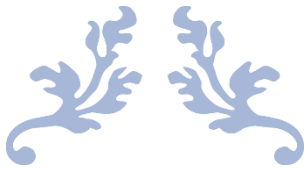
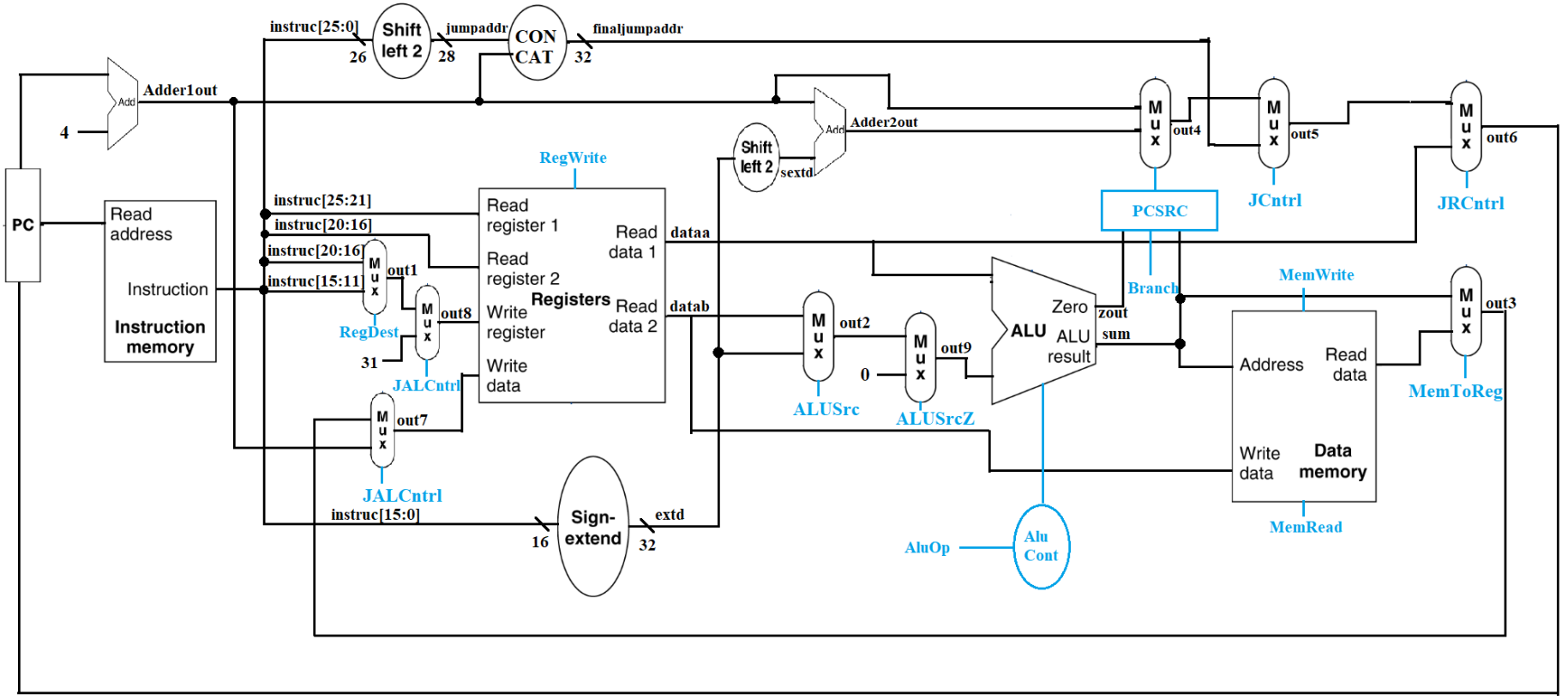


ASSIGNMENT 2 REPORT

Sebahattin Can Yavuzkurt - 240201040

Revised Datapath



Control Signals

	IN	FUNC	BRANCHF	REG DEST	ALU SRC	MEM TO REG	REG WRITE	MEM READ	MEM WRITE	BRANCH	ALU OP	J CNTRL	JR CNTRL	JAL CNTRL	ALUSRCZ
jr	0	00100	xxxxx	0	0	0	0	0	0	000	0000	0	1	0	0
rformat	0	xx0xx	xxxxx	1	0	0	1	0	0	000	0000	0	0	0	0
Beq	000100	xxxxx	xxxxx	0	0	0	0	0	0	001	0010	0	0	0	0
Bneq	000101	xxxxx	xxxxx	0	0	0	0	0	0	010	0010	0	0	0	0
Bgez	000001	xxxxx	xxxx1	0	0	0	0	0	0	011	0010	0	0	0	1
Bgtz	000111	xxxxx	xxxxx	0	0	0	0	0	0	100	0010	0	0	0	1
Blez	000110	xxxxx	xxxxx	0	0	0	0	0	0	101	0010	0	0	0	1
Bltz	000001	xxxxx	xxxx0	0	0	0	0	0	0	110	0010	0	0	0	1
Addi	001000	xxxxx	xxxxx	0	1	0	1	0	0	000	0000	0	0	0	0
Andi	001100	xxxxx	xxxxx	0	1	0	1	0	0	000	0100	0	0	0	0
Ori	001101	xxxxx	xxxxx	0	1	0	1	0	0	000	1000	0	0	0	0
Jump	000010	xxxxx	xxxxx	0	0	0	0	0	0	000	0000	1	0	0	0
jal	000011	xxxxx	xxxxx	0	0	0	0	0	0	000	0000	1	0	1	0

Note On PSRC: PSRC is a bit complex, but what it does is gets the branch output, zout from alu and sum’s most significant(sign) bit from alu. Than through logical calculations, determines whether it satisfies branch equations ad allow the branch offset added value to pass throught and change programming counter according to it.

