

# Calculatoare Numerice

– Cursul 10 –

## De la CISC la RISC

Facultatea de Automatică și Calculatoare  
Universitatea Politehnica București

# Din cursurile anterioare

---

- ISA este interfața hardware/software
  - Definește cum este văzut procesorul de către programator
  - Definește formatul instrucțiunilor (bit encoding) și semnificația 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

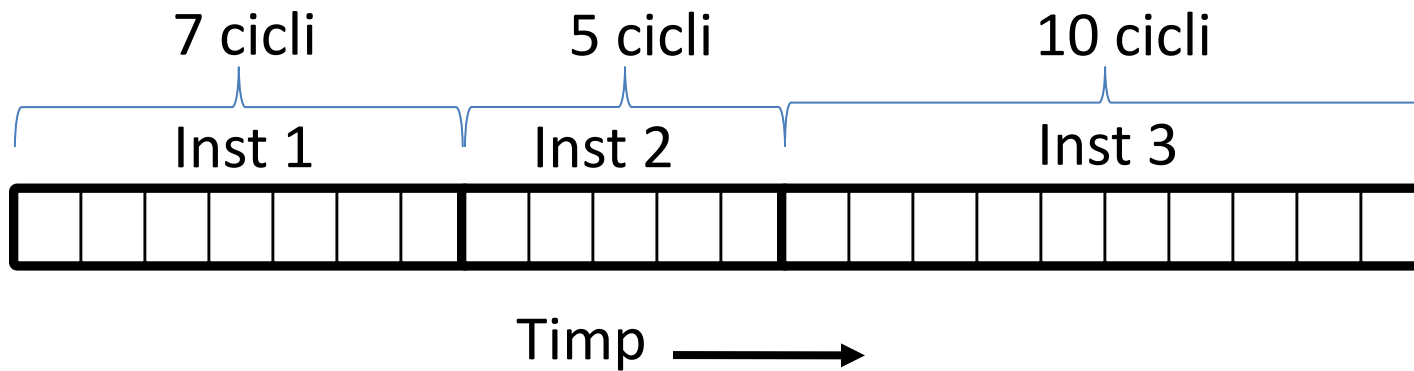
# “Legea de fier” a performanței calculatoarelor

---

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

- Numărul de instrucțiuni din program depinde de codul sursă, compilator și ISA
- Cicli pe instrucțiune (CPI) depinde de ISA și de arhitectură
- Timpul pe ciclu depinde de arhitectură și de tehnologia în care procesorul este construit.

# CPI pentru procesoare cu microcod



Numărul total de cicluri 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.

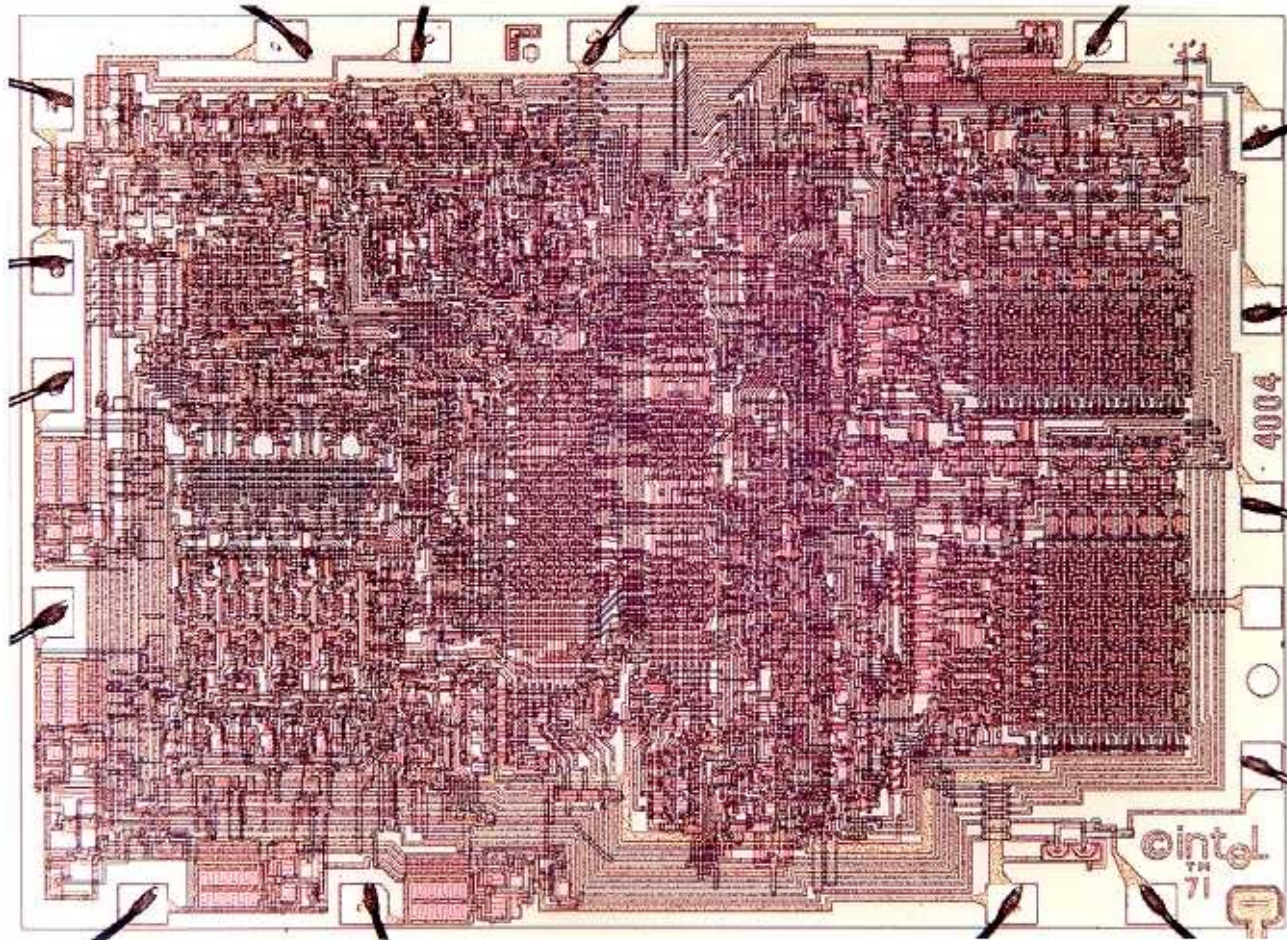
# Influențe tehnologice

---

- 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
- 8 $\mu$ m pMOS
- 2,300 tranzistoare
- 3 x 4 mm<sup>2</sup>
- Ceas 750kHz
- 8-16 cicli/inst.

Posibil datorită noii tehnologii de fabricație a circuitelor integrate

# Microprocesoarele în anii 70

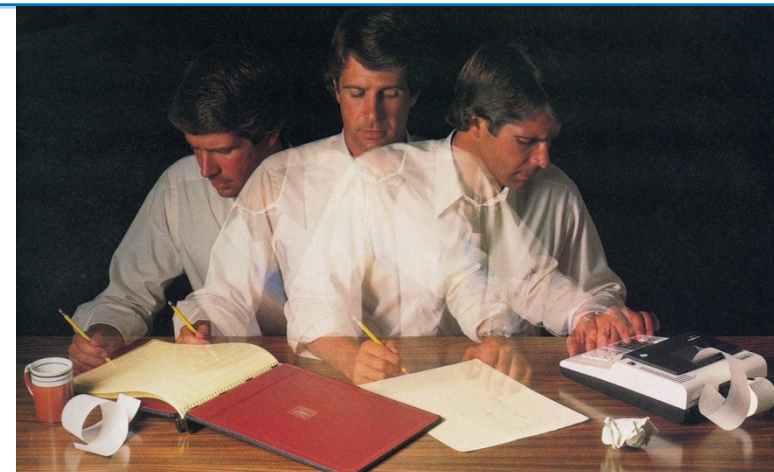
---

- 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!*



## Solve your personal energy crisis. Let VisiCalc™ Power do the work.

With a calculator, pencil and paper you can spend hours planning, projecting, writing, estimating, calculating, revising, erasing and recalculating as you work toward a decision.

Or with VisiCalc and your Apple® II you can explore many more options with a fraction of the time and effort you've spent before.

VisiCalc is a new breed of problem-solving software. Unlike prepackaged software that forces you into a computerized straight jacket, VisiCalc adapts itself to any numerical problem you have. You enter numbers, alphabetic titles and formulas on your keyboard. VisiCalc organizes and displays this information on the screen. You don't have to spend your time programming.

