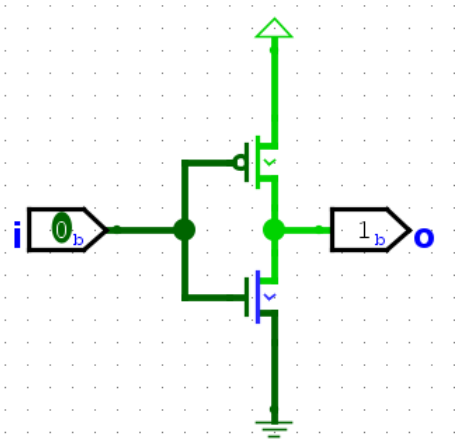
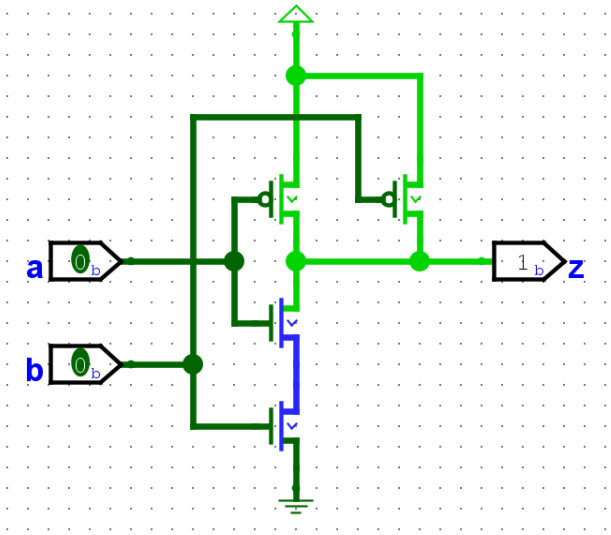
Xetroc 1.0

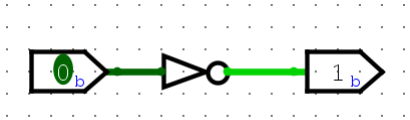
F.B.

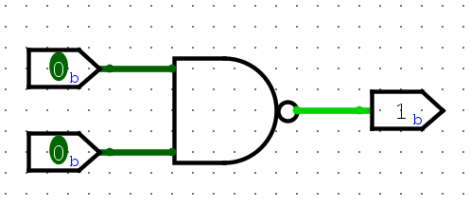
From 0 and 1 to CPU

Inverter & NAND Gate

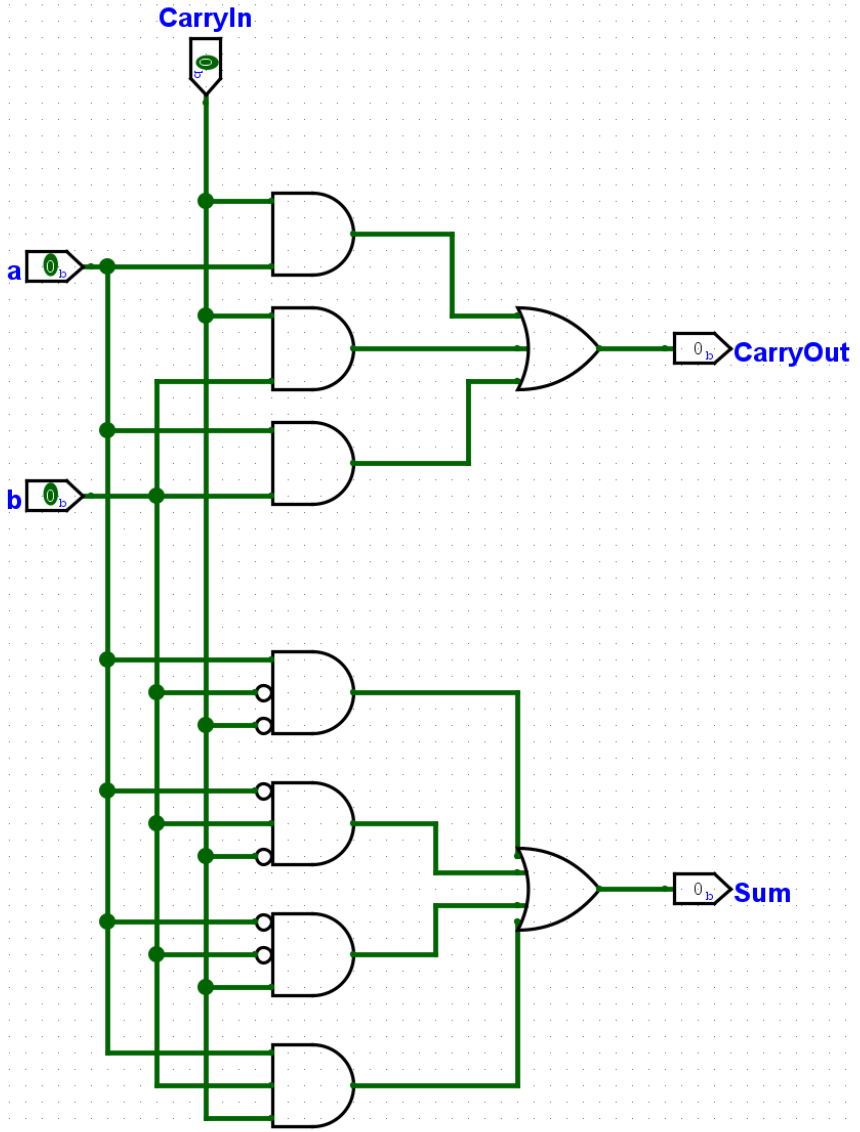


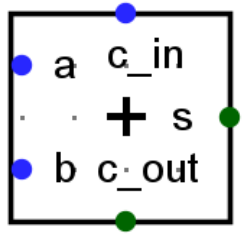






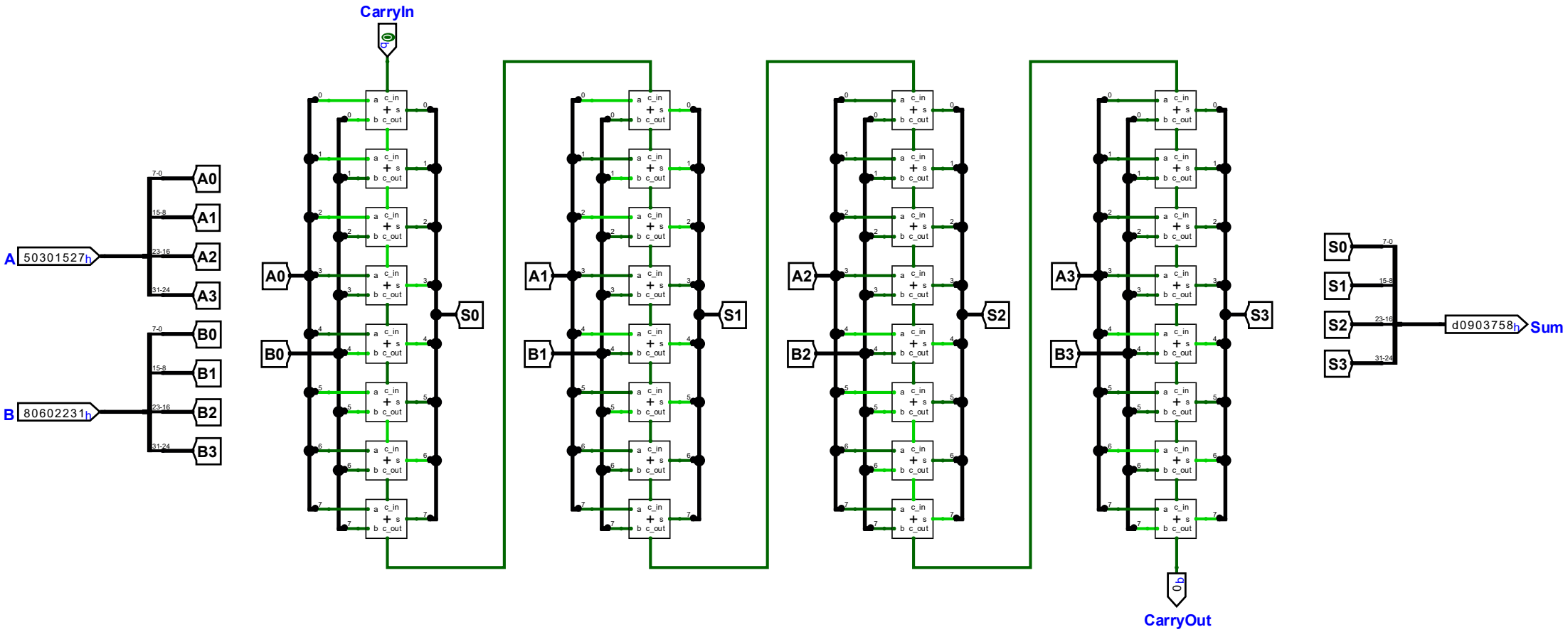
1-Bit Adder





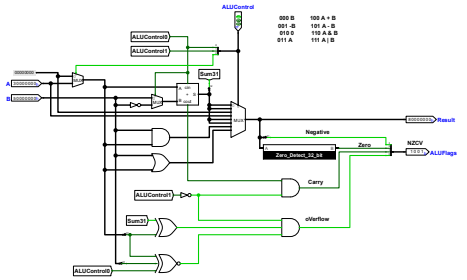
CarryIn	a	b	CarryOut	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