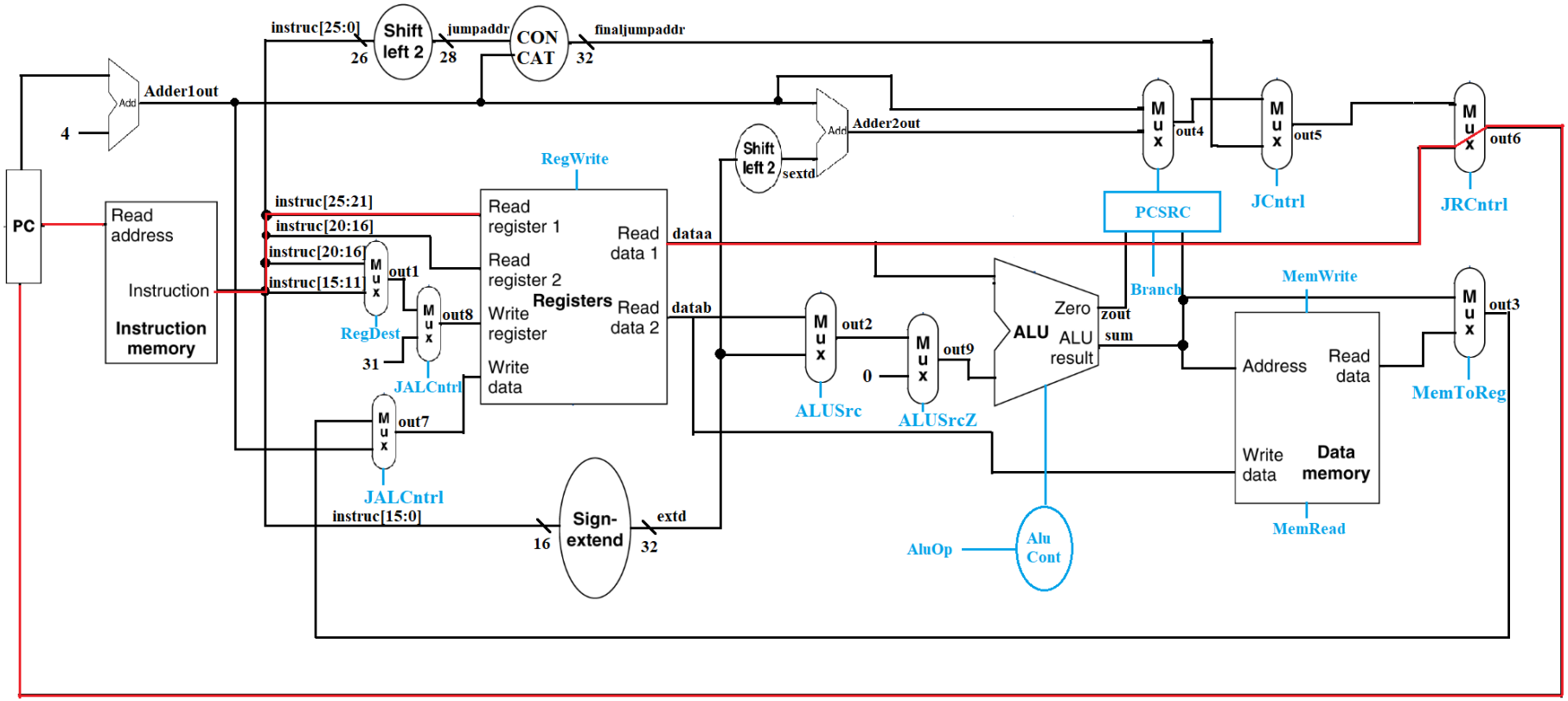
Compatibility With Original Code

First image below is a screenshot of untouched original code running on the original files. Second is the extended code running on the same files.