Your energy is better spent using the results than getting them.

Say you're a business manager and want to project your annual sales. Using the calculator, pencil and paper method, you'd lay out 12 months across a sheet and fill in lines and columns of figures on products, outlets, salespeople, etc. You'd calculate by hand the subtotals and summary figures. Then you'd start revising, erasing and recalculating.

With VisiCalc, you simply fill in the same figures on an electronic "sheet of paper" and let the computer do the work.

Once your first projection is complete, you're ready to use VisiCalc's unique, powerful recalculation feature. It lets you ask "What if?", examining new options and planning for contingencies. "What if" sales drop 20 percent in March? Just type in the sales figure. VisiCalc instantly updates all other figures affected by March sales.

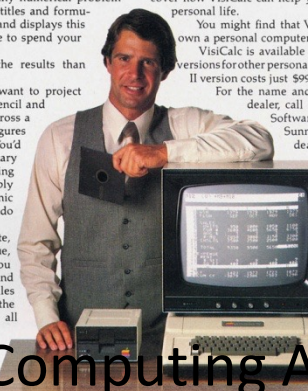
Or say you're an engineer working on a design problem and are wondering "What if that oscillation were damped by another 10 percent?" Or you're working on your family's expenses and wonder "What will happen to our entertainment budget if the heating bill goes up 15 percent this winter?" VisiCalc responds instantly to show you all the consequences of any change.

Once you see VisiCalc in action, you'll think of many more uses for its power. Ask your dealer for a demonstration and discover how VisiCalc can help you in your professional work and personal life.

You might find that VisiCalc alone is reason enough to own a personal computer.

VisiCalc is available now for Apple II computers, with versions for other personal computers coming soon. The Apple II version costs just \$99.50 and requires a 32k disk system.

For the name and address of your nearest VisiCalc dealer, call (408) 745-7841 or write to Personal Software, Inc., Dept. P, 592 Weddell Dr., Sunnyvale, CA 94086. If your favorite dealer doesn't already carry Personal Software products, ask him to give us a call.



**PERSONAL SOFTWARE**

TM-VisiCalc is a trademark of Personal Software, Inc.  
\*Apple is a registered trademark of Apple Computer, Inc.

[ 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

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



The IBM Personal Computer and me.



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

CIRCLE 3

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

[ 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  $\mu$ cod
  - Necesitatea de-a repara defectele din programele de control era în conflict cu natura read-only a  $\mu$ ROM-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

# Schimbările tehnologice pentru IC-uri

---

- Logică, RAM, ROM, toate implementate cu tranzistoare MOS
- RAM semiconductor  $\sim$  aceeași viteză ca ROM

# Nanocoding

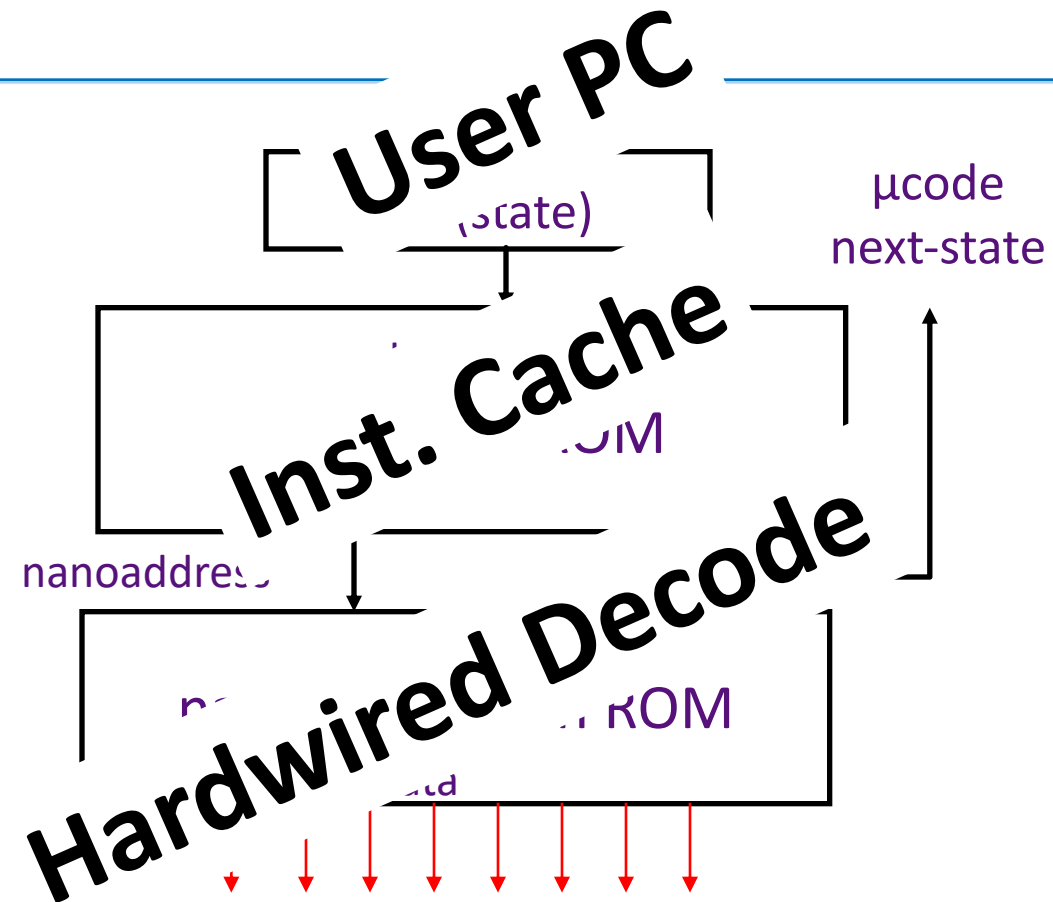
Exploare RISC  
 recurențele semnale de  
 comandă din  $\mu$ cod, e.g.,

ALU<sub>0</sub> A ← Reg[rs1]

...

ALU<sub>i</sub> A ← Reg[rs1]

...



- MC68000 avea un  $\mu$ cod de 17 biți ce conținea ori un  $\mu$ jump 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

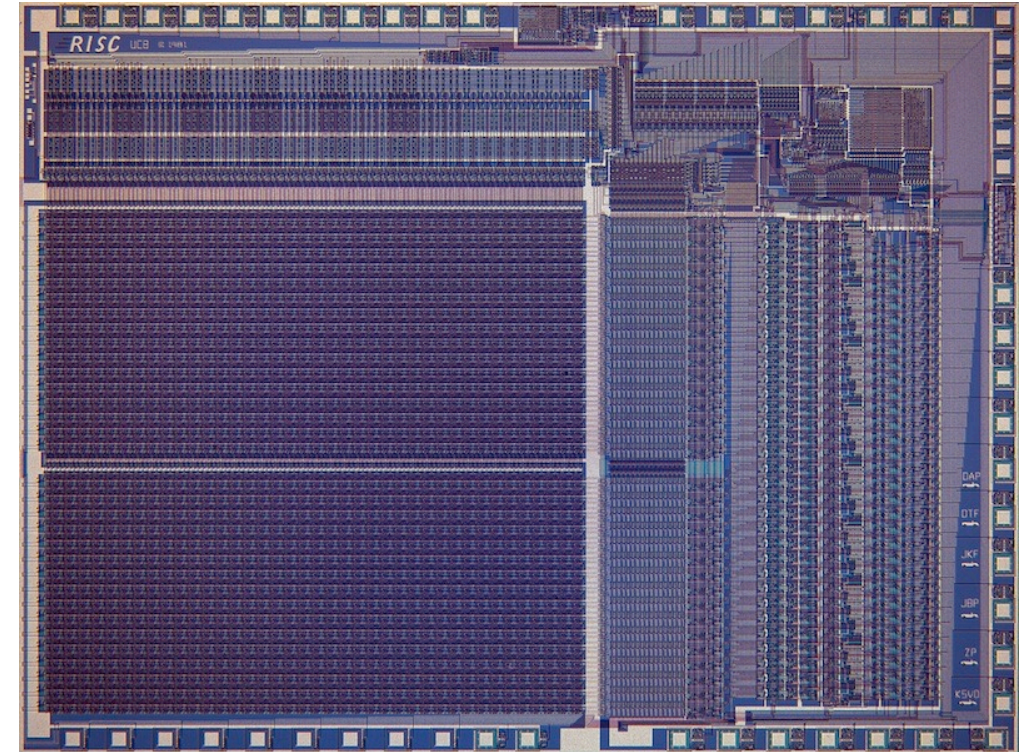
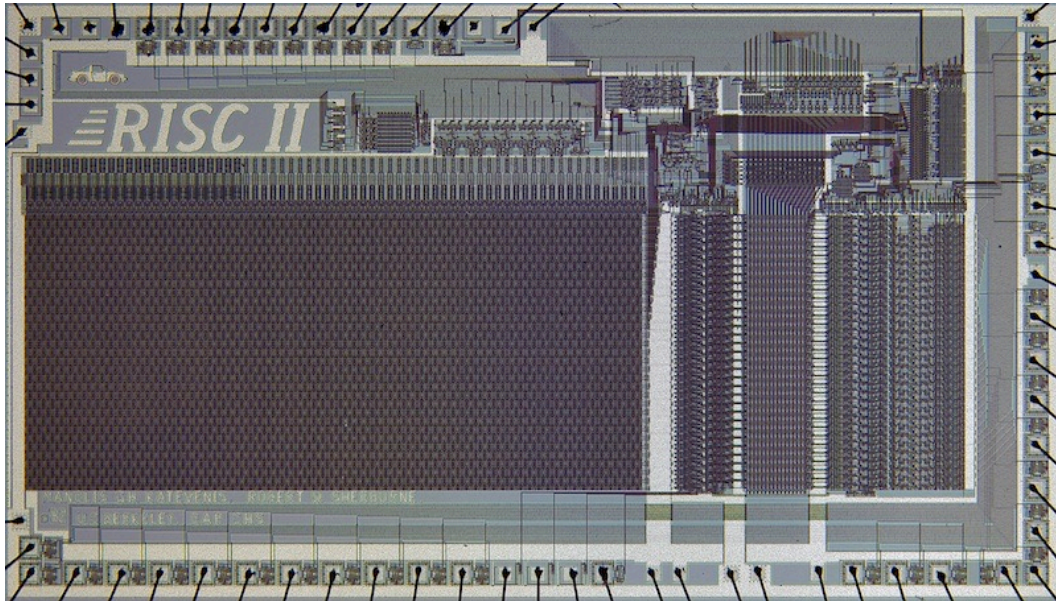
---

