Arhitektura računara

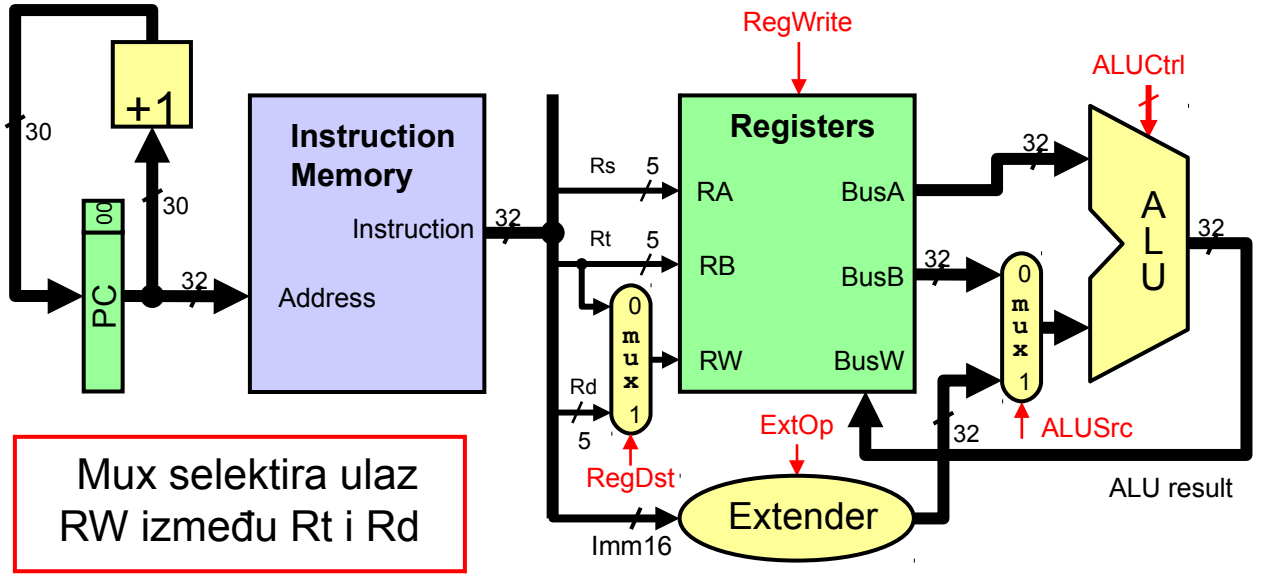
dr.sc. Amer Hasanović

Pregled

- CPU
  - Jednosiklusni Datapath i Kontrola

U predavanju korišteni segmenti iz prezentacije autora M. Mudawar, PhD: http://faculty.kfupm.edu.sa/coe/mudawar/