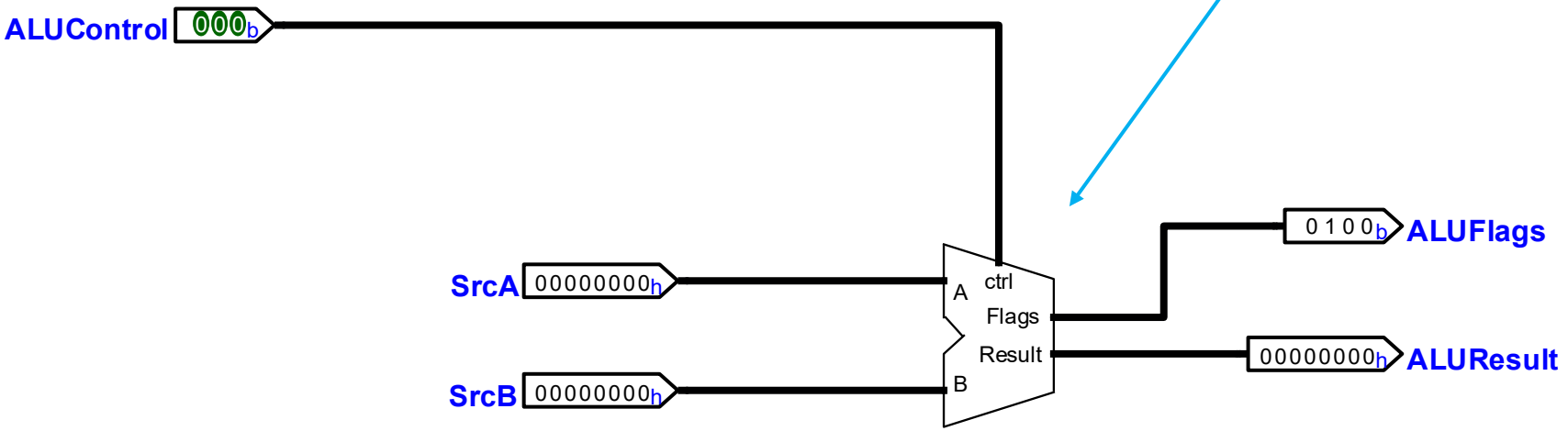
32-Bit Adder (Carry Ripple Adder)



ALU

ALUControl			
0	0	0	B
0	0	1	-B
0	1	0	0
0	1	1	A
1	0	0	A + B
1	0	1	A – B
1	1	0	A & B
1	1	1	A | B

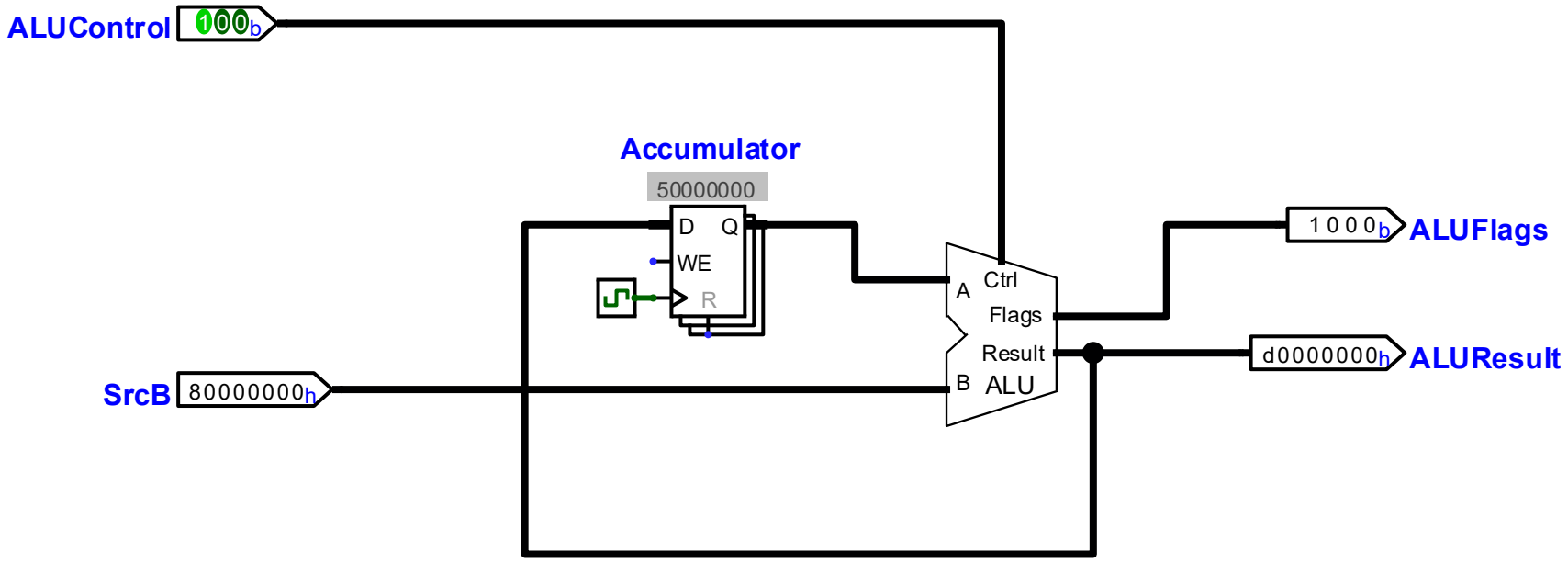




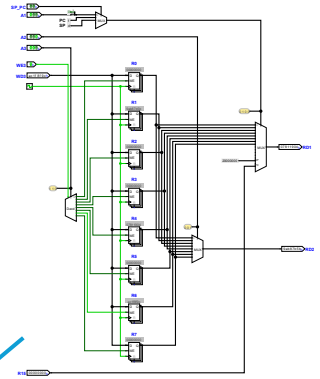
ALU with Accumulator

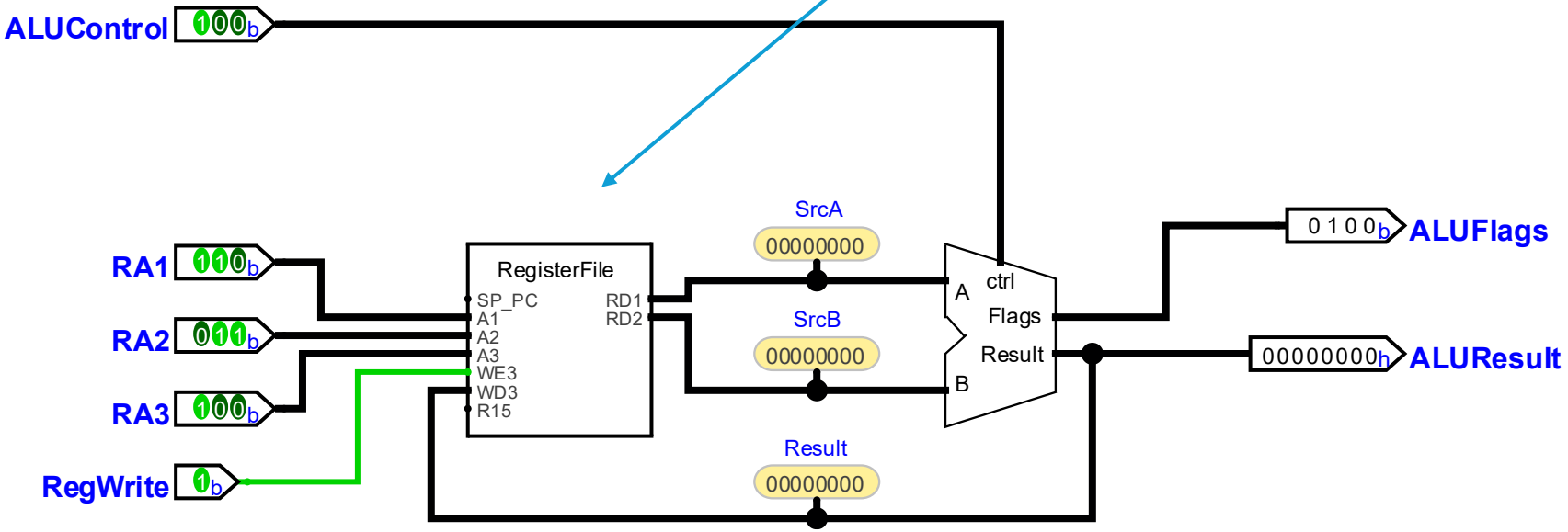
D-Flipflop (MS)
(used for all registers)

D	WE	R	CLK	Q'	
0	0	0	x	Q	no change
1	0	0	x	Q	no change
x	x	1	x	0	static reset
0	1	0	↑	0	edge triggered write
1	1	0	↑	1	edge triggered write