- Folosește RAM rapid pentru a construi un instruction *cache* rapid pentru instrucțiuni vizibile utilizatorului, nu microrutine fixate în 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 simplificată 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

**RISC-I (1982)** Conține 44,420 tranzistore, fabricat în 5  $\mu\text{m}$  NMOS, suprafață 77  $\text{mm}^2$ , ceas de 1 MHz. Probabil primul RISC VLSI.



**RISC-II (1983)** conține 40,760 tranzistoare, fabricat în 3  $\mu\text{m}$  NMOS, ceas de 3 MHz, și suprafața de 60  $\text{mm}^2$ .

# “Iron Law” of Processor Performance

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

- Numărul de instrucțiuni din program depinde de codul sursă, compiler și ISA
- Cicli pe instrucțiune (CPI) depinde de ISA și de μarchitectură
- Timpul pe ciclu depinde de μarchitectură și de tehnologia în care procesorul este construit.

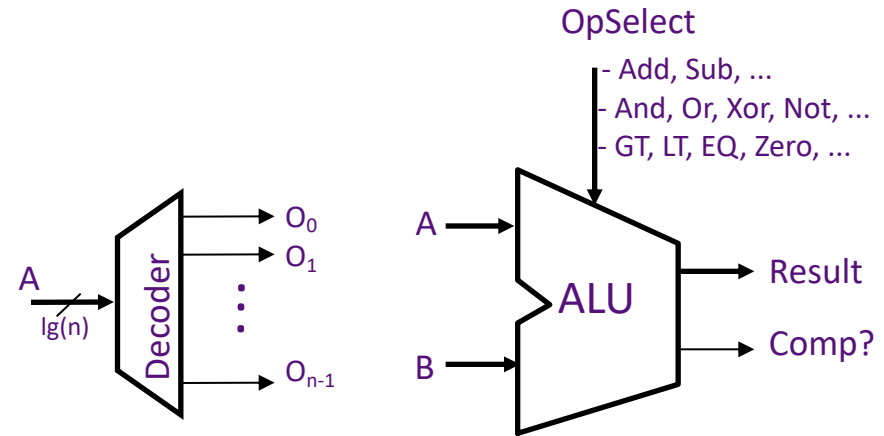
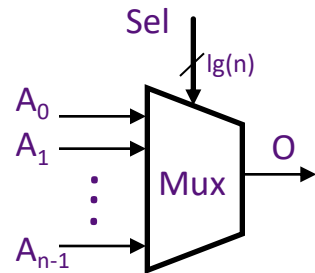
Acest curs →

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

# Elemente hardware

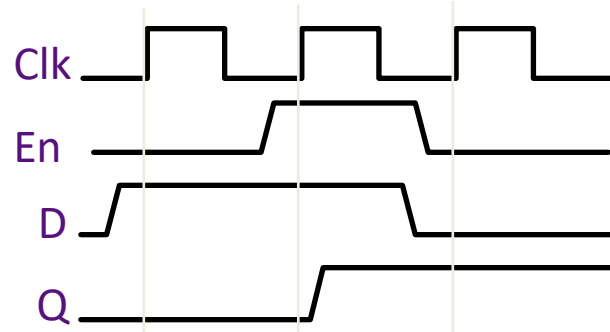
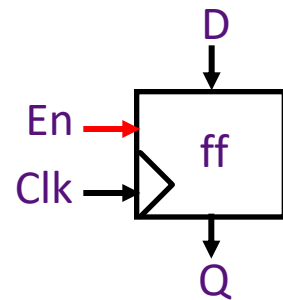
- Circuite combinaționale

- Mux, Decoder, ALU, ...



- Elemente sincrone

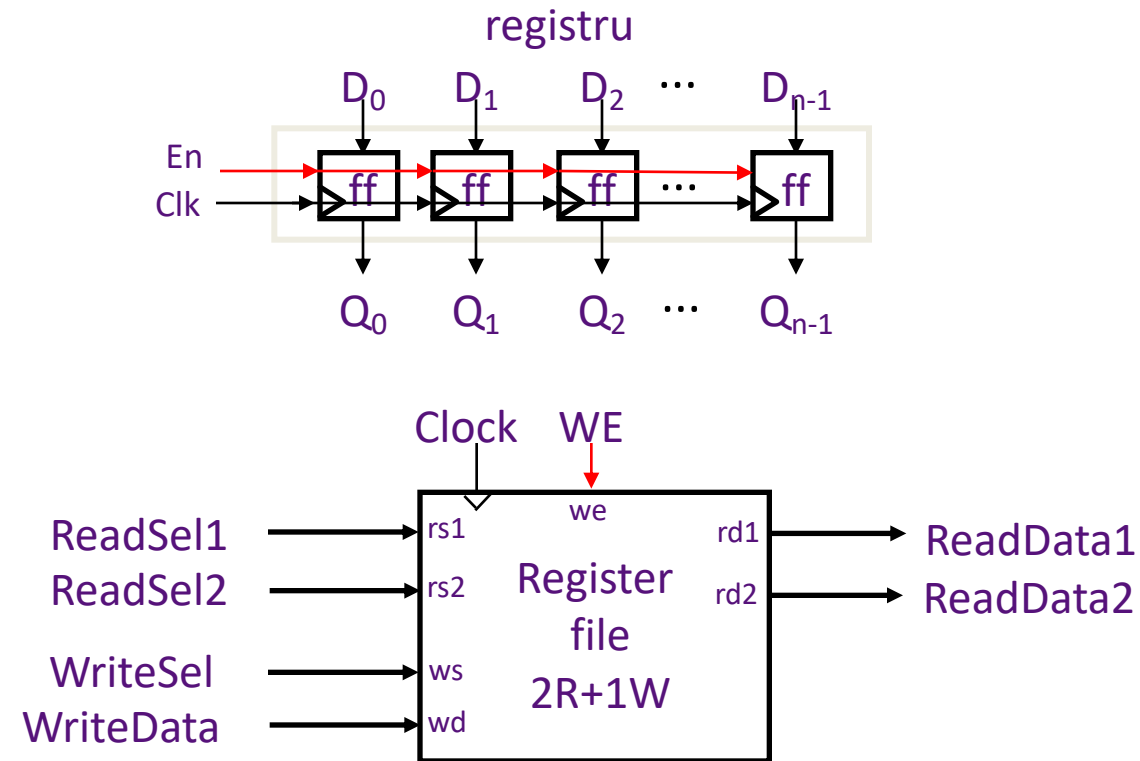
- Flipflop, Register, Register file, SRAM, DRAM