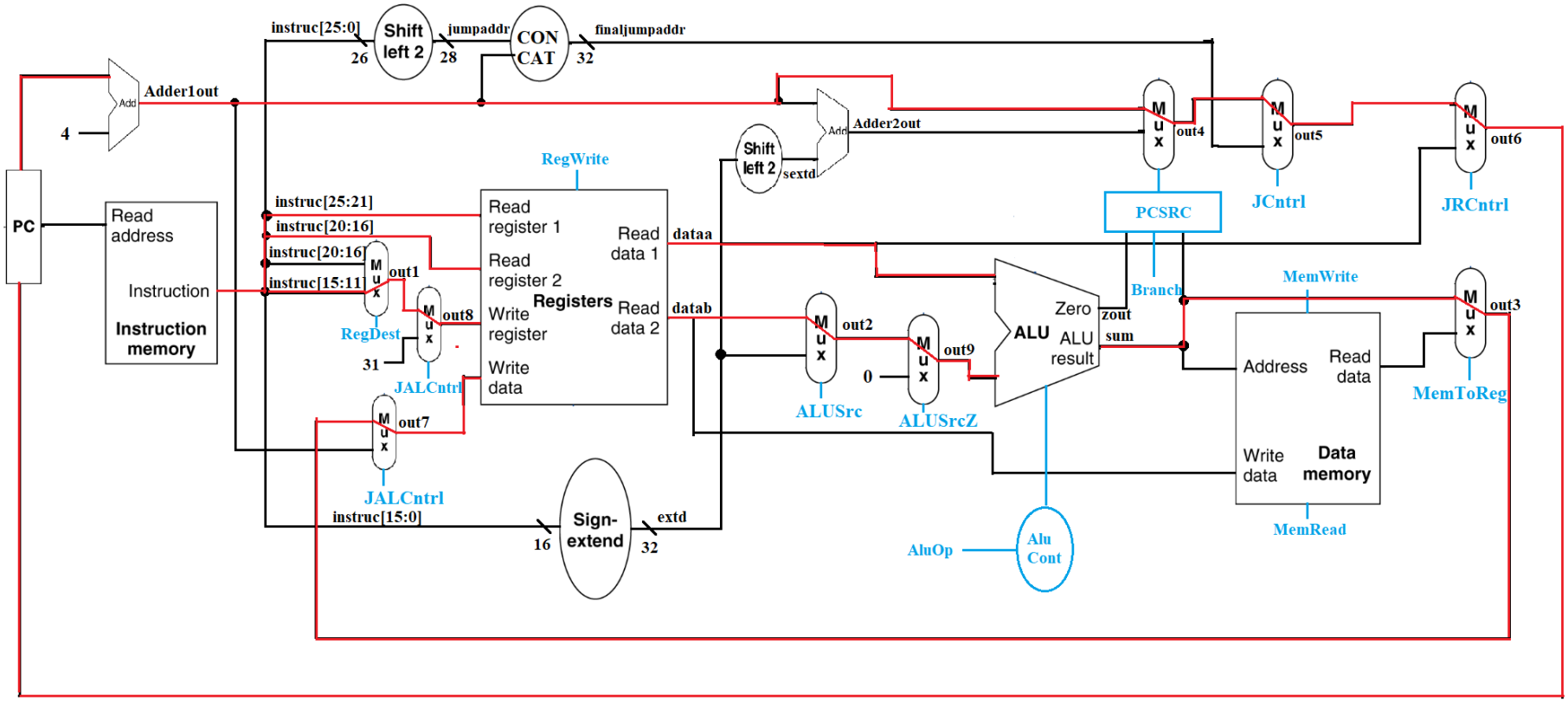
```
  0PC 00000004  SUM ffffffd0   INST 00a33022   REGISTER 00000010 00000030 00000032 00000014
 20PC 00000004  SUM ffffffd0   INST 00a33022   REGISTER 00000010 00000030 ffffffd0 00000014

 40PC 00000008  SUM 00000014   INST 8c240000   REGISTER 00000010 00000030 ffffffd0 00000014
 60PC 00000008  SUM 00000014   INST 8c240000   REGISTER 00000025 00000030 ffffffd0 00000014

 80PC 0000000c  SUM 00000018   INST ac240004   REGISTER 00000025 00000030 ffffffd0 00000014

120PC 00000010  SUM 00000000   INST 1063fffb   REGISTER 00000025 00000030 ffffffd0 00000014

160PC 00000000  SUM 000000a0   INST 00432820   REGISTER 00000025 00000030 ffffffd0 00000014
180PC 00000000  SUM 000000a0   INST 00432820   REGISTER 00000025 000000a0 ffffffd0 00000014

200PC 00000004  SUM 00000040   INST 00a33022   REGISTER 00000025 000000a0 ffffffd0 00000014
220PC 00000004  SUM 00000040   INST 00a33022   REGISTER 00000025 000000a0 00000040 00000014

240PC 00000008  SUM 00000014   INST 8c240000   REGISTER 00000025 000000a0 00000040 00000014

280PC 0000000c  SUM 00000018   INST ac240004   REGISTER 00000025 000000a0 00000040 00000014
```

```
  0PC 00000004  SUM ffffffd0   INST 00a33022   REGISTER R4:00000010 R5:00000030 R6:00000032 R1:00000014 R31:xxxxxxxx
 20PC 00000004  SUM ffffffd0   INST 00a33022   REGISTER R4:00000010 R5:00000030 R6:ffffffd0 R1:00000014 R31:xxxxxxxx

 40PC 00000008  SUM 00000014   INST 8c240000   REGISTER R4:00000010 R5:00000030 R6:ffffffd0 R1:00000014 R31:xxxxxxxx
 60PC 00000008  SUM 00000014   INST 8c240000   REGISTER R4:00000025 R5:00000030 R6:ffffffd0 R1:00000014 R31:xxxxxxxx

 80PC 0000000c  SUM 00000018   INST ac240004   REGISTER R4:00000025 R5:00000030 R6:ffffffd0 R1:00000014 R31:xxxxxxxx

120PC 00000010  SUM 00000000   INST 1063fffb   REGISTER R4:00000025 R5:00000030 R6:ffffffd0 R1:00000014 R31:xxxxxxxx

160PC 00000000  SUM 000000a0   INST 00432820   REGISTER R4:00000025 R5:00000030 R6:ffffffd0 R1:00000014 R31:xxxxxxxx
180PC 00000000  SUM 000000a0   INST 00432820   REGISTER R4:00000025 R5:000000a0 R6:ffffffd0 R1:00000014 R31:xxxxxxxx

200PC 00000004  SUM 00000040   INST 00a33022   REGISTER R4:00000025 R5:000000a0 R6:ffffffd0 R1:00000014 R31:xxxxxxxx
220PC 00000004  SUM 00000040   INST 00a33022   REGISTER R4:00000025 R5:000000a0 R6:00000040 R1:00000014 R31:xxxxxxxx

240PC 00000008  SUM 00000014   INST 8c240000   REGISTER R4:00000025 R5:000000a0 R6:00000040 R1:00000014 R31:xxxxxxxx

280PC 0000000c  SUM 00000018   INST ac240004   REGISTER R4:00000025 R5:000000a0 R6:00000040 R1:00000014 R31:xxxxxxxx
```

