Data and control

Digital Systems – Lecture 22

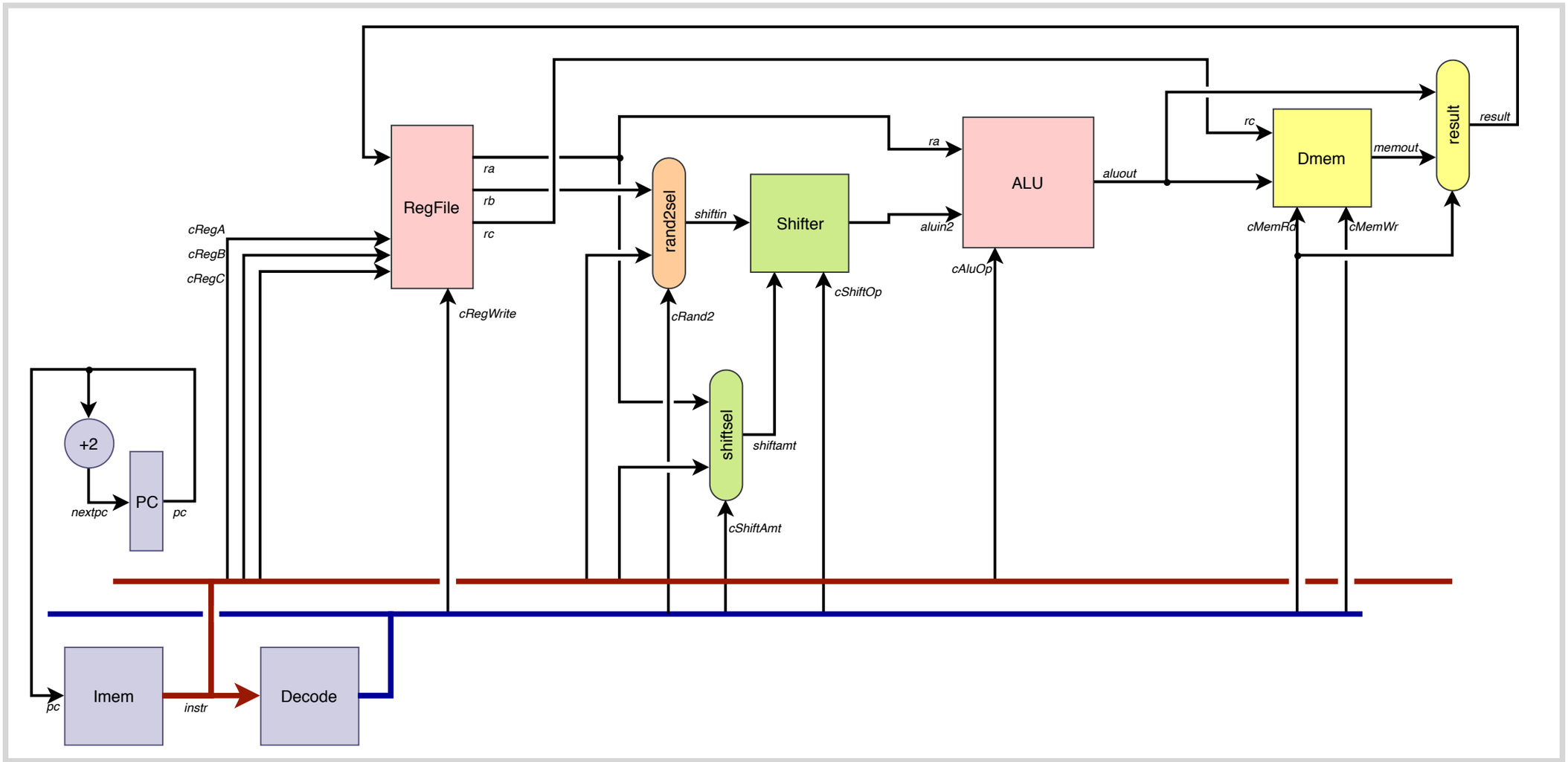
Department of
COMPUTER
SCIENCE

In this lecture

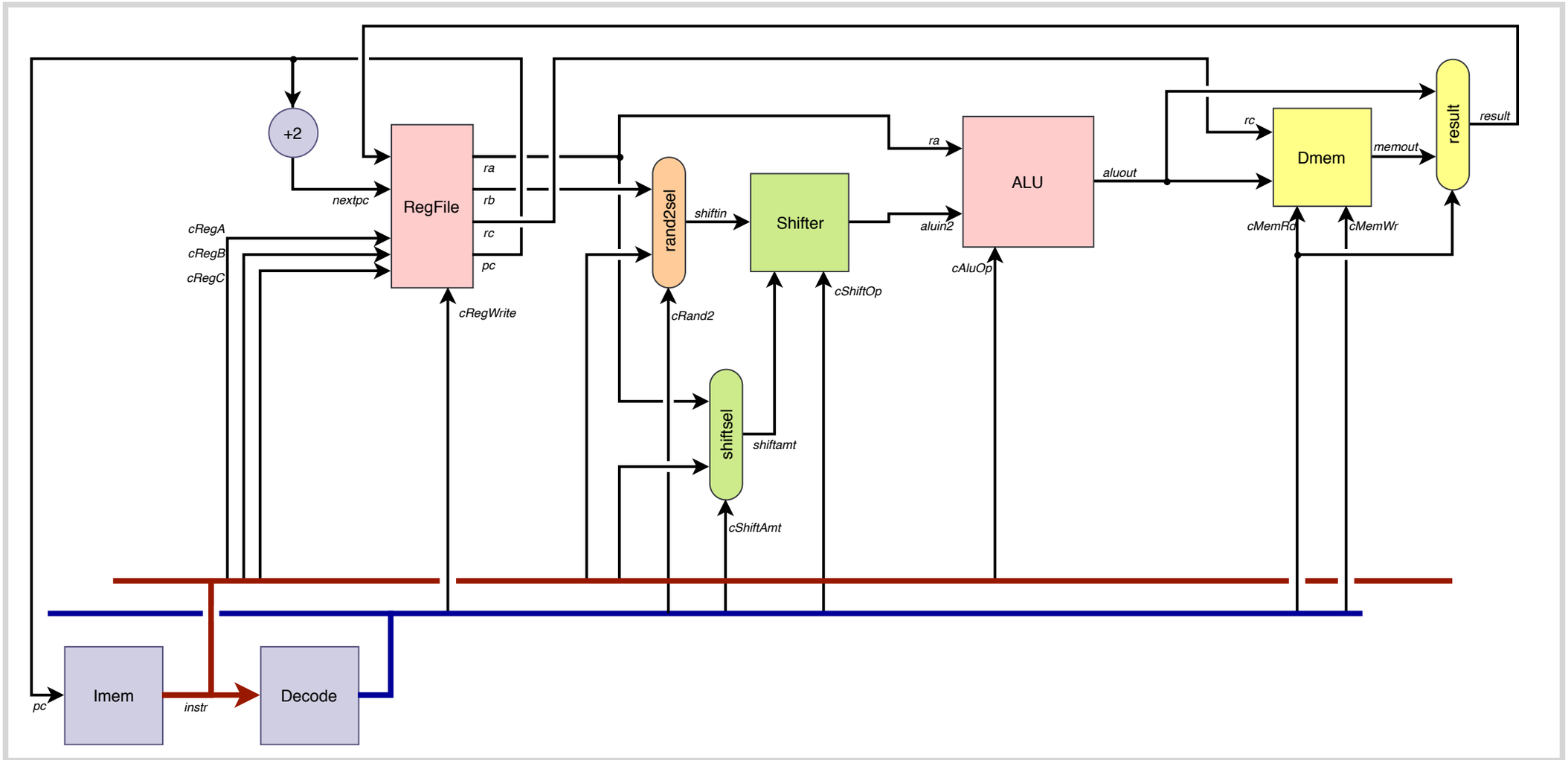
Further stages that enhance control elements of the datapath