Kombinovani Datapath za R i I tip

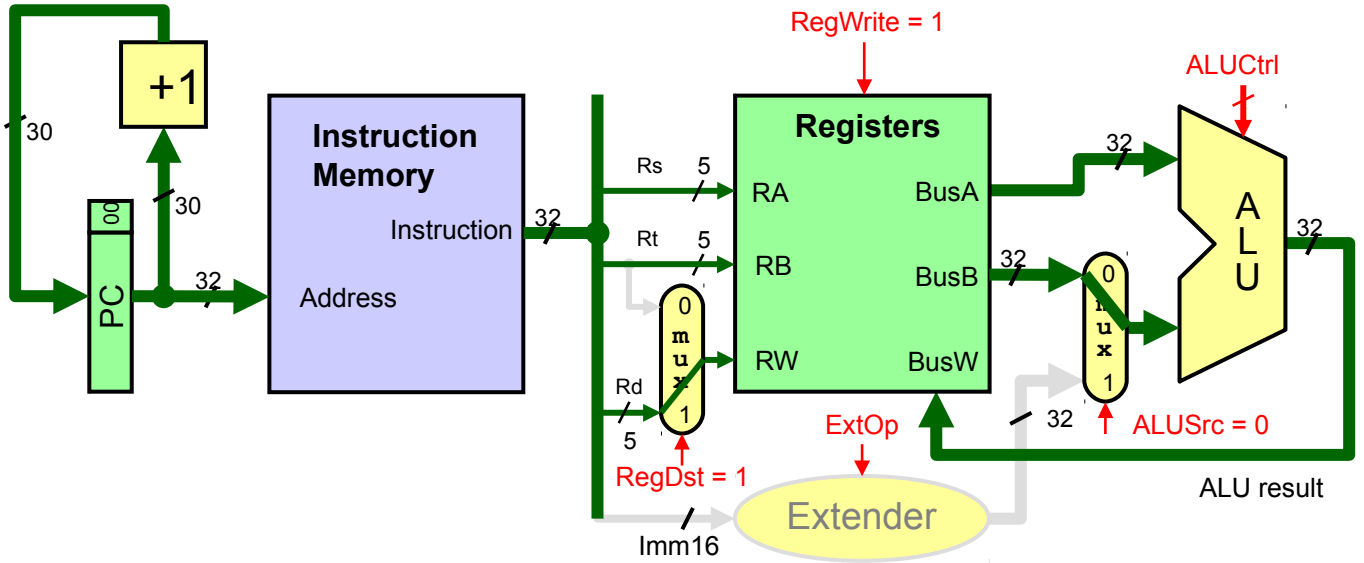


Mux selektira ulaz RW između Rt i Rd

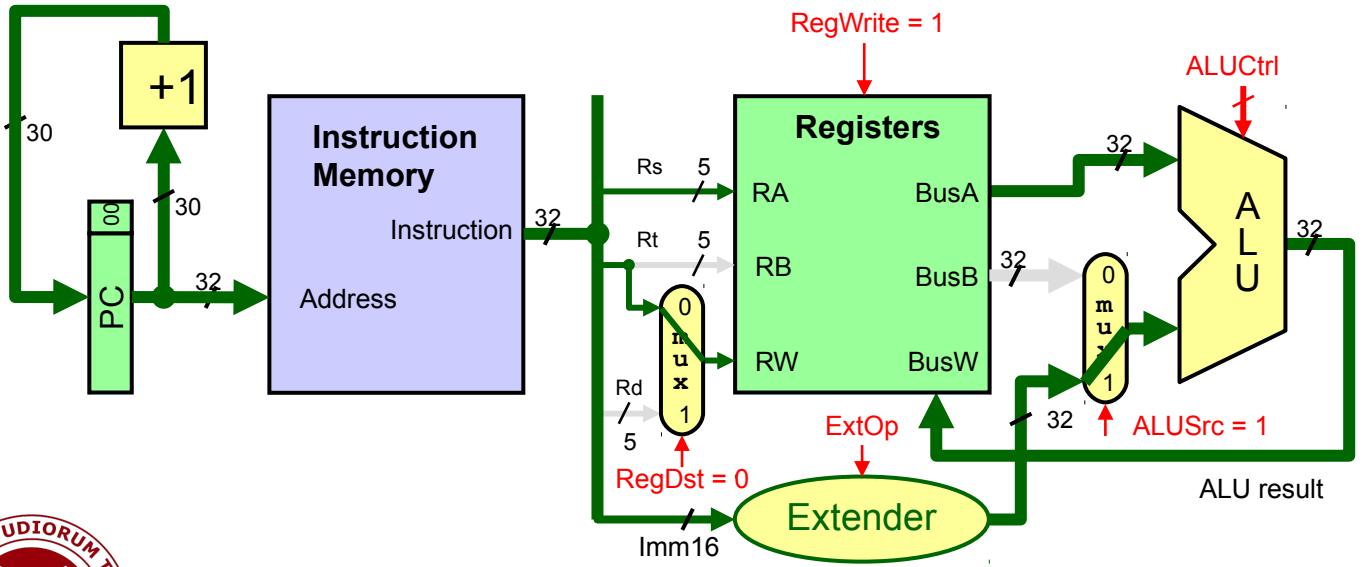
Drugi mux selektira drugi ALU ulaz između BusB izlaza registar fajla i konstante Imm16 iz instrukcije

- Kontrolni signali
  - ALUCtrl nastaje na osnovu polja Op ili funct
  - RegWrite omogućava zapisivanje ALU result
  - ExtOp kontrolira proširivanje 16-bitne konstante
  - RegDst 0 za Rt, 1 za Rd
  - ALUSrc 0 za BusB, 1 za Imm16

