Subiect → Explicați execuția instrucțiunii de tip R pentru un procesor MIPS care rulează pe un singur ciclu de ceas, punând în evidență formatul instrucțiunii, unitățile funcționale implicate, ~~unitățile funcționale simplificate~~, registrii cheie

Instrucțiuni de tip R

INS. ARITMETICE LOGICE

Tipul R → ~~formatul~~ instrucțiunilor aritmetice-logice
↳ add, sub, and, or, slt (set-on-less-than)

Formatul instrucțiunii

op	rs	~~rd~~ rt	rd	shamt	funct
31-26	25-21	20-16	15-11	10-6	5-0

op = codul de operație al instrucțiunii
rs, rt, rd = adresele registrelor sursă și destinație
shamt = numărul de biți cu care se efectuează deplasarea
funct = selectează varianta de operație specificată de către op

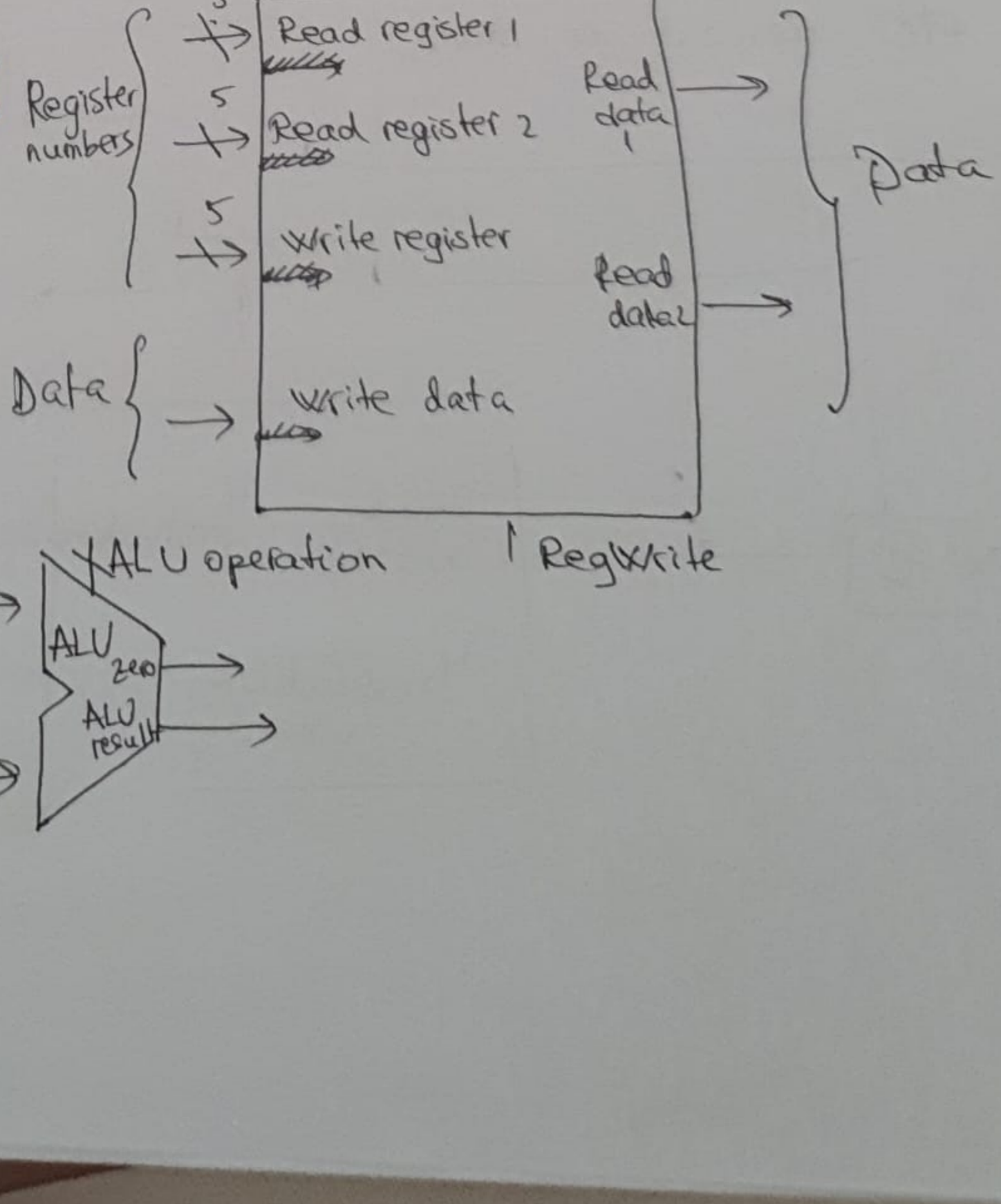
Rezultatul din UAL trebuie scris într-un registru.

