Calculatoare Numerice

– Cursul 10 –

De la CISC la RISC

Facultatea de Automatică și Calculatoare
Universitatea Politehnica București

Comic of the day

http://dilbert.com/strips/comic/2010-05-20/

Din cursurile anterioare

Embedded Systems Laboratory

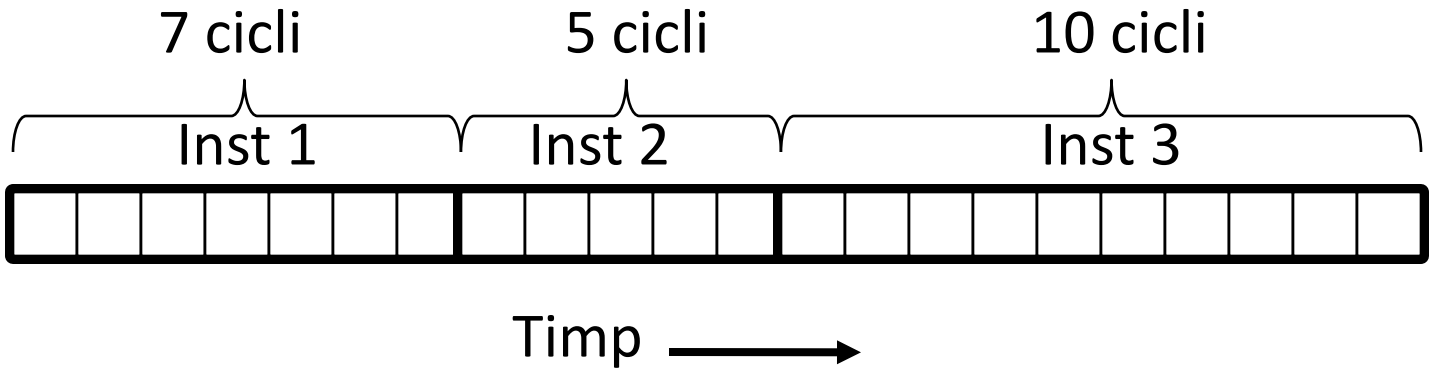
- ISA este interfața hardware/software
  - Definește cum este văzut procesorul de către programator
  - Definește formatul instrucțiunilor (bit encoding) și semnatica lor
  - Exemple: IBM 360, MIPS, RISC-V, x86, JVM
- Mai multe implementări posibile pentru aceeași ISA
  - IBM360: model 30 (c. 1964), z12 (c. 2012)
  - X86: 8086 (c. 1978), 80186, 286, 386, 486, Pentium, Pentium Pro, Pentium-4 (c. 2000), Core 2 Duo, Nehalem, Sandy Bridge, Ivy Bridge, Atom, AMD Athlon, Transmeta Crusoe, SoftPC
  - MIPS: R2000, R4000, R10000, R18K, …
  - JVM: HotSpot, PicoJava, ARM Jazelle, …
- Microcodificare: metodă directă de a implementa mașini de calcul cu puține porți logice și instrucțiuni complexe

“Legea de fier” a performanței calculatoarelor

$$\frac{\text{Time}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} * \frac{\text{Cycles}}{\text{Instruction}} * \frac{\text{Time}}{\text{Cycle}}$$

- Numărul de instrucțiuni per program depinde de codul sursă, compilator și ISA
- Cycles per instructions (CPI) depinde de ISA și de μarchitectură
- Timpul per ciclu depinde de μarchitectură și de tehnologia în care procesorul este construit.

CPI pentru procesoare cu microcod



Numărul total de cicli de ceas = 7+5+10 = 22

Numărul total de instrucțiuni = 3

CPI = 22/3 = 7.33

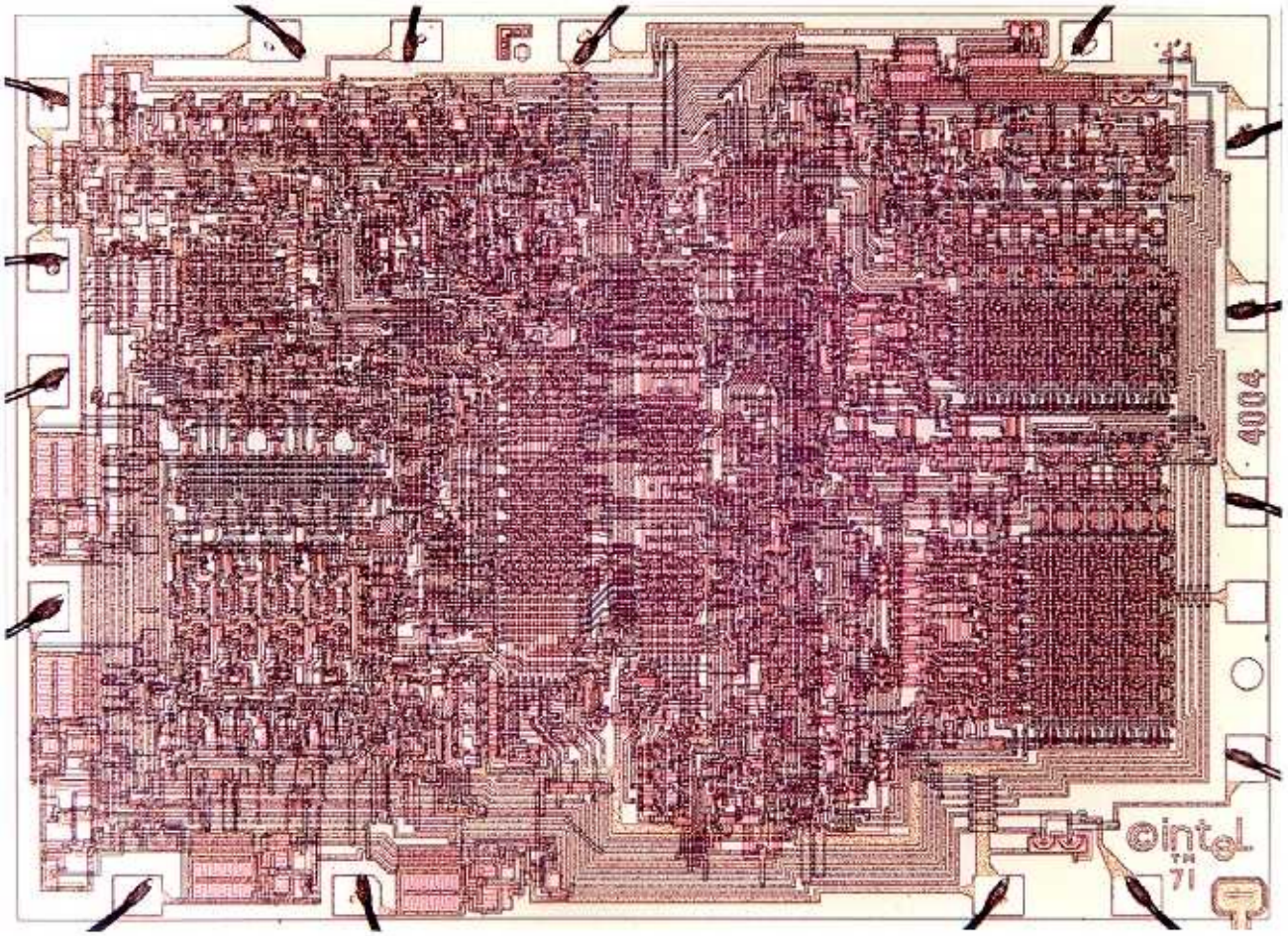
CPI este întotdeauna o valoare medie peste un număr foarte mare de instrucțiuni.

Influențe tehnologice

- Când microprogramarea a apărut pentru prima dată în anii 50, erau tehnologii diferite pentru:
  - Logică: Tuburi cu vid
  - Memorie principală: Miez de ferită
  - Memorie Read-Only: matrice diode, cartele perforate,…
- Logica era foarte scumpă, comparativ cu ROM sau RAM
- ROM mai ieftin ca RAM
- ROM mult mai rapid ca RAM

Anii 70 au adus un avans important în tehnologia de fabricație a circuitelor integrate și a memoriilor semiconductoare...

Primul microprocesor Intel 4004, 1971



- Arhitectură 4-biți cu acumulator
- 8µm pMOS
- 2,300 tranzistoare
- 3 x 4 mm2
- Ceas 750kHz
- 8-16 cicli/inst.