Kontroliranje kombinovanog Datapath-a

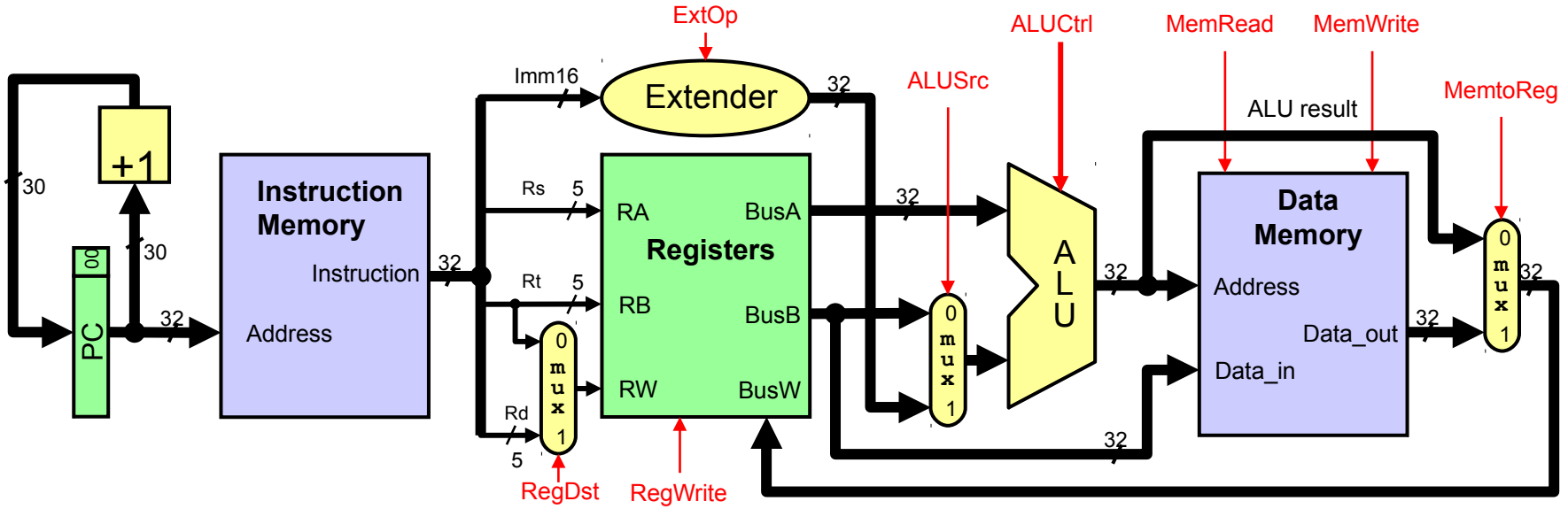


R-tip instrukcije. Aktivni dio datapath-a je prikazan **zelenom** bojom



I-tip instrukcije. Aktivni dio datapath-a je prikazan **zelenom** bojom

Datapath sa pristupom memoriji



ALU računa memorijsku adresu

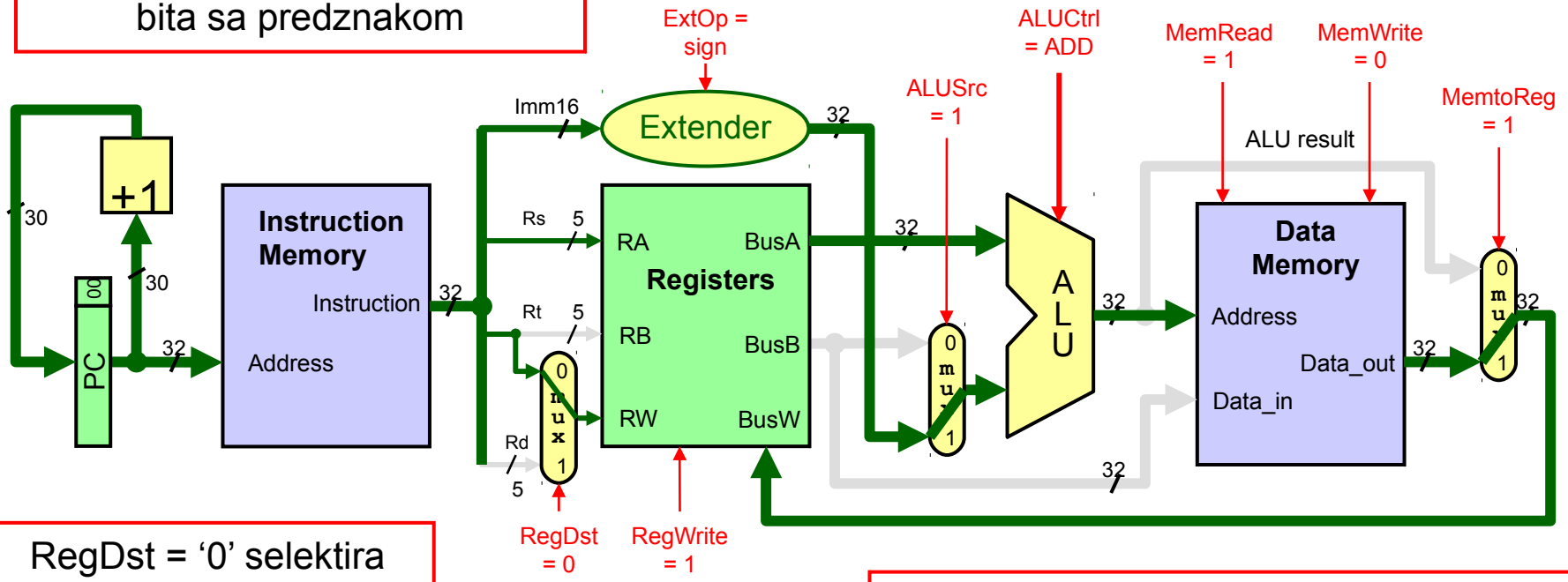
Treći mux selektira između izlaza iz memorije ili izlaza iz ALU jedinice

- Dodatni kontrolni signali
  - MemRead za load instrukciju
  - MemWrite za store instrukciju
  - MemtoReg bira između ALU result ili Memory Data_out