- PC as a register
- Instruction decoding
- Subroutine calls
- Conditional execution

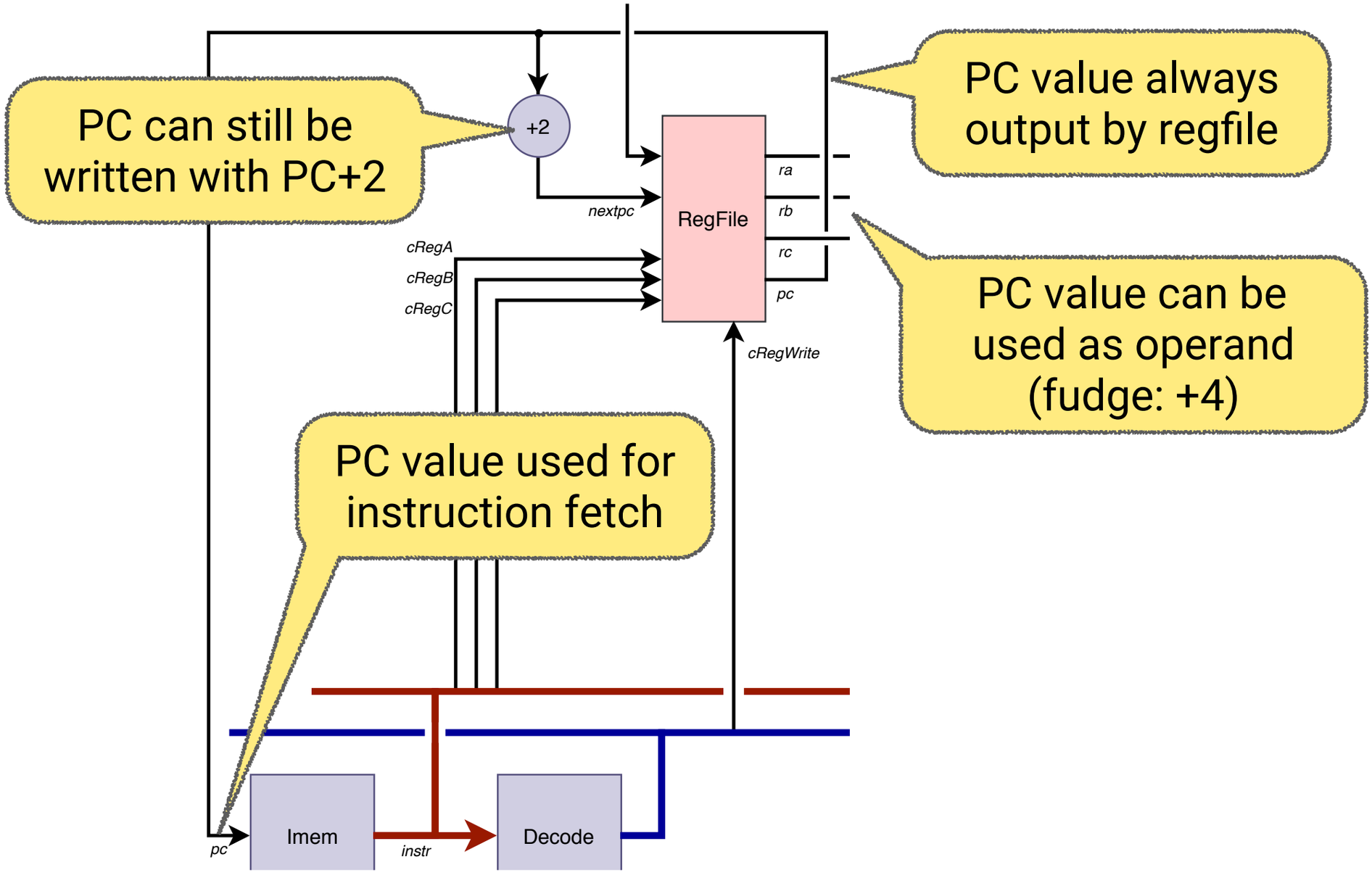
The story from last time



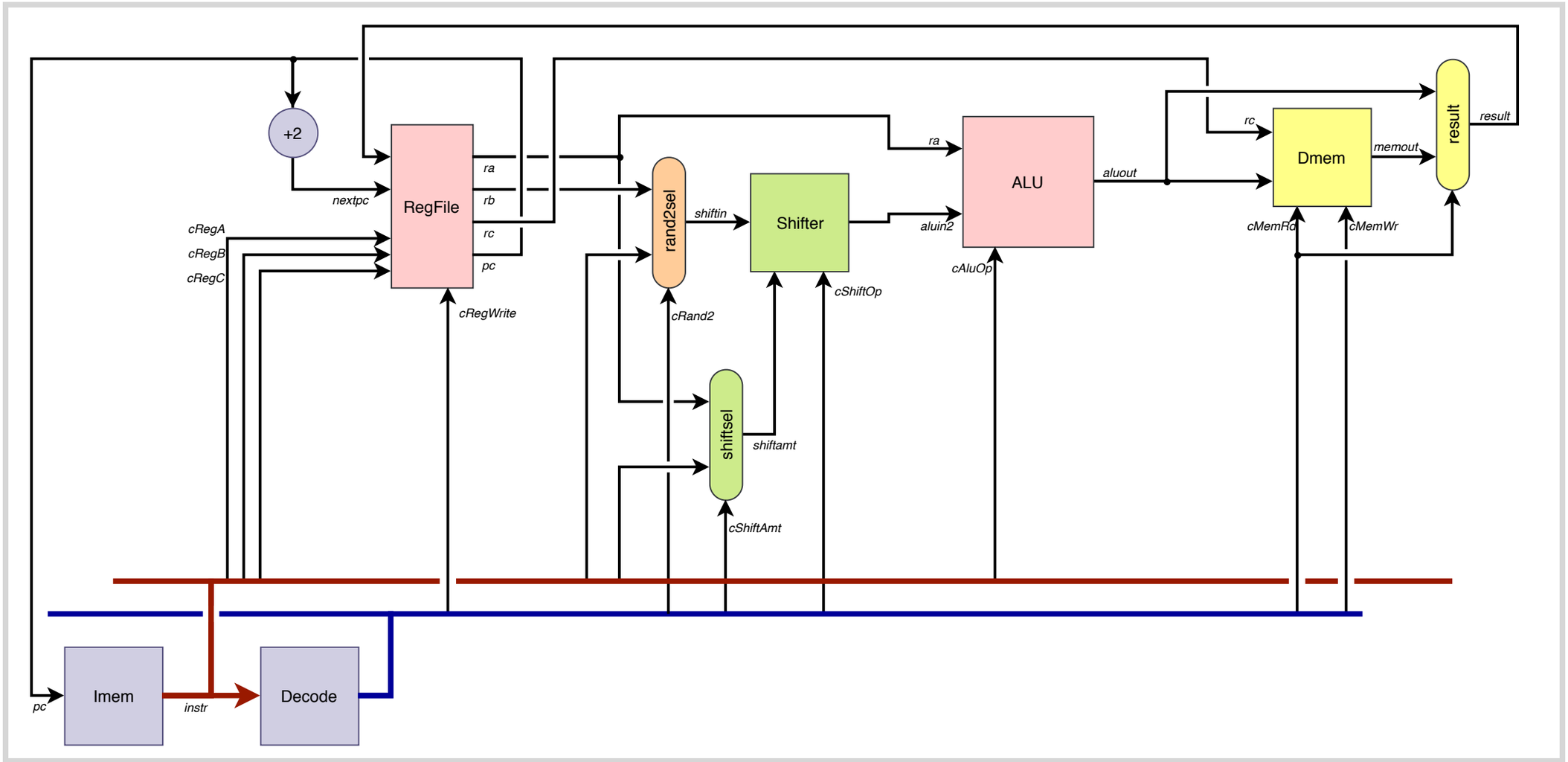
Stage 6: PC as a register



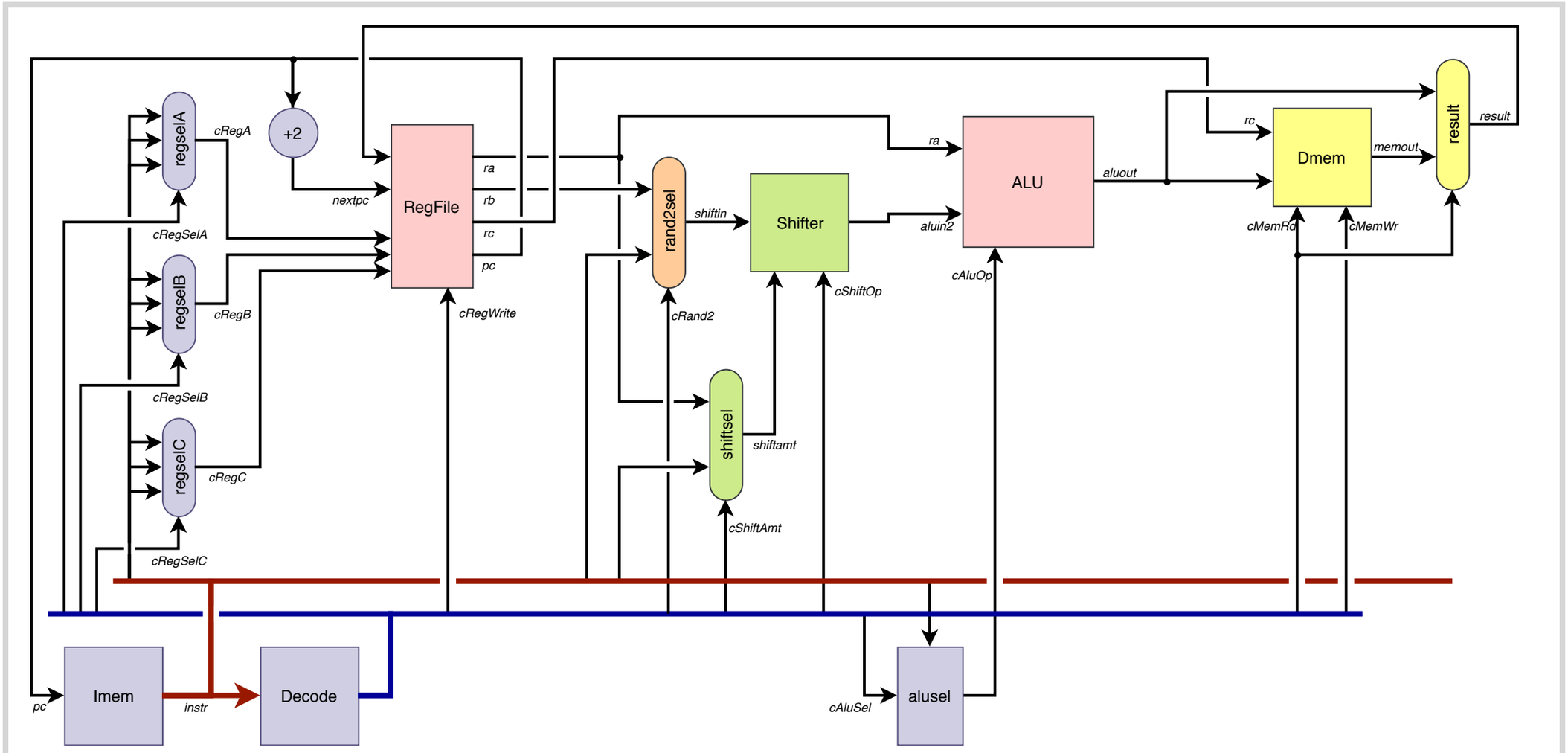
Stage 6: PC as a register



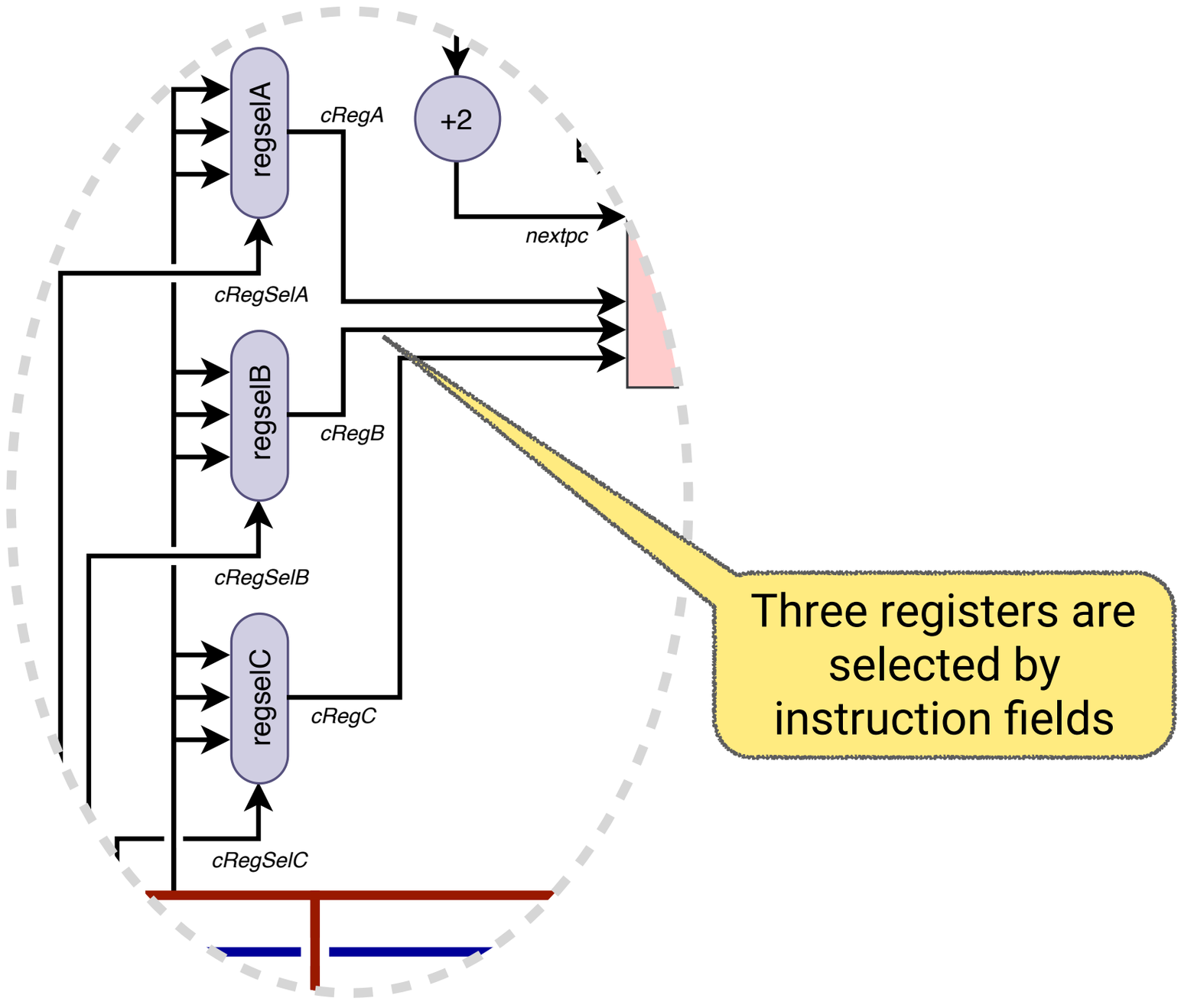
Stage 6: PC as a register



Stage 7: Instruction decoding



Stage 7: Instruction decoding



Some instruction formats

adds ⟨Rx⟩,⟨Ry⟩,⟨Rz⟩

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	0	0	Rz			Ry			Rx		

adds ⟨Rx⟩,⟨Ry⟩,#⟨imm3⟩

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	1	0	imm3			Ry			Rx		

subs ⟨Rx⟩,⟨Ry⟩,⟨Rz⟩

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	0	1	Rz			Ry			Rx		

adds ⟨Rw⟩,#⟨imm8⟩

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	1	0	Rw		imm8								

