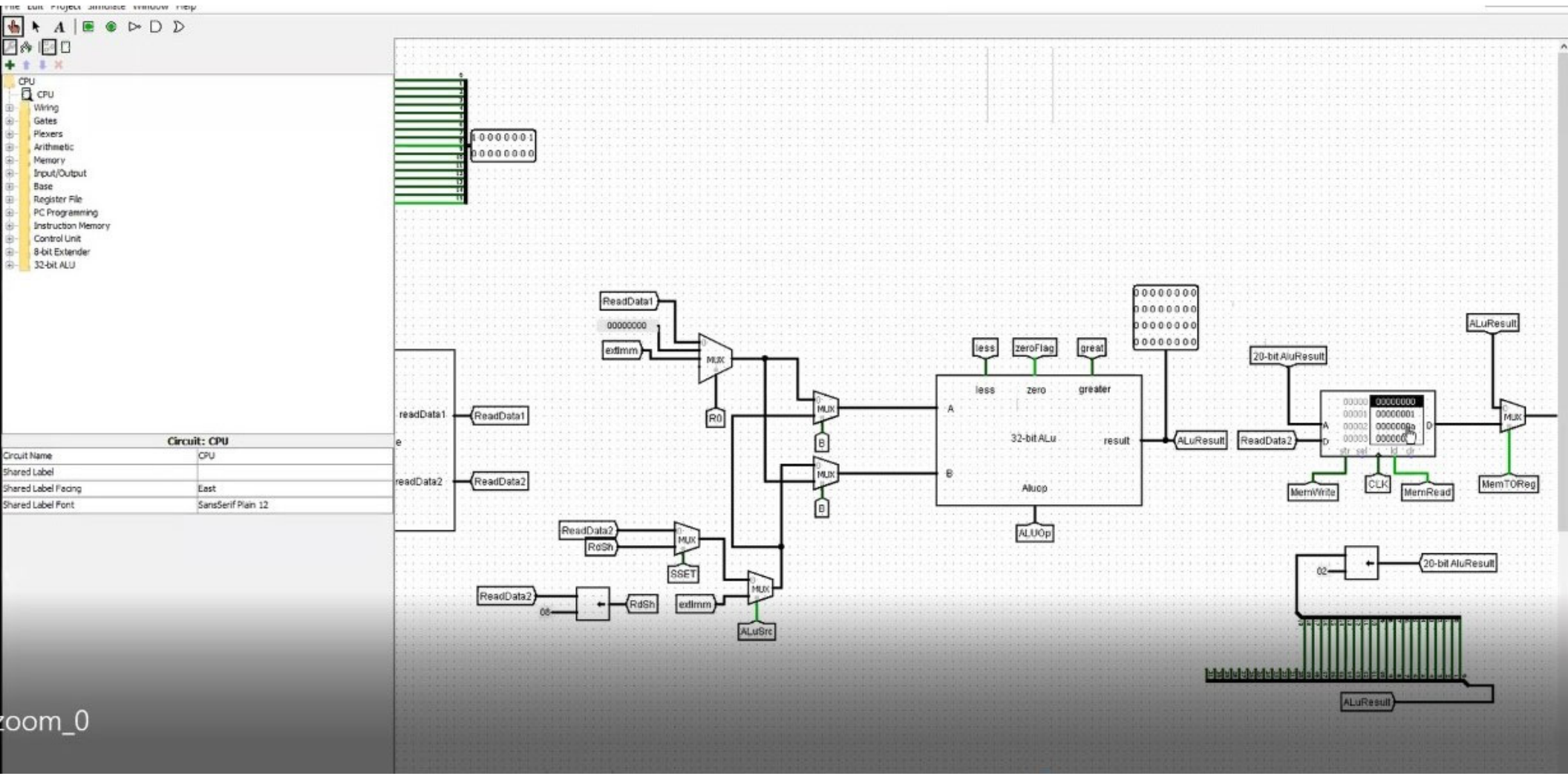
$ALU\text{-}A = SEQI$

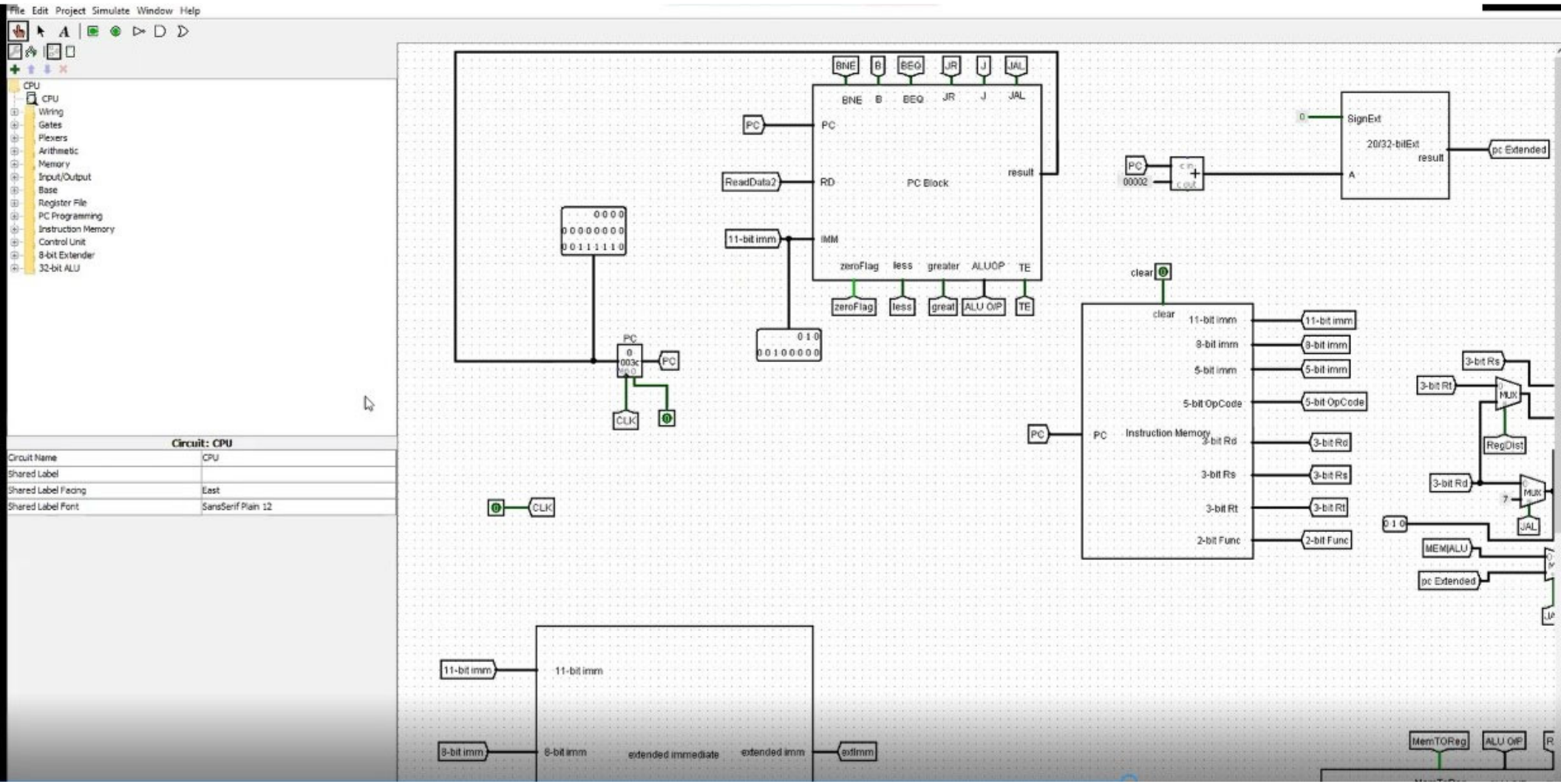
$ALU\text{-}B = (ANDI + ORI + XORI + EQVI + SEQI + SLL + SRL + ROR)$