~~Instr~~ → aritmetice: sub, add
→ logice: and, or, xor, slt (set-on-less-than)

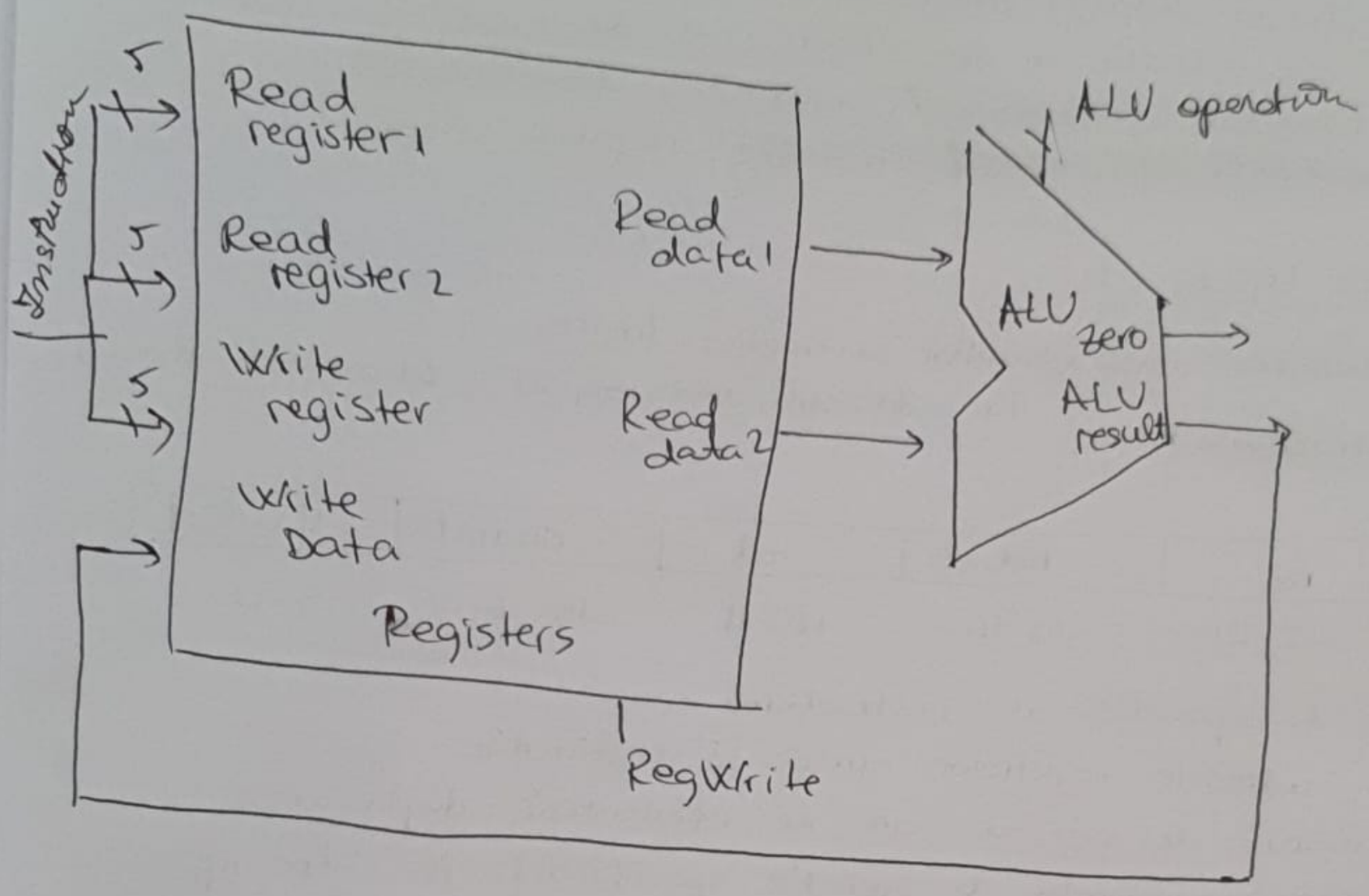
Instrucțiunile de format R au ca operanzi 3 registre → 2 sunt citite, unul este scris

Pentru realizarea căii de date avem nevoie de:

a) Registrele generale



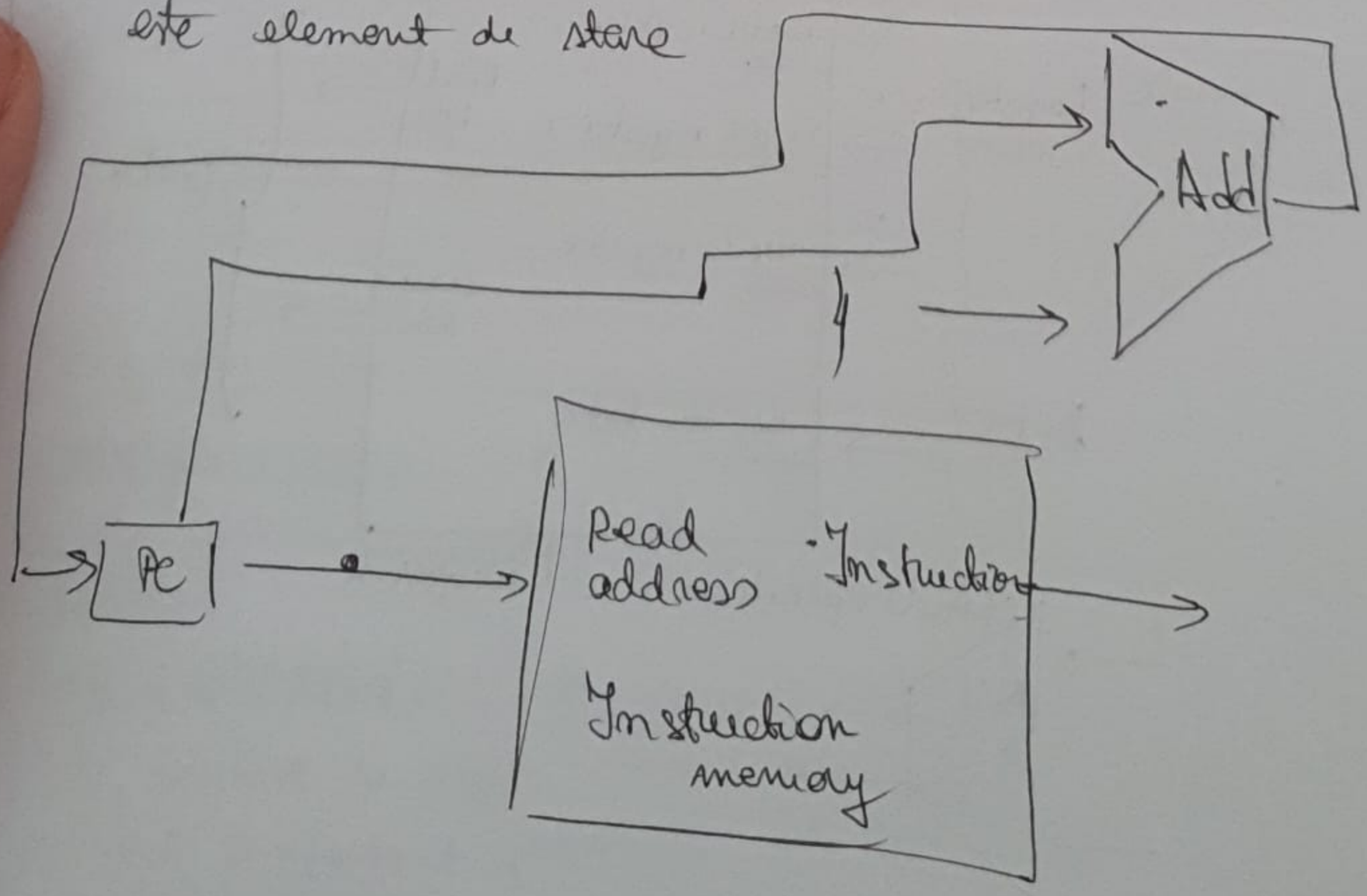
b) UAL unitate aritmetică logică
↓
operează cu valorile citite din registre



Obținerea instrucțiunii curente și trecerea la următoarea instrucțiune MIPS cu un singur ciclu de ceas

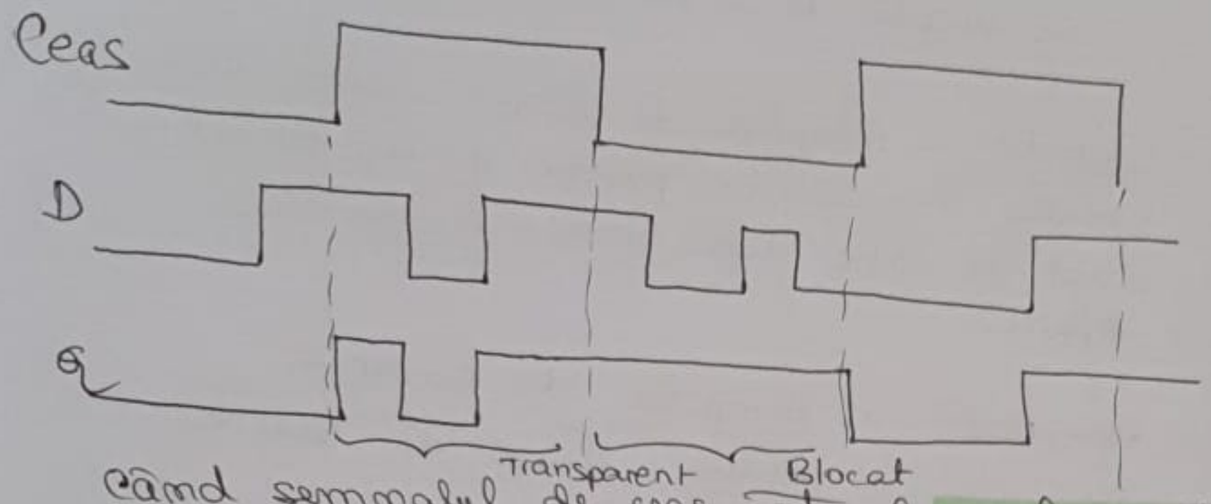
→ se extrage instrucțiunea din memorie

→ pentru trecerea la următoarea instrucțiune, aflată la 4 octeți distanță, se incrementează PC-ul cu 4, care este element de stare