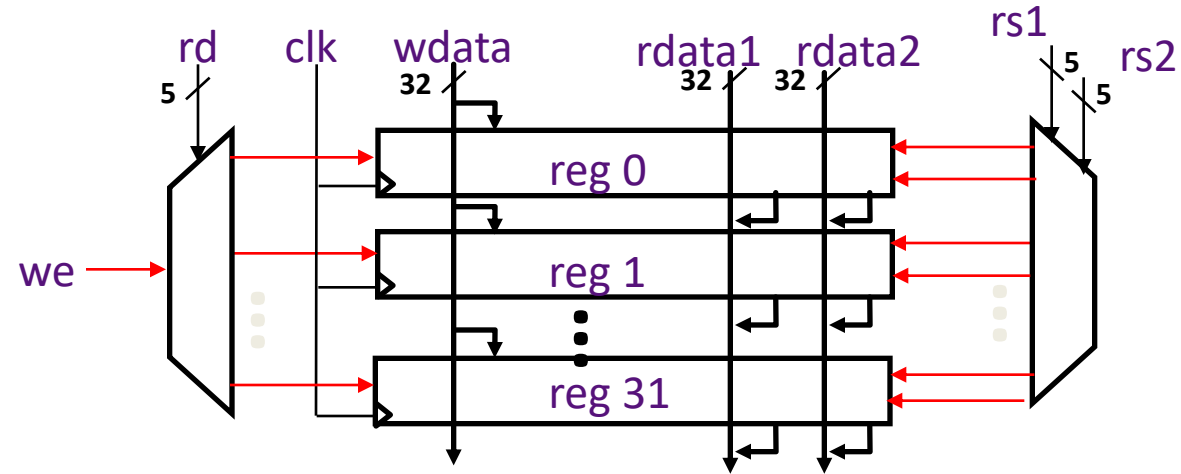
*Edge-triggered: Datele eșantionate pe front crescător*