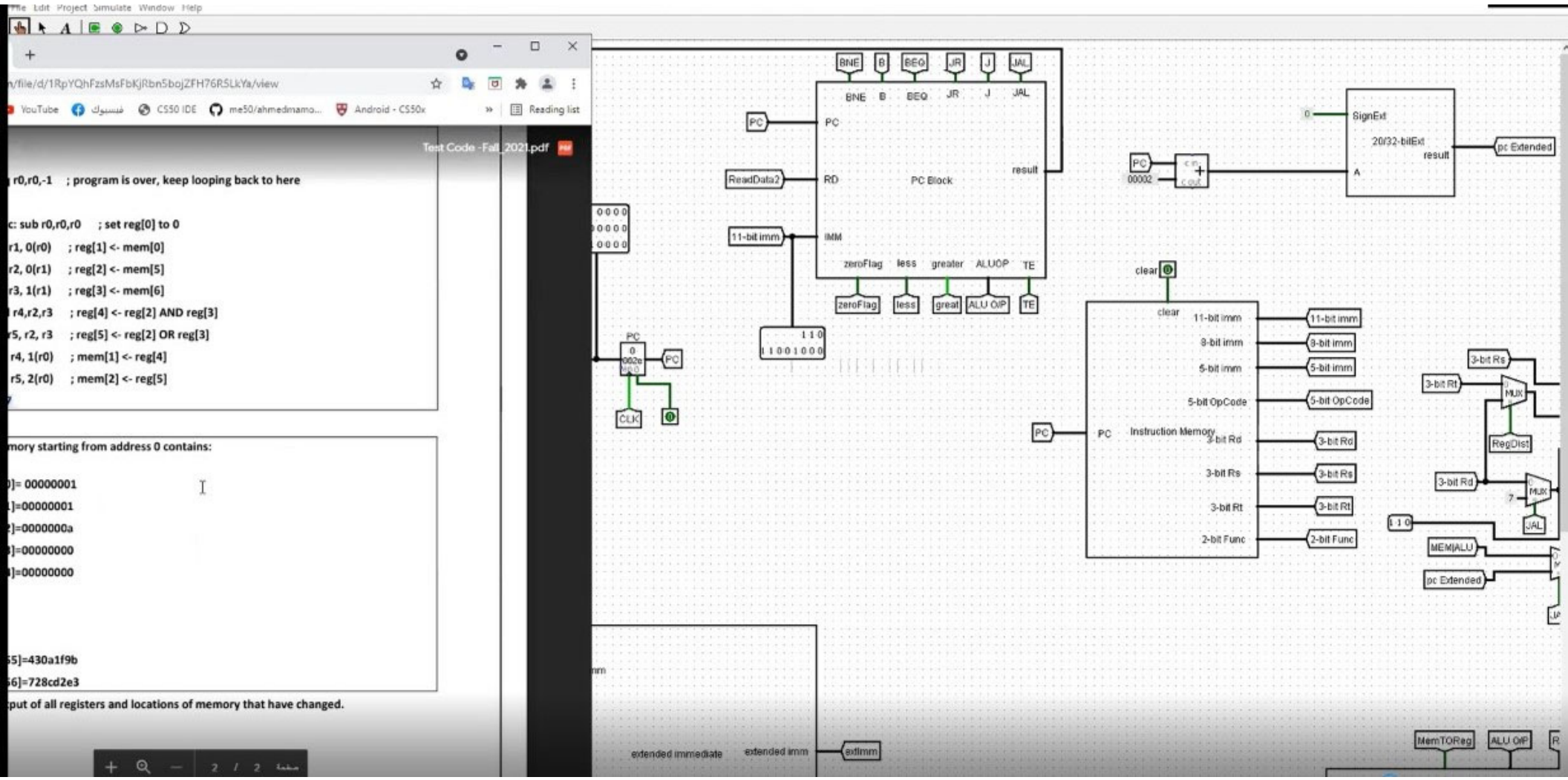
$ALU\text{-}C = (SLTI + SLL + SRL + ROR)$, $ALU\text{-}d = (XORI + EQVI + ROR)$