BusB je podatak koji se zapisuje u memoriju za slučaj store instrukcije

Izvršavanje load instrukcije

ExtOp se postavlja na ekstenziju bita sa predznakom



RegDst = ‘0’ selektira destinaciju na osnovu Rt

MemRead = ‘1’ za čitanje memorije

ALUSrc = ‘1’ bira konstantu iz instrukcije za drugi input u ALU jedinicu

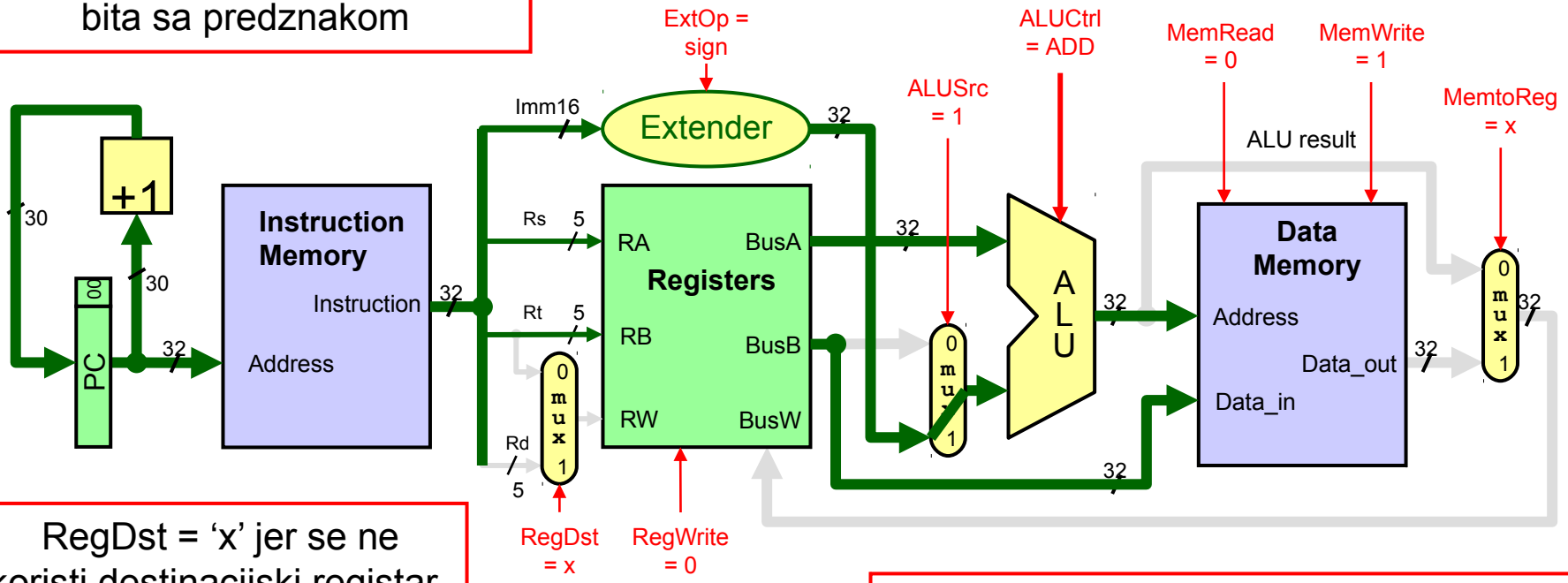
MemtoReg = ‘1’ postavlja pročitani podatak iz memorije za ulaz na BusW

ALUCtrl = ‘ADD’ da bi se sračunala adresa u memoriji za čitanje: Reg(Rs) + sign-extend(Imm16)

RegWrite = ‘1’ indicira da se vrši pisanje vrijednosti BusW u registar definiran sa RW

Izvršavanje store instrukcije

ExtOp se postavlja na ekstenziju bita sa predznakom



RegDst = ‘x’ jer se ne koristi destinacijski registar

MemWrite = ‘1’ jer se piše u memoriju

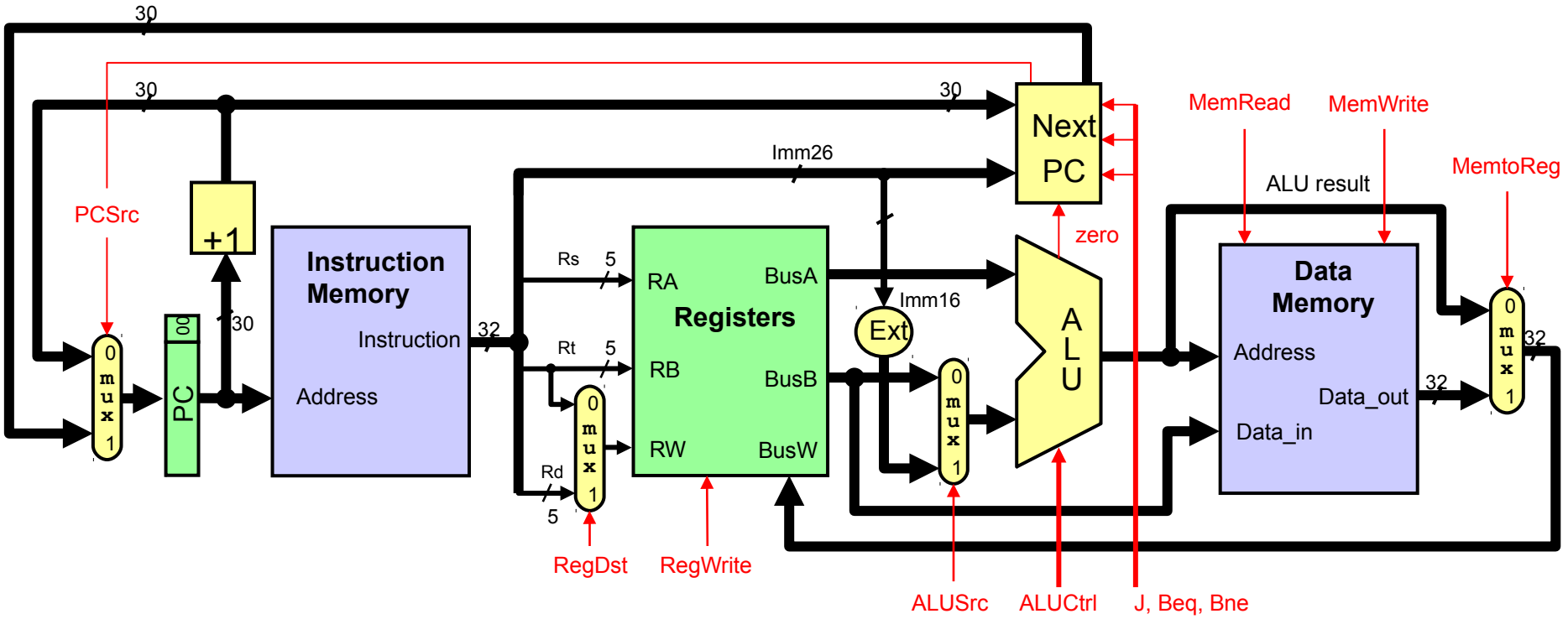
ALUSrc = ‘1’ bira konstantu iz instrukcije za drugi input u ALU jedinicu