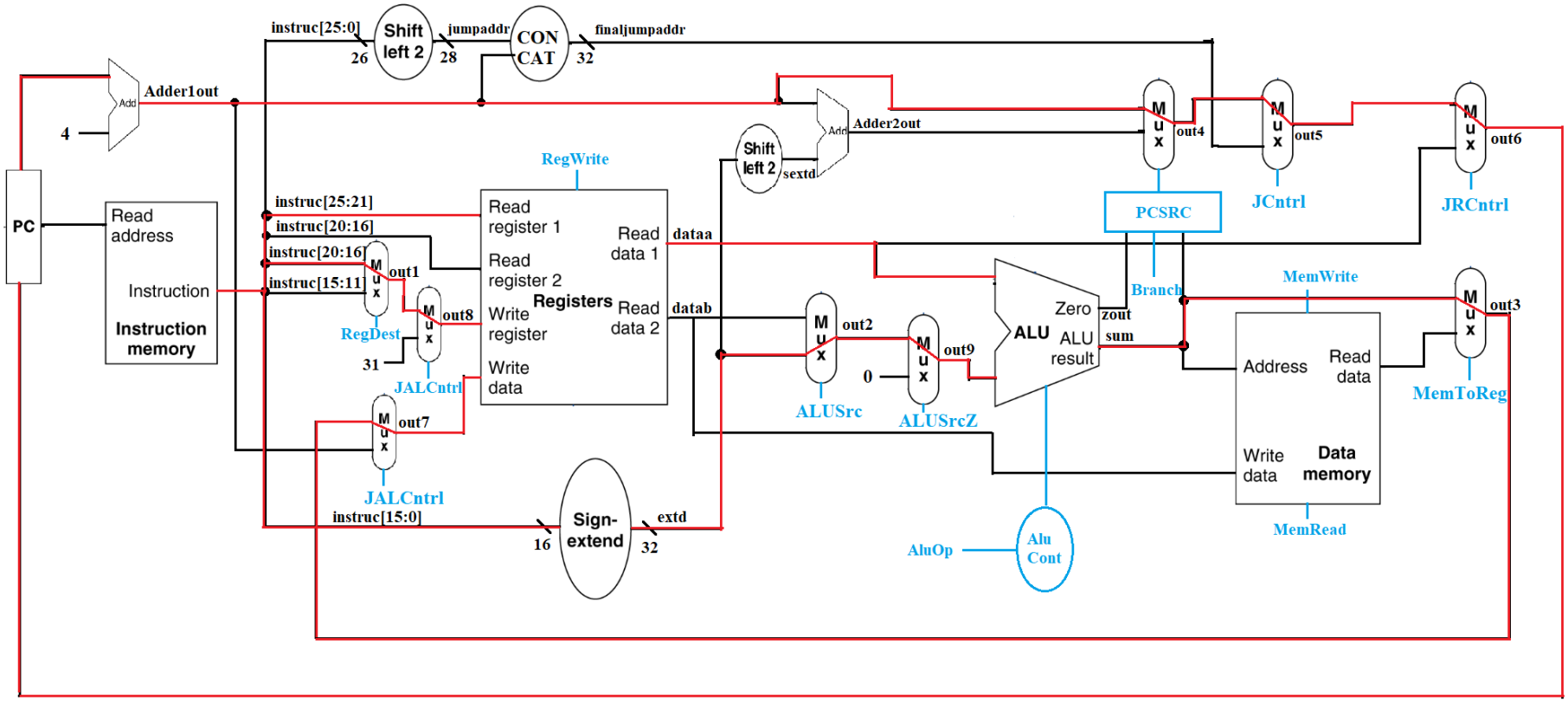
JR Datapath


instruc[25:0]
Shift left 2
26
jumpaddr
28
CON CAT
32
finaljumpaddr
Add
Adder1out
4
Adder2out
Shift left 2
sextd
RegWrite
PC
Read address
Instruction
Instruction memory
instruc[25:21]
instruc[20:16]
instruc[20:16]
instruc[15:11]
Mux
out1
RegDest
31
out8
JALCntrl
out7
JALCntrl
Read register 1
Read register 2
Registers
Write register
Write data
Read data 1
Read data 2
dataa
datab
instruc[15:0]
16
Sign-extend
32
extd
Mux
out2
ALUSrc
0
out9
ALUSrcZ
ALU
Zero
ALU result
zout
sum
AluOp
Alu Cont
Mux
out4
Mux
out5
JCntrl
Mux
out6
JRCntrl
PCSRC
Branch
MemWrite
Address
Read data
Write data
Data memory
MemRead
Mux
out3
MemToReg