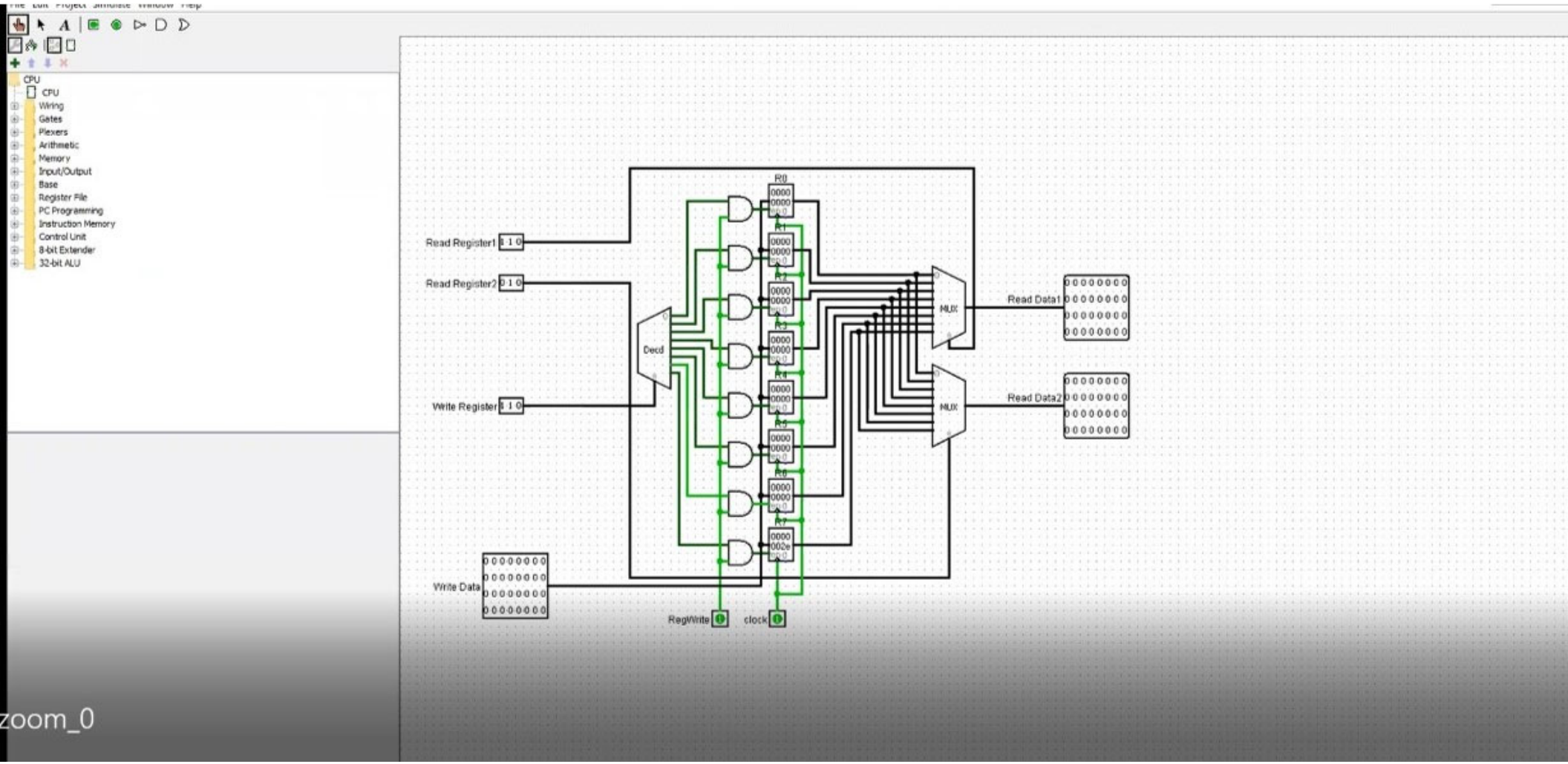
$ALU\text{-}E = (ORI + EQVI + SLTI + SEQI + SRL + BEQ + BNE + BEQZ + BNEZ + BLTZ + BGEZ + BGTZ + BLEZ)$