Posibil datorită noii tehnologii de fabricație a circuitelor integrate

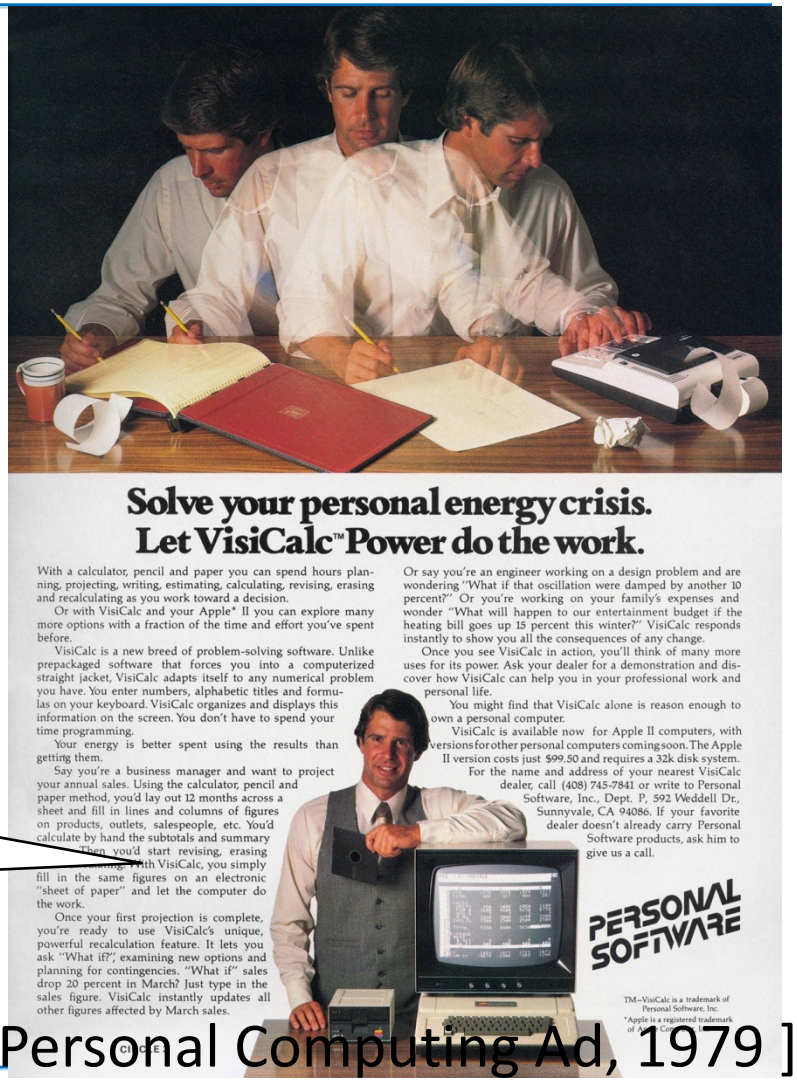
Microprocesoarele în anii 70

- Ținta inițială era controlul disp. embedded
  - Primul micro pe 4 biți, 4004 de la Intel, proiectat pentru un calculator matematic de birou
  - Constrâns de ce se putea integra într-un singur chip
  - Arhitectură cu acumulator, similară cu primele calculatoare
  - Logică cablată de control
- Micro pe 8 biți (8085, 6800, 6502) folosite în PC-uri pentru consumer market
  - Micral, Altair, TRS-80, Apple-II
  - De obicei spațiu de adresă de 16 biți (puteau să adreseze direct maxim 64KB)
  - De obicei veneau cu interpretor BASIC preîncărcat în ROM sau de pe casetă audio.

VisiCalc – primul "killer" app pentru microprocesoare

- Microprocesoarele au avut un impact mic în piața de calculatoare convenționale până la apariția VisiCalc spreadsheet pentru Apple-II
- Apple-II folosea un procesor Mostek 6502 la 1MHz

Discurile floppy au fost inventate de IBM și erau folosite inițial ca un mod de a trimite clienților patch-uri pentru microcodul IBM360!



[Personal Computing Ad, 1979]

Memoria DRAM în anii 70

- Progres dramatic pentru tehnologia de fabricație a memoriilor semiconductoare
- 1970, Intel produce primul DRAM, 1Kbit 1103
- 1979, Fujitsu produce DRAM de 64Kbit

=> La mijlocul anilor șaptezeci era evident că PC-urile vor avea în curând memorii fizice >64KBytes

Evoluția microprocesoarelor

- Progres rapid în anii 70, alimentat de tehnologii noi pentru MOSFET și o piață în continuă creștere
- Intel i432
  - Cel mai ambițios micro al anilor șaptezeci; pornit 1975 - lansat 1981
  - Arhitectură pe 32 de biți orientată obiect
  - Instrucțiuni de lungime variabilă
  - Probleme severe de performanță, complexitate și utilizare
- Motorola 68000 (1979, 8MHz, 68,000 tranzistoare)
  - Puternic microcodat (și nanocodat)
  - Arhitectură pe 32-biți cu registre generale (24 biți adresă)
  - 8 registre adresă, 8 registre de date
- Intel 8086 (1978, 8MHz, 29,000 tranzistoare)
  - Procesor “Stopgap” 16-biți, proiectat în 10 săptămâni
  - Arhitectură cu acumulator, compatibil în assembly cu 8080
  - Adresare segmentată pe 20-biți

IBM PC, 1981

- Hardware
  - Echipa de la IBM ce construia prototipuri de PC-uri în 1979
  - Motorola 68000 ales inițial, dar 68000 nu a fost disponibil la timp
  - IBM construiește prototipuri cu “stopgap” folosind plăci cu 8088 de la procesorul text Display Writer
  - 8088 este versiunea pe 8 biți a lui 8086 => sistem mai ieftin
  - Vânzări estimate inițial la maxim 250,000
  - Vânzări efective: sute de milioane!
- Software
  - Microsoft negociază să producă un OS pentru IBM. Mai târziu cumpără și modifică QDOS de la Seattle Computer Products.
- Sistem deschis
  - Procesor standard, Intel 8088
  - Interfețe standard
  - OS standard, MS-DOS
  - IBM permite clonarea sistemelor lor și existența de software third-party

Presenting the IBM® of Personal Computers.

IBM is proud to announce a product *you* may have a personal interest in. It's a tool that could soon be on your desk, in your home or in your child's schoolroom. It can make a surprising difference in the way you work, learn or otherwise approach the complexities (and some of the simple pleasures) of living.

It's the computer we're making for you.

In the past 30 years, the computer has become faster, smaller, less complicated and less expensive. And IBM has contributed heavily to that evolution.

Today, we've applied what we know to a new product we believe in: the IBM Personal Computer.

IBM PERSONAL COMPUTER SPECIFICATIONS
*ADVANCED FEATURES FOR PERSONAL COMPUTERS

User Memory
16K - 256K bytes*
Permanent Memory
(ROM) 40K bytes*
Microprocessor
High speed, 8088*
Auxiliary Memory
2 optional internal diskette drives, 5¼", 160K bytes per diskette
Keyboard
83 keys, 6 ft. cord attaches to system unit*
10 function keys*
10-key numeric pad
Tactile feedback*
Display Screen
High-resolution (720h x 350v)*
80 characters x 25 lines
Upper and lower case
Green phosphor screen*
Diagnostics
Power-on self testing*
Parity checking
Languages
BASIC, Pascal
Printer
Bidirectional*
80 characters/second
12 character styles, up to 132 characters/line*
9 x 9 character matrix*
Color/Graphics
Text mode:
16 colors*
256 characters and symbols in ROM*
Graphics mode:
4-color resolution: 320h x 200v*
Black & white resolution: 640h x 200v*
Simultaneous graphics & text capability*
Communications
RS-232-C interface
Asynchronous (start/stop) protocol
Up to 9600 bits per second

It's a computer that has reached a truly personal scale in size and in price: starting at less than $1,600† for a system that, with the addition of one simple device, hooks up to your home TV and uses your audio cassette recorder.

For flexibility, performance and ease of use, no other personal computer offers as many advanced features to please novice and expert alike (see the box).

Features like high resolution color graphics. Ten, user-defined function keys. The kind of expandability that lets you add a printer for word processing, or user memory up to 256KB. Or BASIC and Pascal languages that let you write your own programs. And a growing list of superior programs like VisiCalc,™ selected by IBM to match the quality and thoughtfulness of the system's total design.

This new system will be sold through channels which meet our professional criteria: the nationwide chain of 150 ComputerLand® stores, and Sears Business Systems Centers. Of course, our own IBM Product Centers will sell and service the system. And the IBM Data Processing Division will serve those customers who want to purchase in quantity.

Experience the IBM Personal Computer. You'll be surprised how quickly you feel comfortable with it. And impressed with what it can do for you.

IBM®

For the IBM Personal Computer dealer nearest you, call (800) 447-4700. In Illinois, (800) 322-4400.

†This price applies to IBM Product Centers. Prices may vary at other stores.
VisiCalc is a trademark of Personal Software, Inc.

CIRCLE 3

[Personal Computing Ad, 11/81]

Microprogramarea: anii optzeci

- Evoluția a născut mașini de calcul și mai complexe
  - Seturile complexe de instrucțiuni au dus la necesitatea de-a implementa subrutine și lucru cu stiva direct în µcod
  - Necesitatea de-a repara defectele din programele de control era în conflict cu natura read-only a µROM-ului
  - →Writable Control Store (WCS) (B1700, QMachine, Intel i432, …)
- Odată cu apariția tehnologiei VLSI limitările de viteză și mărime pentru ROM & RAM devin invalide →complexitate sporită
- Compilatoarele mai bune făceau ca instrucțiunile complexe să fie mai puțin importante.
- Numeroase inovații la nivel de micro-arhitectură, e.g., pipelining, caches & buffers, au făcut ca operațiile de transfer între registre pe mai mulți cicli de ceas să fie perimate

Analiza mașinilor cu microcod

- John Cocke (& grupul lui) IBM
  - Lucrau pe un procesor cu pipeline simplu, 801 cu compilatoare avansate la IBM
  - Au portat compilatorul experimental PL.8 pe IBM 370; foloseau DOAR instrucțiuni simple de transfer reg-reg și load/store similare 801
  - Codul produs rula mai repede decât codul produs de orice alt compilator existent pe seria 370 care folosea TOATE instrucțiunile!(până la 6MIPS în loc de 2MIPS cu compilatoarele standard)
- Emer, Clark, at DEC
  - Măsura performanța VAX-11/780 folosind hardware extern
  - A descoperit ca era defapt o mașină de calcul de 0.5MIPS, deși se presupunea de obicei că e de 1MIPS
  - 20% din instrucțiunile VAX erau responsabile pentru 60% din microcod, și doar 0.2% din timpul de execuție!
- VAX8800
  - Control Store: 16K*147b RAM, Unified Cache: 64K*8b RAM
  - 4.5x mai mult RAM pentru microcod decât pentru cache RAM!