ALU with Register File

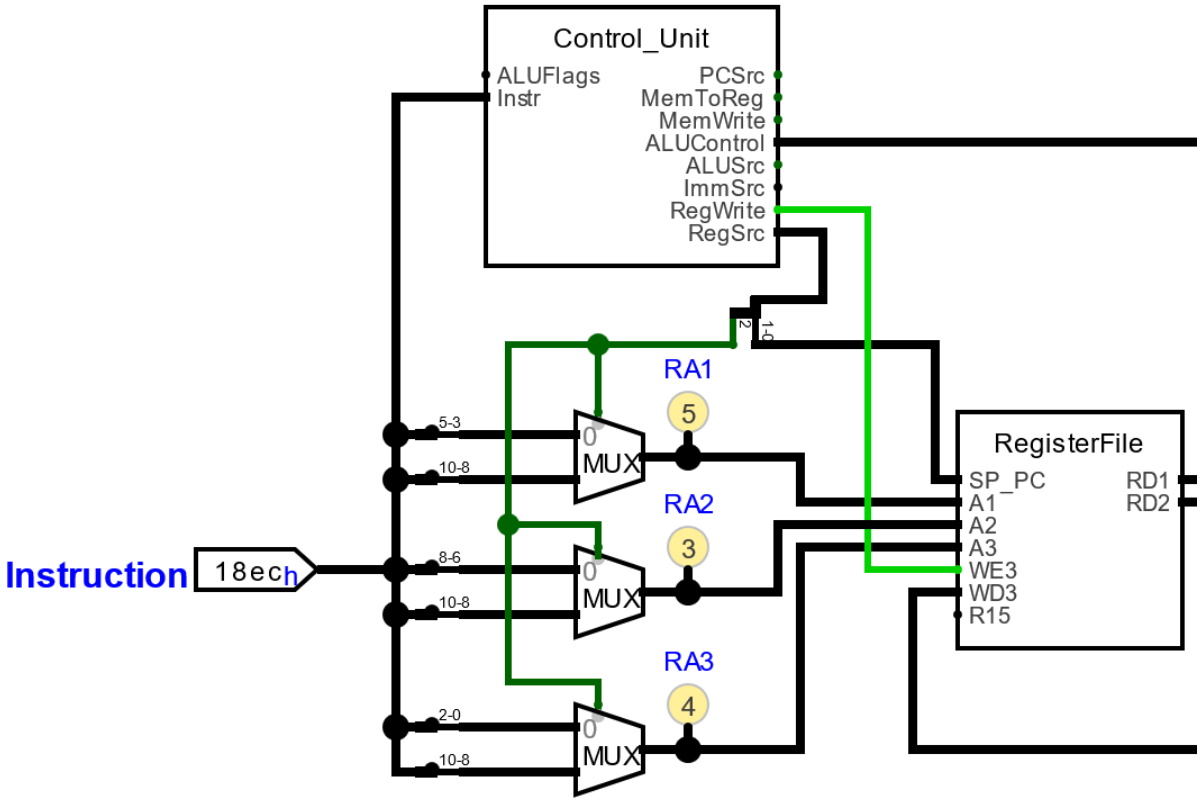




Instruction Encoding

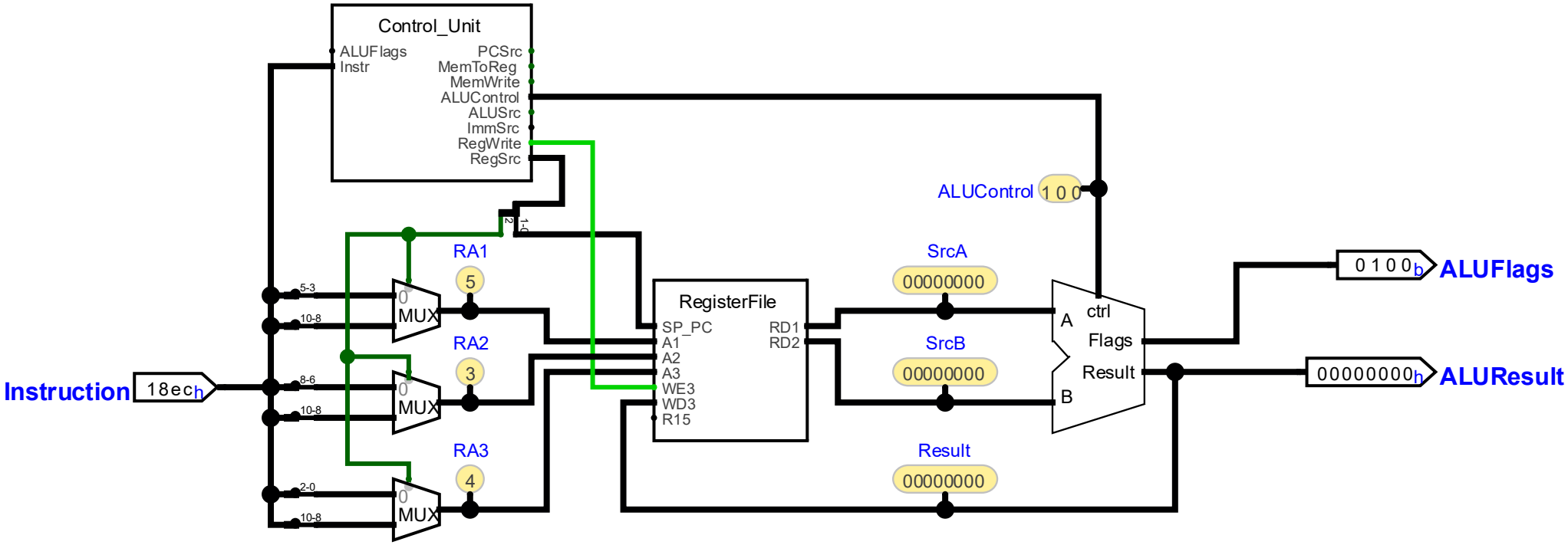
RegSrc			
	0	0	use RA1 for A1
	0	1	use PC for A1
	1	0	use SP for A1
0			use individual bits for RAi
1			use bits 10..8 for all RAi

instruction encoding 15 … 8 … 0																assembler
0	0	0	0	0	0	0	0	0	0	n	n	n	d	d	d	MOVS <Rd>,<Rn>
0	0	0	1	1	0	0	m	m	m	n	n	n	d	d	d	ADDS <Rd>,<Rn>,<Rm>
0	0	0	1	1	0	1	m	m	m	n	n	n	d	d	d	SUBS <Rd>,<Rn>,<Rm>
0	0	1	0	0	d	d	d	i	i	i	i	i	i	i	i	MOVS <Rd>, #<imm8>
0	0	1	1	0	dn	dn	dn	i	i	i	i	i	i	i	i	ADDS <Rdn>,#<imm8>
0	0	1	1	1	dn	dn	dn	i	i	i	i	i	i	i	i	SUBS <Rdn>,#<imm8>
1	0	0	1	1	t	t	t	i	i	i	i	i	i	i	i	LDR <Rt>,[<SP>,#<imm8>]
1	0	0	1	0	t	t	t	i	i	i	i	i	i	i	i	STR <Rt>,[<SP>,#<imm8>]
1	1	0	1	c	c	c	c	i	i	i	i	i	i	i	i	Bcc #<simm8>
1	1	1	0	0	i	i	i	i	i	i	i	i	i	i	i	B #<simm11>



Instruction Decoder & Control Unit

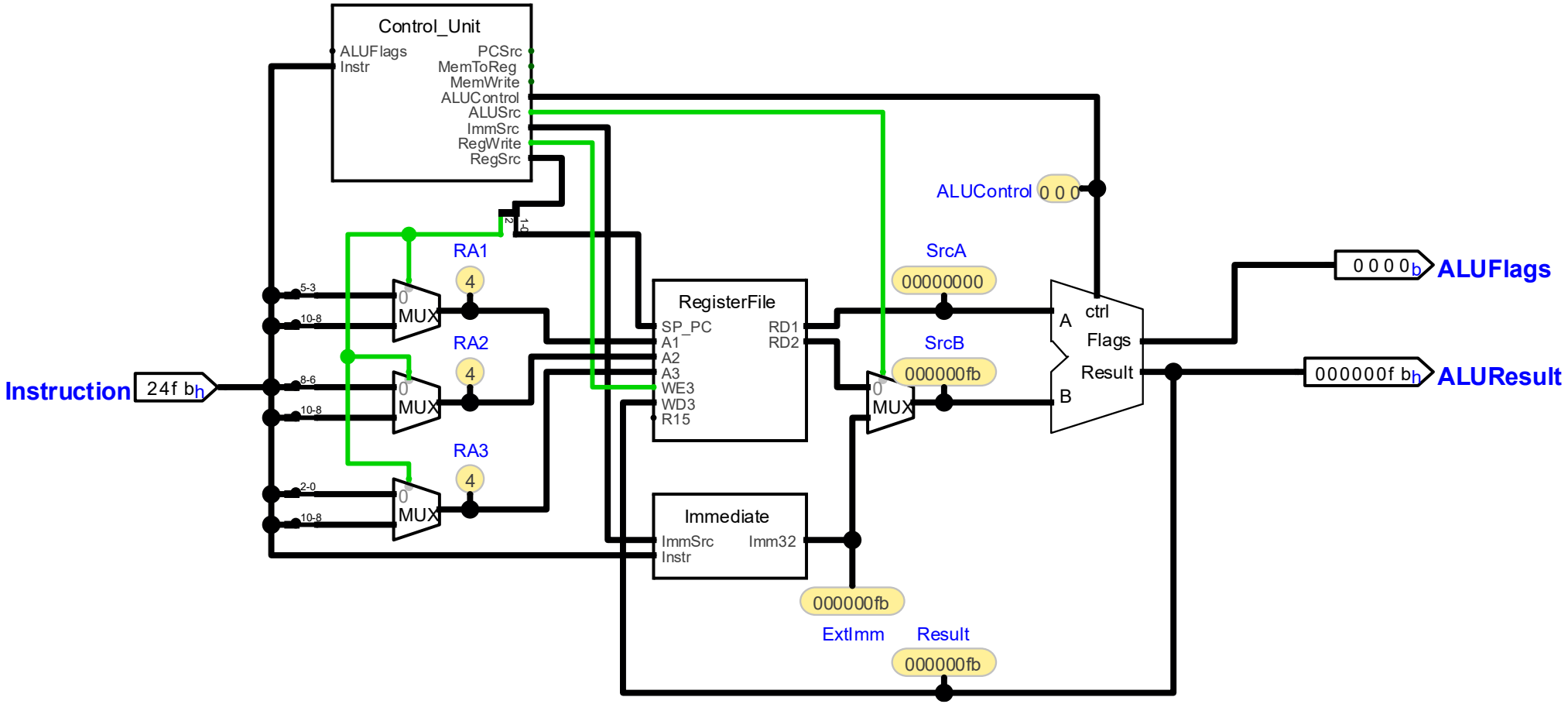
0	0	0	0	0	0	0	0	0	0	n	n	n	d	d	d	MOVS <Rd>,<Rn>
0	0	0	1	1	0	0	m	m	m	n	n	n	d	d	d	ADDS <Rd>,<Rn>,<Rm>
0	0	0	1	1	0	1	m	m	m	n	n	n	d	d	d	SUBS <Rd>,<Rn>,<Rm>