Controls for register selection

Instruction	cRegSelA	cRegSelB	cRegSelC	cRand2	cShiftOp/Amt	cAluSel	cMemRd/Wr	cRegWrite
adds/subsr/i3	Ry	Rz	Rx	RImm3	Lsl/Sh0	Sg9	F/F	T
movs i8	–	–	Rw	Imm8	Lsl/Sh0	Mov	F/F	T
ldr r	Ry	Rz	Rx	RegB	Lsl/Sh0	Add	T/F	T
str r	Ry	Rz	Rx	RegB	Lsl/Sh0	Add	F/T	F
lsls i5	–	Ry	Rx	RegB	Lsl/ShImm	Mov	F/F	T
ands r	Rx	Ry	Rx	RegB	Lsl/Sh0	And	F/F	T
rors r	Ry	Rx	Rx	RegB	Ror/ShReg	Mov	F/F	T
ldr i5	Ry	–	Rx	Imm5	Lsl/Sh2	Add	T/F	T
str i5	Ry	–	Rx	Imm5	Lsl/Sh2	Add	F/T	F

Instructions with fixed registers

add sp,sp,#⟨imm7⟩

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	0	0	0	0	0	imm7						

sub sp,sp,#⟨imm7⟩

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	0	0	0	0	1	imm7						

add ⟨Rw⟩,sp,#⟨imm8⟩

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	0	1	Rw			imm8							

ldr ⟨Rw⟩,[sp,#⟨imm8⟩]