# Tabele de registre

- Citirile se fac combinațional

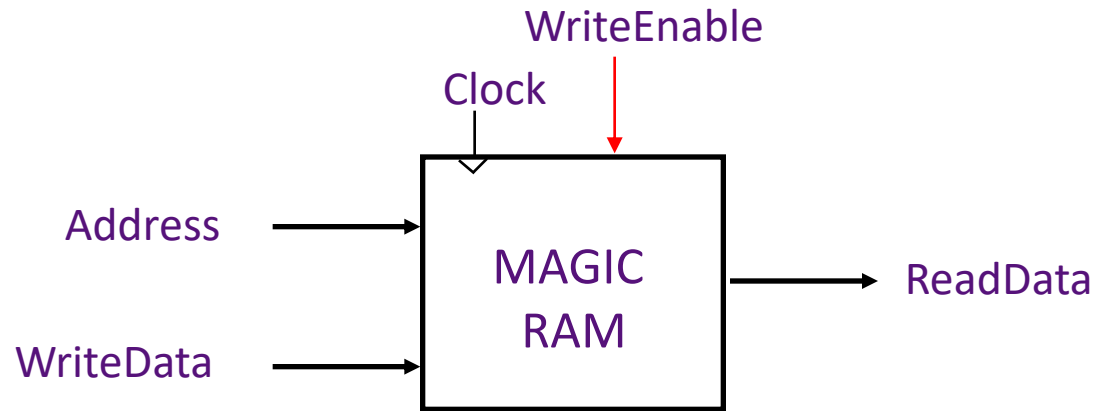


# Implementarea unei tabele de registre



- 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

*Semestrul următor vom prezenta un model mai realist de memorie*

---

# Implementarea RISC-V

Single-cycle per instruction  
datapath & control logic

# Execuția instrucțiunilor

---

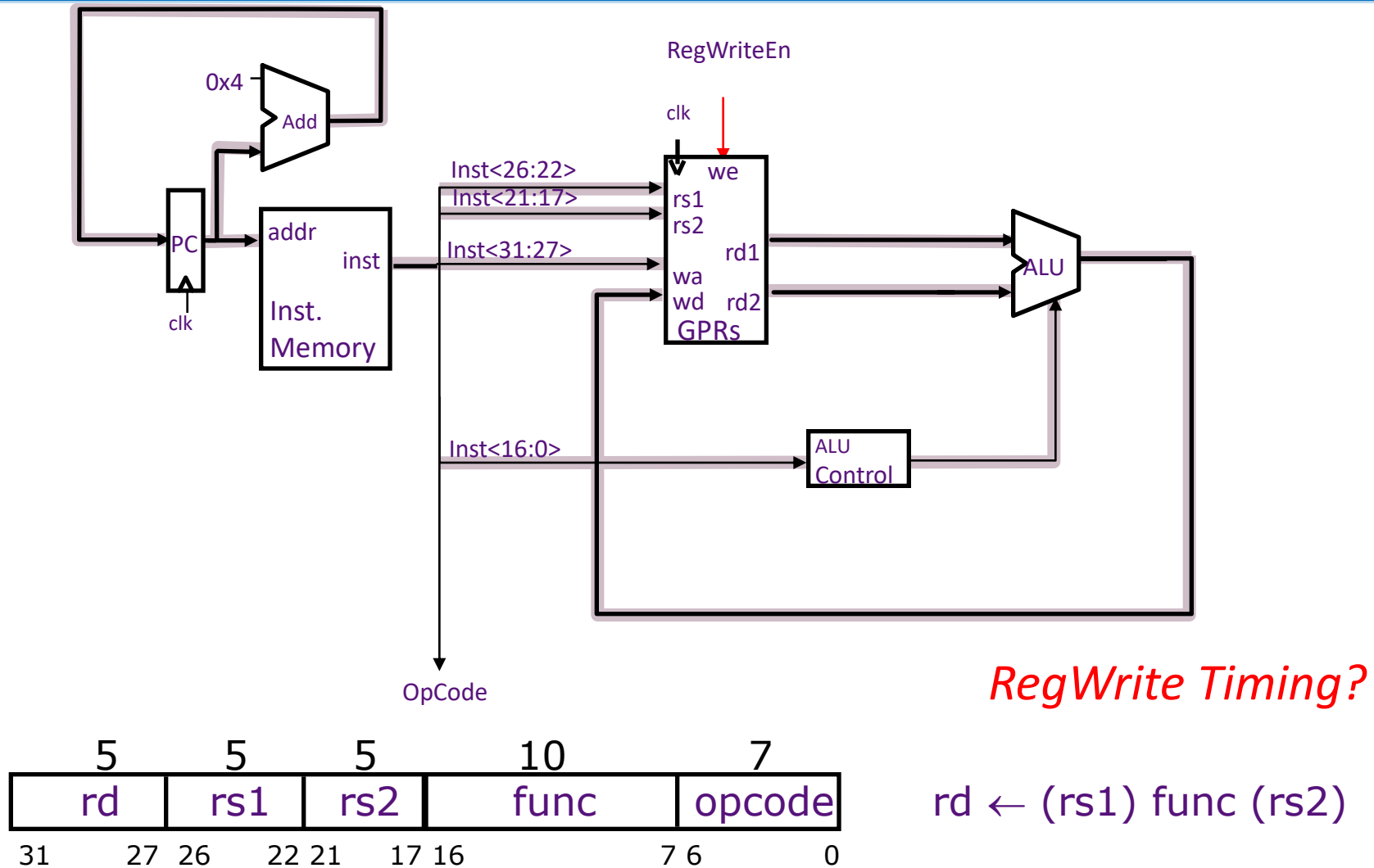
O execuție implică

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

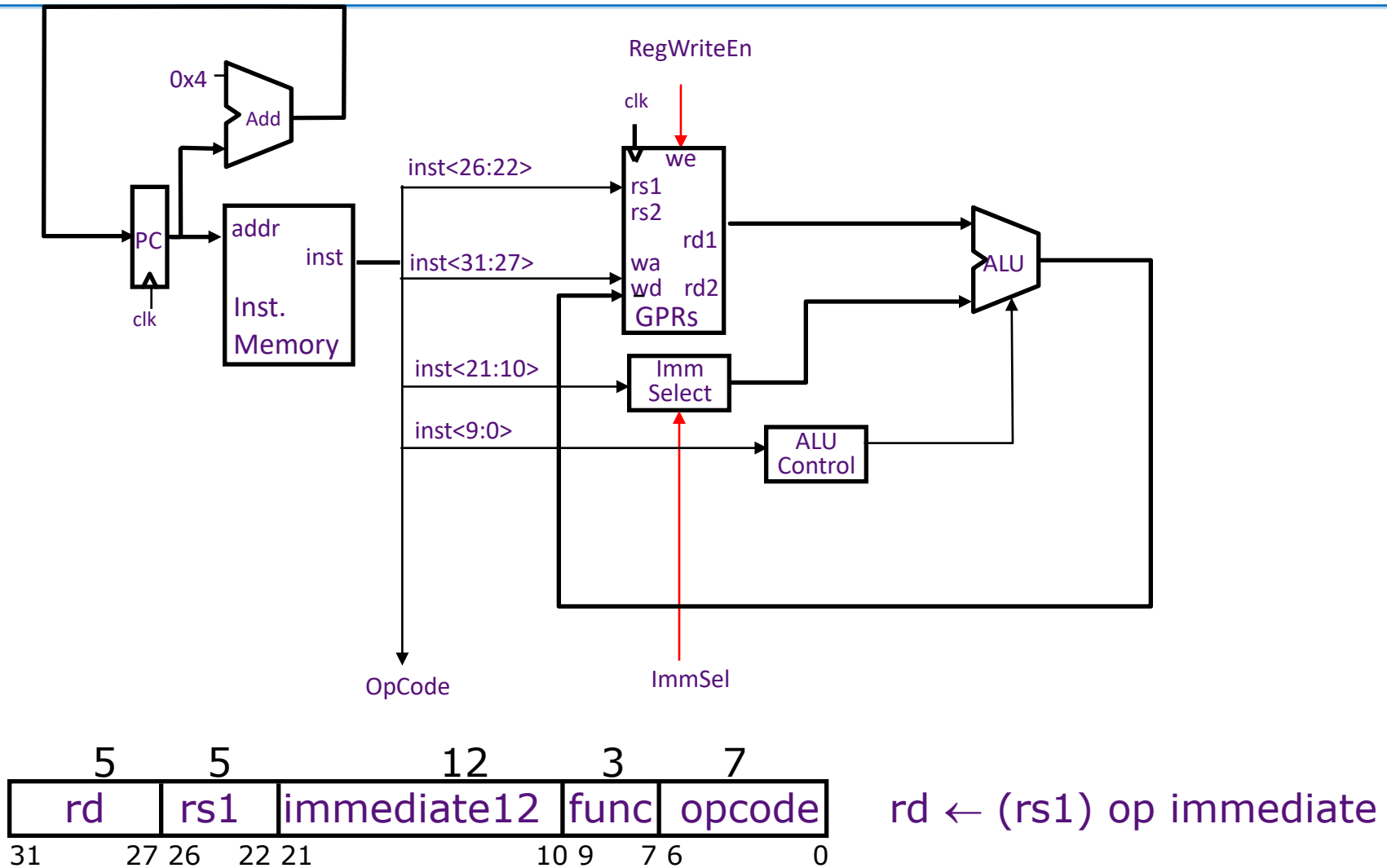
și calculul adresei instrucțiunii următoare



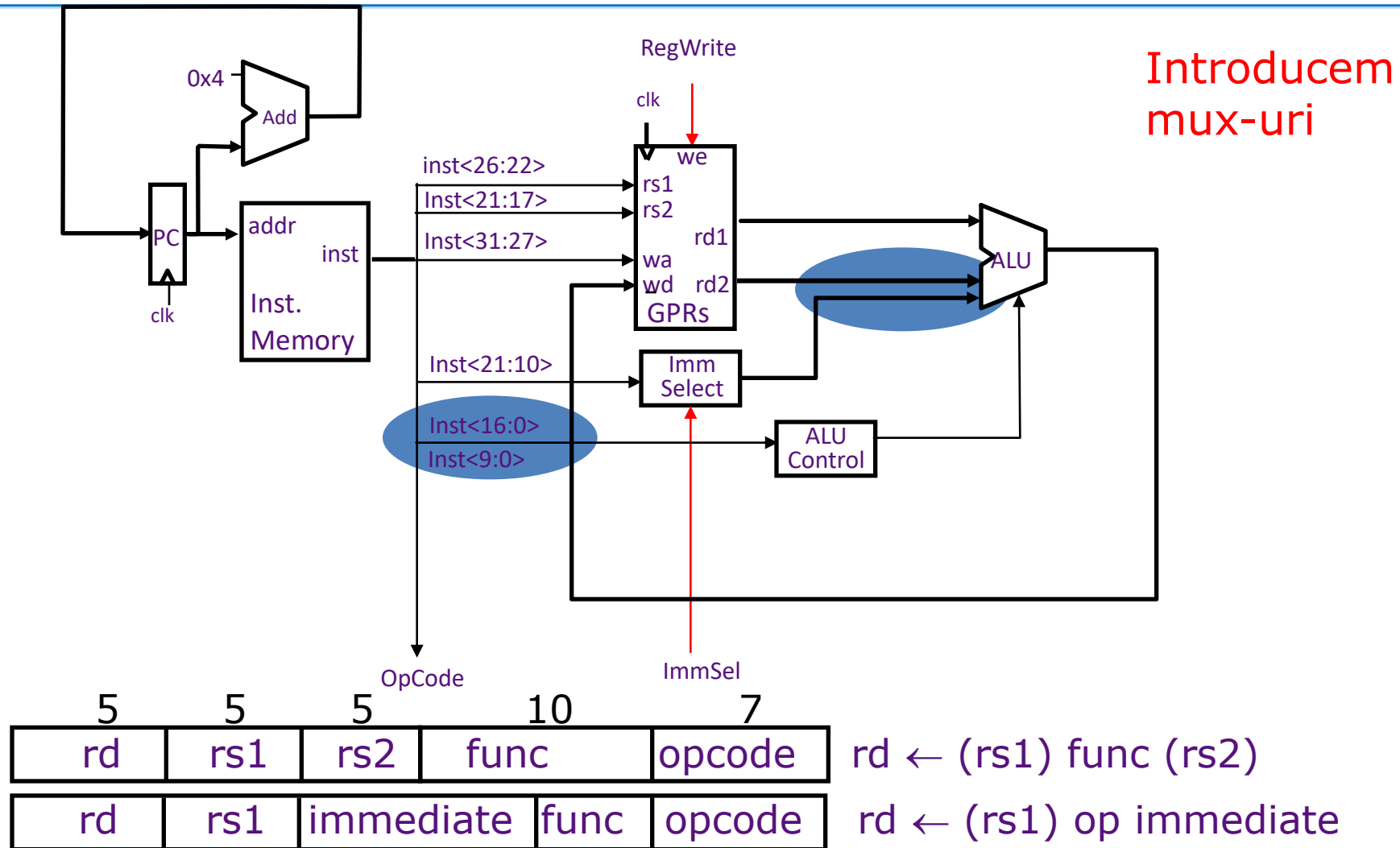
# Transfer date: Instrucțiuni Reg-Reg ALU