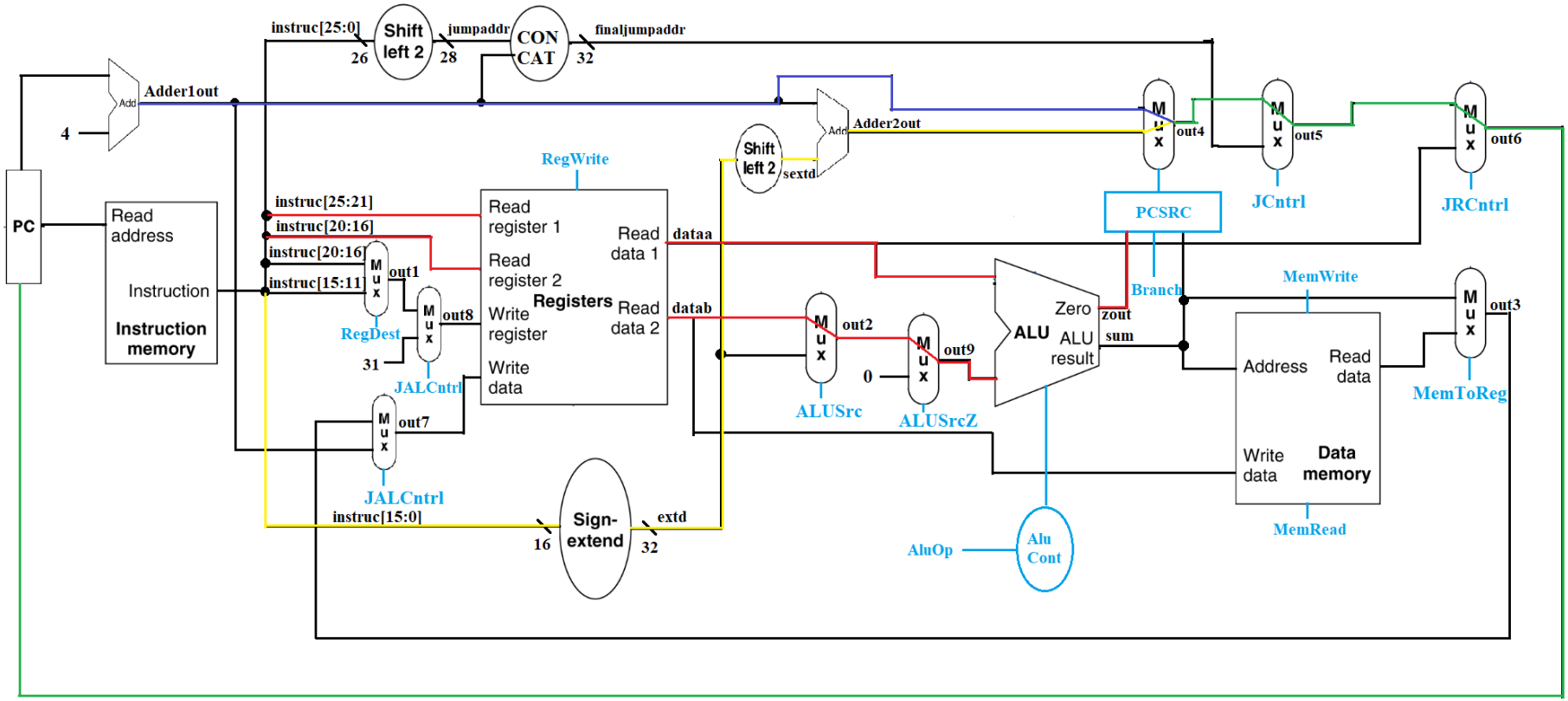
Nor Datapath



Addi,Andi,Ori Datapath

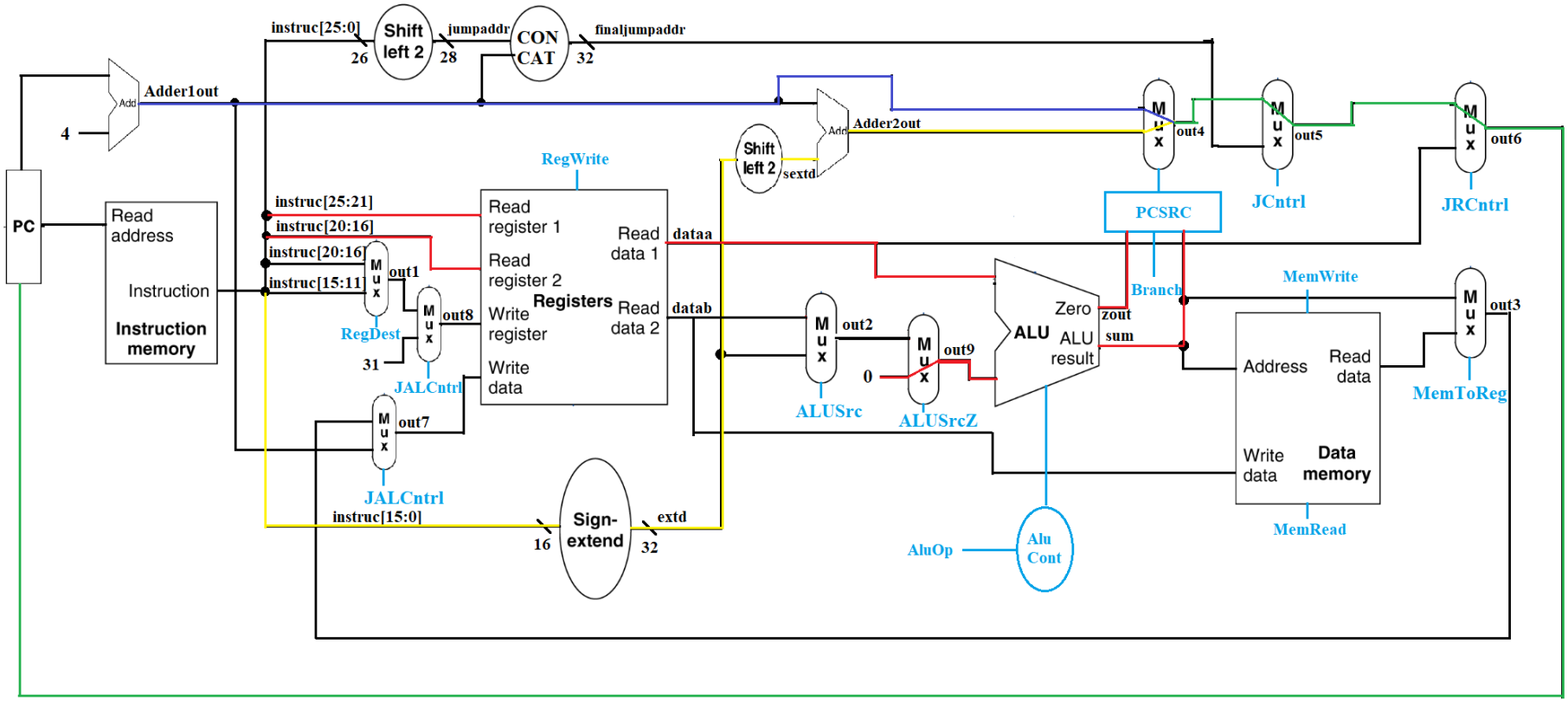


BNEQ Datapath



RedWire:Follows the path to calculate whether bneq is satisfied or not.
GreenWire:If bneq is satisfied, yellow wire continues, otherwise blue wire continues.