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	1	Rw			imm8							

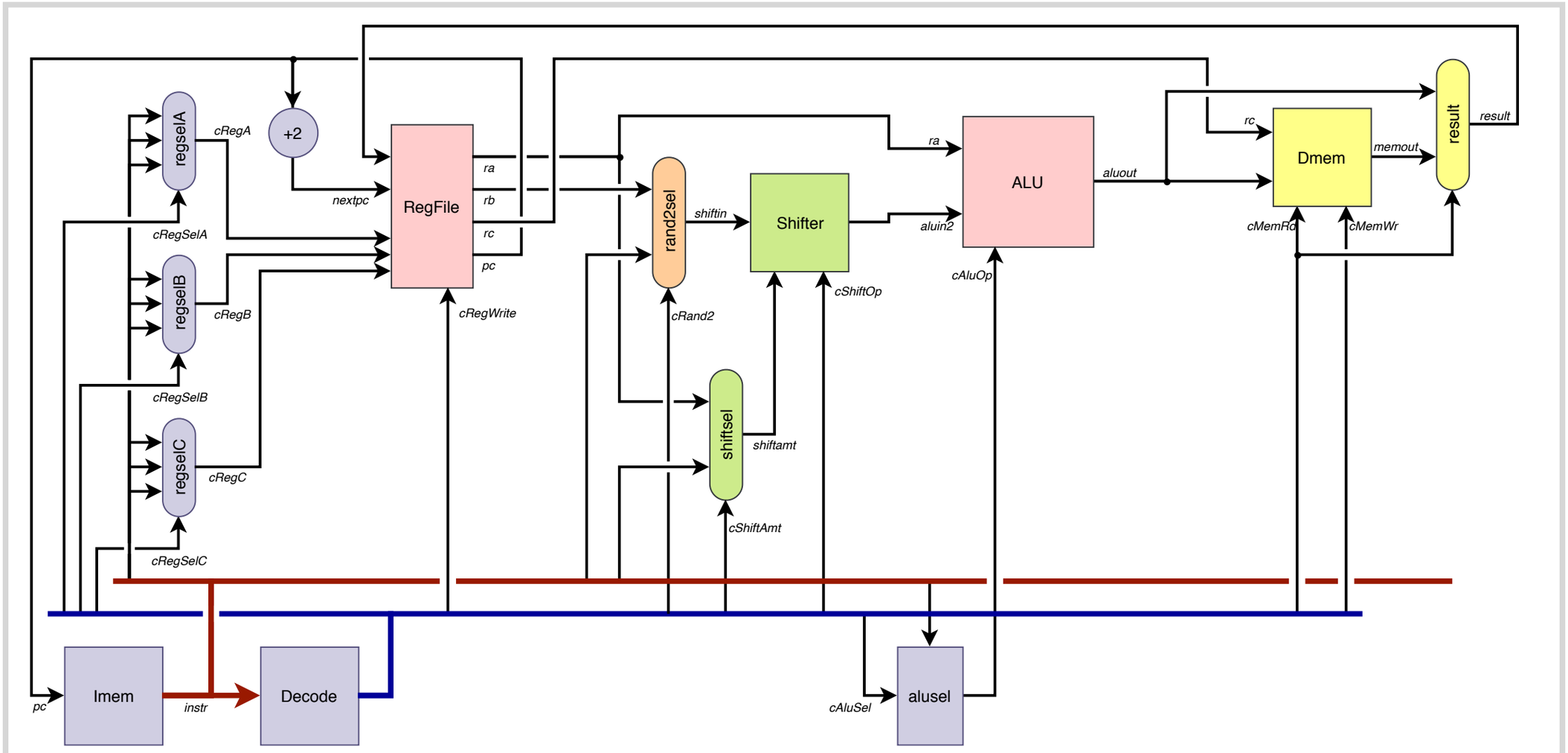
b ⟨disp11⟩

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	0	disp11										

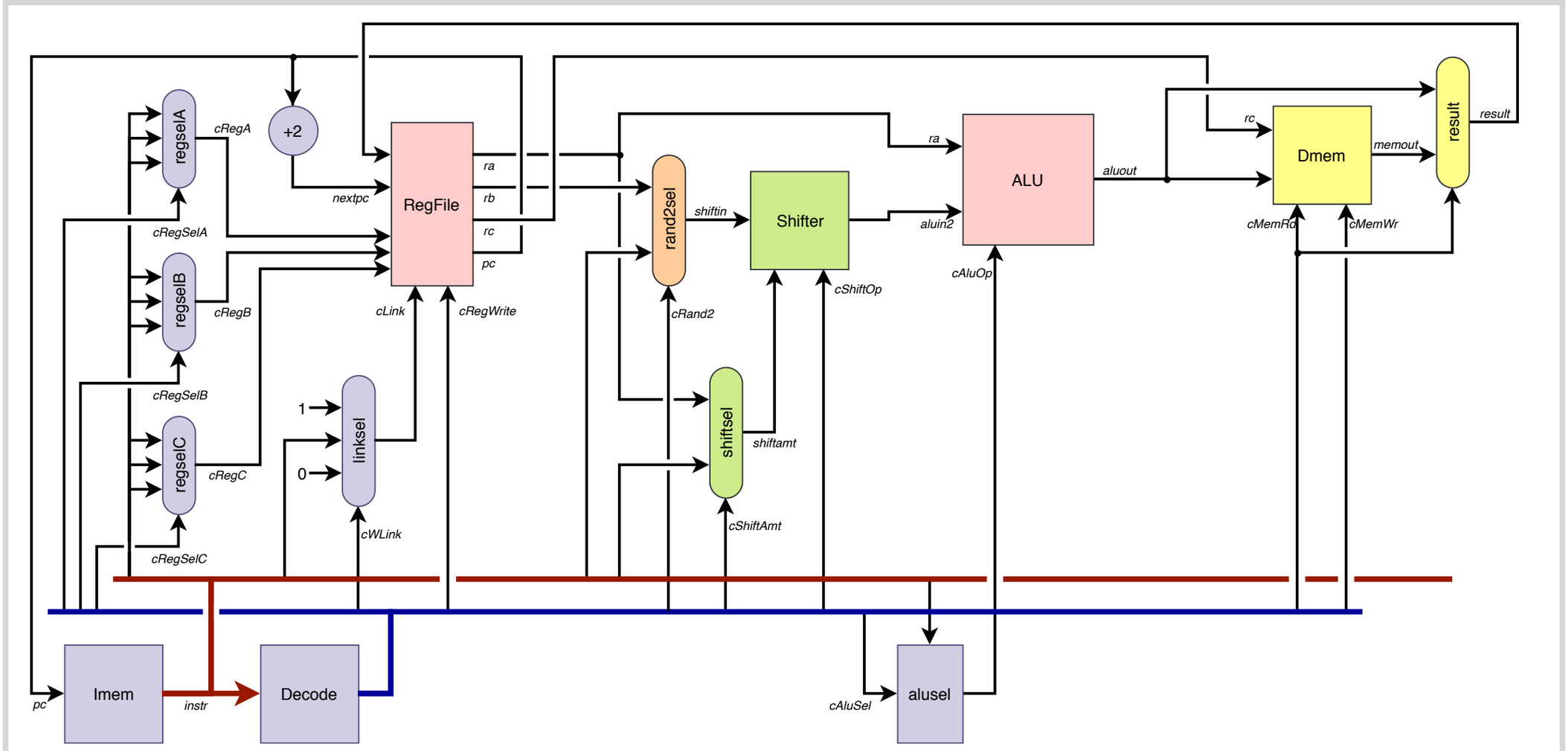
More control rules

Instruction	cReg SelA	cReg SelB	cReg SelC	cRand2	cShiftOp/ Amt	cAlu Sel	cMem Rd/Wr	cReg Write
add/sub sp	Rsp	–	Rsp	Imm7	Lsl/Sh2	Sg7	F/F	T
add rsp	Rsp	–	Rw	Imm8	Lsl/Sh2	Add	F/F	T
ldr sp	Rsp	–	Rw	Imm8	Lsl/Sh2	Add	T/F	T
str sp	Rsp	–	Rw	Imm8	Lsl/Sh2	Add	F/T	F
b	Rpc	–	Rpc	SImm11	Lsl/Sh1	Add	F/F	T