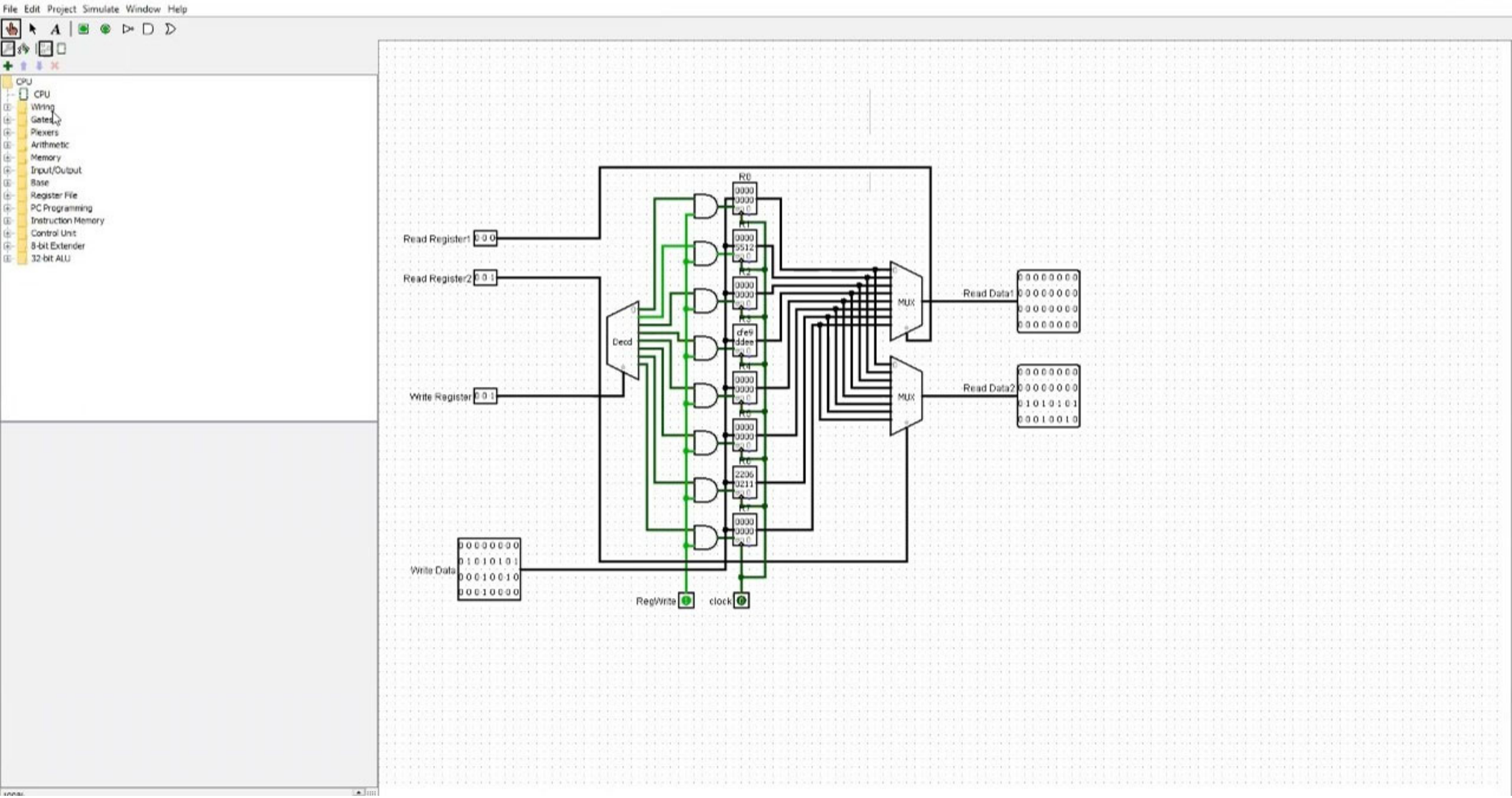
Testing Outputs

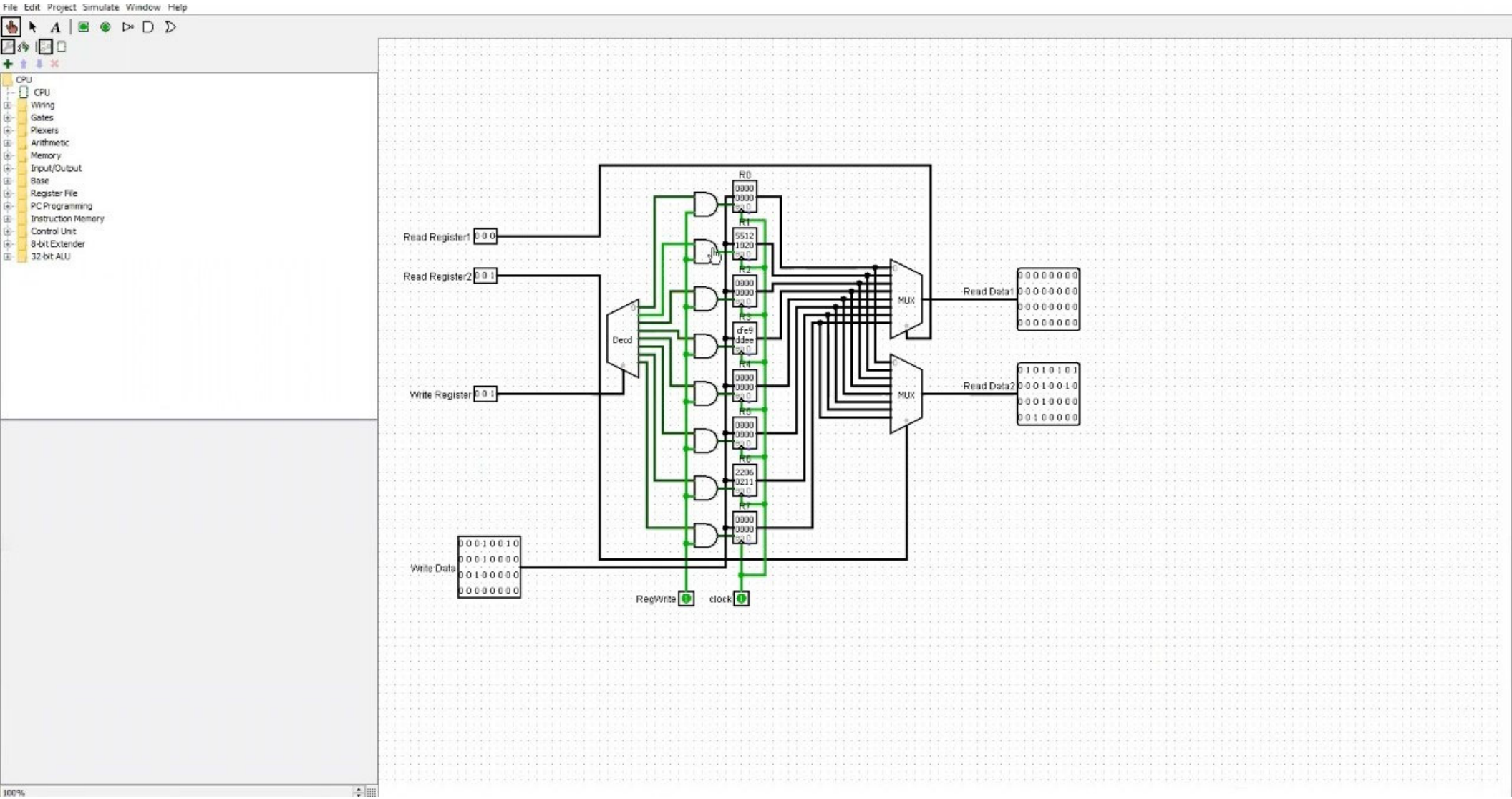

CPU
CPU
Wiring
Gates
Plexers
Arithmetic
Memory
Input/Output
Base
Register File
PC Programming
Instruction Memory
Control Unit
8-bit Extender
32-bit ALU
Circuit: CPU
Circuit Name
CPU
Shared Label
Shared Label Facing
East
Shared Label Font
SansSerif Plain 12
ReadData1
extImm
MUX
R0
B
less
zeroFlag
great
zero
greater
A
32-bit ALu
result
ALuop
ALUOp
ALuResult
20-bit AluResult
ReadData2
MemWrite
CLK
MemRead
MemTOReg
readData1
readData2
RdSh
SSET
ALuSrc
02
08
zoom_0

