**Calculatoare Numerice 1**

**Test Curs 6**

**Grigore Lucian-Florin 324CD**

**Cerinta**: Pe baza schemei de execuție și control din fig. 4.17 din notele de curs MIPS single-cycle

de săptămâna trecută, vă rog să răspundeți printr-un document postat în care să completați în tabelul din fig. 4.18 biții de control pentru o instrucțiune de tip SET, la alegerea voastră și apoi să explicați în cuvinte cum operează instrucțiunea aleasă pe cele 5 faze de execuție precizând resursele folosite sau nu în fiecare fază pe schema din fig. 4.17.

Voi exemplifica procesul pentru intructiunea SLT (set less than, instructiune R-Type).

Cele 5 faze de executie sunt:

* Citirea: in care se citeste instructiunea (IF – Instruction Fetch) si se pune in registrul de instructiuni
* Decodificarea: in care se decodifica instructiunea si se acceseaza operanzii (TS1 si TS2)
* Executia: unitatea aritmetica logica utilizeaza operanzii pregatiti la pasul anterior si se executa

instructiunea: o UALies ← TS1 op TS2, unde op este ‚<’

* Acces la memorie: o R[$rd] ← R[$rs] < R[$rt]
* Scrierea rezultatului: write back

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instructiune | RegDst | ALUSrc | Memto-Reg | Reg-Write | Mem-Read | Mem-write | Branch | ALUOp1 | ALUOp2 |
| SLT | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |

Unde:

**RegDst** = 1, deoarece registrul de scris este rd, nu rt (daca ar fi fost setat pe 0)

**ALUSrc** = 0, deoarece al doilea operand ALU vine din al doilea registru

**MemtoReg** = 0, deoarece nu avem date din memorie de scris in registri

**RegWrite** = 1, operatie R-Type scriu un registru (RD). In schimb ele nu citesc si nici nu scriu data (in memorie), motiv pentru care MemRead si MemWrite sunt 0

**MemRead** = 0,

**MemWrite** = 0

**Branch** = 1, deoarece nu avem nevoie ca PC sa fie inlocuit de PC + 4

**ALUOp** = 10, deoarece avem o instructiune R-Type si vrem ca controlul ALU sa fie generat din campul funct (care ar trebui sa fie 101010 pentru SLT)