MemtoReg = ‘x’ jer nije bitno koji podatak se dovodi kao ulaz za BusW

ALUCtrl = ‘ADD’ da bi se sračunala adresa u memoriji za čitanje: Reg(Rs) + sign-extend(Imm16)

RegWrite = ‘0’ jer se ne vrši pisanje u registre tokom ove instrukcije

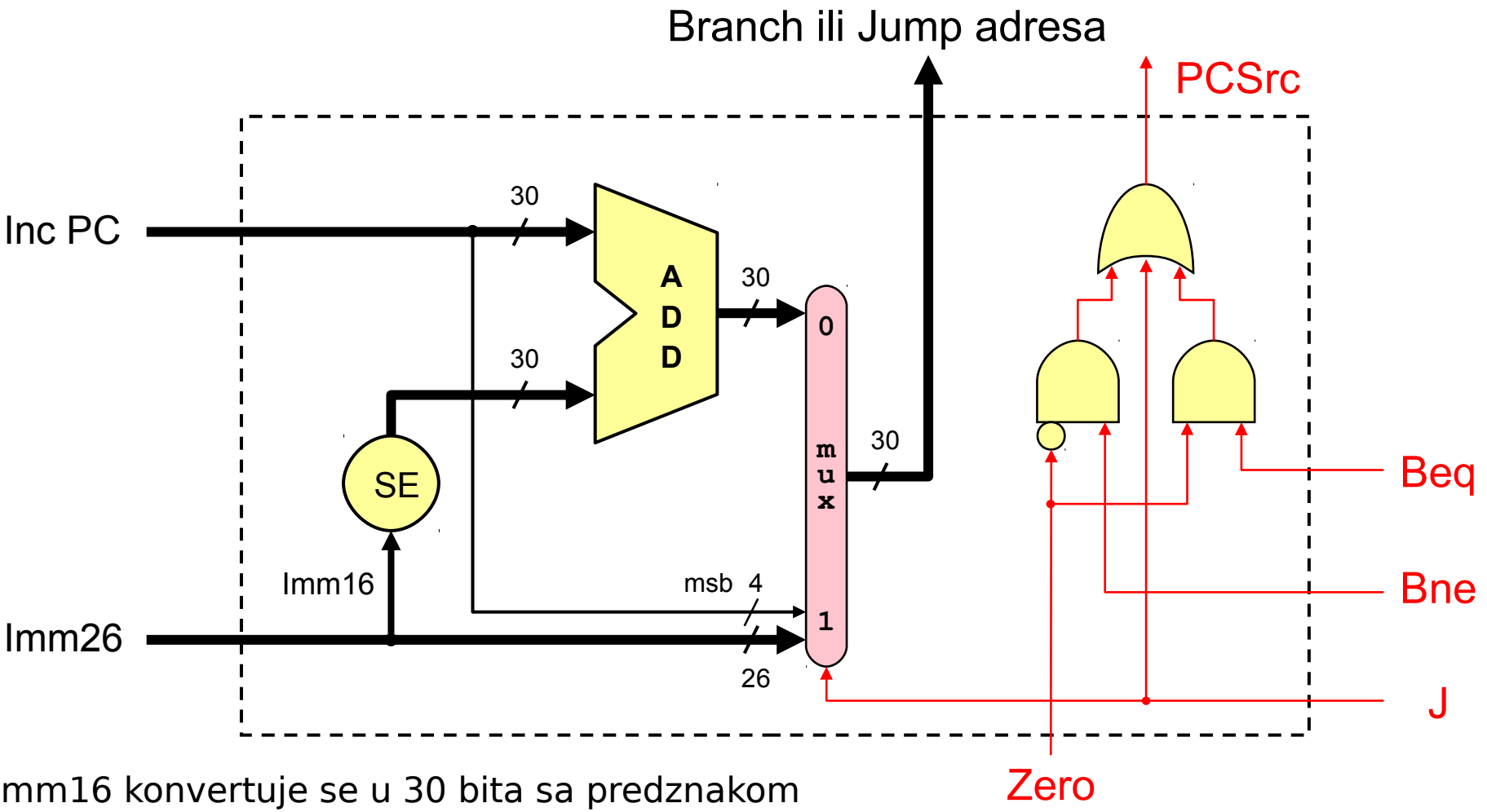
Datapath sa jump i branch instrukcijama



- Dodatni kontrolni signali
  - ✧ J, Beq, Bne
  - ✧ Zero uslov za grananje
  - ✧ PCSrc = 1 u slučaju kada se obavlja J ili branch promjena PC-a