Immediate Operands

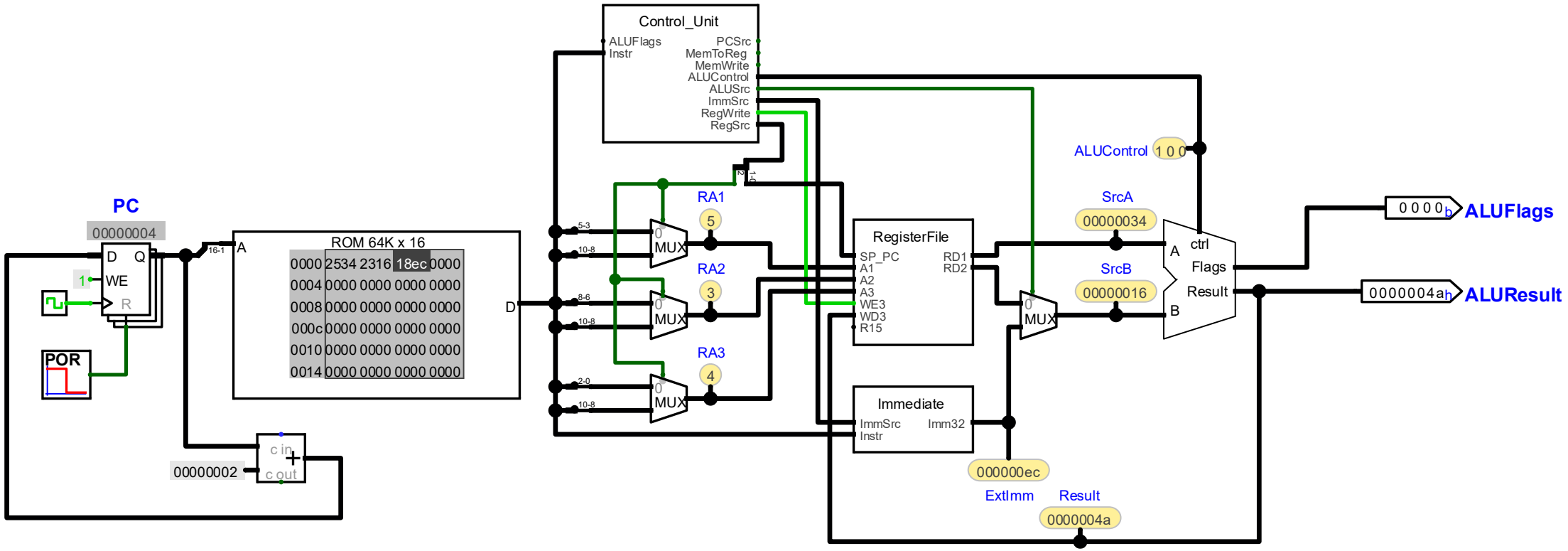
0	0	1	0	0	d	d	d	i	i	i	i	i	i	i	i	MOVS <Rd>, #<imm8>
0	0	1	1	0	dn	dn	dn	i	i	i	i	i	i	i	i	ADDS <Rdn>,#<imm8>
0	0	1	1	1	dn	dn	dn	i	i	i	i	i	i	i	i	SUBS <Rdn>,#<imm8>

ImmSrc		
	0 0	imm8
	0 1	imm8 << 2
	1 0	simm8 << 1
	1 1	simm11 << 1



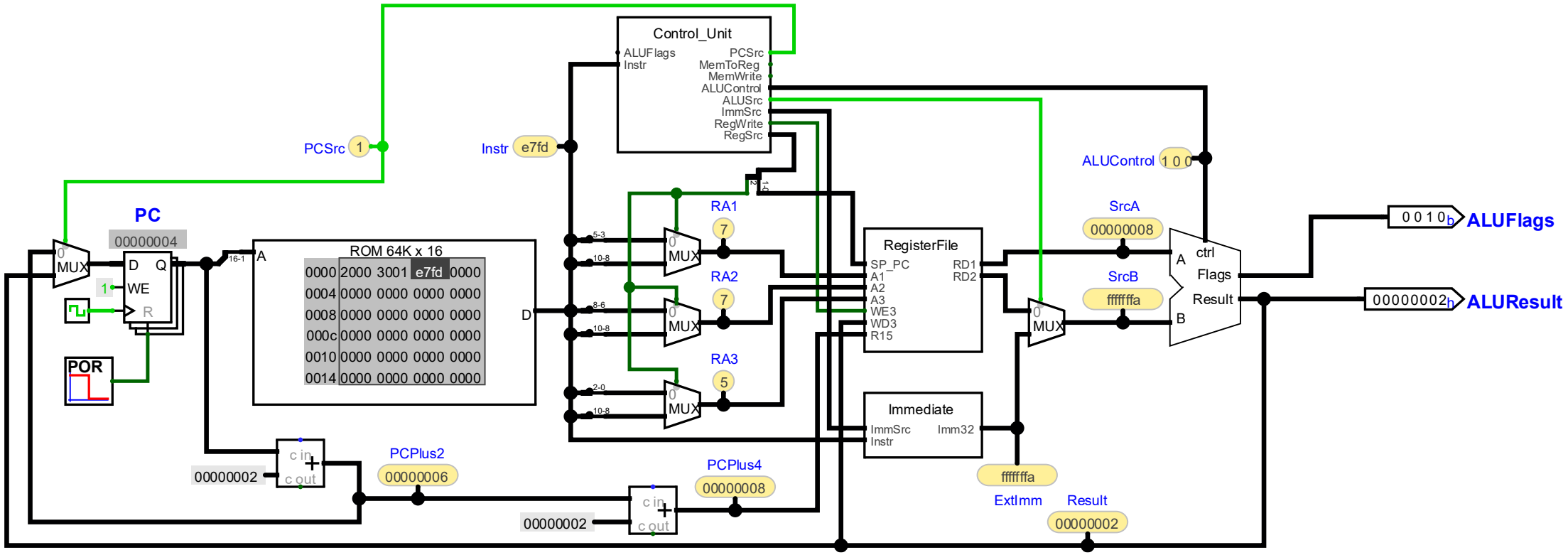
Instruction Memory

```
ROM | A   D          |
    | a   ROM[a]     | static read
```



Branch

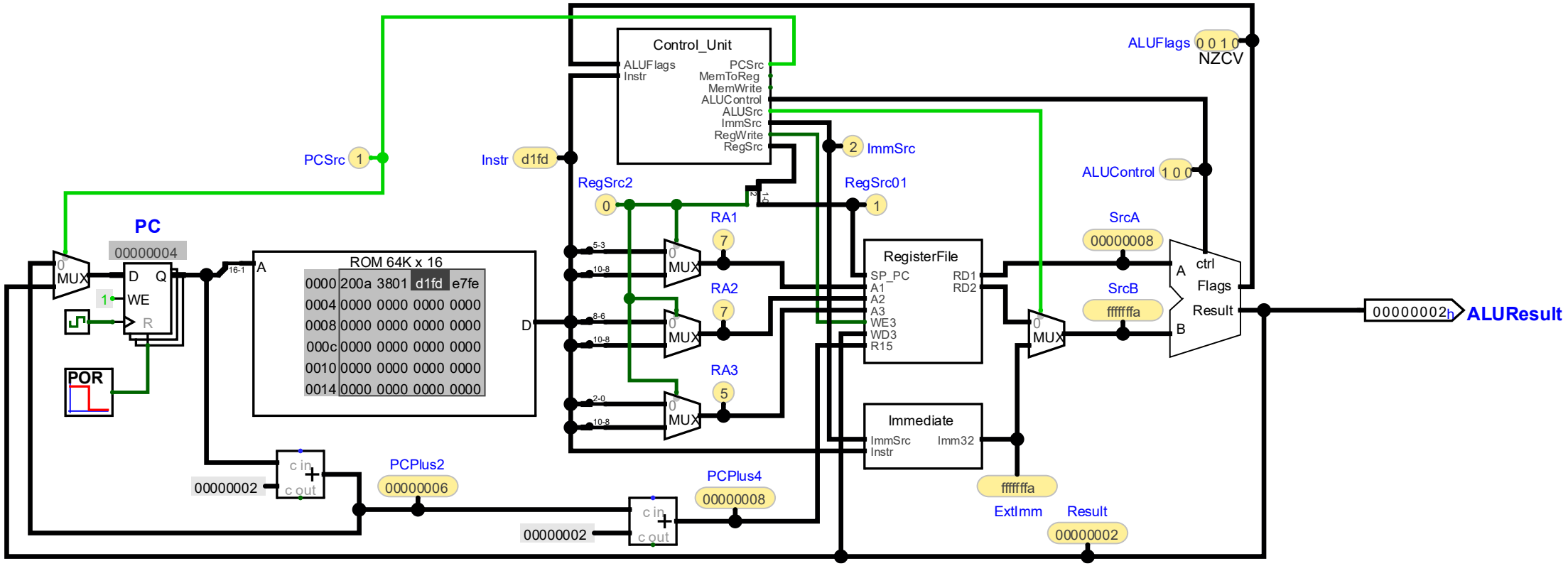
1 1 1 0 0	i i i i i i i i i i i



Conditional Branch

1	1	0	1	c	c	c	c	i	i	i	i	i	i	i	i

Code	Condition	Meaning	Status of Flags
0 0 0 0	EQ	Equal	Z==1
0 0 0 1	NE	Not Equal	Z==0
0 0 1 0	CS or HS	Unsigned Higher or Same (or Carry Set)	C==1
0 0 1 1	CC or LO	Unsigned Lower (or Carry Clear)	C==0
0 1 0 0	MI	Negative (or Minus)	N==1
0 1 0 1	PL	Positive (or Plus)	N==0
0 1 1 0	VS	Signed Overflow	V==1
0 1 1 1	VC	No signed Overflow	V==0
1 0 0 0	HI	Unsigned Higher	(C==1) && (Z!=0)
1 0 0 1	LS	Unsigned Lower or same	(C==0) || (Z==0)
1 0 1 0	GE	Signed Greater Than or Equal	N==V
1 0 1 1	LT	Signed Less Than	N!=V
1 1 0 0	GT	Signed Greater Than	(Z==0) && (N==V)
1 1 0 1	LE	Signed Less Than or Equal	(Z==1) || (N!=V)
1 1 1 0	AL	Always executed	true
1 1 1 1	NV	Never executed	false