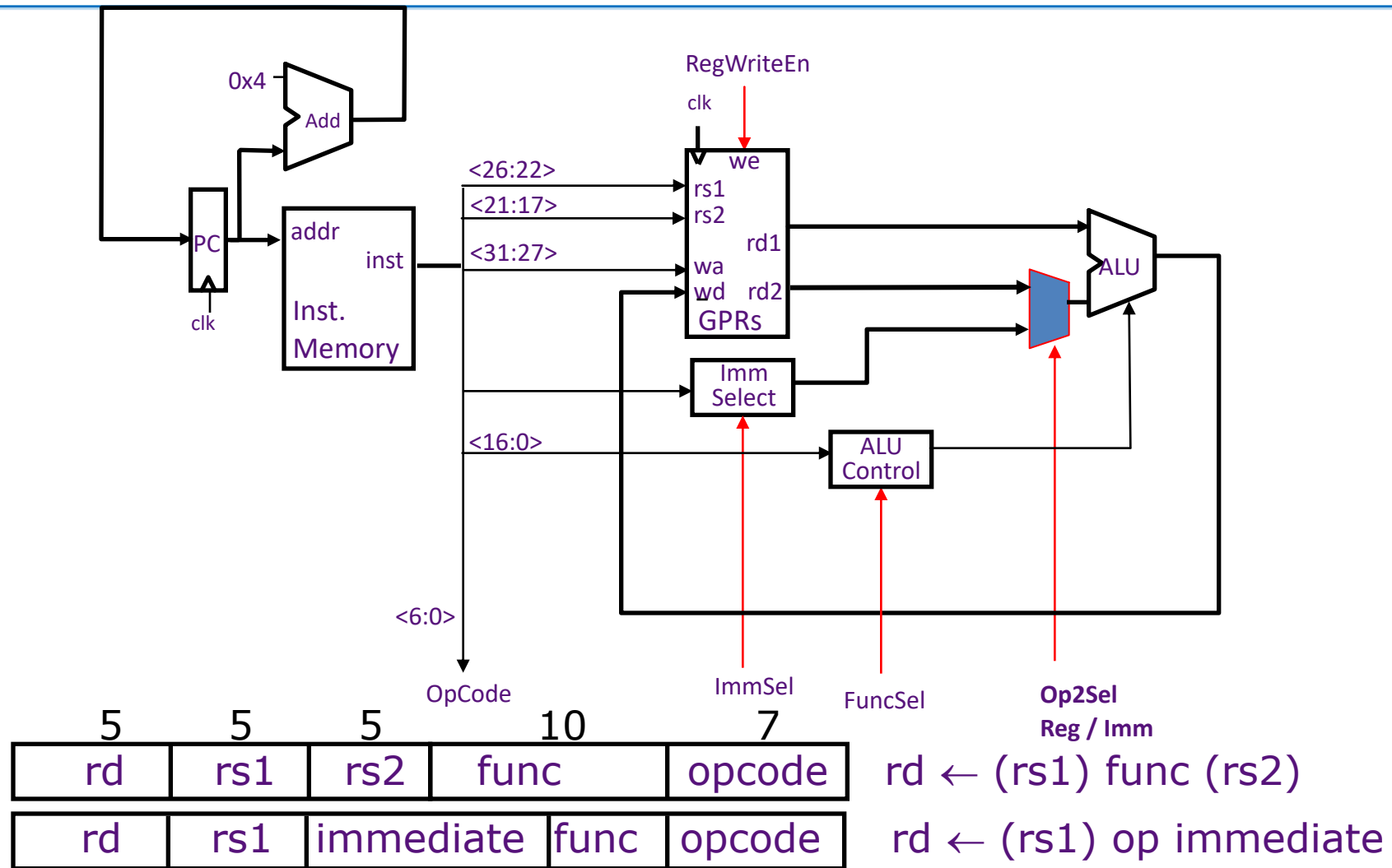
# Transfer de date: Instrucțiuni Reg-Imm ALU