Stage 7: Instruction decoding



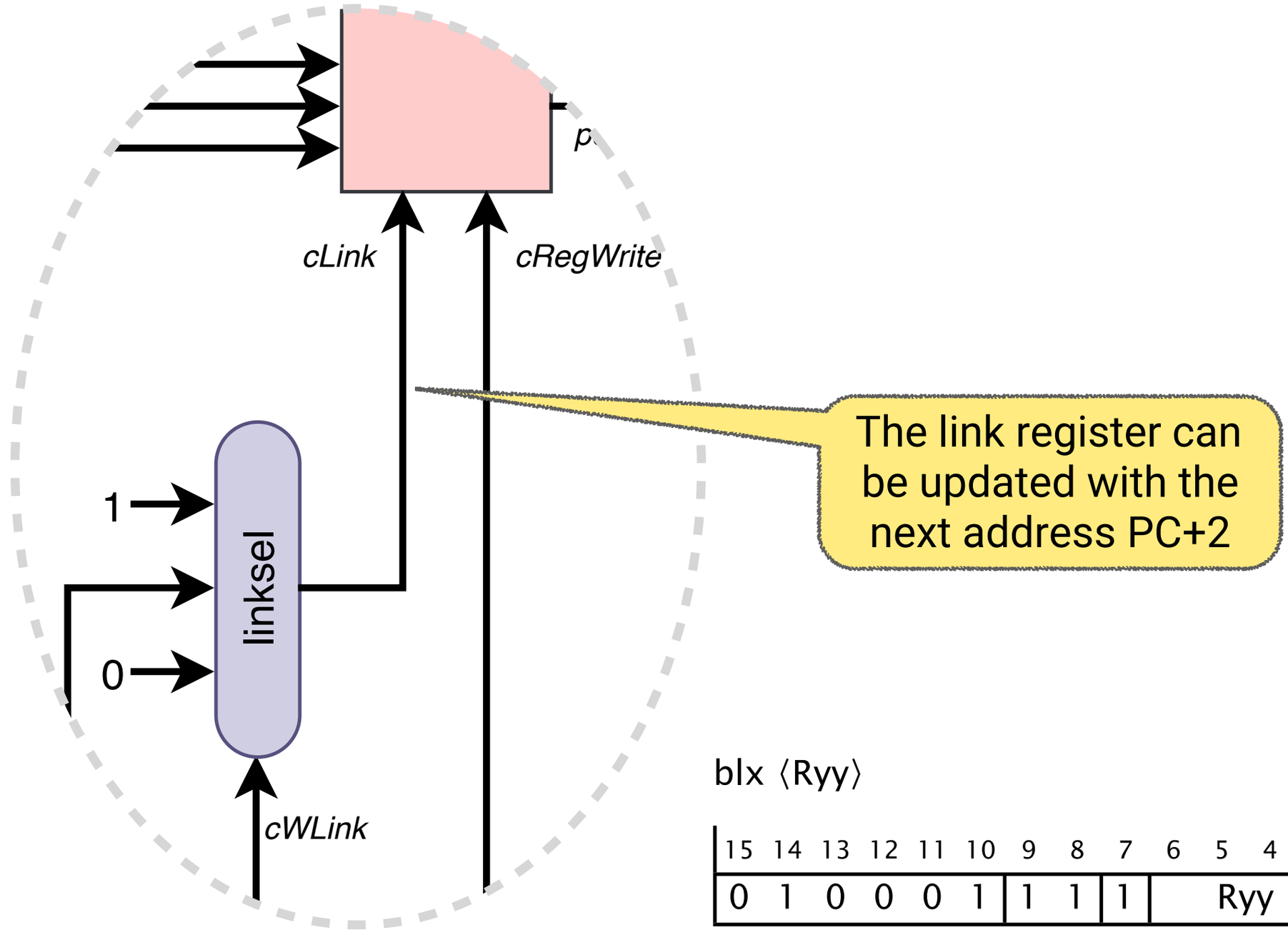
Stage 8: Subroutine calls



blx ⟨Ryy⟩

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	1	1	1	1	Ryy				0	0	0

Stage 8: Subroutine calls

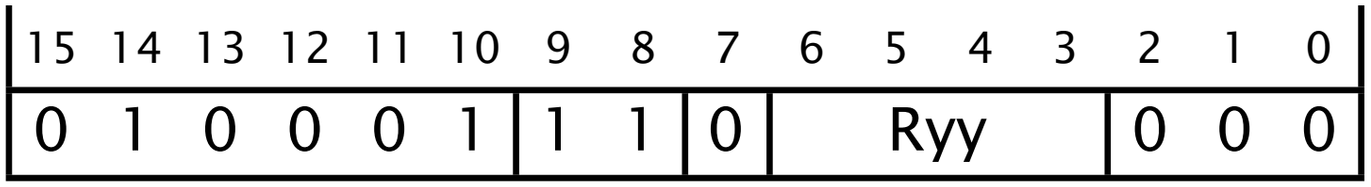


blx ⟨Ryy⟩

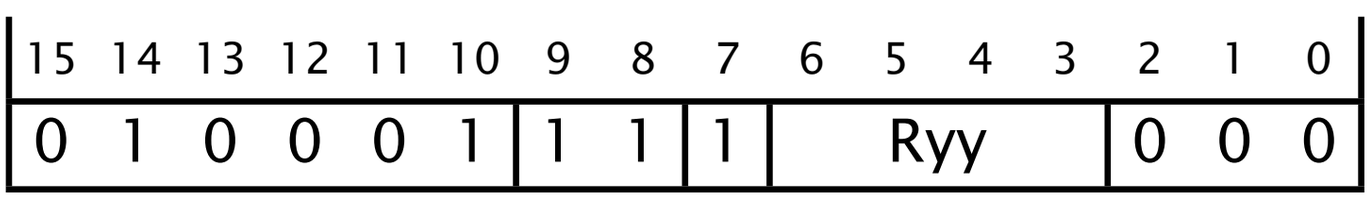
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	1	1	1	1	Ryy				0	0	0

Subroutine call and return

bx ⟨Ryy⟩



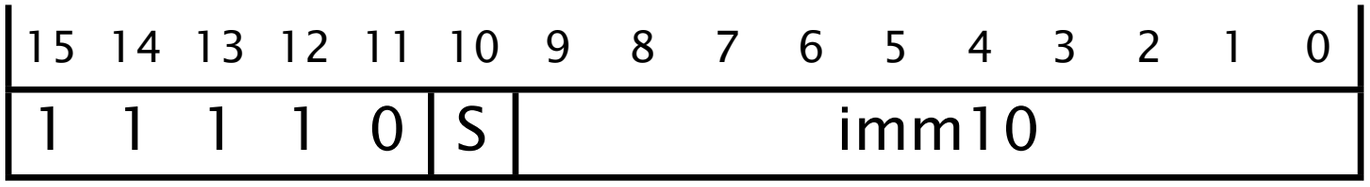
blx ⟨Ryy⟩



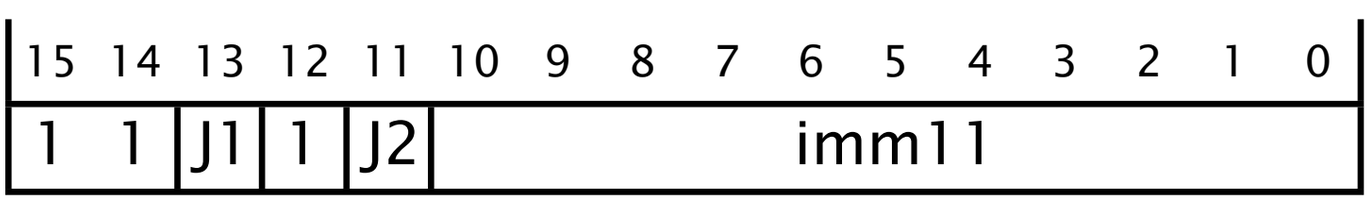
Instruction	cRegSel A/B/C	cRand2	cShiftOp/ Amt	cAlu Sel	cMem Rd/Wr	cReg Write	cWLink
`bx/blx r`	`-/Ryy/Rpc`	`RegB`	`Lsl/Sh0`	`Mov`	`F/F`	`T`	`C`