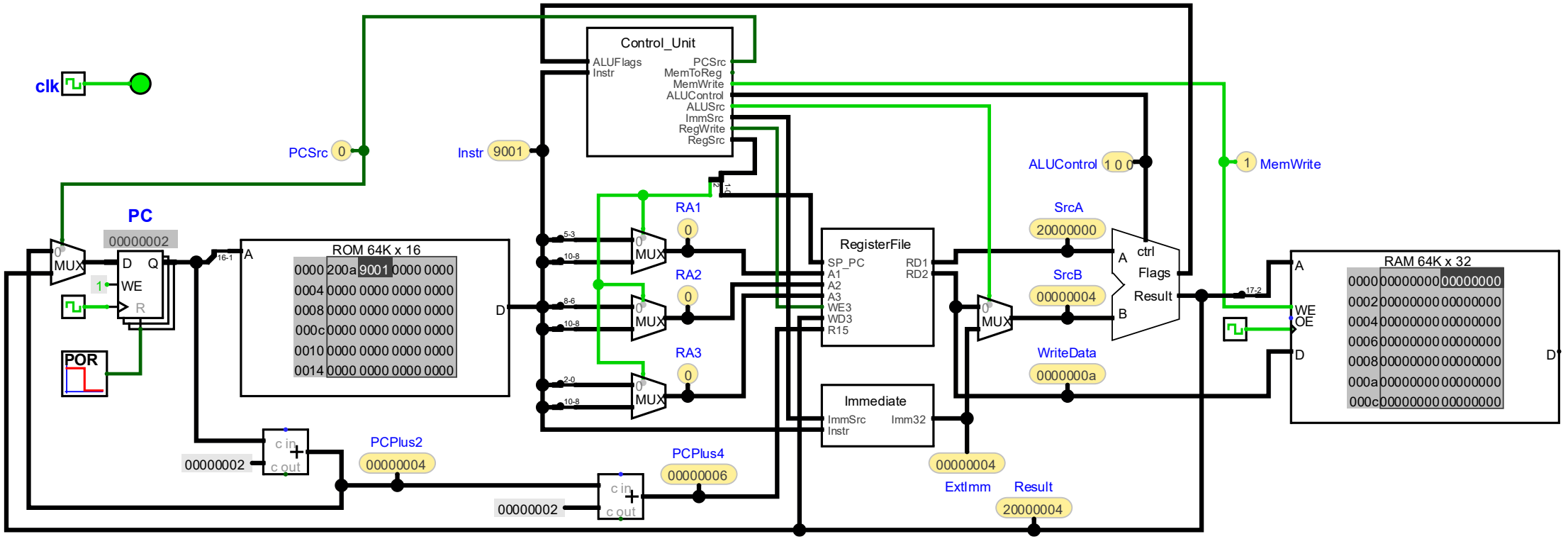


Store to Data Memory

1 0 0 1 0	t t t	i i i i i i i i

RAM

C	WE	OE	A	Din	Dout	
x	x	1	a		RAM[a]	static read
↑	1	x	a	RAM[a] = Din		rising edge triggered write

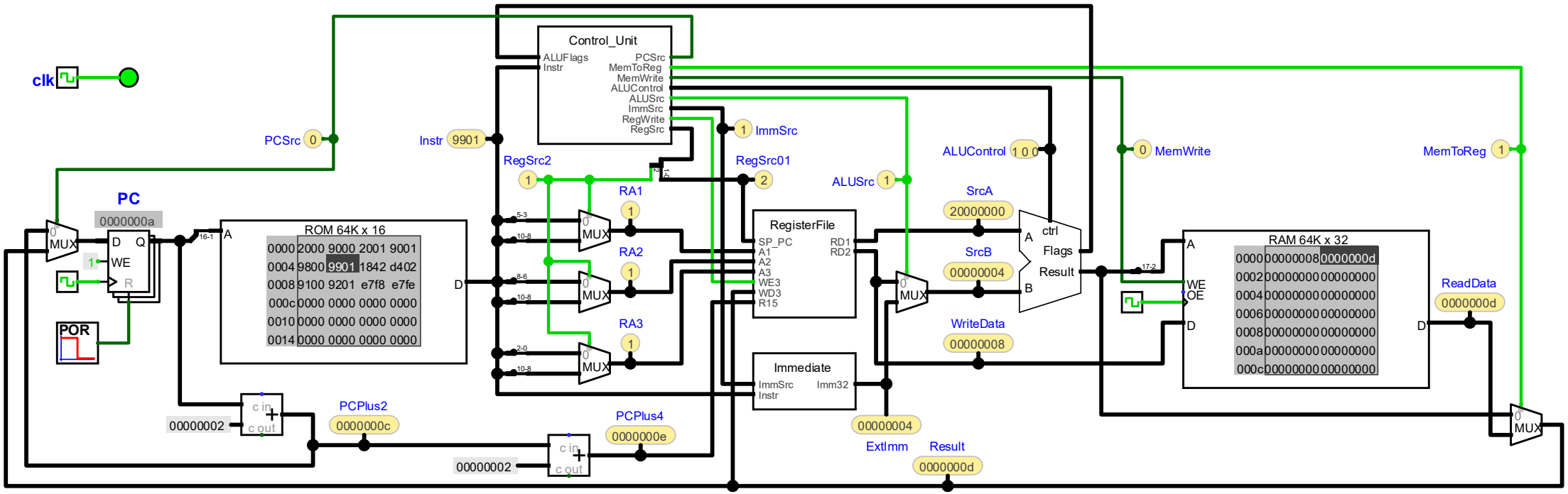


Load from Data Memory

1	0	0	1	1	t	t	t	i	i	i	i	i	i	i	i

RAM

C	WE	OE	A	Din	Dout	
x	x	1	a		RAM[a]	static read
↑	1	x	a	RAM[a] = Din		rising edge triggered write



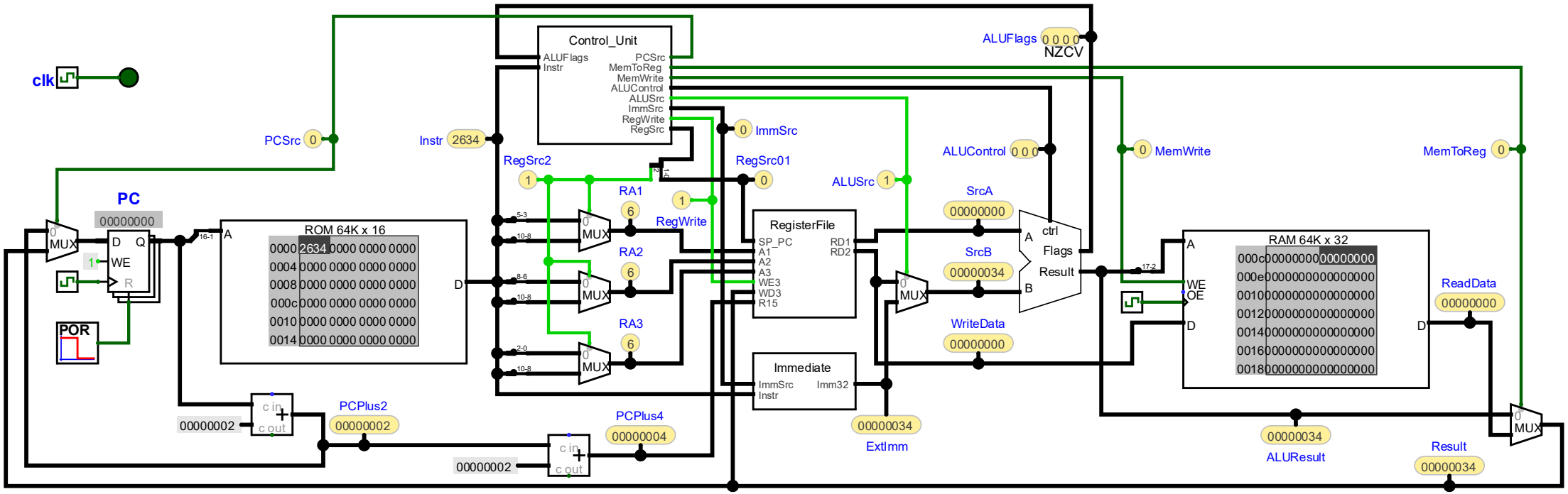
Instructions

MOVS <Rd>, #<imm8>

00 10 0 Rd. imm8

ImmSrc	0	0	imm8
	0	1	imm8 << 2
	1	0	simm8 << 1
	1	1	simm11 << 1

ALUControl	0	0	0	B
	0	0	1	-B
	0	1	0	0
	0	1	1	A
	1	0	0	A+B
	1	0	1	A-B
	1	1	0	A&B
	1	1	1	A|B