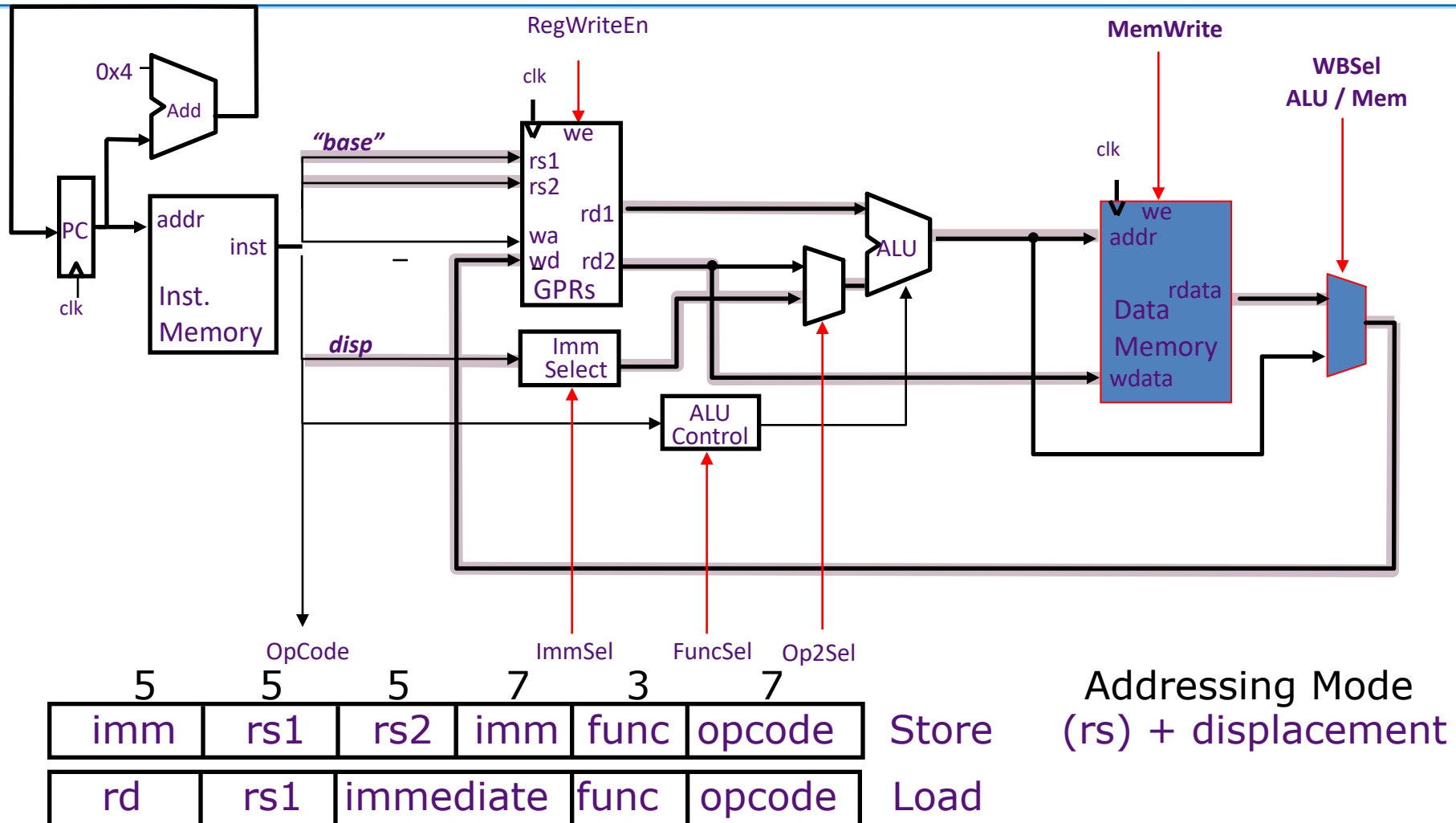
# Conflicte în transferul de date



# Transfer date pentru instrucțiuni cu UAL

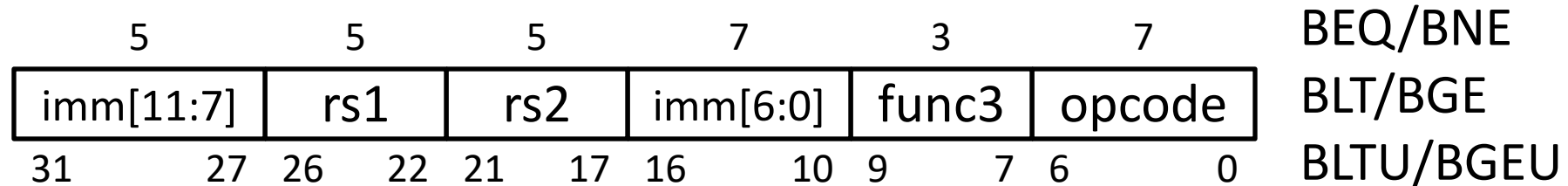


# Instrucțiuni Load/Store



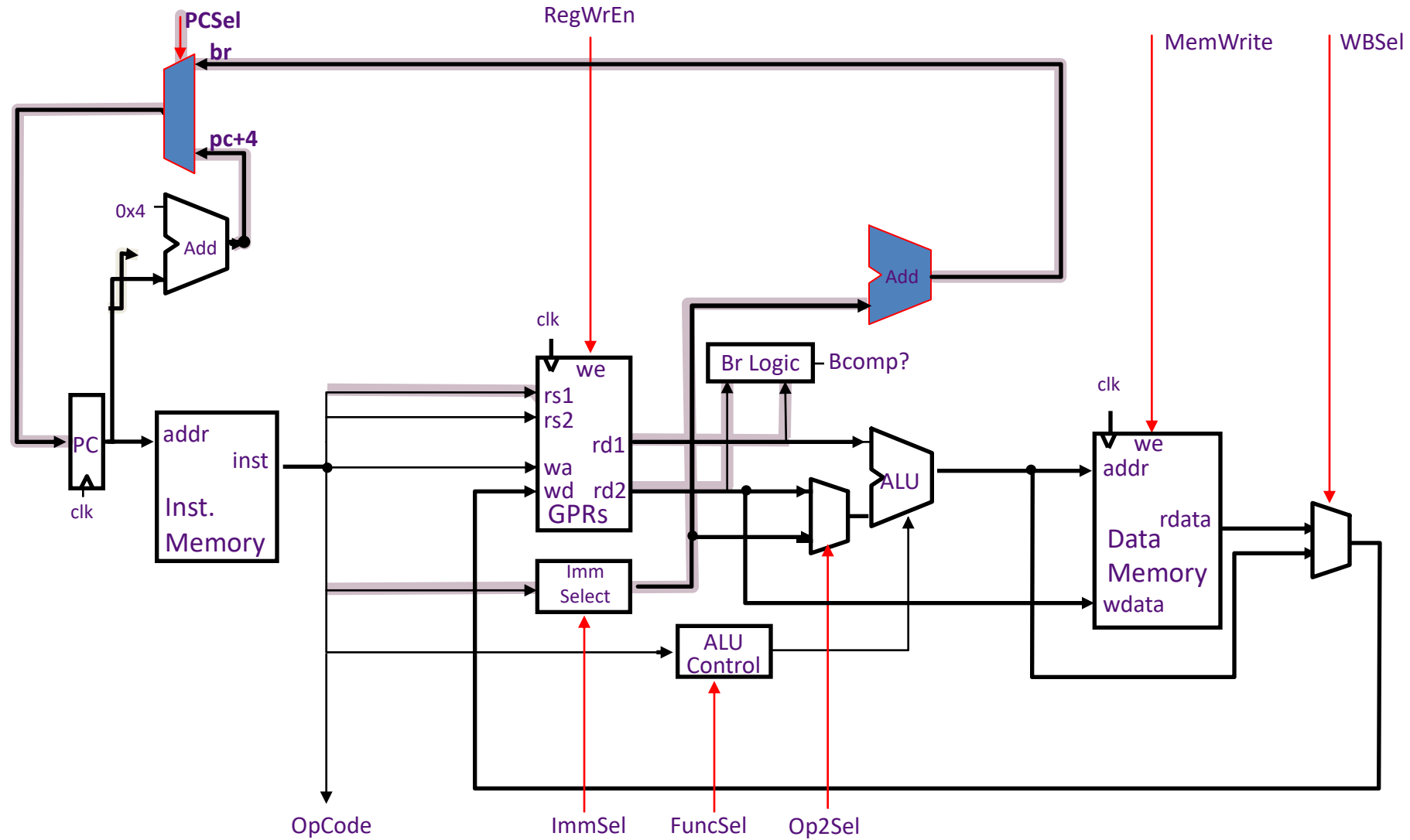
rs1 este registrul de bază  
rd este destinația pt Load, rs2 este sursa de date pentru un Store

# Salturi condiționale la RISC-V

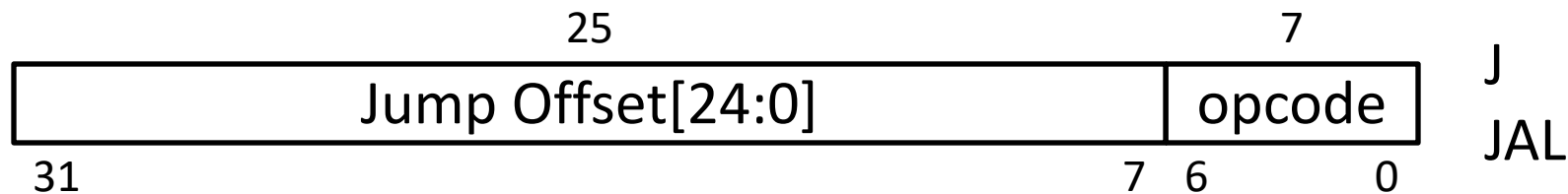


- Compară două registre întregi pentru egalitate (BEQ/BNE) sau mărime cu semn (BLT/BGE) sau fără semn (BLTU/BGEU)
- Valoarea imediată de 12 biți stipulează adresa țintă de salt ca fiind un offset cu semn pentru PC, în unități de 16 biți (i.e., shift left cu 1 apoi adaugă la PC).