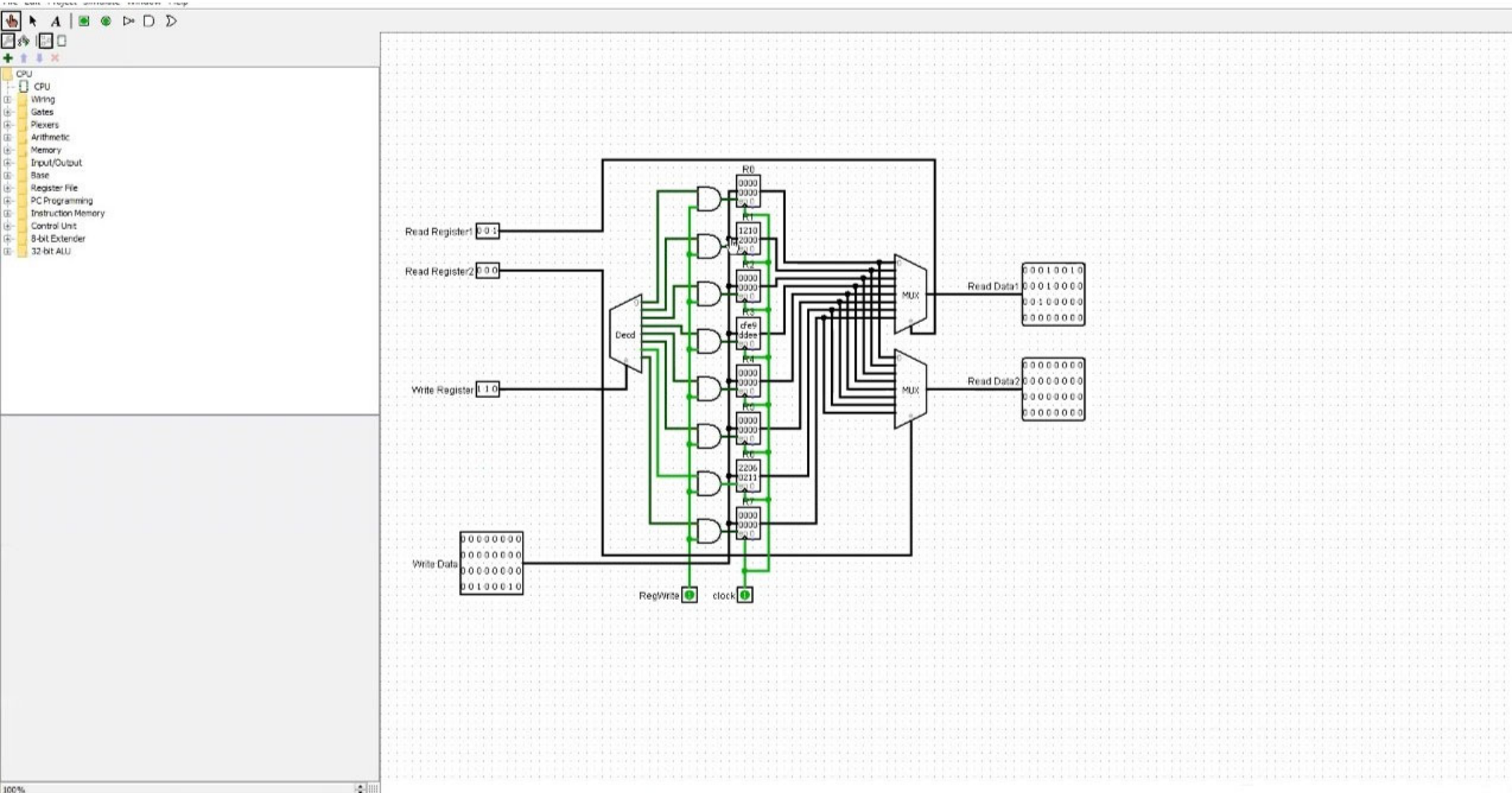

File Edit Project Simulate Window Help
CPU
CPU
Wiring
Gates
Plexers
Arithmetic
Memory
Input/Output
Base
Register File
PC Programming
Instruction Memory
Control Unit
8-bit Extender
32-bit ALU
Circuit: CPU
Circuit Name
CPU
Shared Label
Shared Label Facing
East
Shared Label Font
SansSerif Plain 12
BNE
B
BEQ
JR
J
JAL
PC
ReadData2
RD
PC Block
result
11-bit imm
IMM
zeroFlag
less
greater
ALUOP
TE
0000
00000000
00111110
010
00100000
PC
0
003c
CLK
clear
11-bit imm
8-bit imm
5-bit imm
5-bit OpCode
3-bit Rd
3-bit Rs
3-bit Rt
2-bit Func
Instruction Memory
SignExt
20/32-bitExt
pc Extended
00002
A
3-bit Rs
3-bit Rt
MUX
RegDist
3-bit Rd
7
JAL
010
MEM|ALU
pc Extended
8-bit imm
extended immediate
extImm
MemTOReg
ALU OP