Za Branch instrukciju, ALU obavlja oduzimanje

Next PC blok implementacija

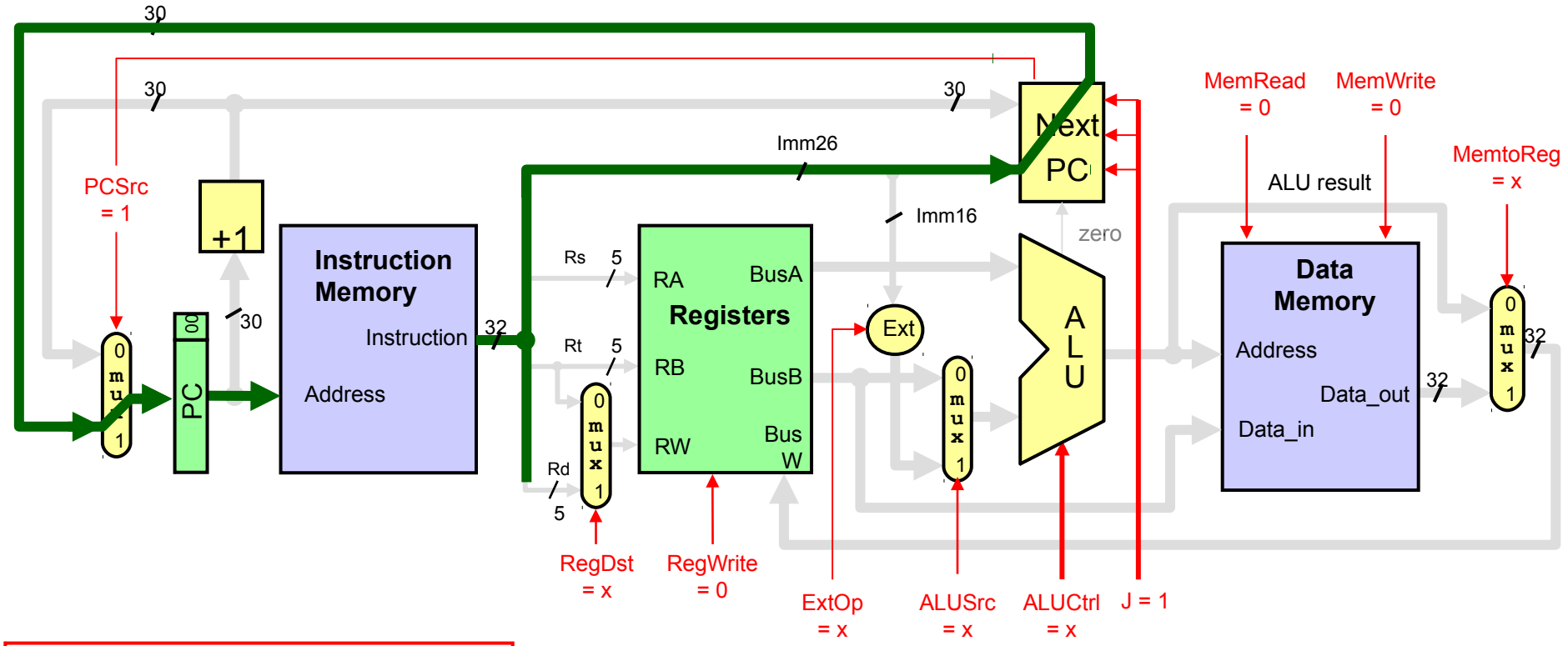


Imm16 konvertuje se u 30 bita sa predznakom

Na Imm26 dodaju se najviša četiri bita iz PC

PCSrc = J ili (Beq **i** Zero) ili (Bne **i** $\overline{\text{Zero}}$)

Izvršavanje jump instrukcije



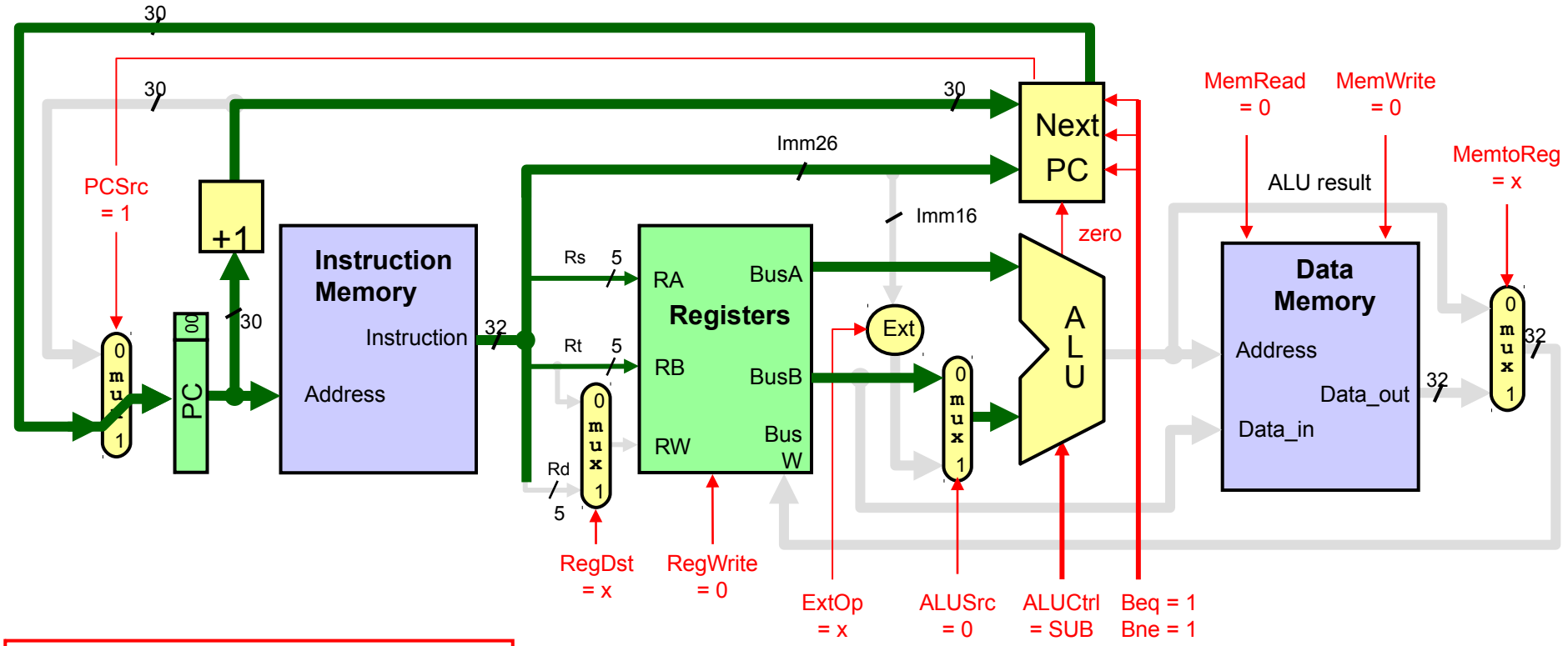
J = 1 selektira Imm26 kao adresu grananja