Bl done the old-fashioned way

bl1, ⟨simm10⟩

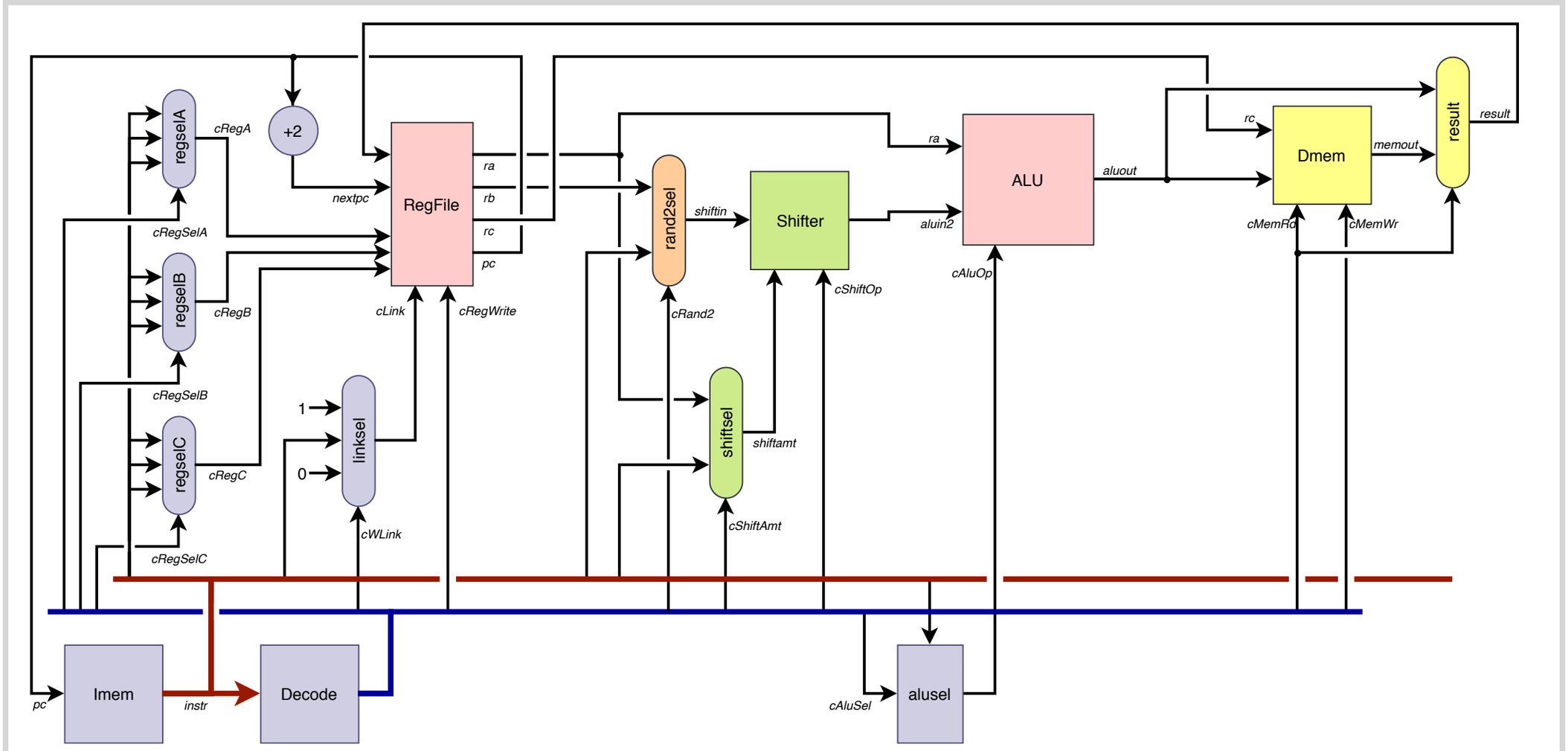


bl2, ⟨imm11⟩

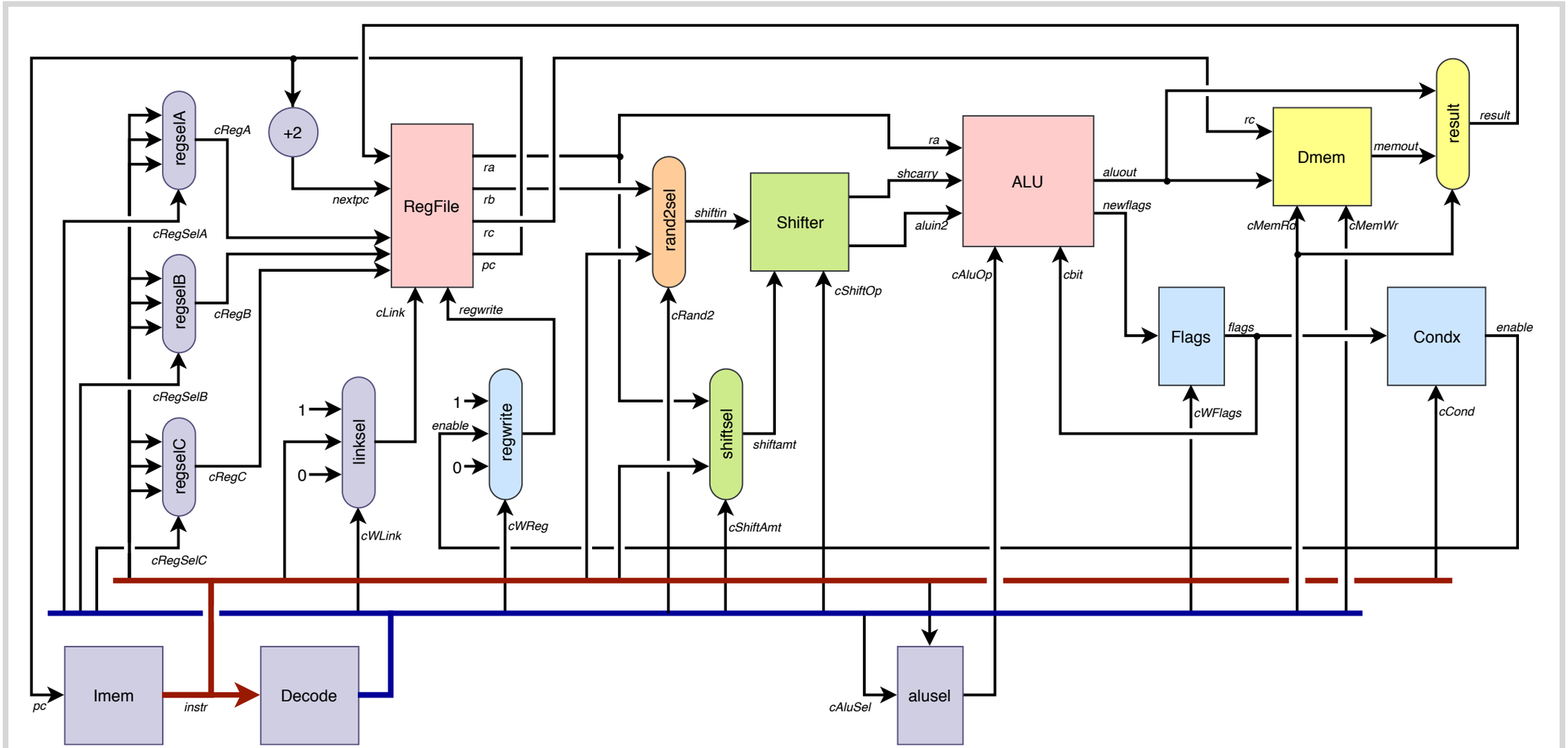


Instruction	cRegSel A/B/C	cRand2	cShiftOp/ Amt	cAlu Sel	cMem Rd/Wr	cReg Write	cWLink
bl1	Rpc/–/Rlr	SImm11	Lsl/Sh12	Add	F/F	T	N
bl2	Rlr/–/Rpc	Imm11	Lsl/Sh1	Add	F/F	T	Y