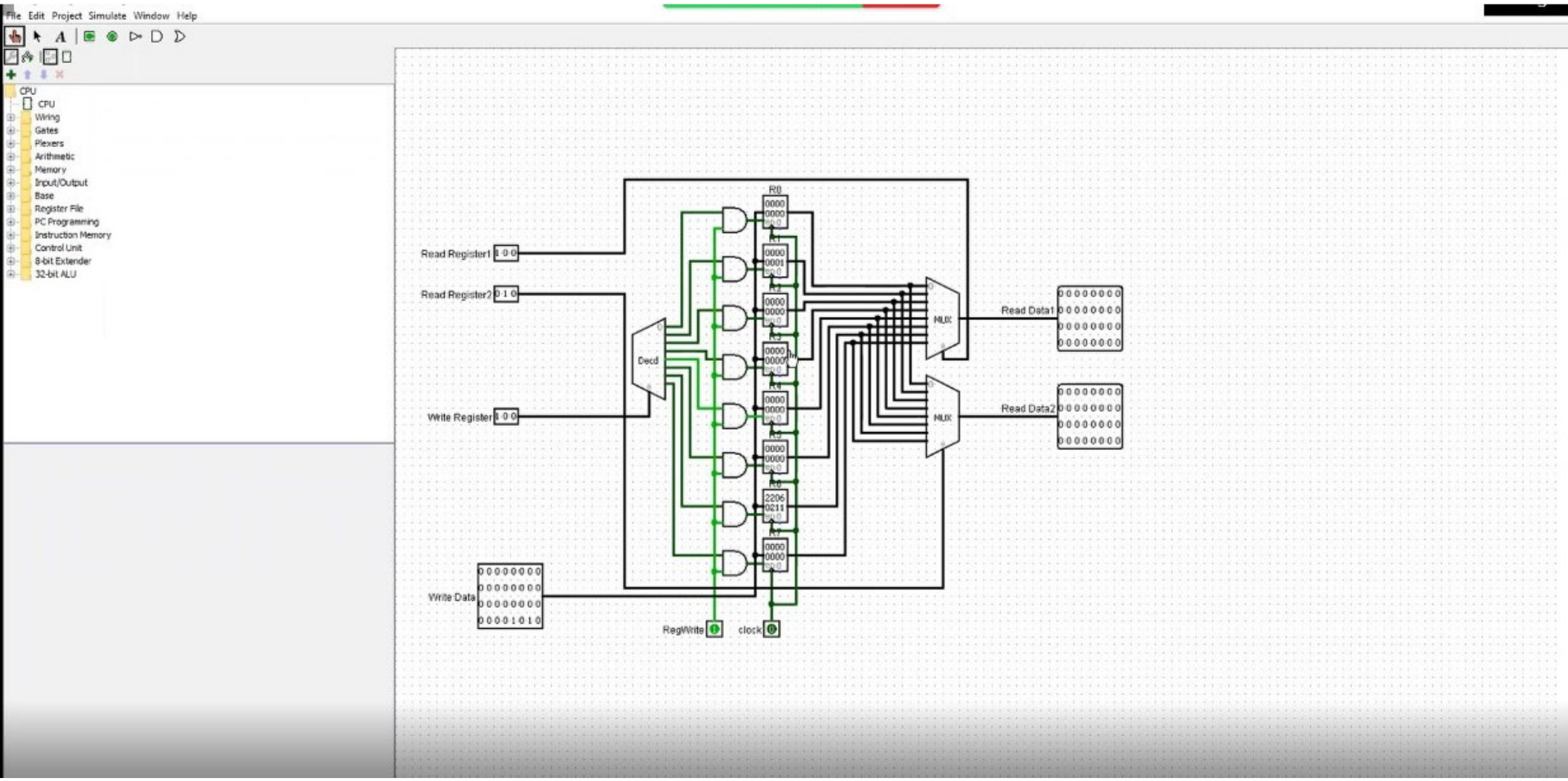

File Edit Project Simulate Window Help
n/file/d/1RpYQhFzsMsFbKjRbn5bojZFH76R5LkYa/view
YouTube
CS50 IDE
me50/ahmedmamo...
Android - CS50x
Reading list
Test Code -Fall_2021.pdf
r0,r0,-1 ; program is over, keep looping back to here
c: sub r0,r0,r0 ; set reg[0] to 0
r1, 0(r0) ; reg[1] <- mem[0]
r2, 0(r1) ; reg[2] <- mem[5]
r3, 1(r1) ; reg[3] <- mem[6]
r4,r2,r3 ; reg[4] <- reg[2] AND reg[3]
r5, r2, r3 ; reg[5] <- reg[2] OR reg[3]
r4, 1(r0) ; mem[1] <- reg[4]
r5, 2(r0) ; mem[2] <- reg[5]
mory starting from address 0 contains:
]= 00000001
]=00000001
]=0000000a
]=00000000
]=00000000
5]=430a1f9b
6]=728cd2e3
put of all registers and locations of memory that have changed.
2 / 2
BNE
B
BEQ
JR
J
JAL
PC
ReadData2
RD
PC Block
result
11-bit imm
IMM
zeroFlag
less
greater
ALUOP
TE
SignExt
20/32-bitExt
pc Extended
clear
8-bit imm
5-bit imm
5-bit OpCode
Instruction Memory
3-bit Rd
3-bit Rs
3-bit Rt
2-bit Func
CLK
RegDist
MUX
JAL
MEM|ALU
extended immediate
extended imm
extImm
MemTOReg
ALU OP

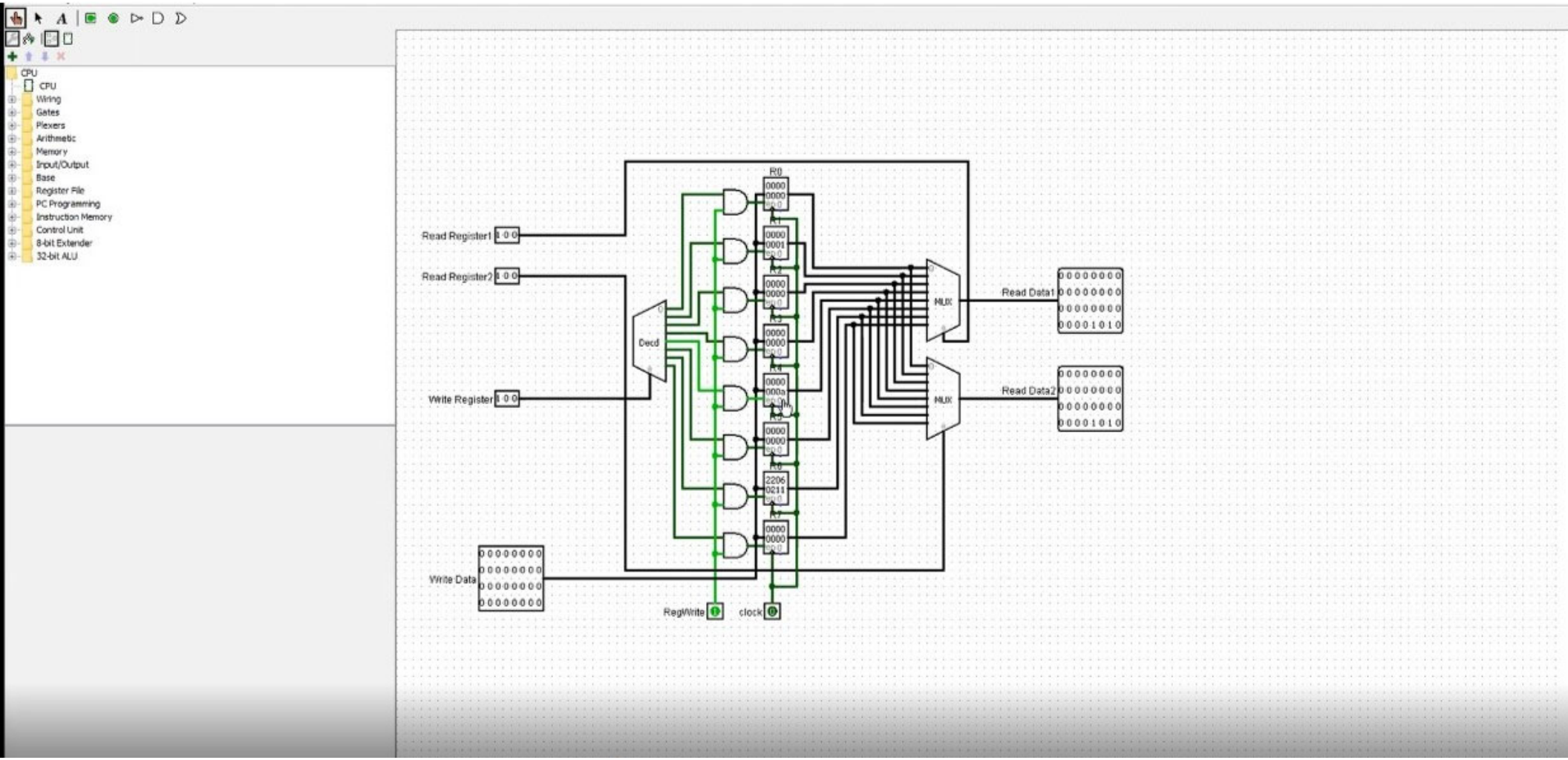

CPU
CPU
Wiring
Gates
Plexers
Arithmetic
Memory
Input/Output
Base
Register File
PC Programming
Instruction Memory
Control Unit
8-bit Extender
32-bit ALU
R0
R1
R2
R3
R4
R5
R6
R7
Read Register1
Read Register2
Write Register
Write Data
Decd
MUX
MUX
Read Data1
Read Data2
RegWrite
clock