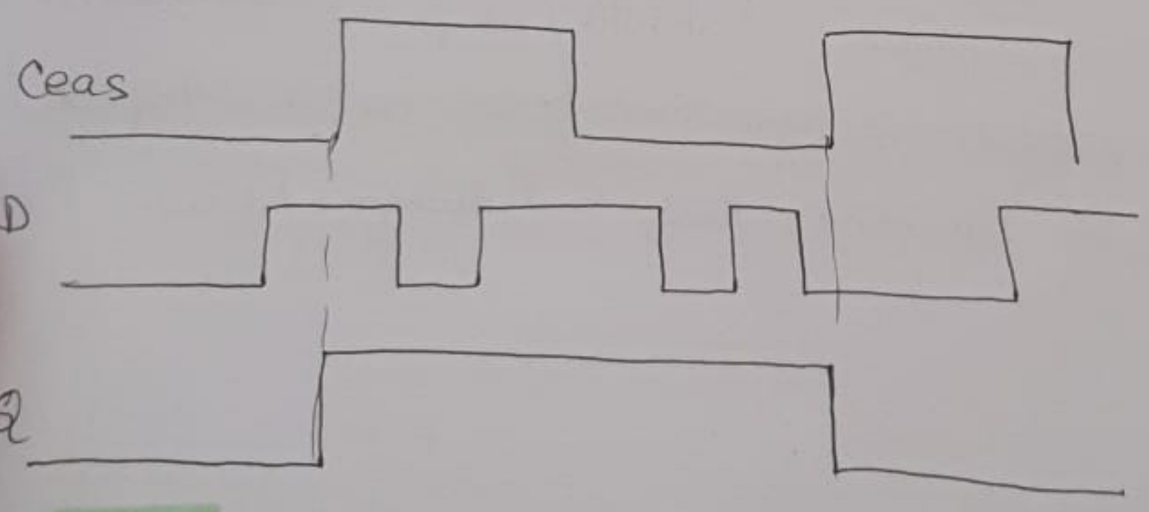
Diferența dintre un latch transparent și un bistabil D comandat pe front

Latch transparent



Când semnalul de ceas este la nivel înalt, datele traversează latch-ul. Când semnalul de ceas este la nivel coborât, datele sunt blocate.

Bistabil de tip D



Datele sunt stocate/capturate pe frontul crescător al circuitului și memorate pe restul circuitului.

Parametrii sincronizării / timing-ului latch-ului

- TCQmin / TCQmax → timpul de propagare a semnalului de la intrare la ieșire atunci când semnalul de ceas deschide latch-ul.
- TDQmin / TDQmax → timpul de propagare a semnalului de la intrare la ieșire atunci când latch-ul este transparent. (cel mai important parametru de sincronizare a ceasului)
- Tstabilire/setup / Tmentinere/hold → definesc o fereastră în jurul frontului posterior al semnalului de ceas pe durata căreia data trebuie să fie stabilită pentru a putea fi eșantionată corect

... de ramificație ⟶ formatul instrucțiunii

FPGA → unitățile structurale

Componentele structurale de bază ale arhitecturilor de porți, de tip FPGA, sunt:

- blocurile logice programabile
- comutatoarele programabile
- traseele de interconectare (routing)

Blocurile logice se pot realiza sub formă de:

→ rețele de perechi de tranzistoare NMOS și PMOS, comutatoare de tip T-gates
→ rețele de porți combinaționale (NAND, XOR etc)
→ multiplexoare
→ tabele asociative (lookup tables) cu n intrări
→ structuri ȘI-SAU cu mai multe intrări