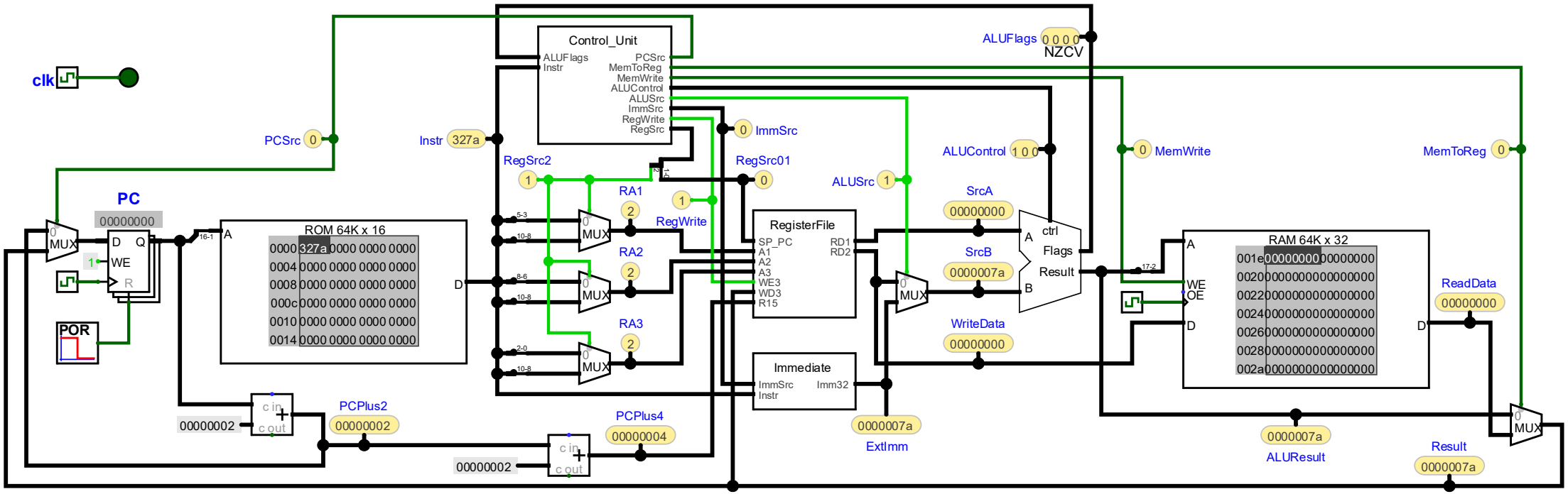


ADDS <Rdn>, #<imm8>

`00 11 0 Rdn ....imm8`

ImmSrc	0	0	imm8	
	0	1	imm8 << 2	
	1	0	simm8 << 1	
	1	1	simm11 << 1	

ALUControl	0	0	0	B
	0	0	1	-B
	0	1	0	0
	0	1	1	A
	1	0	0	A+B
	1	0	1	A-B
	1	1	0	A&B
	1	1	1	A|B

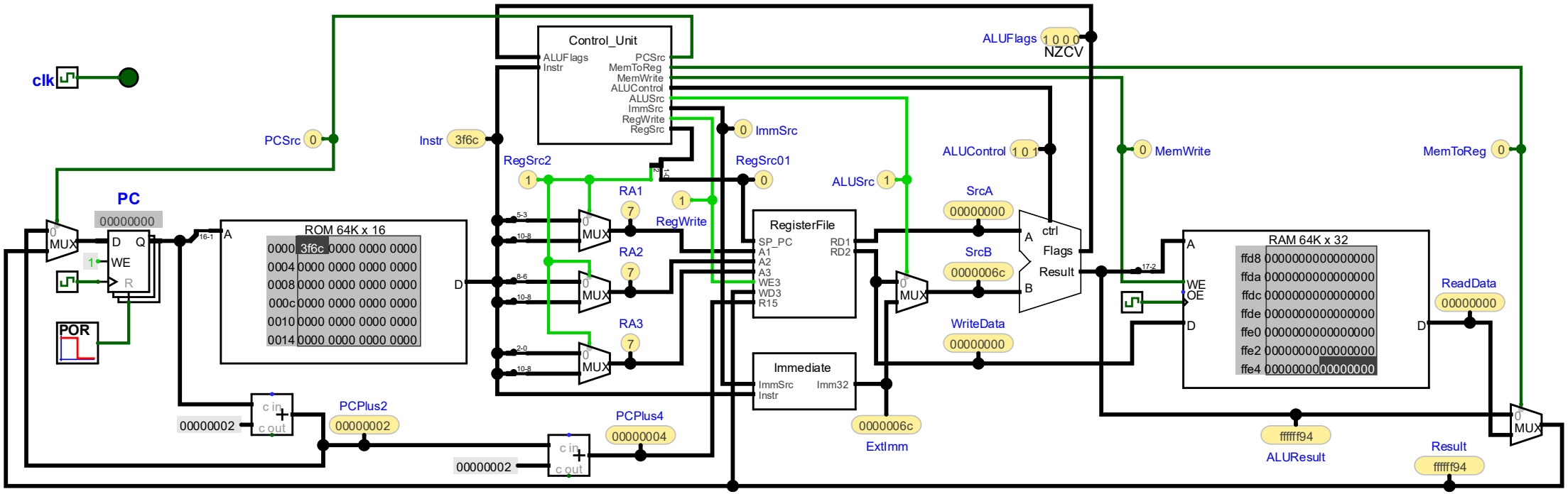


SUBS <Rdn>, #<imm8>

00 11 1 Rdn imm8

ImmSrc	0	0	imm8	
	0	1	imm8 << 2	
	1	0	simm8 << 1	
	1	1	simm11 << 1	

ALUControl	0	0	0	B
	0	0	1	-B
	0	1	0	0
	0	1	1	A
	1	0	0	A+B
	1	0	1	A-B
	1	1	0	A&B
	1	1	1	A|B

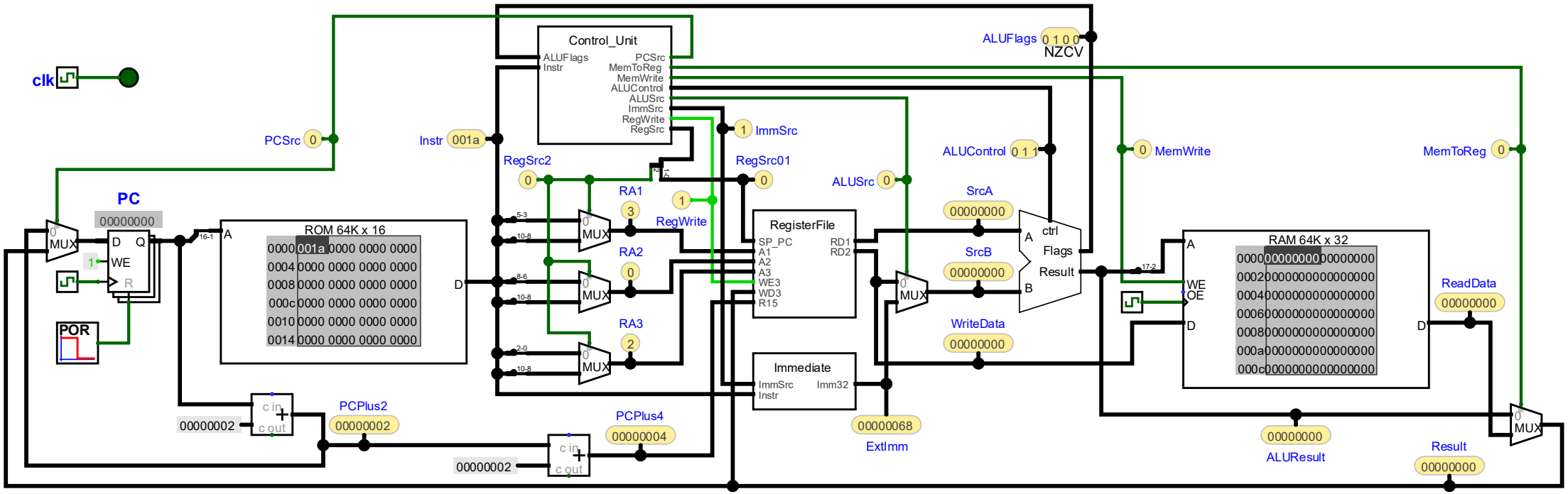


MOVS <Rd>,<Rm>

00 00 0 00000 Rm. Rd.

ImmSrc	0	0	imm8	
	0	1	imm8 << 2	
	1	0	simm8 << 1	
	1	1	simm11 << 1	

ALUControl	0	0	0	B
	0	0	1	-B
	0	1	0	0
	0	1	1	A
	1	0	0	A+B
	1	0	1	A-B
	1	1	0	A&B
	1	1	1	A|B

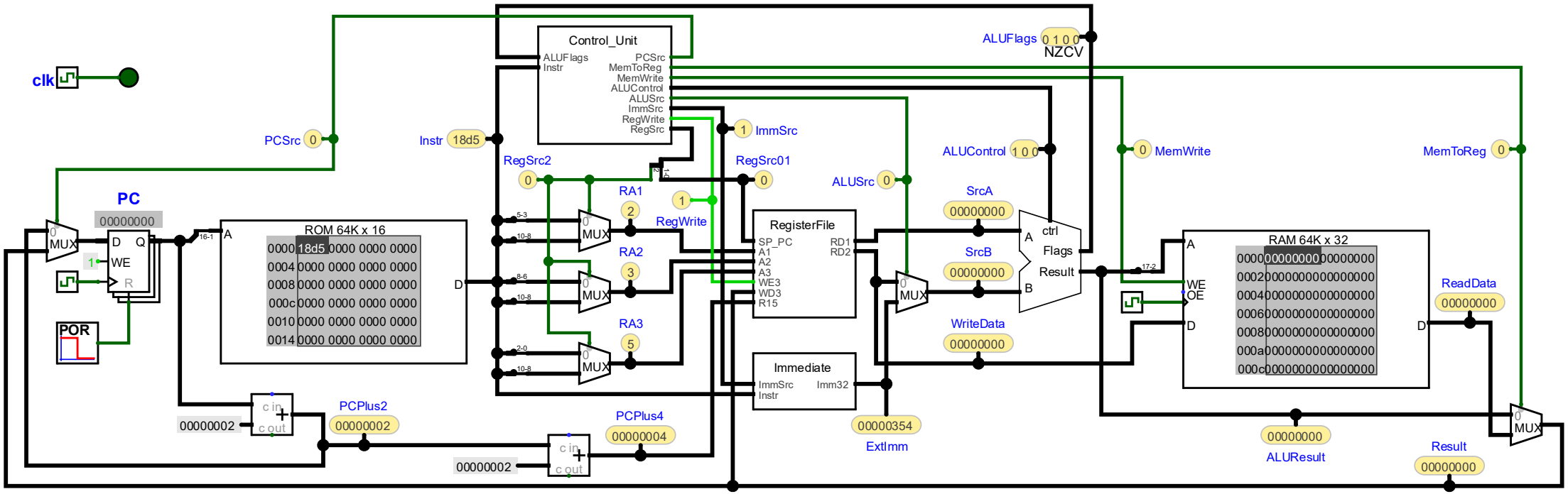


ADDS <Rd>,<Rn>,<Rm>

00 01 1 00 Rm. Rn. Rd.