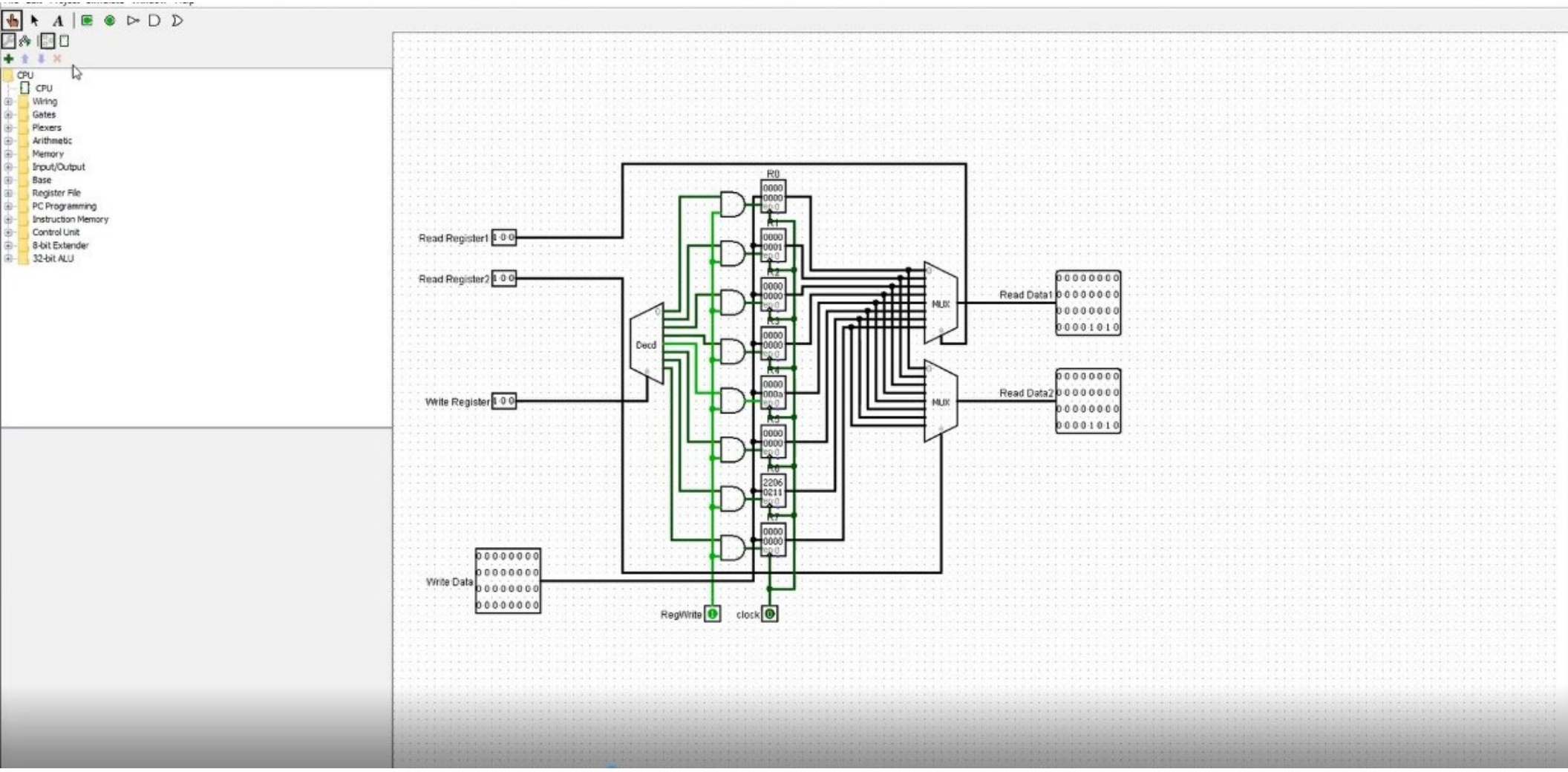

File Edit Project Simulate Window Help
CPU
CPU
Wiring
Gates
Plexers
Arithmetic
Memory
Input/Output
Base
Register File
PC Programming
Instruction Memory
Control Unit
8-bit Extender
32-bit ALU
R0
Read Register1
Read Register2
Write Register
Decd
MUX
MUX
Read Data1
Read Data2
Write Data
RegWrite
clock

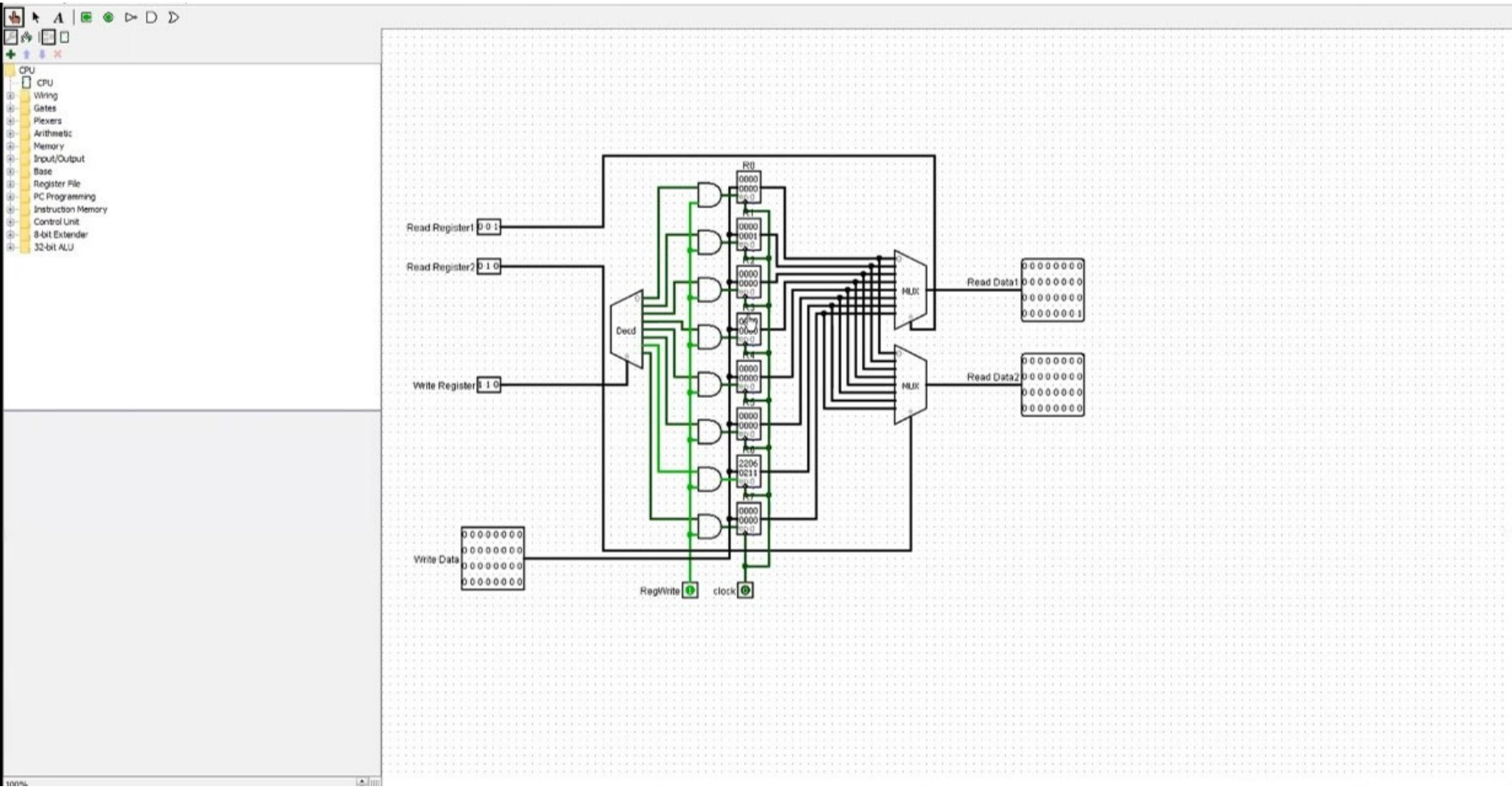

File Edit Project Simulate Window Help
CPU
CPU
Wiring
Gates
Plexers
Arithmetic
Memory
Input/Output
Base
Register File
PC Programming
Instruction Memory
Control Unit
8-bit Extender
32-bit ALU
100%
R0
R1
R2
R3
R4
R5
R6
R7
5512
1020
cfe9
ddee
2206
0211
Read Register1
Read Register2
Write Register
Write Data
Decd
MUX
MUX
Read Data1
Read Data2
RegWrite
clock