PCSrc = 1 da bi se izvršilo grananje

MemRead, MemWrite i RegWrite su 0

RegDst, ExtOp, ALUSrc, ALUCtrl, i MemtoReg su proizvoljne vrijednosti

Izvršavanje branch instrukcije



Beq ili Bne je 1

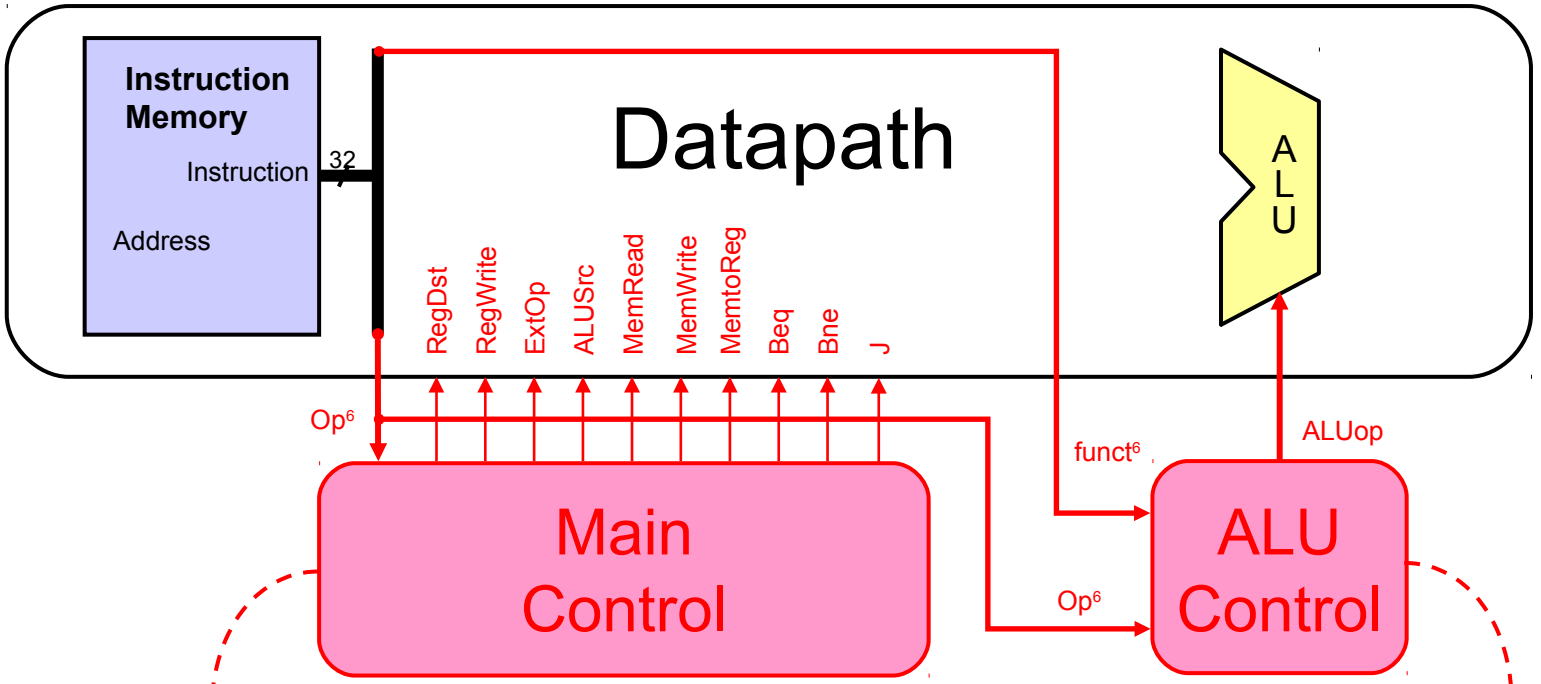
ALUSrc = ‘0’ a
ALUCtrl = ‘SUB’ jer ALU radi oduzimanje