Schimbările tehnologice pentru IC-uri

- Logică, RAM, ROM, toate implementate cu tranzistoare MOS
- RAM semiconductor ~ aceeași viteză ca ROM

Nanocoding

Embedded Systems Laboratory

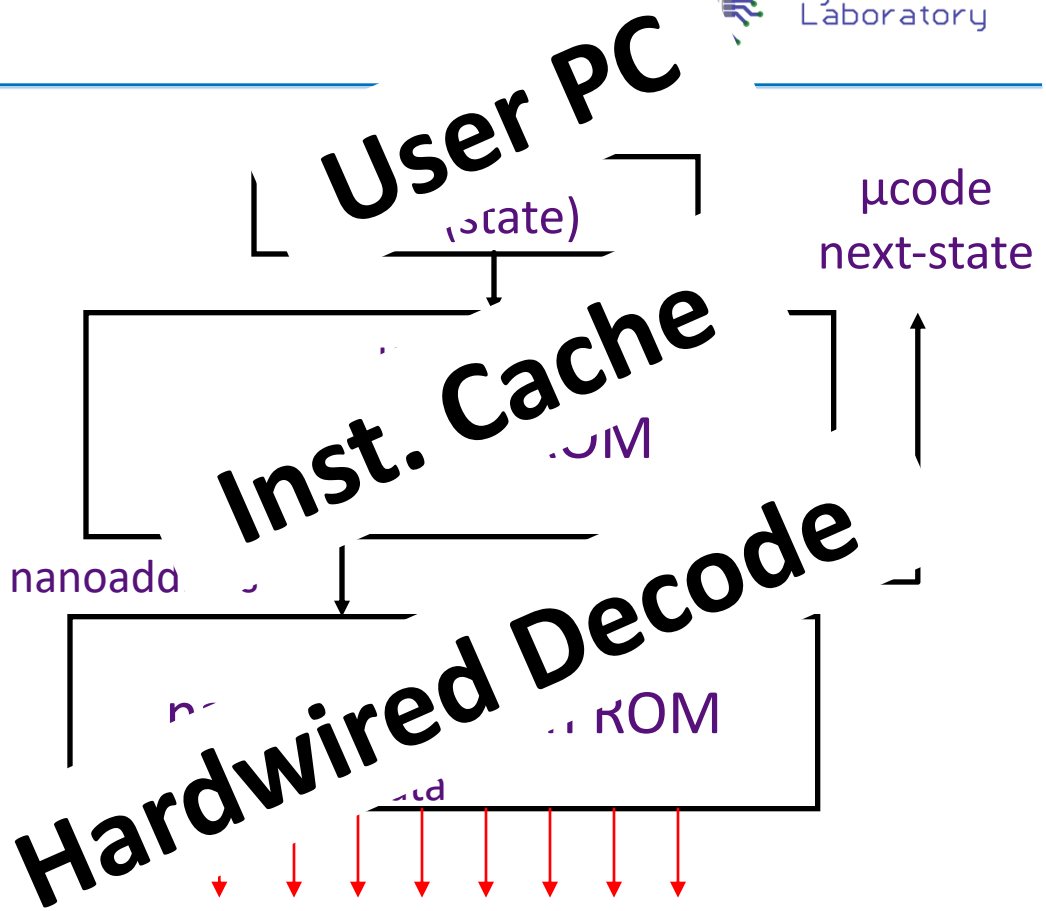
Explo[illegible] nțele recure[illegible] semnale de comanda din µcod, e.g.,

ALU_0 A <- Reg[rs1]

...

$ALUi_0$ A <- Reg[rs1]

...



- MC68000 avea un µcod [illegible] 17 biți ce conținea ori un µjump pe 10 biți, ori un pointer de 9-biți la o nanoinstrucțiune
  - Nanoinstrucțiunile aveau 68 biți lățime și decodate dădeau 196 de semnale de control

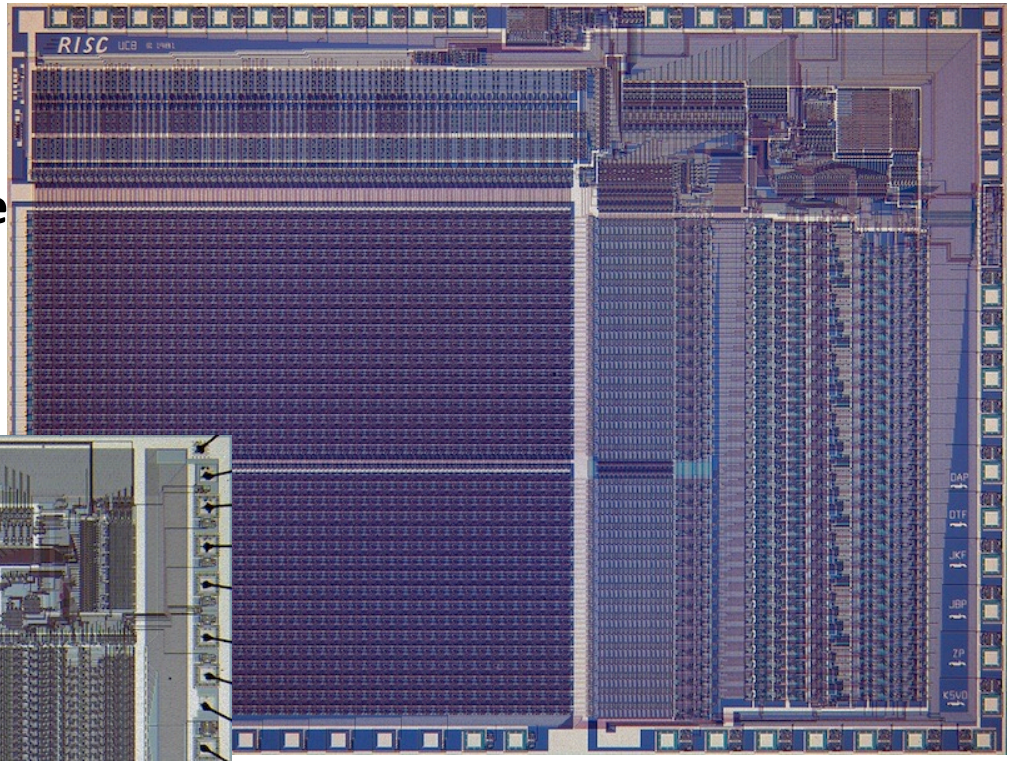
De la CISC la RISC

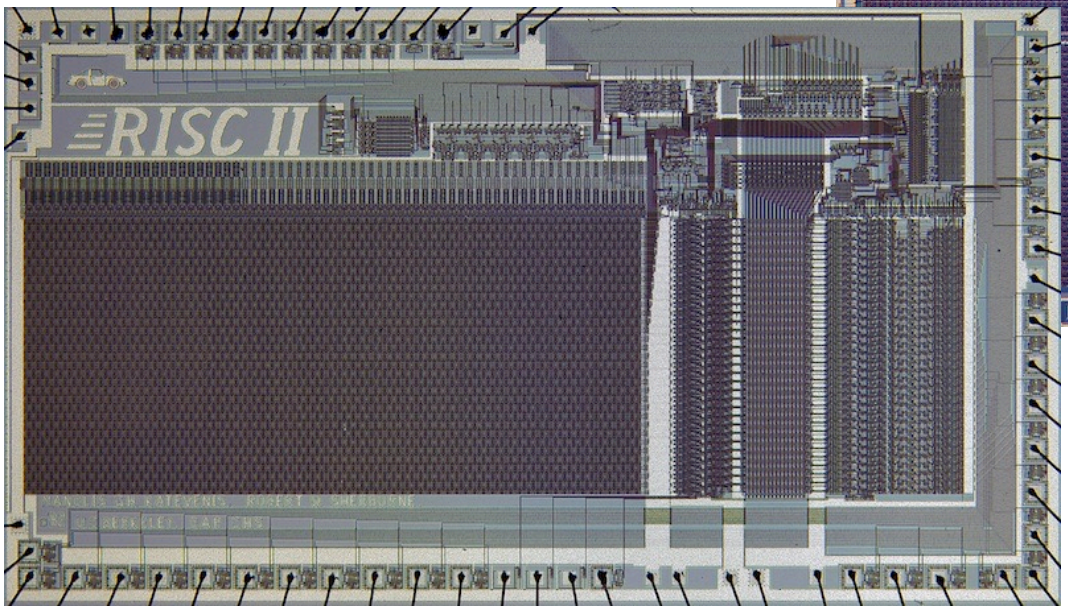
- Folosește RAM rapid pentru a construi un instruction *cache* rapid pentru instrucțiuni vizibile utilizatorului, nu microrutine fixate in hardware
  - Conținutul memoriei rapide se schimbă ca să acomodeze noile nevoi ale aplicațiilor
- Folosește un ISA simplu pentru a permite o implementare cablată de bandă de asamblare
  - Majoritatea codului compilat folosea doar o mică parte din instrucțiunile CISC disponibile
  - Codificarea simpificată permitea implementarea cu pipeline
- Beneficii suplimentare
  - La începutul anilor ‘80, se puteau în sfârșit integra o magistrală de date de 32 de biți și memorii cache de mici dimensiuni într-un singur chip

Chip-uri RISC Berkeley

Embedded Systems Laboratory

RISC-I (1982) Conține 44,420 tranzisotare, fabricat în 5 µm NMOS, suprafață 77 mm², ceas de 1 MHz. Probabil primul RISC VLSI.





RISC-II (1983) conține 40,760 tranzistoare, fabricat în 3 µm NMOS, ceas de 3 MHz, și suprafața de 60 mm².

"Iron Law" of Processor Performance

$$\frac{\text{Time}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} * \frac{\text{Cycles}}{\text{Instruction}} * \frac{\text{Time}}{\text{Cycle}}$$

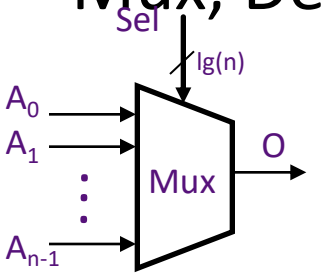
- Instructions per program depends on source code, compiler technology, and ISA
- Cycles per instructions (CPI) depends on ISA and µarchitecture
- Time per cycle depends upon the µarchitecture and base technology

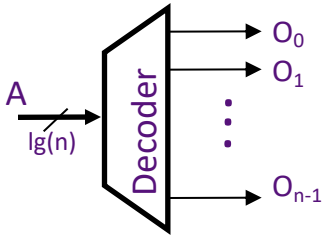
Acest curs →

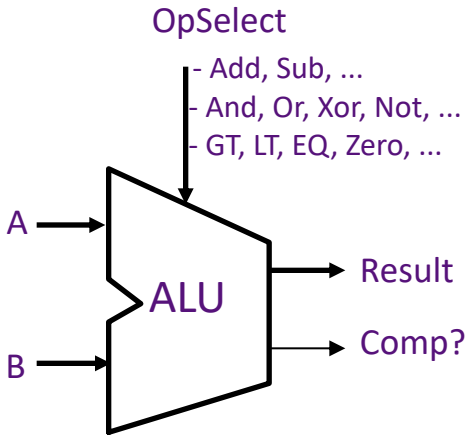
Microarchitectură	CPI	cycle time
Microcodată	>1	short
Single-cycle unpipelined	1	long
Pipelined	1	short

Elemente hardware

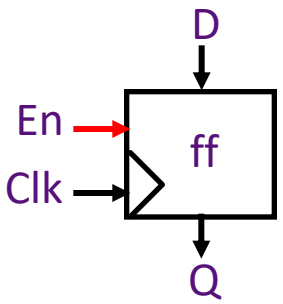
- Circuite combinaționale
  - Mux, Decoder, ALU, ...