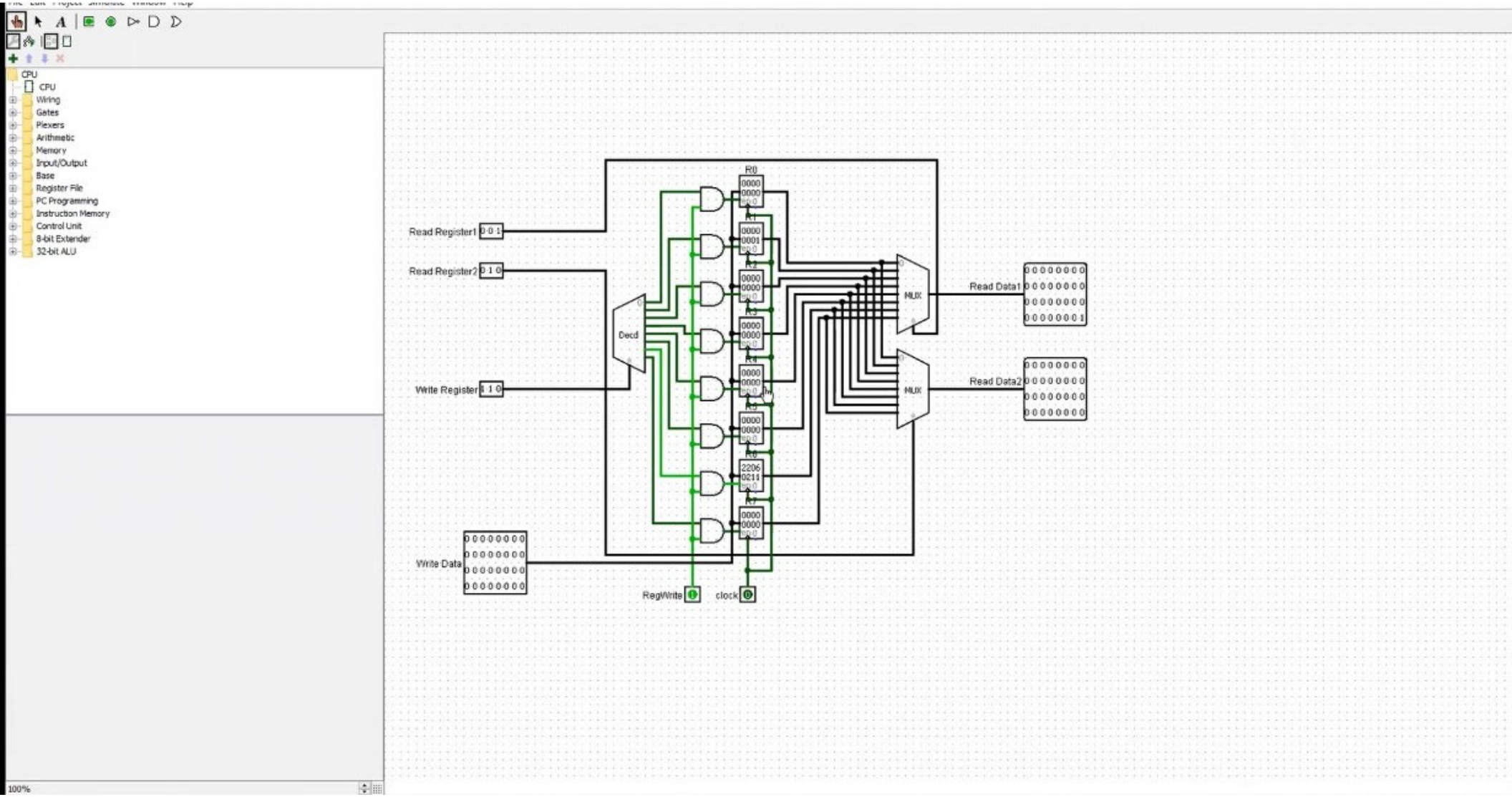

CPU
CPU
Wiring
Gates
Plexers
Arithmetic
Memory
Input/Output
Base
Register File
PC Programming
Instruction Memory
Control Unit
8-bit Extender
32-bit ALU
Read Register1
Read Register2
Write Register
Write Data
RegWrite
clock
Decd
R0
R1
R2
R3
R4
R5
R6
R7
MUX
MUX
Read Data1
Read Data2


File Edit Project Simulate Window Help
CPU
CPU
Wiring
Gates
Plexers
Arithmetic
Memory
Input/Output
Base
Register File
PC Programming
Instruction Memory
Control Unit
8-bit Extender
32-bit ALU
R0
Read Register1
Read Register2
Write Register
Write Data
Decd
MUX
MUX
Read Data1
Read Data2
RegWrite
clock


CPU
CPU
Wiring
Gates
Plexers
Arithmetic
Memory
Input/Output
Base
Register File
PC Programming
Instruction Memory
Control Unit
8-bit Extender
32-bit ALU
R0
Read Register1
Read Register2
Write Register
Decd
MUX
MUX
Read Data1
Read Data2
Write Data
RegWrite
clock


CPU
CPU
Wiring
Gates
Plexers
Arithmetic
Memory
Input/Output
Base
Register File
PC Programming
Instruction Memory
Control Unit
8-bit Extender
32-bit ALU
R0
Read Register1
Read Register2
Write Register
Write Data
Decd
MUX
MUX
Read Data1
Read Data2
RegWrite
clock


CPU
CPU
Wiring
Gates
Plexers
Arithmetic
Memory
Input/Output
Base
Register File
PC Programming
Instruction Memory
Control Unit
8-bit Extender
32-bit ALU
R0
Read Register1
Read Register2
Write Register
Decd
MUX
MUX
Read Data1
Read Data2
Write Data
RegWrite
clock


CPU
CPU
Wiring
Gates
Plexers
Arithmetic
Memory
Input/Output
Base
Register File
PC Programming
Instruction Memory
Control Unit
8-bit Extender
32-bit ALU
R0
Read Register1
Read Register2
Write Register
Write Data
Decd
MUX
MUX
Read Data1
Read Data2
RegWrite
clock
100%