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



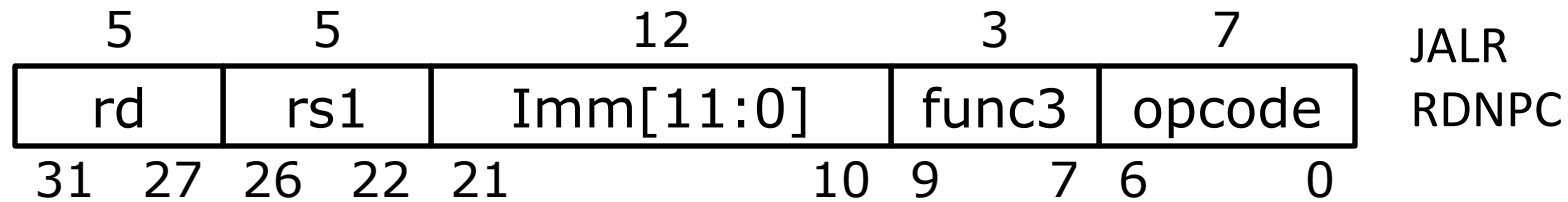
# Salturi necondiționale la RISC-V



- 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

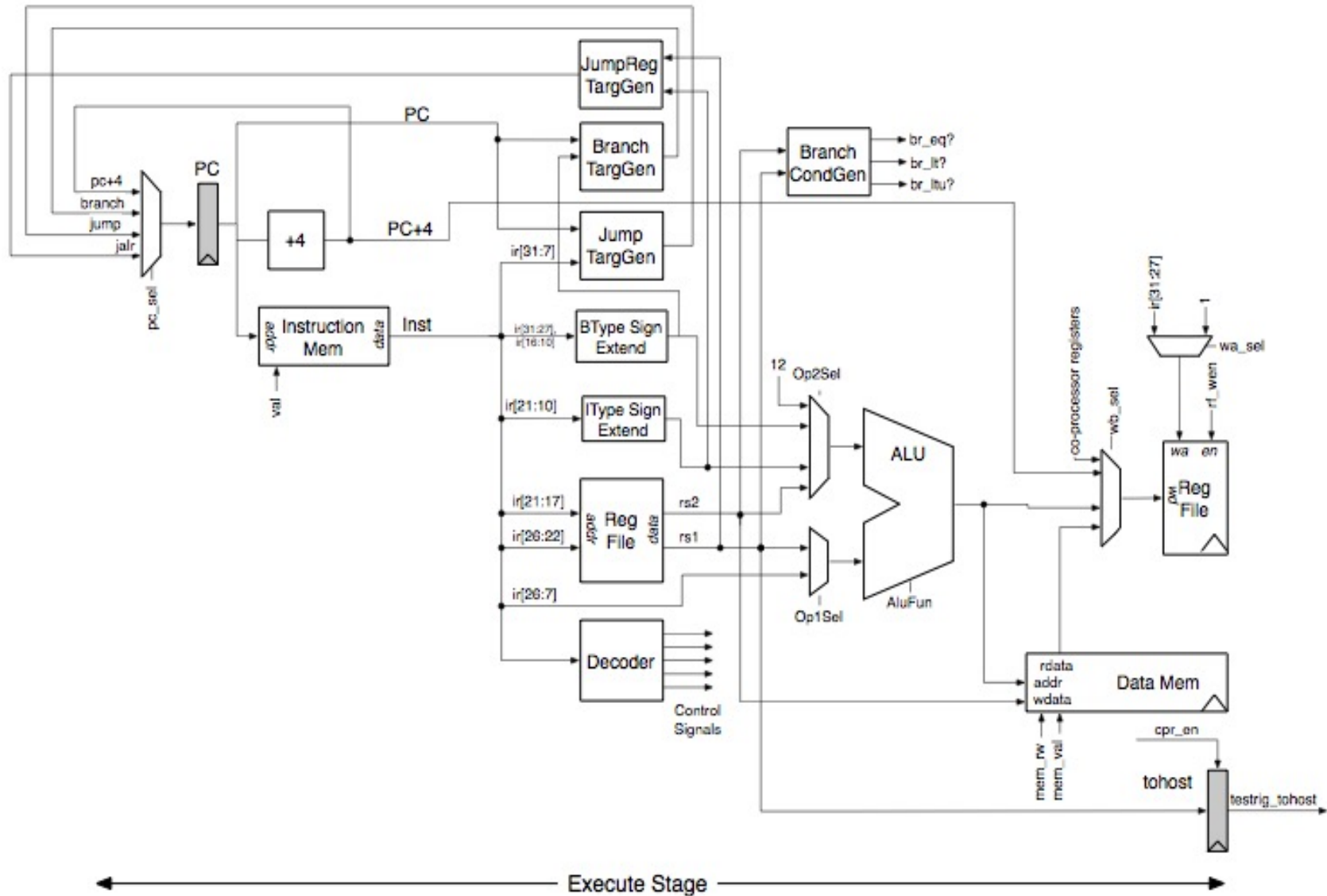


# Salturi indirecte prin registre la RISC-V

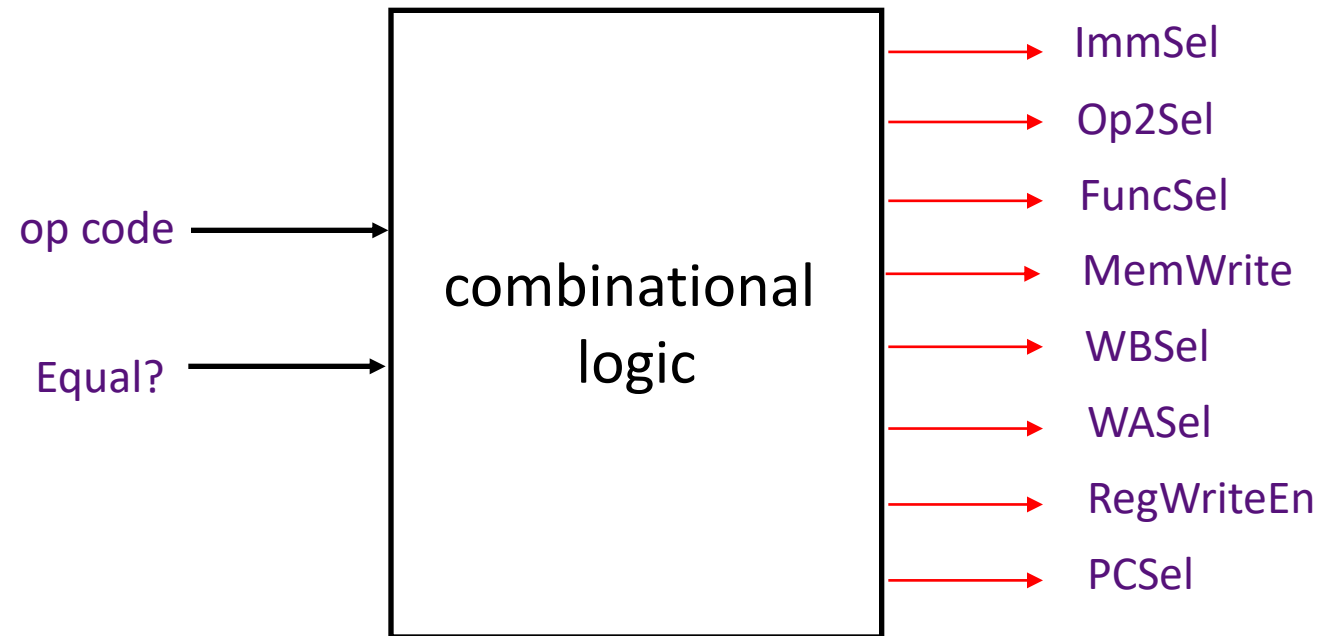


- Salturile la adresa țintă sunt făcute prin adunarea unui offset de 12 biți (*ne-shiftat* cu 1) registrului rs1
- Adresa de întoarcere (PC+4) este scrisă în registrul rd (poate fi **x0** dacă valoarea nu este necesară)
- Instrucțiunea RDNPC scrie adresa de întoarcere în rd fără să facă saltul efectiv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

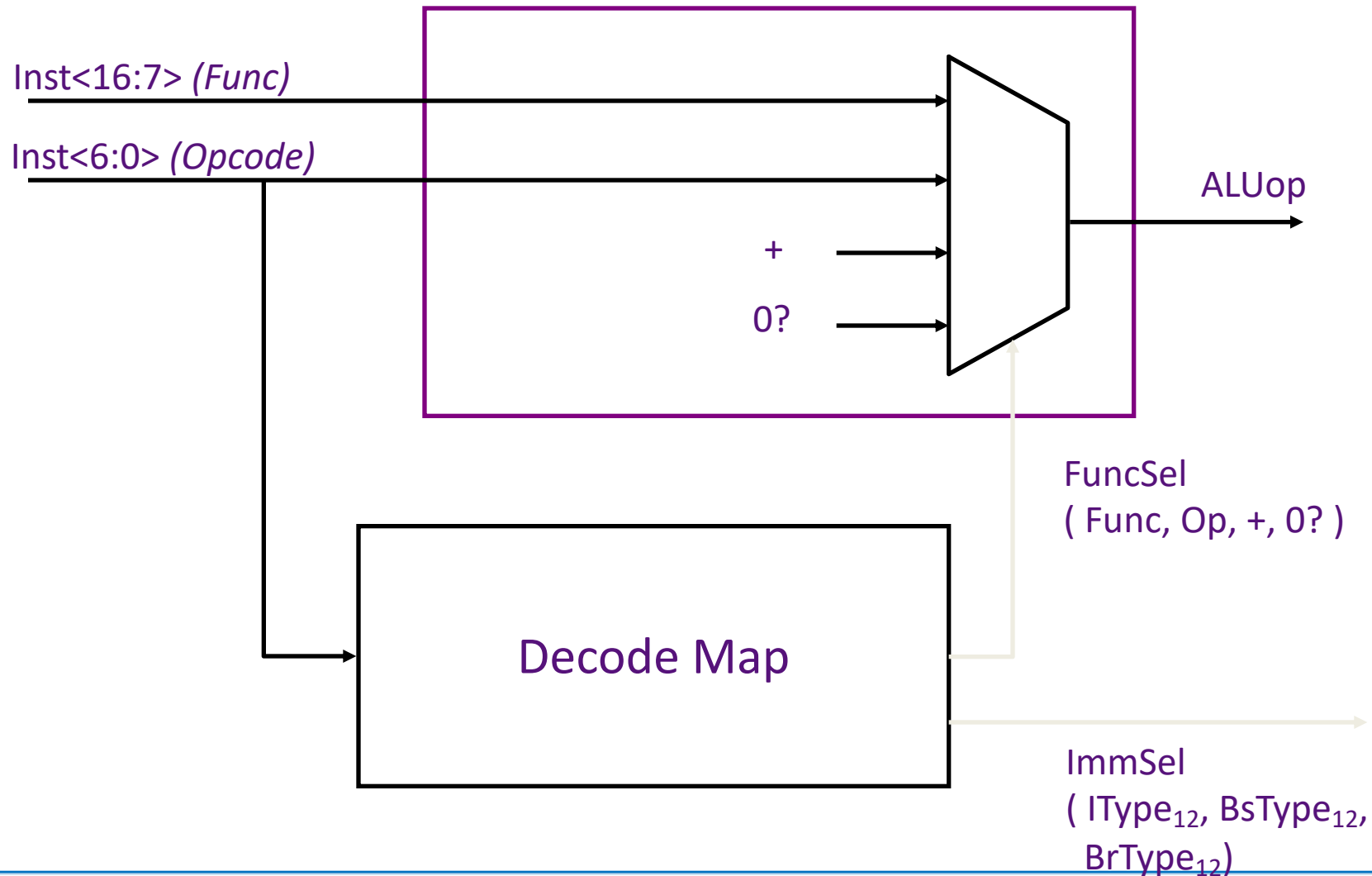
# Full RISC-V1 Stage Datapath



# Controlul cablat este logică combinațională pură



# Controlul UAL și extensia pentru valori imediate



# Tabela pentru controlul cablat

Opcode	ImmSel	Op2Sel	FuncSel	MemWr	RFWen	WBSel	WASel	PCSel
ALU	*	Reg	Func	no	yes	ALU	rd	pc+4
ALUi	IType <sub>12</sub>	Imm	Op	no	yes	ALU	rd	pc+4
LW	IType <sub>12</sub>	Imm	+	no	yes	Mem	rd	pc+4
SW	BsType <sub>12</sub>	Imm	+	yes	no	*	*	pc+4
BEQ <sub>true</sub>	BrType <sub>12</sub>	*	*	no	no	*	*	br
BEQ <sub>false</sub>	BrType <sub>12</sub>	*	*	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

---

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

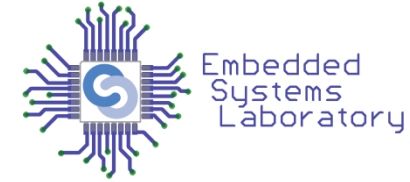
---

# Recapitulare

---

- Microcodarea a devenit mai neatrăgătoare 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 într-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
  - Interschimbă 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