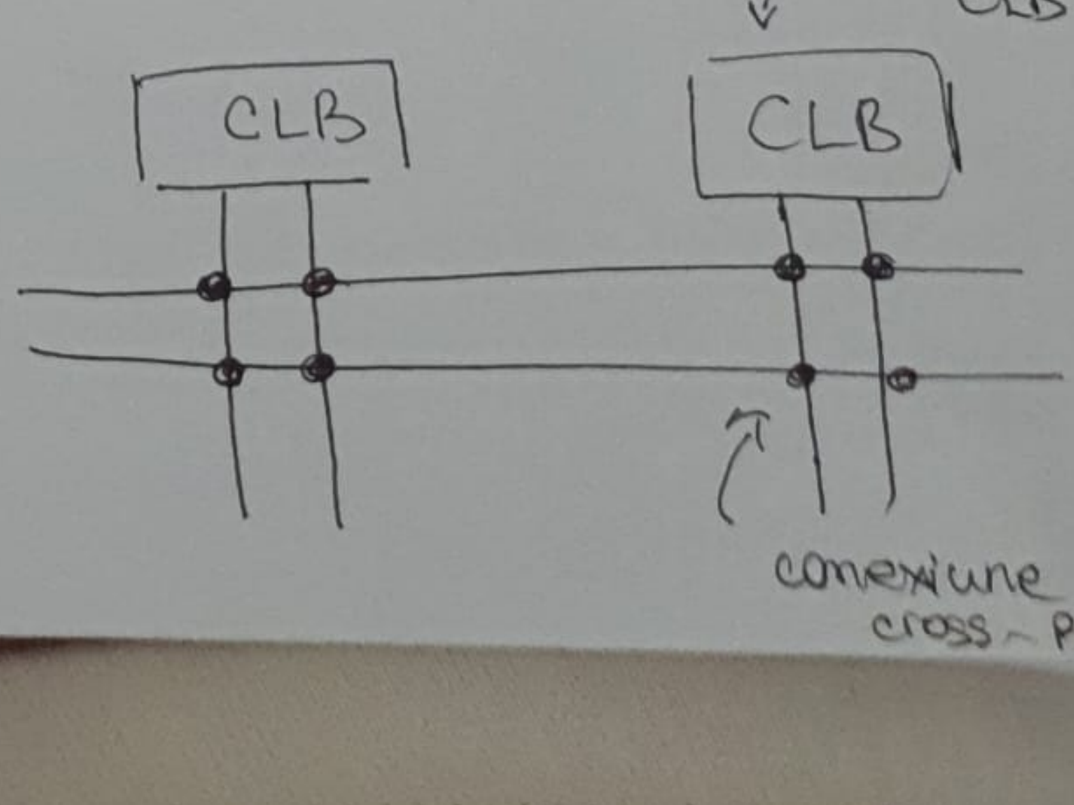
Trasee de interconectare ⟶ Xilinx, Actel, Altera } producători

FPGA → conexiune cross-point

Familiile de FPGA-uri diferă prin:
→ mijloacele fizice de realizare a programării de către utilizator
→ organizarea traseelor de interconectare
→ funcțiile de bază ale blocurilor logice combinaționale (CLB)

Cele mai mari diferențe se regăsesc la tehnicile folosite pentru realizarea unor interconexiuni cât mai flexibile în cadrul blocurilor și între blocuri.

Conexiuni de tip „punte de intersecție" în cadrul unor trasee de tip „magistrale intersectate" sau cross-bar

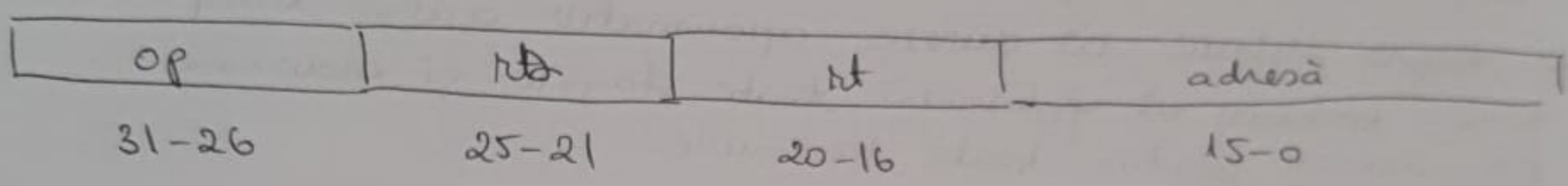


Subiect: Instrucțiunea de ramificație

Procesor MIPS – un singur ciclu de ceas
- → formatul instrucțiunii
- → unitățile funcționale implicate
- → registrii cheie

→ instrucțiune de tip I

FORMAT



op → codul de operație al instrucțiunii

rs → adresa registrului sursă (source)

rt → adresa registrului sursă (target)

Se folosește ieșirea UAL în determinarea adresei următoare instrucțiuni de executat → se introduce o logică de control

Ramificație → egal / salt

Instrucțiunea beq (ramificație condiționată)

Instrucțiunea este formată din 2 registre care sunt comparate pentru o egalitate și o deplasare de 16 biți, folosită în calculul adresei obiectiv pentru ramificație:

beq $t1, $t2, offset

În vederea implementării acestei instrucțiuni trebuie determinată adresa obiectiv pentru ramificație → PC + câmpul de deplasare al instrucțiunii cu semn extins

Offset este folosit în calculul adresei obiectiv pentru ramificație, care este relativă la adresa instrucțiunii de ramificație.

Câmpul deplasării trebuie mutat cu 2 poziții la stânga

1) Anulează conținutul registrului în care se acumulează sumele produselor parțiale
2) Inițializează nr. rangului biților înmulțitorului j=0
3) Formează produsul parțial x·yj
4) Adună produsul parțial la jumătatea superioară a registrului sumei produselor parțiale
5) Efectuează j=j+1 Și dacă j=n trece la 8
6) Deplasează la dreapta cu un rang conținutul registrului sumei produselor parțiale
7) Trece la pas 3
8) Produsul cu 2(n-1) rangui s-a obținut în registrul de deplasare.