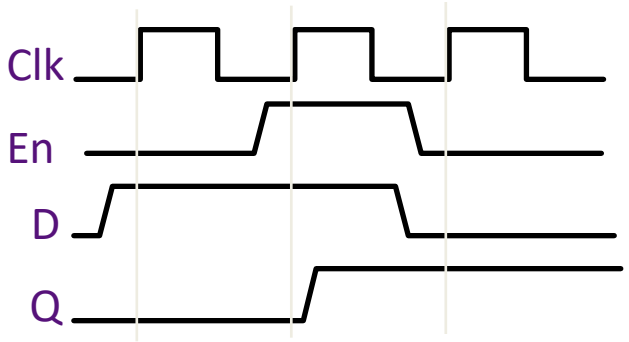






- Elemente sincrone
  - Flipflop, Register, Register file, SRAM, DRAM

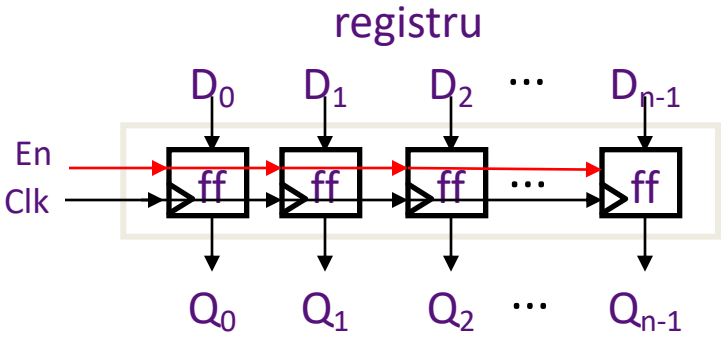


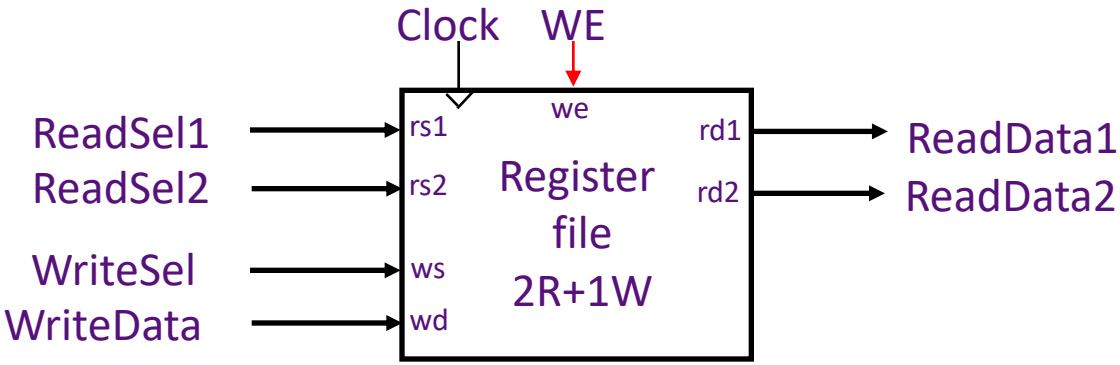


Edge-triggered: Datele eșantionate pe front crescător

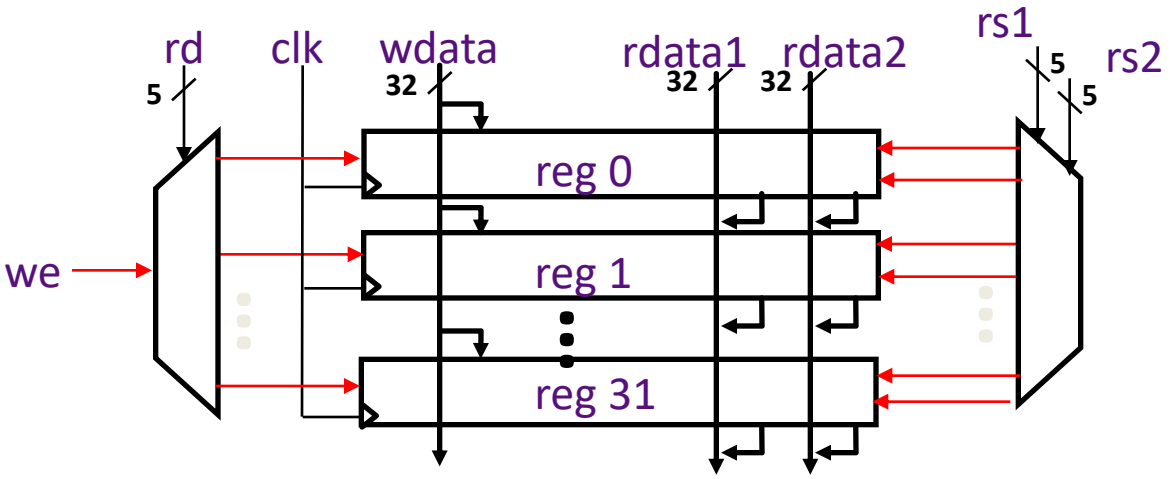
Tabele de registre

- Citirile se fac combinațional



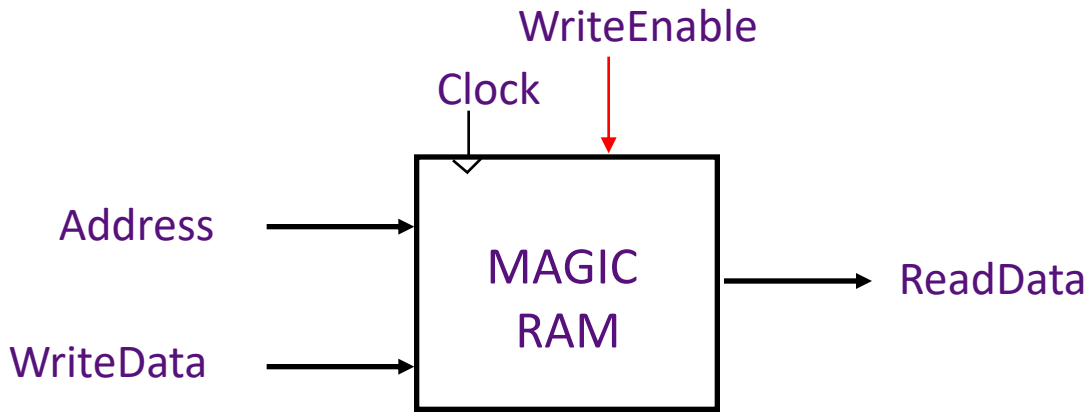


Implementarea unei tabele de registre



- Instrucțiunile cu întregi pentru RISC-V au cel mult doi operanzi sursă în registrele generale

Un model simplu pentru memorie



Citirile și scrierile se fac întotdeauna într-un singur ciclu

- o citire poate fi făcută oricând (i.e. Combinațional)
- o scriere este făcută pe frontul pozitiv al ceasului