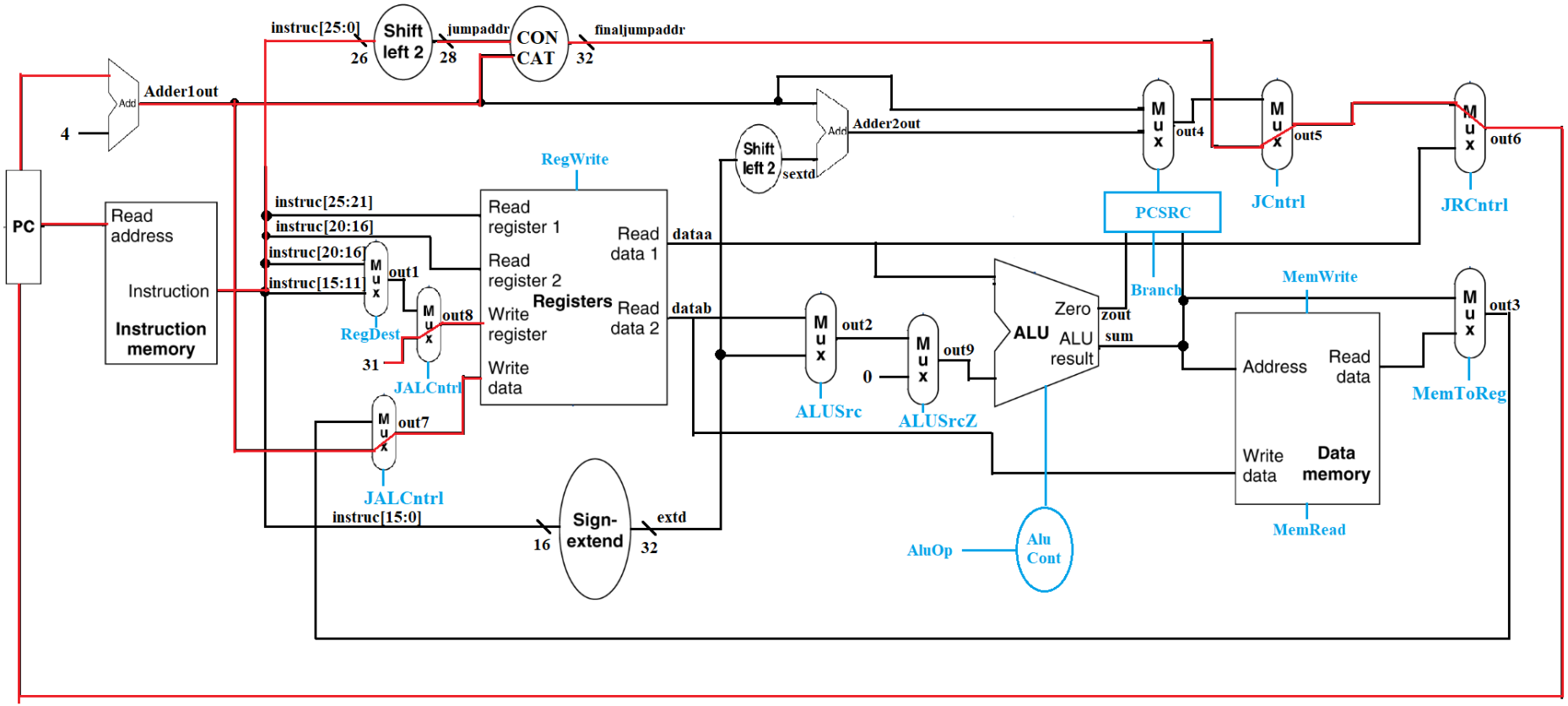
BQEZ,BQLT,BLEZ,BLTZ Datapath



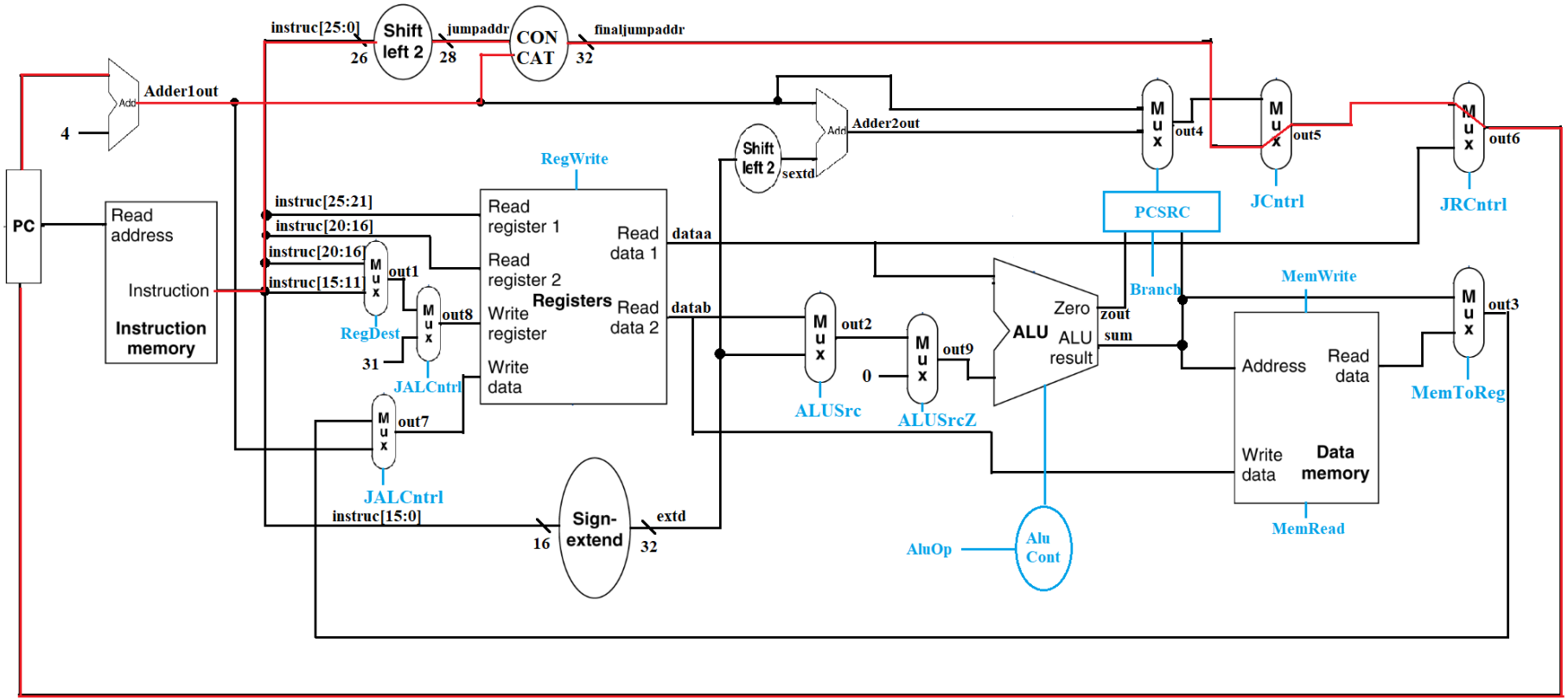
RedWire:Follows the path to calculate whether branch equation is satisfied or not.
GreenWire:If branch equation is satisfied, yellow wire continues, otherwise blue wire continues.

Note: Difference from beq and bneq lies in ALUSrcZ mux, which makes the alu calculation to be made on 0. It also requires the sign bit of the value in the register, which it gets through sum wire.

JAL Datapath



J Datapath


instruc[25:0]
Shift left 2
26
jumpaddr
28
CON CAT
finaljumpaddr
32
Add
Adder1out
4
Adder2out
Shift left 2
sextd
RegWrite
PC
Read address
Instruction
Instruction memory
instruc[25:21]
instruc[20:16]
instruc[20:16]
instruc[15:11]
Mux
out1
RegDest
31
out8
JALCntrl
out7
JALCntrl
Read register 1
Read register 2
Write register
Write data
Registers
Read data 1
Read data 2
dataa
datab
instruc[15:0]
16
Sign-extend
extd
32
out2
ALUSrc
0
out9
ALUSrcZ
ALU
Zero
ALU result
zout
sum
AluOp
Alu Cont
PCSRC
out4
JCntrl
out5
JRCntrl
out6
Branch
MemWrite
Address
Read data
Write data
Data memory
MemRead
out3
MemToReg

Test Runs

Run1:Jal, Jr

#1 Jal 3 -> 0C000003
#2 add $6 $5 $4 -> 00A43020
#3 sub $6 $5 $4 -> 00A43022
#4 jr $31 -> 03E00008

```