Calea de date pentru instrucțiunile de încărcare sau memorare realizează un acces la un registru, urmat de calcularea unei adrese de memorie, iar în continuare o citire sau o scriere în memorie și o scriere în fișierul de regiștri, dacă instrucțiunea este de încărcare.

Codul operațiilor:
→ încărcare: 35
→ memorare: 43

MIPS cu mai multe cicluri de ceas

→ versiune abstractă
↳ diferențe față de varianta cu un singur ciclu

Fiecare pas al execuției va necesita o perioadă de ceas ⇒ permite unei unități funcționale să fie utilizată mai mult decât o singură dată pe instrucțiune, cât timp este utilizată în cicluri definite de ceas.

Această utilizare multiplă reduce cantitatea de hardware.

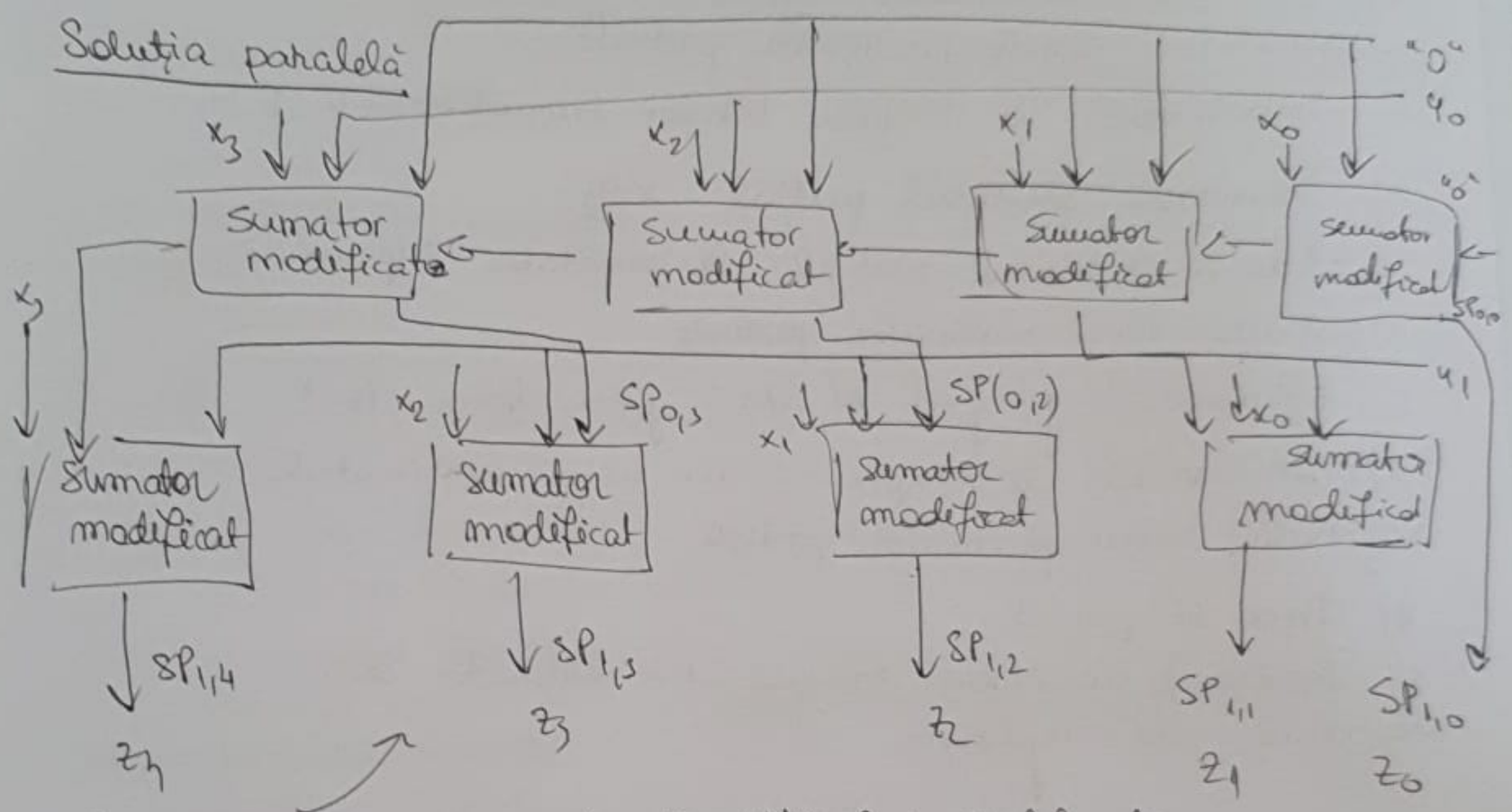
Avantaje MIPS cu mai multe cicluri de ceas:
- posibilitatea de a folosi aceleași unități funcționale de mai multe ori pe durata execuției unei singure instrucțiuni
- posibilitatea de a permite instrucțiunilor să folosească un număr definit de cicluri de timp

Diferențe față de versiunea cu un singur ciclu

- o singură unitate de memorie, atât pentru instrucțiuni, cât și pentru date
- un singur UAL
- sunt adăugate unul sau mai multe regiștri după fiecare unitate funcțională majoră pentru a păstra ieșirea acelei unități până când valoarea va fi folosită într-un ciclu ulterior de ceas.