liniile de adresă și de date trebuie să fie stabile pe frontul ceasului

Mai târziu vom prezenta un model mai realist de memorie

Implementarea RISC-V

Single-cycle per instruction datapath & control logic

Execuția instrucțiunilor

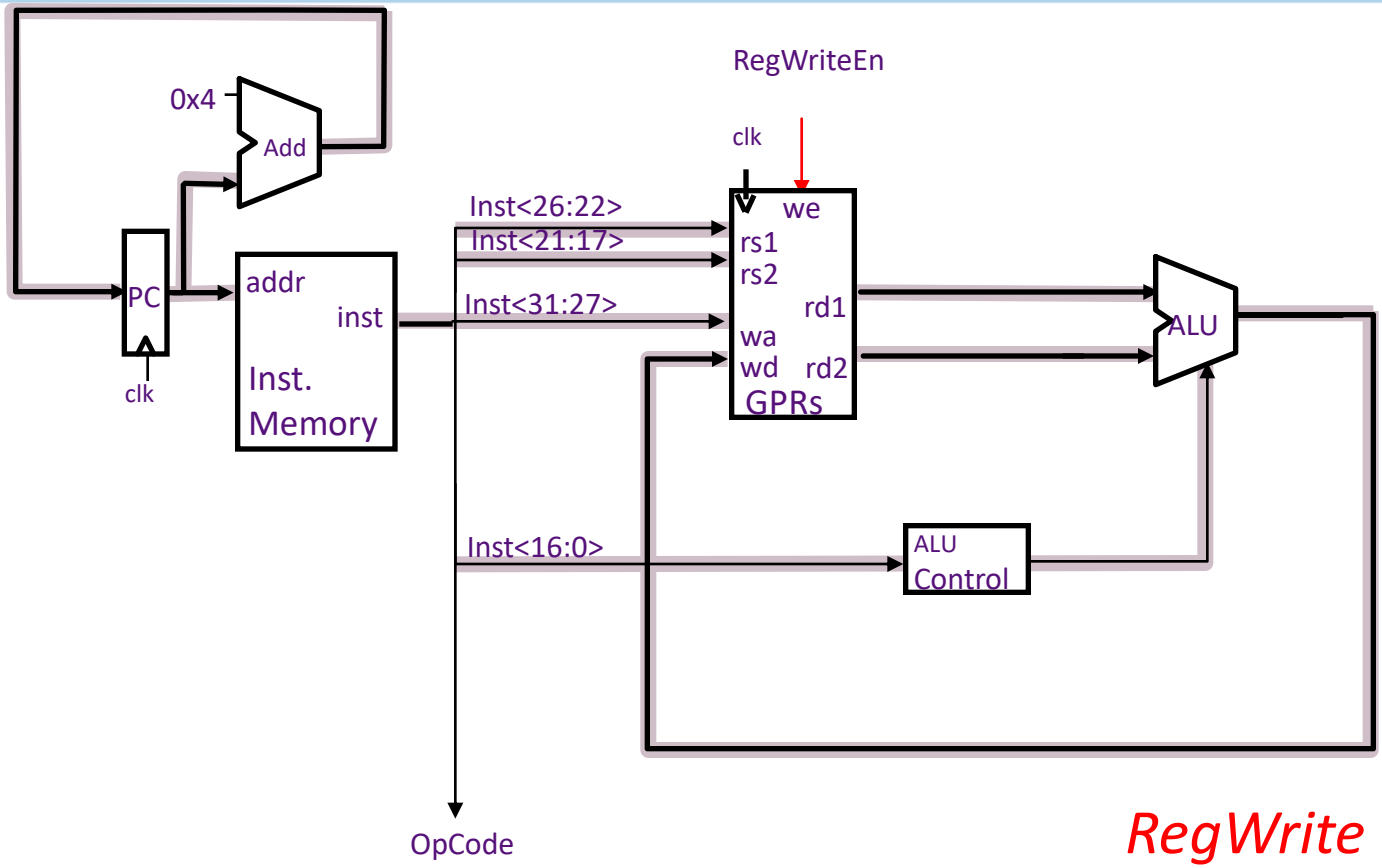
Embedded Systems Laboratory

O execuție implică

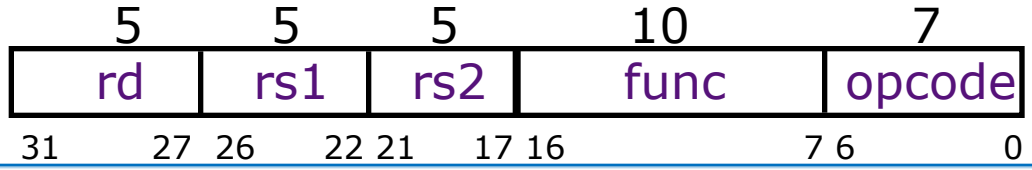
1. Instruction fetch
2. Decode and register fetch
3. Operații UAL
4. Operații cu memoria (opțional)
5. Write back (opțional)

și calculul adresei instrucțiunii următoare

Transfer date: Instrucțiuni Reg-Reg ALU

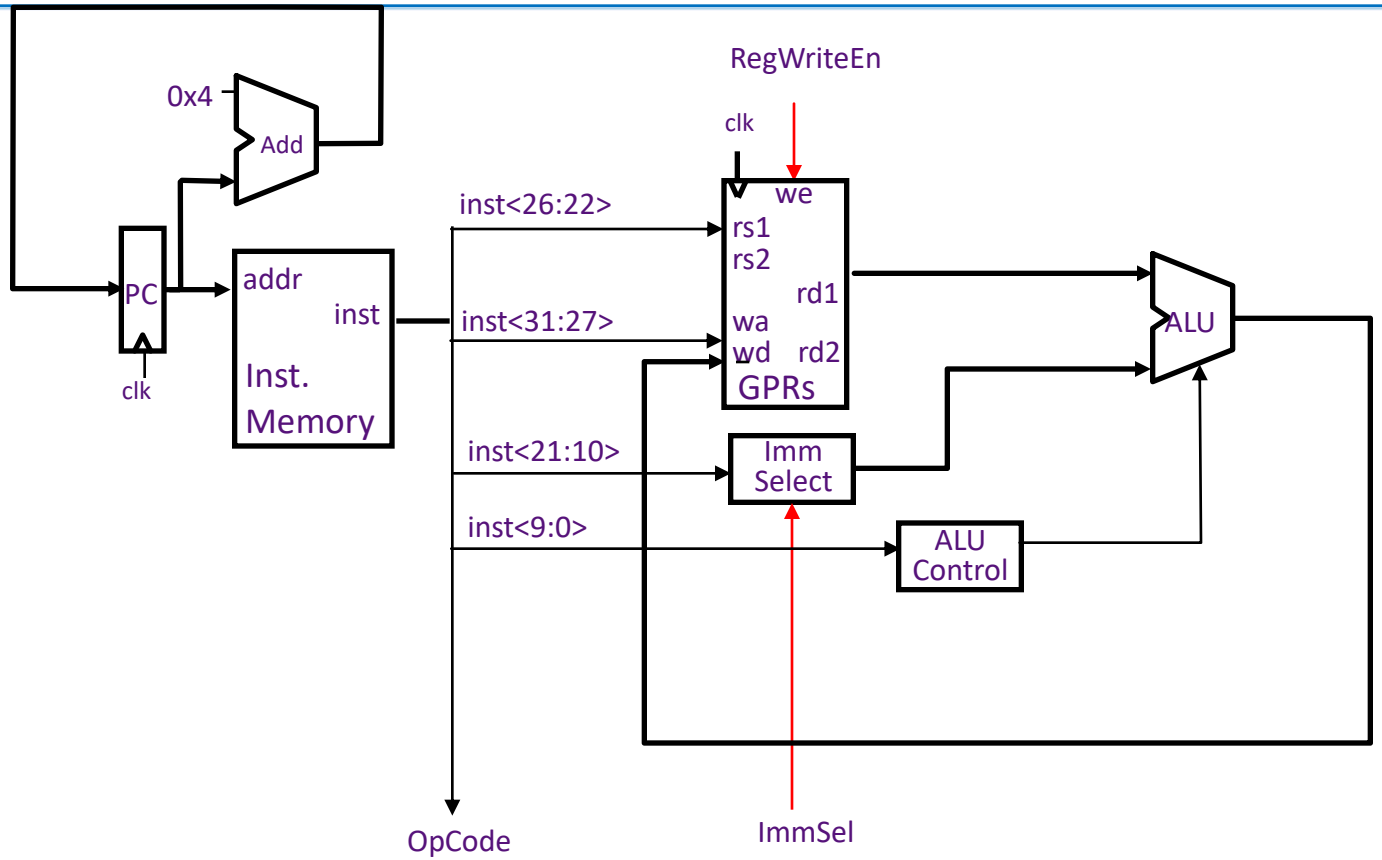


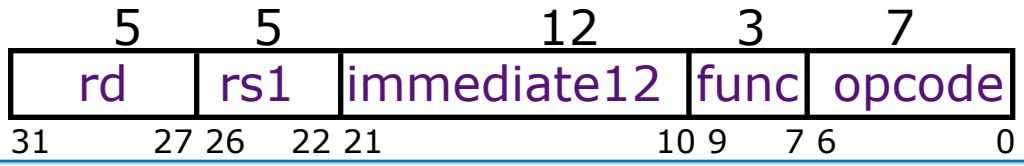
RegWrite Timing?



rd ← (rs1) func (rs2)

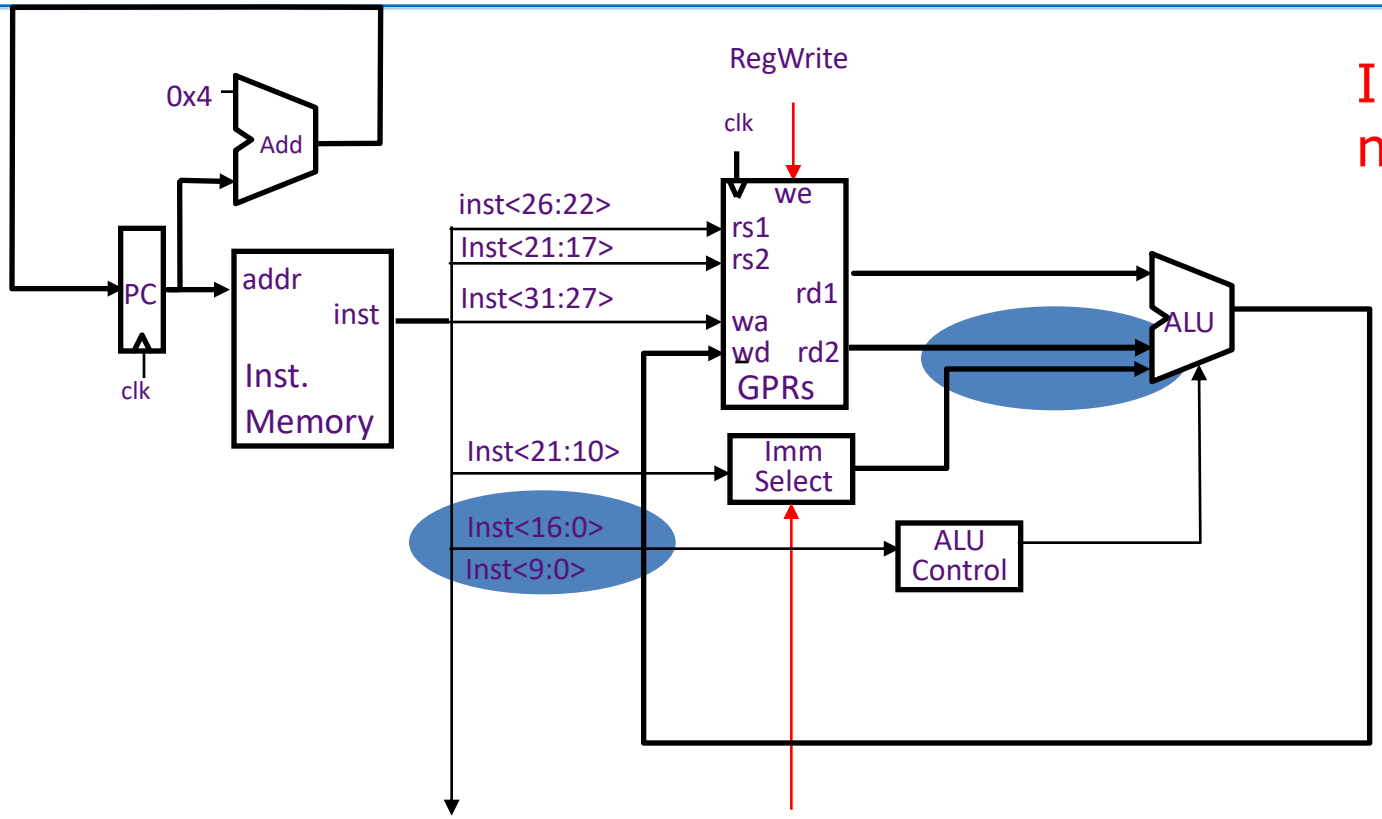
Transfer de date: Instrucțiuni Reg-Imm ALU