Stage 8: Subroutine calls



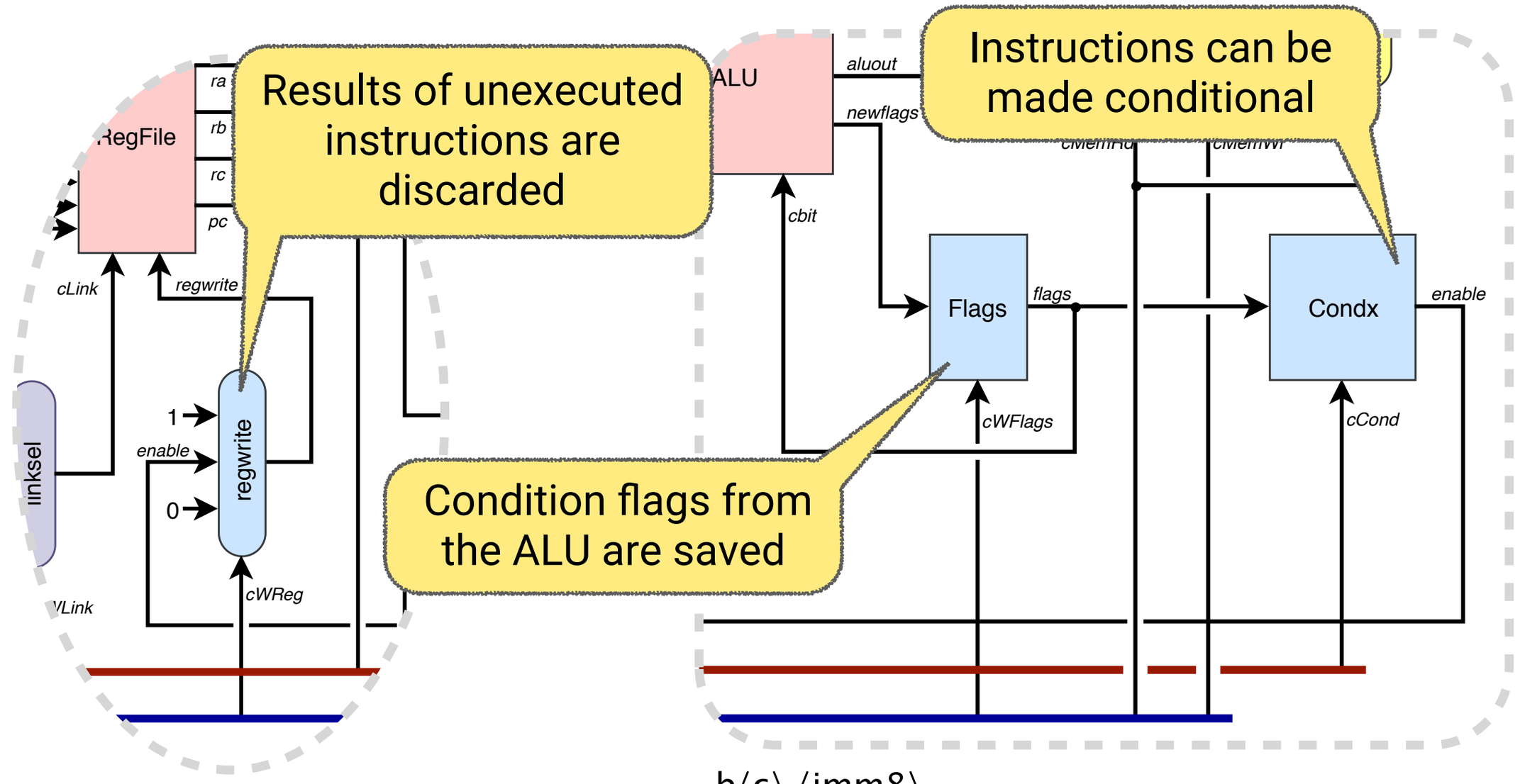
Stage 9: Conditional execution



b⟨c⟩ ⟨imm8⟩

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	1	cond				imm8							

Stage 9: Conditional execution

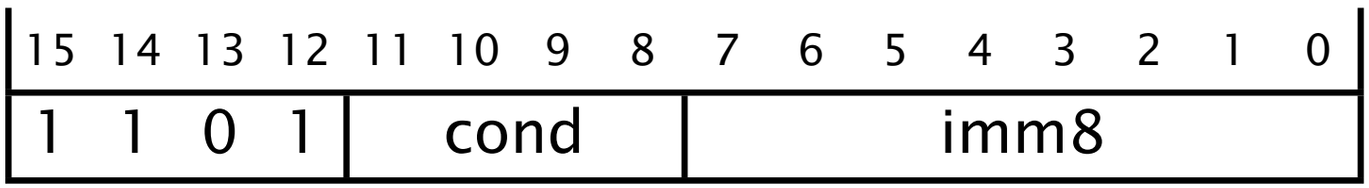


b⟨c⟩ ⟨imm8⟩

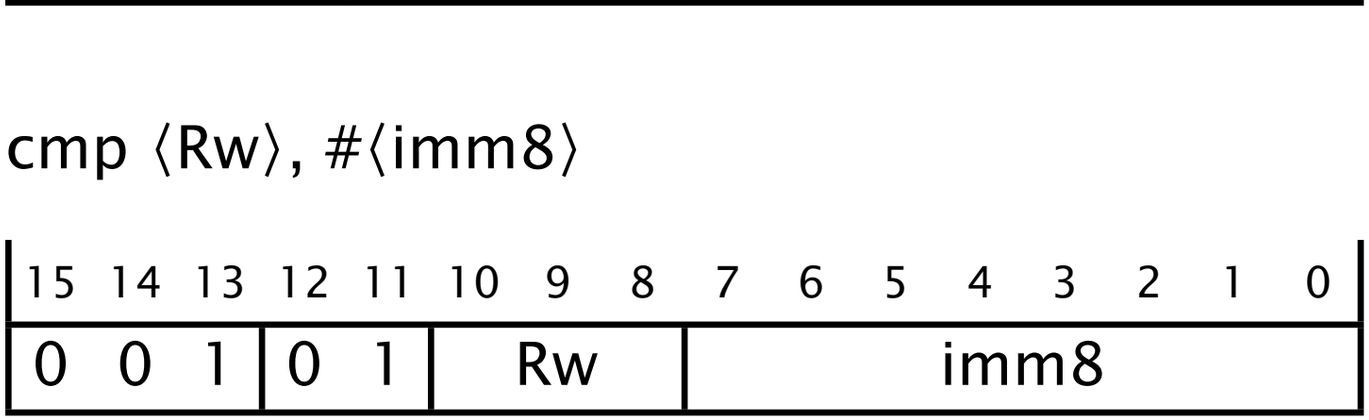
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	1	cond				imm8							

Compare and branch

b⟨c⟩ ⟨imm8⟩

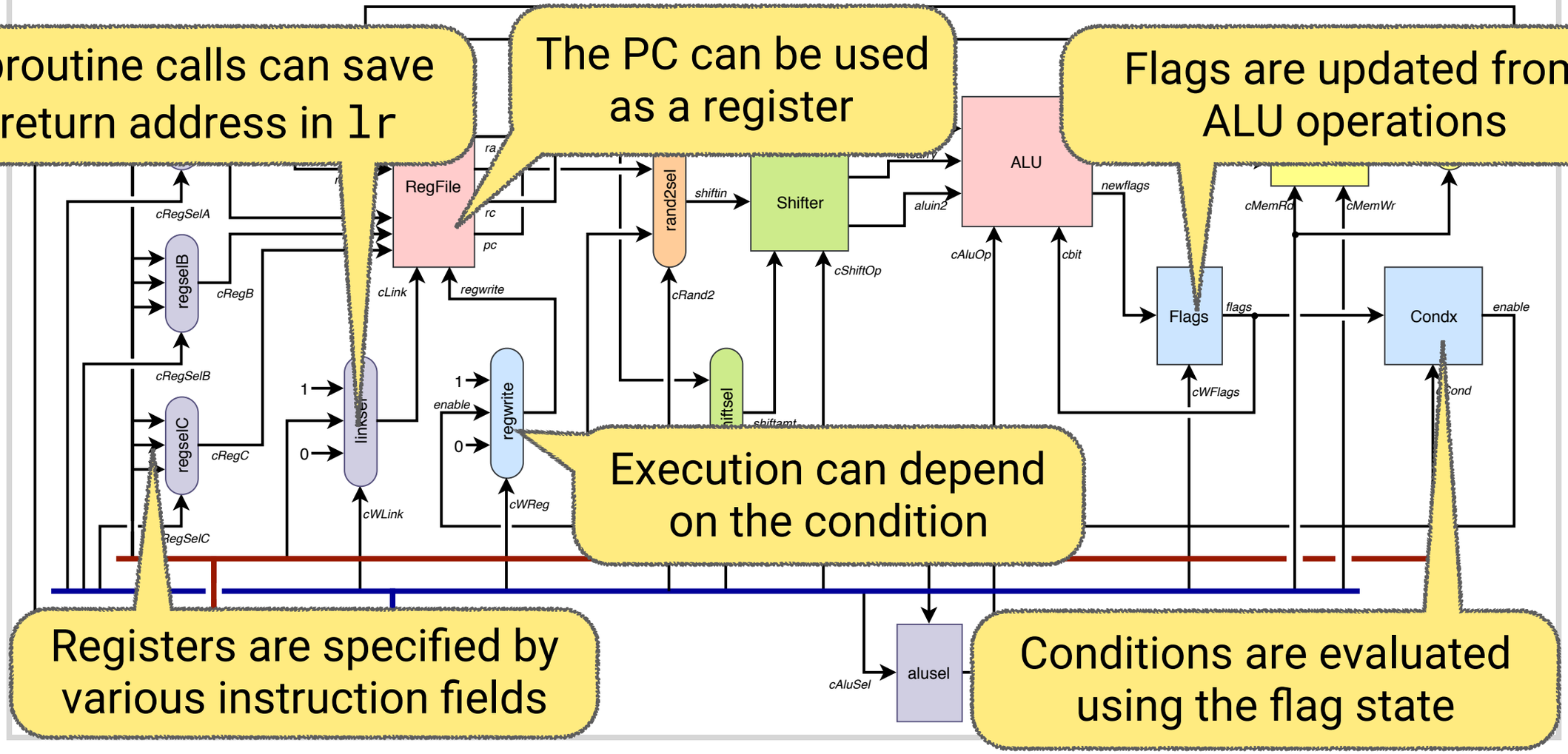


cmp ⟨Rw⟩, #⟨imm8⟩



Instruction	cRegSel A/B/C	cRand2	cShiftOp/ Amt	cAlu Sel	cMem Rd/Wr	cWFlags/ Reg/Link
b<c>	Rpc/–/Rpc	SImm8	Lsl/Sh1	Add	F/F	F/C/N
cmp ri	Rw/–/–	Imm8	Lsl/Sh0	Sub	F/F	T/N/N
subs ri	Rw/–/Rw	Imm8	Lsl/Sh0	Sub	F/F	T/Y/N

The complete datapath



A simulator

"Lab 5" consists of a simulator following the design given in these lectures.

It can load and execute programs prepared with the ARM assembler and linker, provided they use no instructions that we haven't implemented.

A document on the wiki presents the program and explanation in 'literate' form.