  0PC 00000004  SUM 00000000   INST 0c000003   REGISTER R4:00000010 R5:00000030 R6:00000032 R1:00000014 R31:xxxxxxxx
 20PC 00000004  SUM 00000000   INST 0c000003   REGISTER R4:00000010 R5:00000030 R6:00000032 R1:00000014 R31:00000008

 40PC 0000000c  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:00000032 R1:00000014 R31:00000008
 60PC 0000000c  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:00000020 R1:00000014 R31:00000008

 80PC 00000010  SUM 00000008   INST 03e00008   REGISTER R4:00000010 R5:00000030 R6:00000020 R1:00000014 R31:00000008

120PC 00000008  SUM 00000040   INST 00a43020   REGISTER R4:00000010 R5:00000030 R6:00000020 R1:00000014 R31:00000008
140PC 00000008  SUM 00000040   INST 00a43020   REGISTER R4:00000010 R5:00000030 R6:00000040 R1:00000014 R31:00000008

160PC 0000000c  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:00000040 R1:00000014 R31:00000008
180PC 0000000c  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:00000020 R1:00000014 R31:00000008

200PC 00000010  SUM 00000008   INST 03e00008   REGISTER R4:00000010 R5:00000030 R6:00000020 R1:00000014 R31:00000008

240PC 00000008  SUM 00000040   INST 00a43020   REGISTER R4:00000010 R5:00000030 R6:00000020 R1:00000014 R31:00000008
260PC 00000008  SUM 00000040   INST 00a43020   REGISTER R4:00000010 R5:00000030 R6:00000040 R1:00000014 R31:00000008
```