rd ← (rs1) op immediate

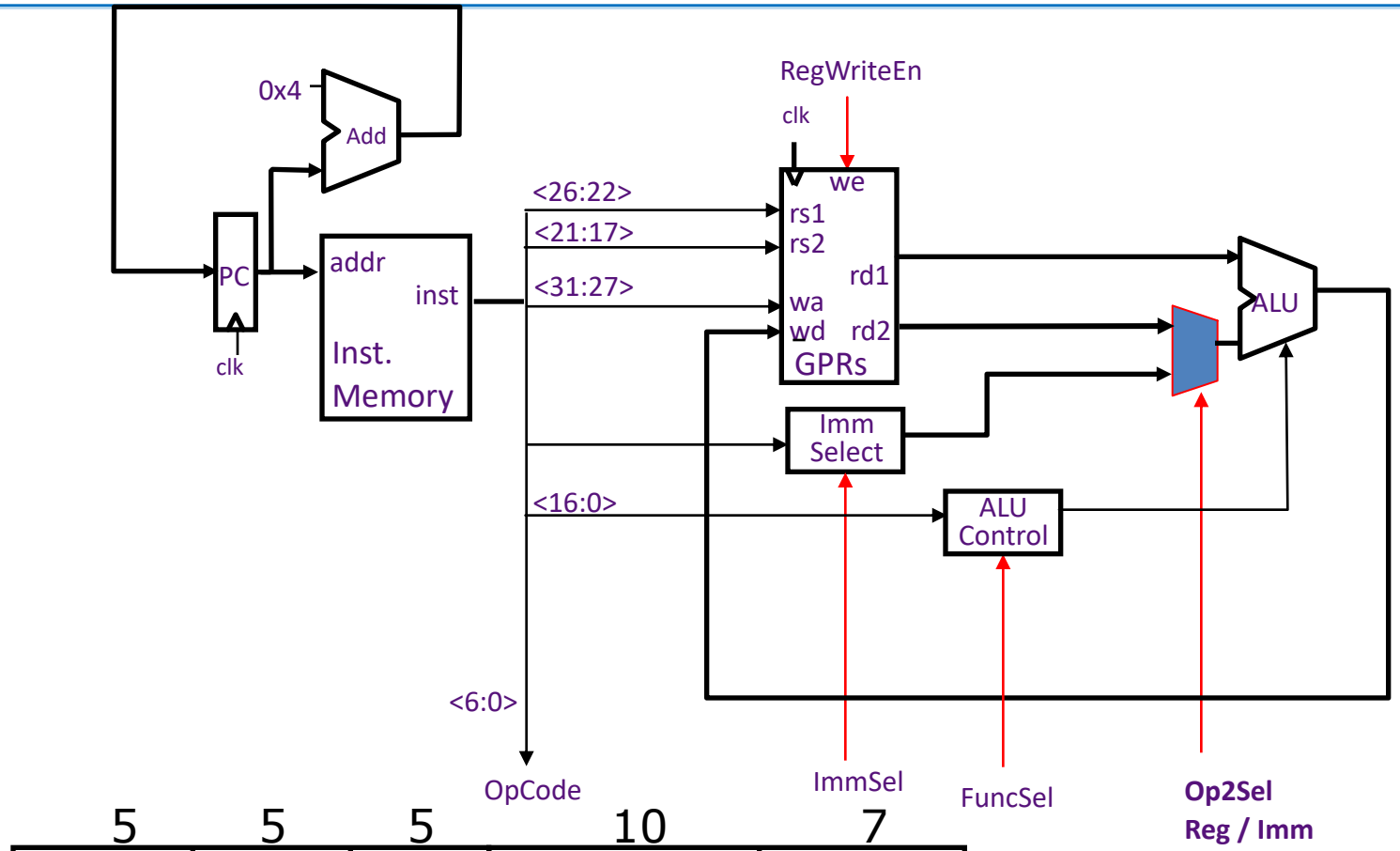
Conflicte în transferul de date



Introducem mux-uri

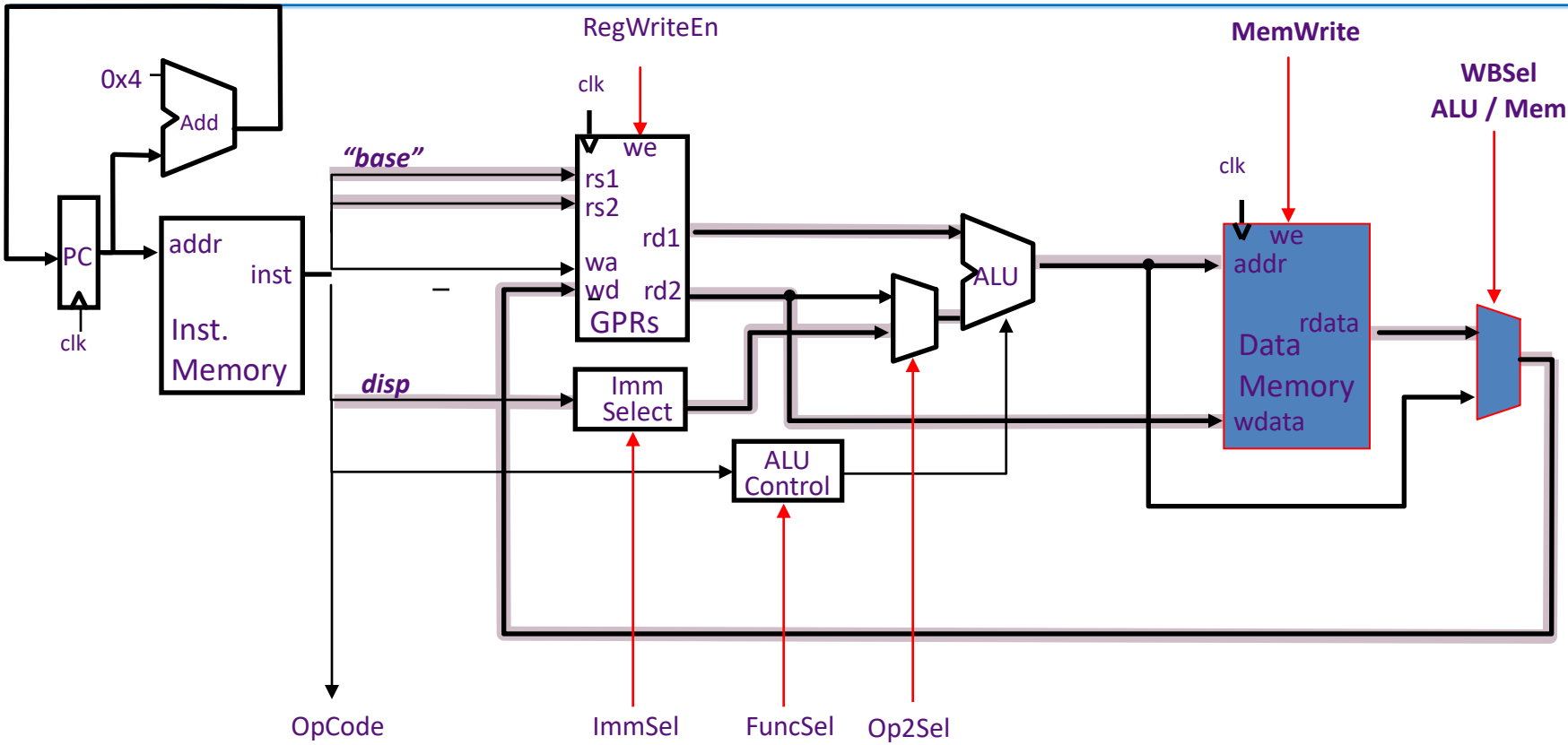
5	5	5	10	7	
rd	rs1	rs2	func	opcode	rd ← (rs1) func (rs2)
rd	rs1	immediate		func	opcode	rd ← (rs1) op immediate

Transfer date pentru instrucțiuni cu UAL



5	5	5	10	7	
rd	rs1	rs2	func	opcode	rd ← (rs1) func (rs2)
rd	rs1	immediate	func	opcode	rd ← (rs1) op immediate

Instrucțiuni Load/Store



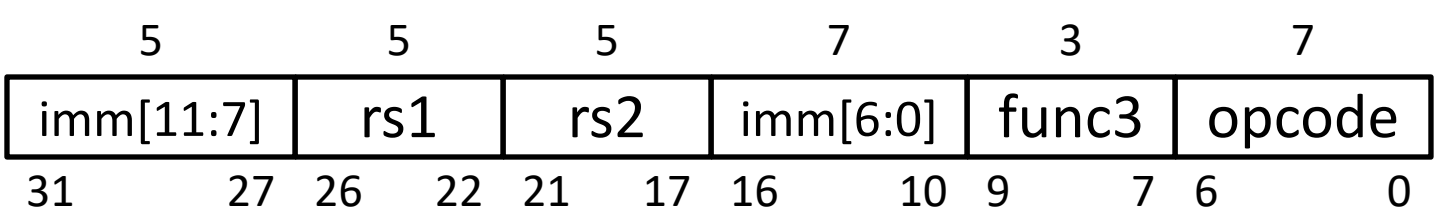
5	5	5	7	3	7	
imm	rs1	rs2	imm	func	opcode	Store
rd	rs1	immediate		func	opcode	Load