Înmulțirea

Soluția paralelă

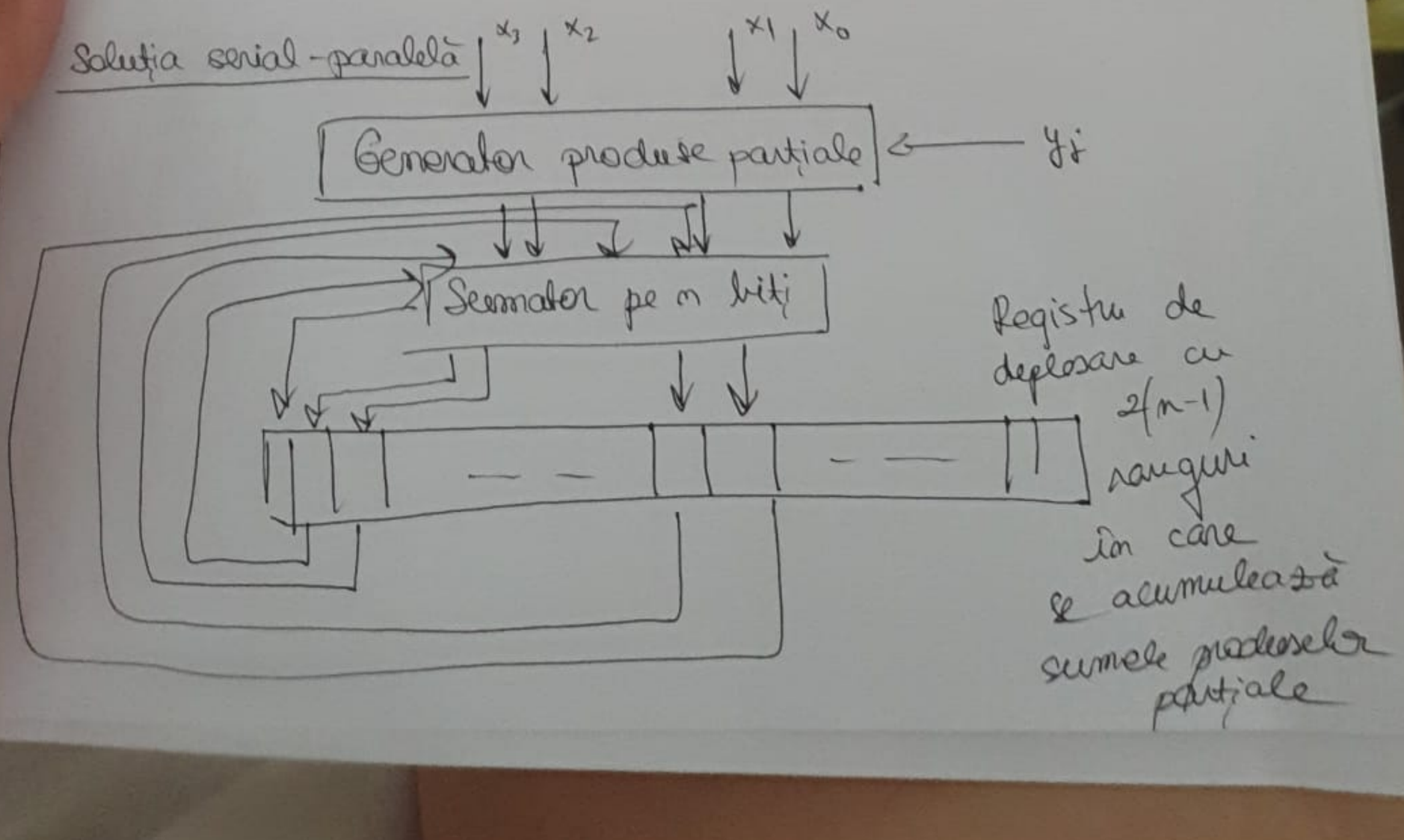


Primele două etaje ale dispozitivului paralel de înmulțire

SP_{ij} = bitul i al sumei produselor parțiale j

Sumatorul modificat poate fi utilizat în cadrul unei structuri de înmulțire paralelă, fiind vorba de o „programare spațială", care conduce la viteze ridicate de operare. Soluția necesită $(n-1)^2$ sumatoare modificate.