ImmSrc	0	0	imm8
	0	1	imm8 << 2
	1	0	simm8 << 1
	1	1	simm11 << 1

ALUControl	0	0	0	B
	0	0	1	-B
	0	1	0	0
	0	1	1	A
	1	0	0	A+B
	1	0	1	A-B
	1	1	0	A&B
	1	1	1	A|B

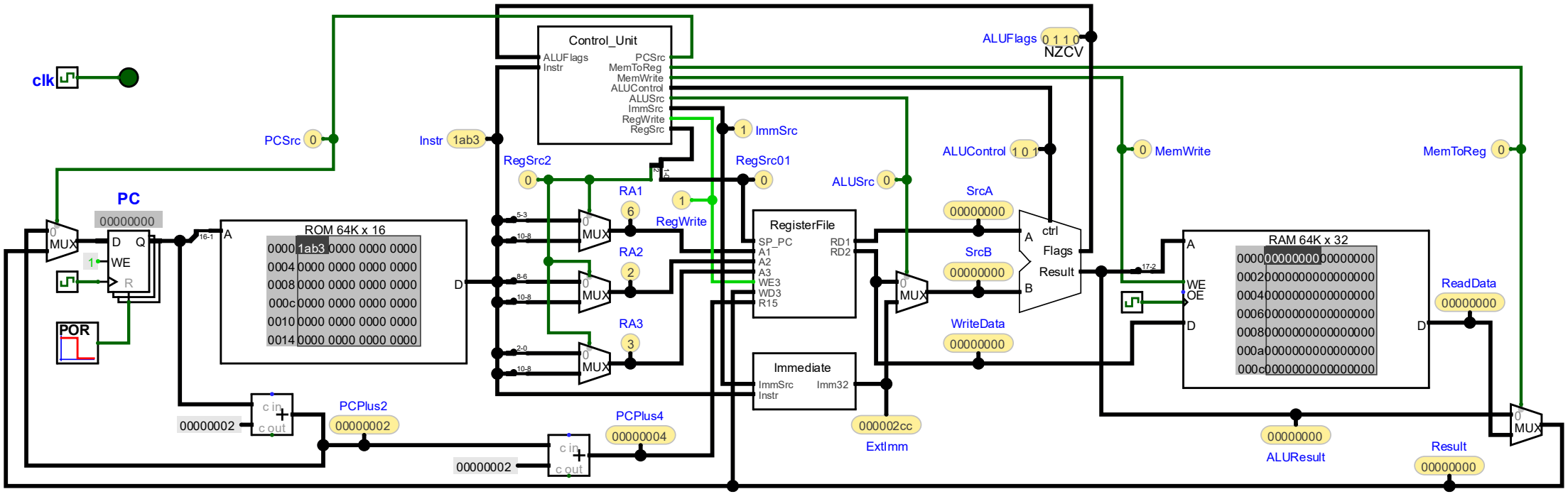


SUBS <Rd>,<Rn>,<Rm>

00 01 1 01 Rm. Rn. Rd.

ImmSrc	0	0	imm8
	0	1	imm8 << 2
	1	0	simm8 << 1
	1	1	simm11 << 1

ALUControl	0	0	0	B
	0	0	1	-B
	0	1	0	0
	0	1	1	A
	1	0	0	A+B
	1	0	1	A-B
	1	1	0	A&B
	1	1	1	A|B

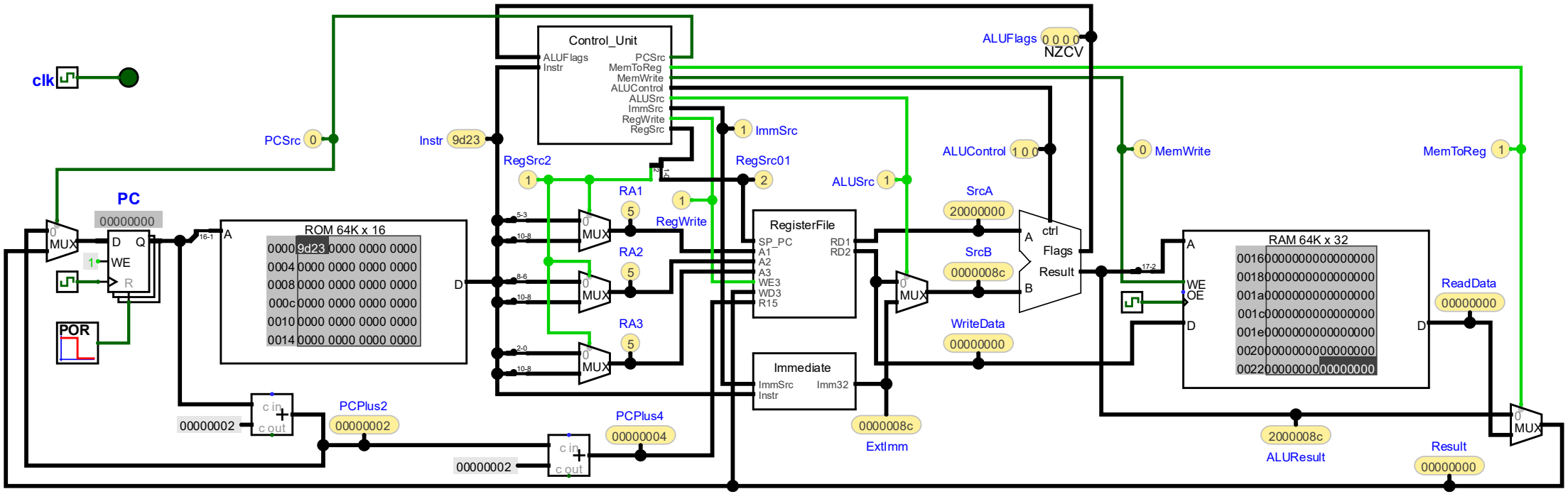


LDR <Rt>, [SP, #<imm8>]

10 01 1 Rt. imm8

ImmSrc	0	0	imm8
	0	1	imm8 << 2
	1	0	simm8 << 1
	1	1	simm11 << 1

ALUControl	0	0	0	B
	0	0	1	-B
	0	1	0	0
	0	1	1	A
	1	0	0	A+B
	1	0	1	A-B
	1	1	0	A&B
	1	1	1	A|B

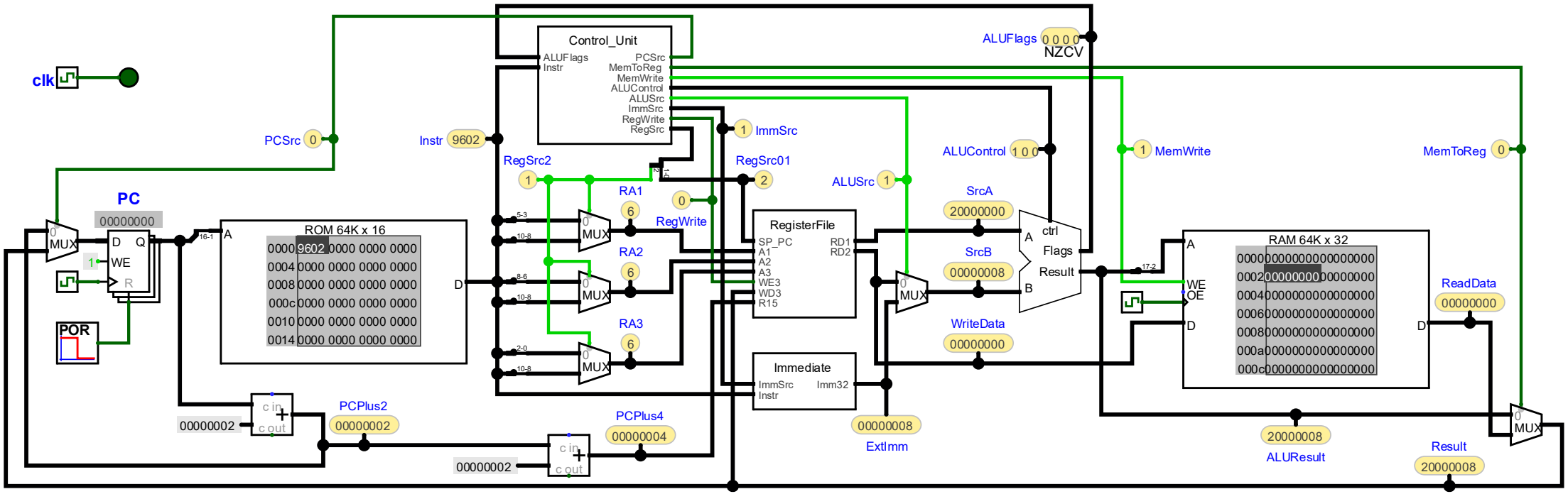


STR <Rt>, [SP, #<imm8>]

`10 01 0 Rt. ....imm8`

ImmSrc	0	0	imm8
	0	1	imm8 << 2
	1	0	simm8 << 1
	1	1	simm11 << 1

ALUControl	0	0	0	B
	0	0	1	-B
	0	1	0	0
	0	1	1	A
	1	0	0	A+B
	1	0	1	A-B
	1	1	0	A&B
	1	1	1	A|B