Addressing Mode
(rs) + displacement

rs1 este registrul de bază

rd este destinația pt Load, rs2 este sursa de date pentru un Store

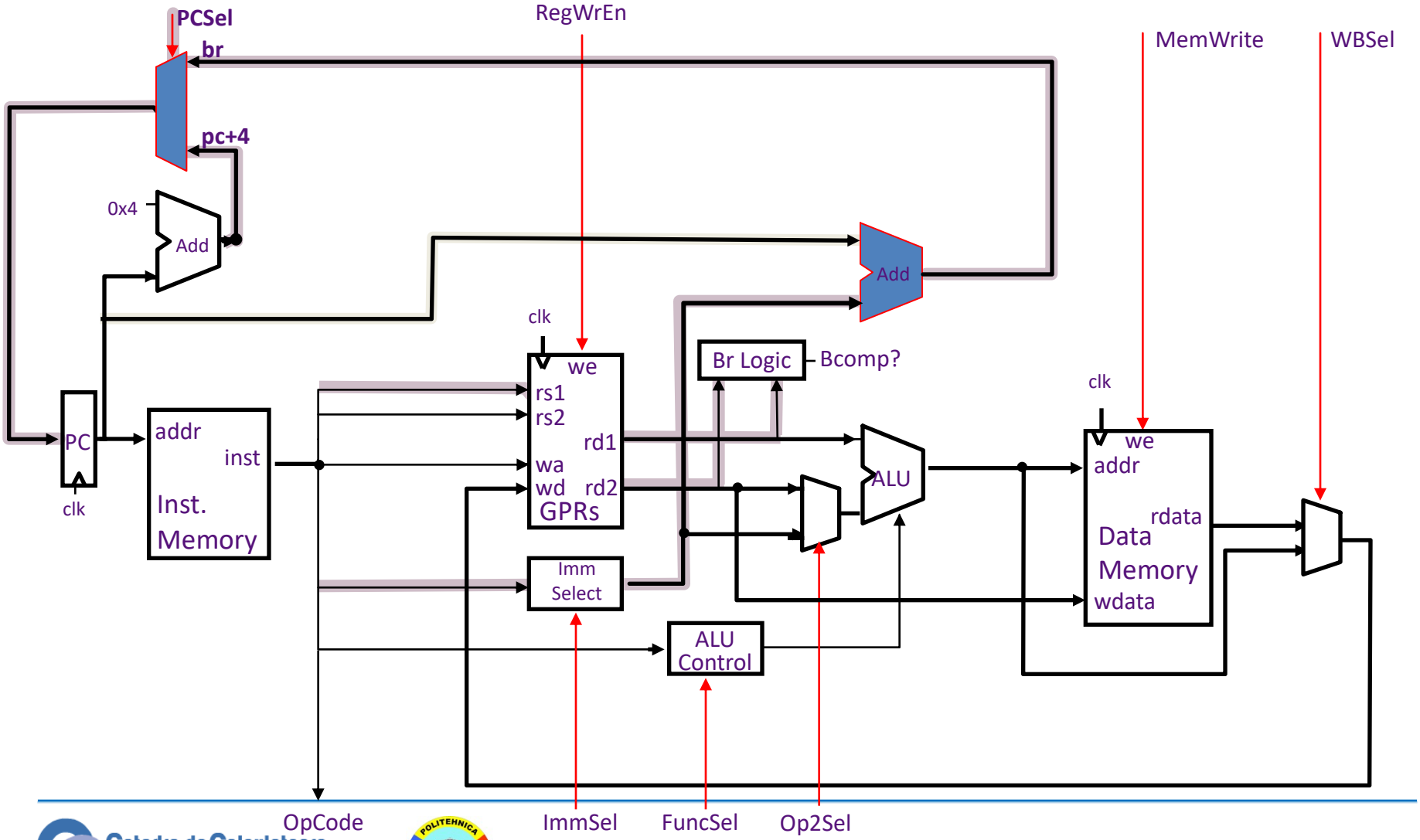
Salturi condiționale la RISC-V



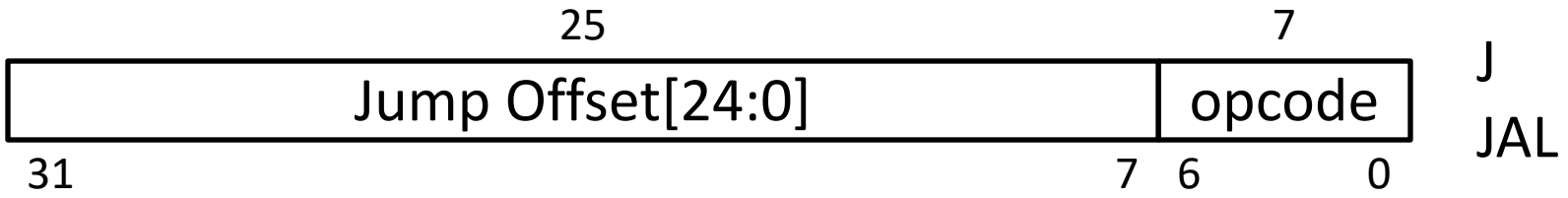
BEQ/BNE
BLT/BGE
BLTU/BGEU

- Compară două registre întregi pentru egalitate (BEQ/BNE) sau mărime cu semn (BLT/BGE) sau fără semn (BLTU/BGEU)
- Valoarea imediată de 12 biți stipulează adresa țintă de salt ca fiind un offset cu semn pentru PC, în unități de 16 biți (i.e., shift left cu 1 apoi adaugă la PC).

Salturi condiționale (BEQ/BNE/BLT/BGE/BLTU/BGEU)

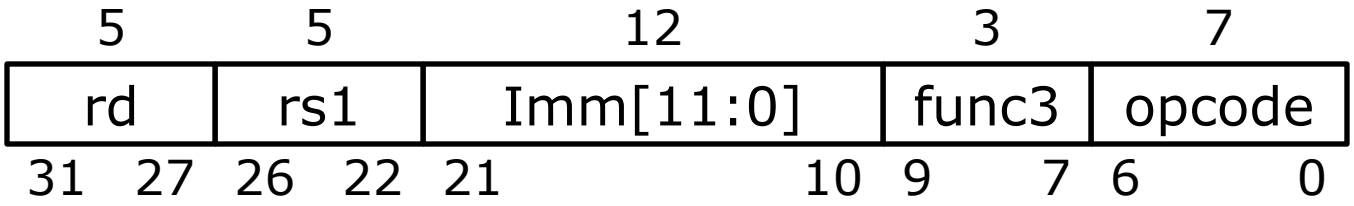


Salturi necondiționale la RISC-V



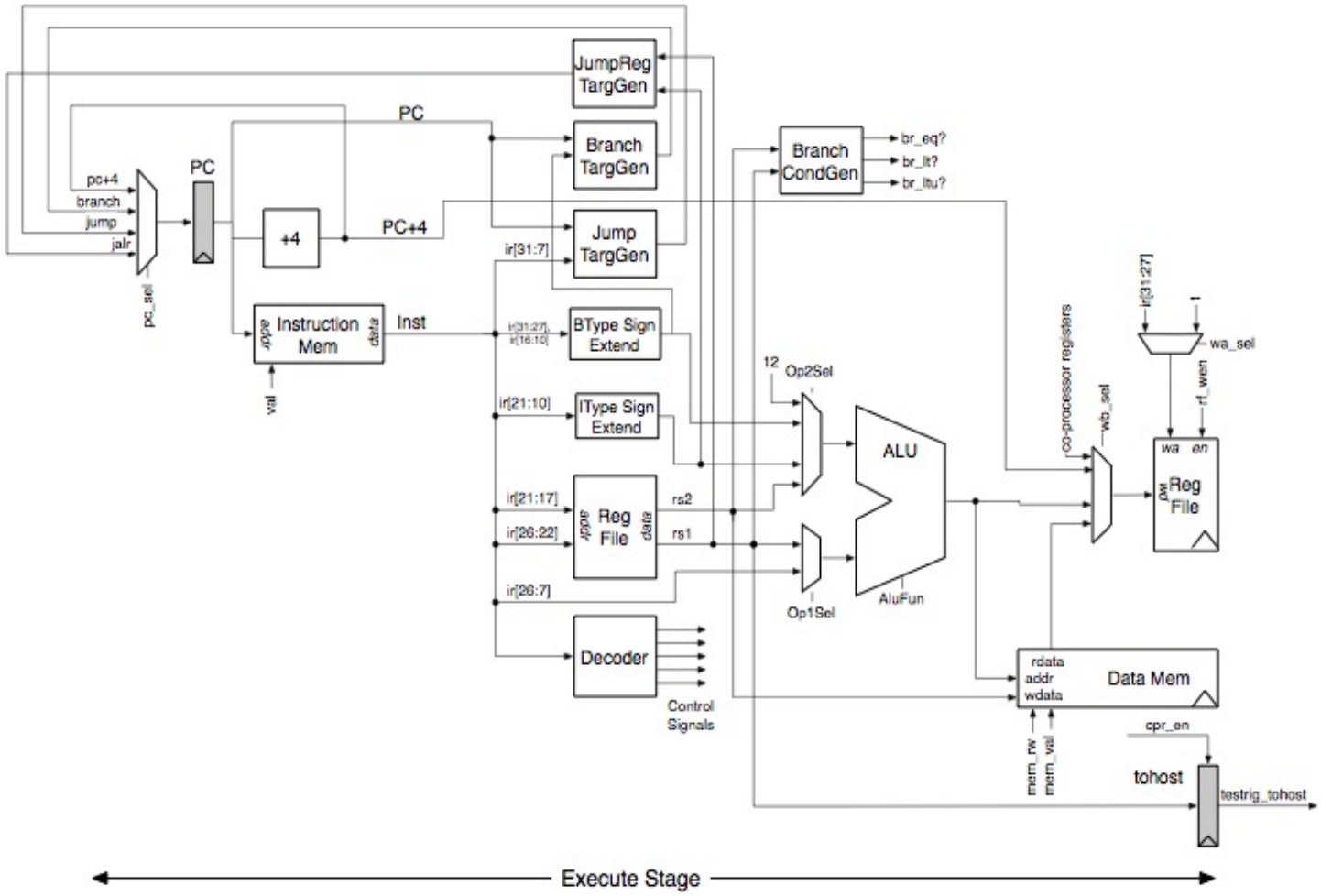
- Valoarea imediată de 25 biți specifică adresa de jump ca fiind un offset cu semn al PC, în unități de 16 biți cu semn (i.e., shift left cu 1 apoi adună la PC). (+/- 16MB)
- JAL este o subrutină care salvează și adresa de întoarcere (PC+4) în registrul x1