Run2:nor, addi, andi, ori

#1 nor $6 $5 $4 ->00A43027
#2 addi $6 $5 0x20 ->20A60020
#3 andi $6 $5 0x20 ->30A60020 (0x30 -> 110000 0x20-> 100000) and result 100000 = 0x20
#4 ori $6 $5 0x20 ->34A60020 (0x30 -> 110000 0x20-> 100000) or result 110000 = 0x30

```
  0PC 00000004  SUM ffffffcf   INST 00a43027   REGISTER R4:00000010 R5:00000030 R6:00000032 R1:00000014 R31:xxxxxxxx
 20PC 00000004  SUM ffffffcf   INST 00a43027   REGISTER R4:00000010 R5:00000030 R6:ffffffcf R1:00000014 R31:xxxxxxxx

 40PC 00000008  SUM 00000050   INST 20a60020   REGISTER R4:00000010 R5:00000030 R6:ffffffcf R1:00000014 R31:xxxxxxxx
 60PC 00000008  SUM 00000050   INST 20a60020   REGISTER R4:00000010 R5:00000030 R6:00000050 R1:00000014 R31:xxxxxxxx

 80PC 0000000c  SUM 00000020   INST 30a60020   REGISTER R4:00000010 R5:00000030 R6:00000050 R1:00000014 R31:xxxxxxxx
100PC 0000000c  SUM 00000020   INST 30a60020   REGISTER R4:00000010 R5:00000030 R6:00000020 R1:00000014 R31:xxxxxxxx

120PC 00000010  SUM 00000030   INST 34a60020   REGISTER R4:00000010 R5:00000030 R6:00000020 R1:00000014 R31:xxxxxxxx
140PC 00000010  SUM 00000030   INST 34a60020   REGISTER R4:00000010 R5:00000030 R6:00000030 R1:00000014 R31:xxxxxxxx
```

Run3:bne

#1 bne $6 $6 0x2 ->14C60002
#2 bne $6 $5 0x1 ->14C50001
#3 add $6 $5 $4 ->00A43020
#4 sub $6 $5 $4 ->00A43022

```
  0PC 00000004  SUM 00000000   INST 14c60002   REGISTER R4:00000010 R5:00000030 R6:00000032 R1:00000014 R31:xxxxxxxx

 40PC 00000008  SUM 00000002   INST 14c50001   REGISTER R4:00000010 R5:00000030 R6:00000032 R1:00000014 R31:xxxxxxxx

 80PC 00000010  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:00000032 R1:00000014 R31:xxxxxxxx
100PC 00000010  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:00000020 R1:00000014 R31:xxxxxxxx
```

Run4:bgez

#1 bgez $6 0x2 ->04C10002
#2 bgez $0 0x1 ->04010002
#3 add $6 $5 $4 ->00A43020
#4 sub $6 $5 $4 ->00A43022

```
  0PC 00000004  SUM f0000010   INST 04c10002   REGISTER R4:00000010 R5:00000030 R6:f0000010 R1:00000014 R31:xxxxxxxx

 40PC 00000008  SUM 00000000   INST 04010001   REGISTER R4:00000010 R5:00000030 R6:f0000010 R1:00000014 R31:xxxxxxxx

 80PC 00000010  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:f0000010 R1:00000014 R31:xxxxxxxx
100PC 00000010  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:00000020 R1:00000014 R31:xxxxxxxx
```

Run5:bgtz

#1 bgtz $0 0x2 ->1C000002
#2 bgtz $5 0x1 ->1CA00001
#3 add $6 $5 $4 ->00A43020
#4 sub $6 $5 $4 ->00A43022

```
  0PC 00000004  SUM 00000000   INST 1c000002   REGISTER R4:00000010 R5:00000030 R6:f0000010 R1:00000014 R31:xxxxxxxx

 40PC 00000008  SUM 00000030   INST 1ca00001   REGISTER R4:00000010 R5:00000030 R6:f0000010 R1:00000014 R31:xxxxxxxx

 80PC 00000010  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:f0000010 R1:00000014 R31:xxxxxxxx
100PC 00000010  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:00000020 R1:00000014 R31:xxxxxxxx
```

Run6:blez

#1 blez $5 0x2 ->18A00002
#2 blez $0 0x1 ->18000001
#3 add $6 $5 $4 ->00A43020
#4 sub $6 $5 $4 ->00A43022

```
  0PC 00000004  SUM 00000030   INST 18a00002   REGISTER R4:00000010 R5:00000030 R6:f0000010 R1:00000014 R31:xxxxxxxx

 40PC 00000008  SUM 00000000   INST 18000001   REGISTER R4:00000010 R5:00000030 R6:f0000010 R1:00000014 R31:xxxxxxxx

 80PC 00000010  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:f0000010 R1:00000014 R31:xxxxxxxx
100PC 00000010  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:00000020 R1:00000014 R31:xxxxxxxx
```

Run7:bltz

#1 bltz $0 0x2 ->04000002
#2 bltz $6 0x1 ->04C00001
#3 add $6 $5 $4 ->00A43020
#4 sub $6 $5 $4 ->00A43022

```
  0PC 00000004  SUM 00000000   INST 04000002   REGISTER R4:00000010 R5:00000030 R6:f0000010 R1:00000014 R31:xxxxxxxx

 40PC 00000008  SUM f0000010   INST 04c00001   REGISTER R4:00000010 R5:00000030 R6:f0000010 R1:00000014 R31:xxxxxxxx

 80PC 00000010  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:f0000010 R1:00000014 R31:xxxxxxxx
100PC 00000010  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:00000020 R1:00000014 R31:xxxxxxxx
```

Run8:j

#1 j 0x3 ->08000003
#2 add $6 $5 $4 ->00A43020
#3 sub $6 $5 $4 ->00A43022

```
  0PC 00000004  SUM 00000000   INST 08000003   REGISTER R4:00000010 R5:00000030 R6:f0000010 R1:00000014 R31:xxxxxxxx

 40PC 0000000c  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:f0000010 R1:00000014 R31:xxxxxxxx
 60PC 0000000c  SUM 00000020   INST 00a43022   REGISTER R4:00000010 R5:00000030 R6:00000020 R1:00000014 R31:xxxxxxxx
```