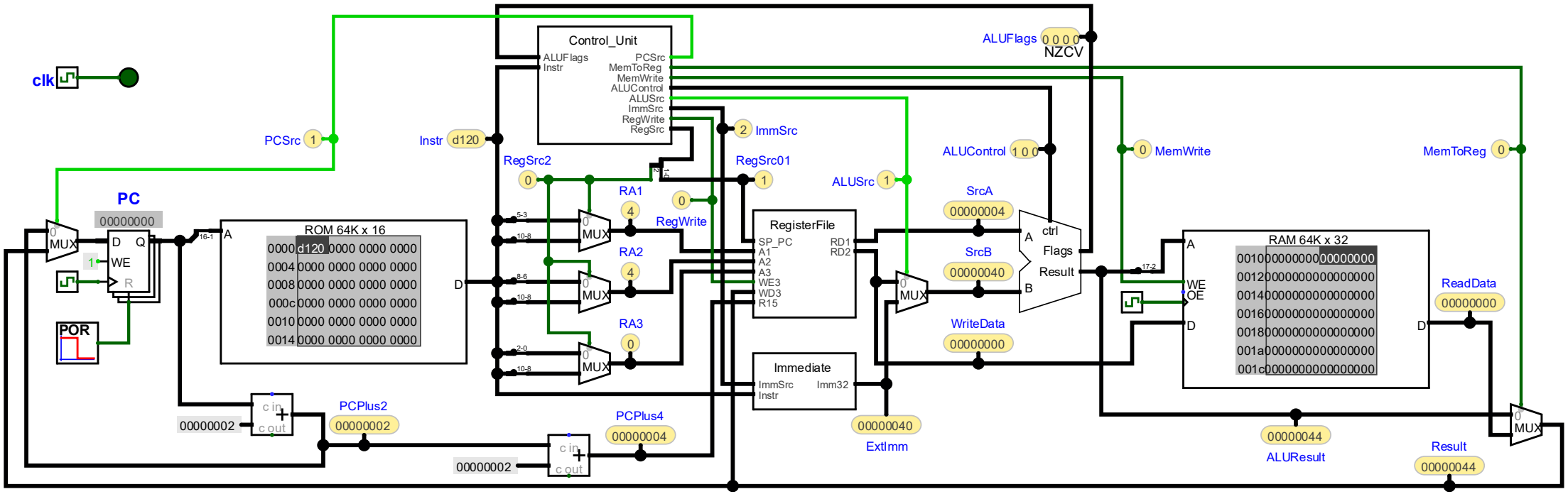


Bcc #<simm8>

11 01 cond ...simm8

ImmSrc		
0	0	imm8
0	1	imm8 << 2
1	0	simm8 << 1
1	1	simm11 << 1

ALUControl			
0	0	0	B
0	0	1	-B
0	1	0	0
0	1	1	A
1	0	0	A+B
1	0	1	A-B
1	1	0	A&B
1	1	1	A|B

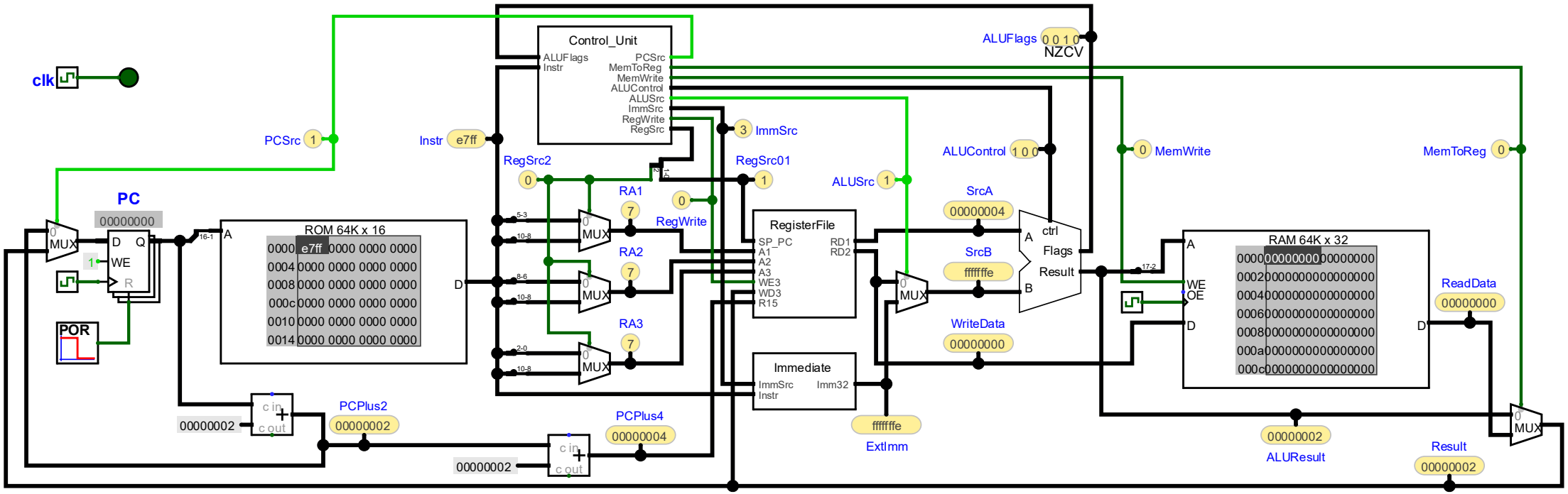


B #<simm11>

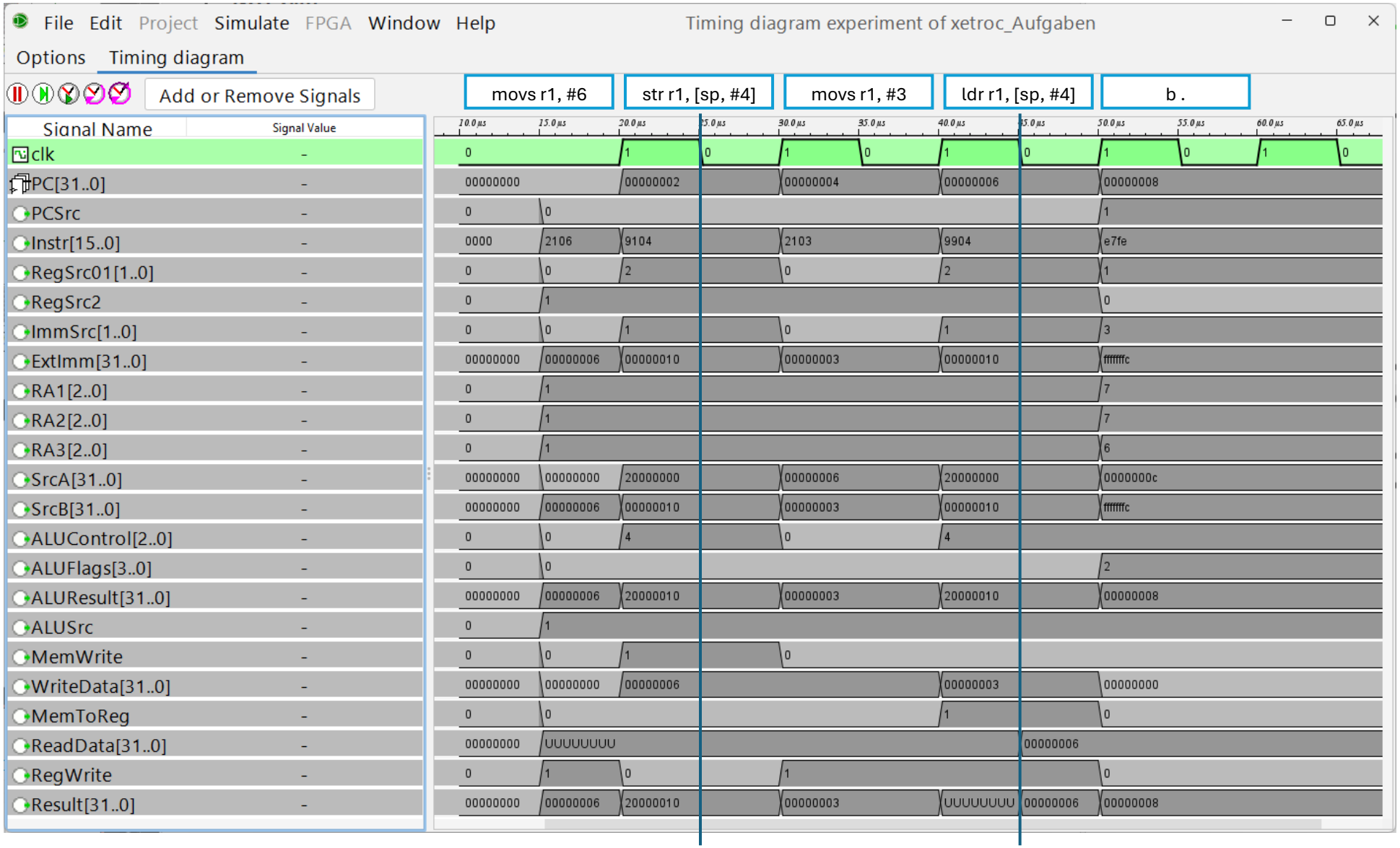
11 10 0 .....simm11

ImmSrc	0	0	imm8
	0	1	imm8 << 2
	1	0	simm8 << 1
	1	1	simm11 << 1

ALUControl	0	0	0	B
	0	0	1	-B
	0	1	0	0
	0	1	1	A
	1	0	0	A+B
	1	0	1	A-B
	1	1	0	A&B
	1	1	1	A|B



Timing Diagrams


File Edit Project Simulate FPGA Window Help
Timing diagram experiment of xetroc_Aufgaben
Options Timing diagram
Add or Remove Signals
movs r1, #6
str r1, [sp, #4]
movs r1, #3
ldr r1, [sp, #4]
b .
Signal Name
Signal Value
clk
PC[31..0]
PCSrc
Instr[15..0]
RegSrc01[1..0]
RegSrc2
ImmSrc[1..0]
ExtImm[31..0]
RA1[2..0]
RA2[2..0]
RA3[2..0]
SrcA[31..0]
SrcB[31..0]
ALUControl[2..0]
ALUFlags[3..0]
ALUResult[31..0]
ALUSrc
MemWrite
WriteData[31..0]
MemToReg
ReadData[31..0]
RegWrite
Result[31..0]