Salturi indirecte prin registre la RISC-V

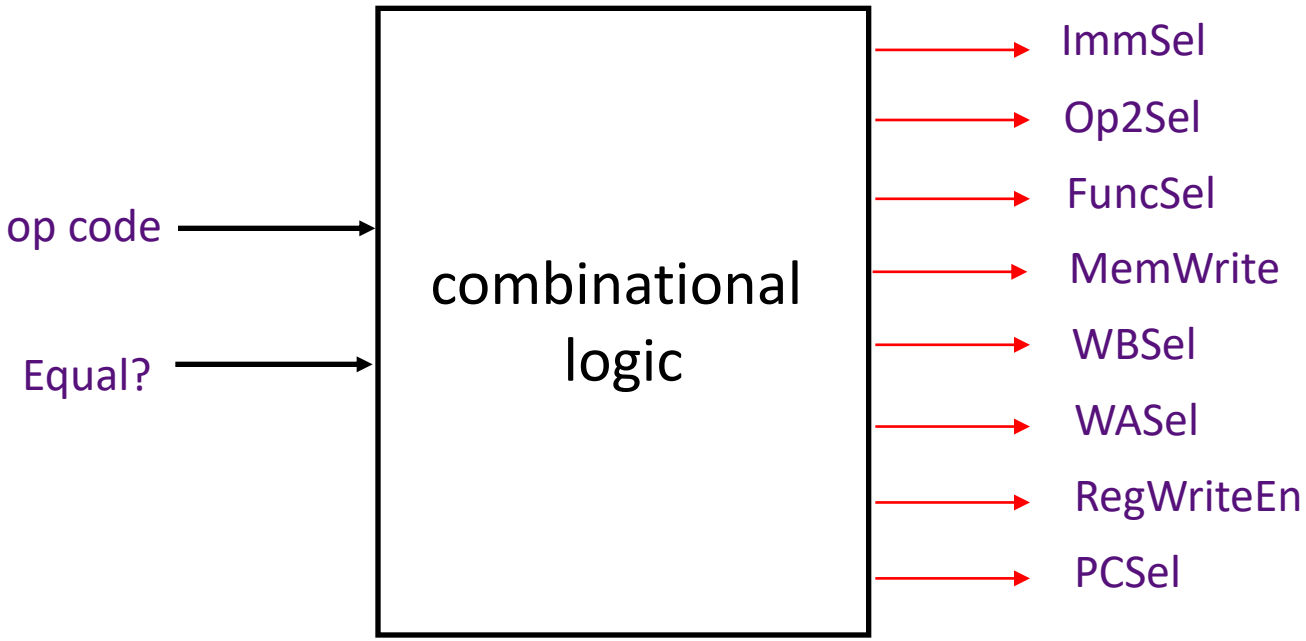


- Salturile la adresa țintă sunt făcute prin adunarea unui offset de 12 biți (*ne-shiftat* cu 1)registrului rs1
- Adresa de întoarcere (PC+4) este scrisă în registrul rd (poate fi **x0** dacă valoarea nu este necesară)
- Instrucțiunea RDNPC scrie adresa de întoarcere în rd fără să facă saltul efectiv (folosită pentru dynamic linking)

Full RISCV1Stage Datapath



Controlul cablat este logică combinațională pură



Controlul UAL și extensia pentru valori imediate

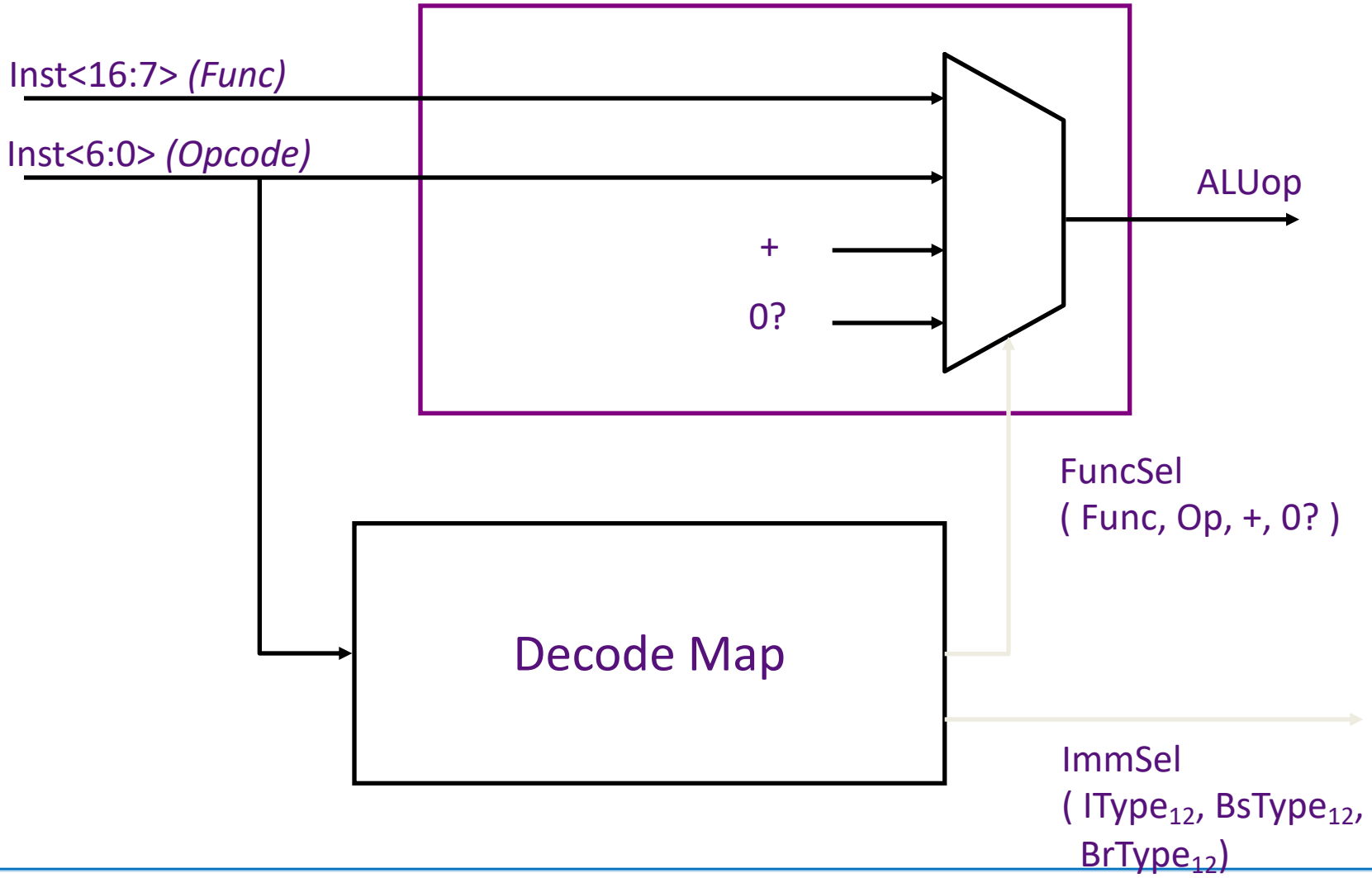


Tabela pentru controlul cablat

Embedded Systems Laboratory

Opcode	ImmSel	Op2Sel	FuncSel	MemWr	RFWen	WBSel	WASel	PCSel
ALU	*	Reg	Func	no	yes	ALU	rd	pc+4
ALUi	$IType_{12}$	Imm	Op	no	yes	ALU	rd	pc+4
LW	$IType_{12}$	Imm	+	no	yes	Mem	rd	pc+4
SW	$BsType_{12}$	Imm	+	yes	no	*	*	pc+4
BEQ_{true}	$BrType_{12}$	*	*	no	no	*	*	br
BEQ_{false}	$BrType_{12}$	*	*	no	no	*	*	pc+4
J	*	*	*	no	no	*	*	jabs
JAL	*	*	*	no	yes	PC	X1	jabs
JALR	*	*	*	no	yes	PC	rd	rind

Op2Sel= Reg / Imm
WASel = rd / X1

WBSel = ALU / Mem / PC
PCSel = pc+4 / br / rind / jabs

Controlul cablat într-un singur ciclu

Embedded Systems Laboratory

Presupunem că perioada ceasului este suficient de lungă pentru ca toți pașii următori să fie executați:

1. Instruction fetch
2. Decode and register fetch
3. ALU operation
4. Data fetch if required
5. Register write-back setup time

$$\Rightarrow t_C > t_{IFetch} + t_{RFetch} + t_{ALU} + t_{DMem} + t_{RWB}$$

Pe frontul pozitiv al semnalului următor de ceas sunt actualizate PC, tabela de registre și memoria

Recapitulare

- Microcodarea a devenit mai neatragatoare pe măsură ce diferențele de viteză dintre RAM și ROM s-au anulat și circuitele logice au putut fi implementate în aceeași tehnologie ca și memoria
- Seturile complexe de instrucțiuni sunt dificil de folosit intr-un pipeline așa că, pe măsură ce complexitatea circuitelor a crescut era foarte greu să crească și performanța lor
- Legea de fier descrie foarte bine diferența dintre RISC și CISC
  - Interschimbi instrucțiuni/program, cicli/instrucțiune și timp/ciclu
- ISA RISC este de tip load/store și e proiectată pentru implementări eficiente de pipeline
  - Foarte similar cu microcodul verticalizat
  - Inspirat de masinile Cray timpurii (CDC 6600/7600)
- ISA RISC-V va fi folosită în curs pentru a descrie restul conceptelor
  - Berkeley RISC chips: RISC-I, RISC-II, SOAR (RISC-III), SPUR (RISC-IV)

Acknowledgements

- Aceste slide-uri conțin materiale aparținând:
  - Arvind (MIT)
  - Krste Asanovic (MIT/UCB)
  - Joel Emer (Intel/MIT)
  - James Hoe (CMU)
  - John Kubiatowicz (UCB)
  - David Patterson (UCB)
  - Behrooz Parhami (UCSB)

- MIT material derived from course 6.823
- UCB material derived from course CS252