Soluția serial-paralelă

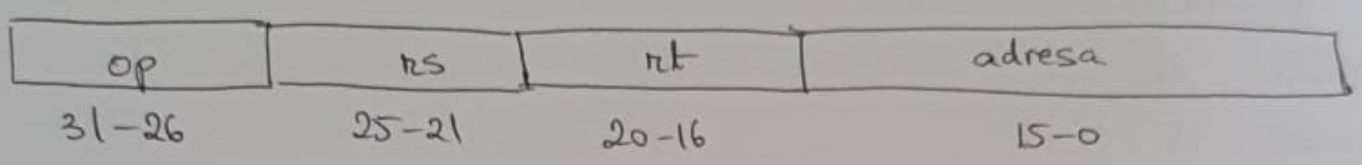


~~Arhitectura de tip RISC~~ INSTR. MEM ÎNCĂRCARE MEMORARE

Subiect → Explicați execuția instrucțiunilor de încărcare/memorare pentru un procesor MIPS care rulează pe un singur ciclu de ceas, punând în evidență formatul instrucțiunii, unitățile funcționale implicate, registrii cheie

Instrucțiune de tipul I → instrucțiuni de referire a memoriei

Formatul instrucțiunii

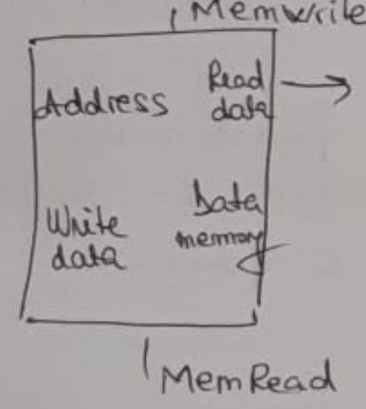


- op = codul de operație al instrucțiunii
- rs = adresa registrului sursă
- rt = adresa registrului destinație
- adresa = deplasarea adresei / offset

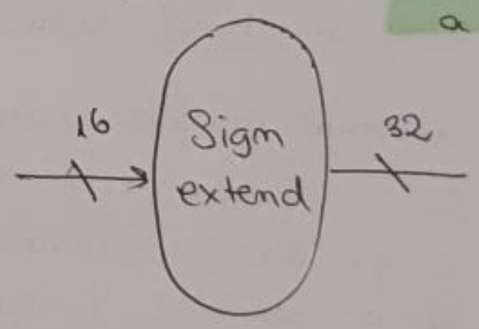
Rezultatul din UAL va fi o adresă

Pe lângă regiștrele generale și UAL, unitățile necesare implementării acestor instrucțiuni sunt:

a) Unitatea memoriei de date



b) Unitatea de extindere a semnului



Instrucțiunile sunt:

- lw $t1, valoare_deplasare ($t2)
- sw $t1, valoare_deplasare ($t2)

Aceste instrucțiuni calculează o adresă de memorie prin adunarea registrului de bază $t2 cu câmpul pentru deplasare cu semn de 16 biți din instrucțiune. Extinderea se face până la o valoare de 32 de biți cu semn.

Dacă instrucțiunea este de memorare, valoarea de memorat trebuie citită din fișierul de registre, unde ea se găsește în $t1.

Dacă instrucțiunea este de încărcare, valoarea citită din memorie trebuie scrisă în $t1 care este registrul specificat din fișierul de registre.

PIPELINE → Stagiile de execuție ale unei instrucțiuni

1) IF - citirea următoarei instrucțiuni oferite de PC
2) ID - decodificarea instrucțiunii
3) EX - execuția instrucțiunii
4) MEM - memorarea rezultatului și incrementarea lui PC
5) WB - scrierea în registre

Toate stagiile trebuie să dureze aproximativ același timp.
Un task trebuie să folosească toate stagiile și ordinea
să fie aceeași pentru toate task-urile.

La intersecțiile între barele verticale și cele orizontale se pot stabili conexiuni permanente sau temporare, în funcție de tehnologia utilizată.

Permanente → se utilizează elemente de tip „anti-fuse" ce contact permanent stabilit ca urmare a aplicării temporare a unei tensiuni ridicate

Avantaje → caracter nevolatile, dimensiunile relativ mici, rezistența și capacitatea reduse

Dezavantaje → conținutul fix, imposibilitatea reprogramării

Temporare → se utilizează în calitate de comutatoare tranzistoare NMOS, cu canal indus, sau tranzistoare cu poartă flotantă

Avantaj → posibilitatea reconfigurării

Dezavantaje → caracter nevolatile, dimensiuni relativ mari ale comutatoarelor

Bistabili → cum este afectat T_{ciclu} de $T_{alunecare}$ (T_{skew})

Cauzele alunecării
- → variații în procesul de producție
- → modificarea temperaturii + zgomotul în sursele de alimentare

Skew → variație spațială a timpilor de sosire a semnalelor de ceas: variația în ceea ce privește același front de ceas, văzut de către două sau mai multe bistabile diferite

Jitter
- → variația temporală a timpilor de sosire - variație în ceea ce privește timpii de sosire a două fronturi de ceas ale aceluiași bistabil
- → cauză: zgomotul sursei de alimentare

Restricția pentru drumul cel mai lung sau calea cea mai lentă

$$T_{ciclu} \geq T_{CQmax} + T_{pmax} + T_{stabilire/setup} + T_{alunecare/skew}$$

Restricția pentru drumul cel mai scurt / calea cea mai rapidă

$$T_{CQmin} + T_{pmin} \geq T_{menținere/hold} + T_{alunecare/skew}$$