MemRead, MemWrite i RegWrite su 0

Next PC logika određuje vrijednost PCSrc na osnovu zero signala

RegDst, ExtOp i MemtoReg su proizvoljne vrijednosti

Kontrola datapath-a



Main kontrolni blok ulazi

- 6-bitno opcode polje iz instrukcije

Main kontrolni blok izlazi

- 10 kontrolnih signala za Datapath

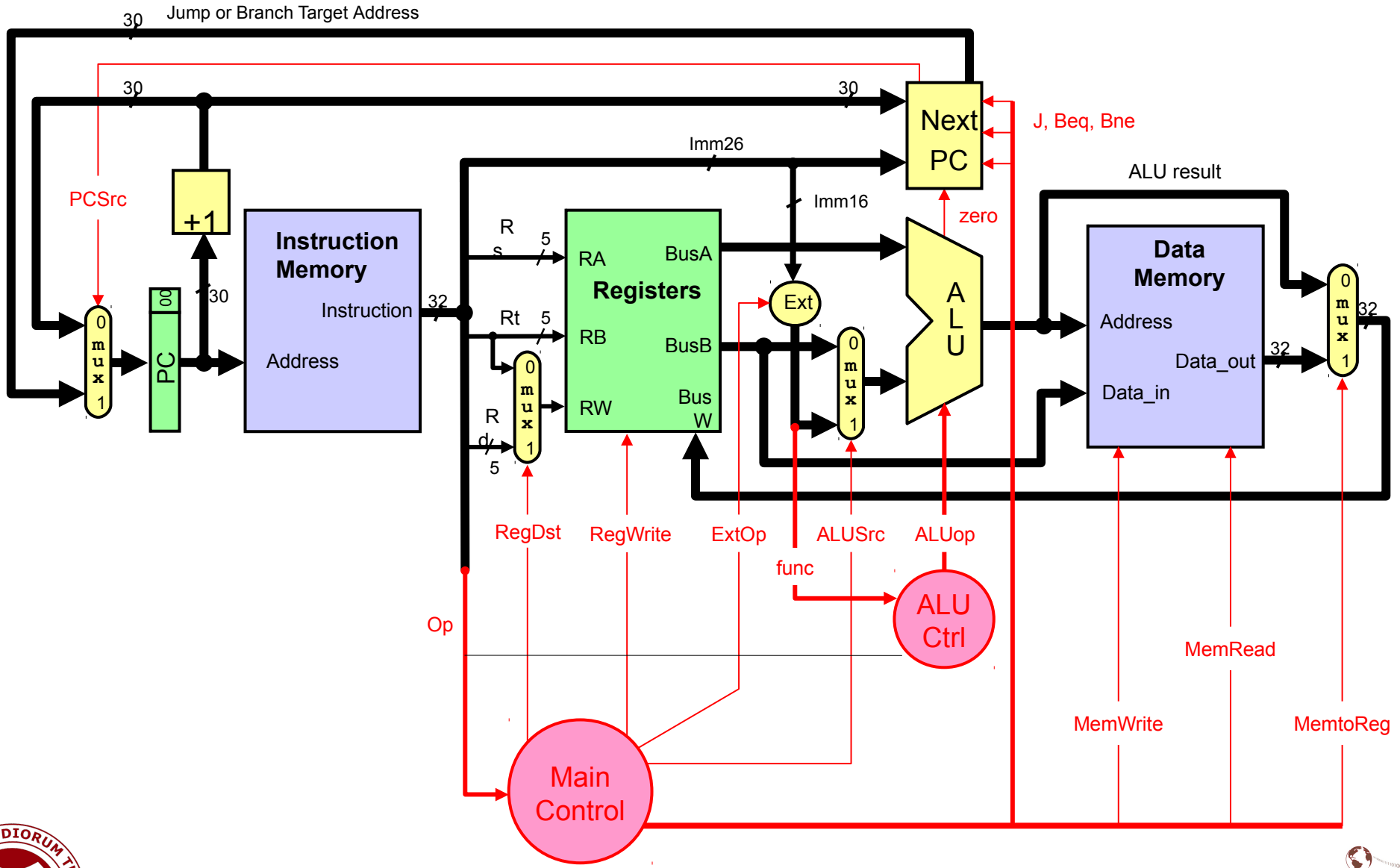
ALU kontrolni blok ulaz:

- 6-bitno opcode polje iz instrukcije
- 6-bitno funct polje iz instrukcije

ALU kontrolni blok izlaz:

- ALUop kontrolni signal za ALU

Jednociklusni CPU



Main blok kontrolni signali za dizajn

Op	Reg Dst	Reg Write	Ext Op	ALU Src	Beq	Bne	J	Mem Read	Mem Write	Mem toReg
R-tip	1 = Rd	1	x	0=BusB	0	0	0	0	0	0
addi	0 = Rt	1	1=sign	1=Imm	0	0	0	0	0	0
slti	0 = Rt	1	1=sign	1=Imm	0	0	0	0	0	0
andi	0 = Rt	1	0=zero	1=Imm	0	0	0	0	0	0
ori	0 = Rt	1	0=zero	1=Imm	0	0	0	0	0	0
xori	0 = Rt	1	0=zero	1=Imm	0	0	0	0	0	0
lw	0 = Rt	1	1=sign	1=Imm	0	0	0	1	0	1
sw	x	0	1=sign	1=Imm	0	0	0	0	1	x
beq	x	0	x	0=BusB	1	0	0	0	0	x
bne	x	0	x	0=BusB	0	1	0	0	0	x
j	x	0	x	x	0	0	1	0	0	x

ALU blok kontrolni signali za dizajn

Input		Output	ALUop
Op^6	$funct^6$	ALUop	kod
R-type	add	ADD	0000
R-type	sub	SUB	0010
R-type	and	AND	0100
R-type	or	OR	0101
R-type	xor	XOR	0110
R-type	slt	SLT	1010
addi	x	ADD	0000
slti	x	SLT	1010
andi	x	AND	0100
ori	x	OR	0101
xori	x	XOR	0110
lw	x	ADD	0000
sw	x	ADD	0000
beq	x	SUB	0010
bne	x	SUB	0010
j	x	x	x