**Semnale control MIPS16 pentru Anexa 5**

<?> ϵ {\_gez, \_ne, \_gtz}

*Tipuri de operații care se pun în paranteză la ALUOp si ALUCtrl:* {(+), (-), (&), (|), (^), (<<*l*), (<<*lv*), (>>*l*), (>>*a*), (<)}, & - AND, | - OR, ^ *- XOR, l* *- logic, a - aritmetic, v - cu variabilă*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Instrucțiune** | **Opcode** *Instr(15-13)* | **RegDst** | **ExtOp** | **ALUSrc** | **Branch** | **<BGEZ>** (opțional) | **Jump** | **JmpR** (opțional) | **MemWrite** | **MemtoReg** | **Reg Write** | **ALUOp (1:0)** | **func**  *Instr(2-0)* | **ALUCtrl (2:0)** |
| add | 000 | 1 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 | 1 | 00(+) | 001 | 001 |
| sub | 000 | 1 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 | 1 | 00(-) | 000 | 000 |
| sll | 000 | 1 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 | 1 | 00(<<l) | 011 | 011 |
| srl | 000 | 1 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 | 1 | 00(>>l) | 110 | 110 |
| and | 000 | 1 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 | 1 | 00(&) | 100 | 100 |
| or | 000 | 1 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 | 1 | 00(|) | 111 | 111 |
| sla | 000 | 1 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 | 1 | 00(<<a) | 101 | 101 |
| sra | 000 | 1 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 | 1 | 00(>>a) | 010 | 010 |
| addi | 001 | 0 | 1 | 1 | 0 | 0 | 0 |  | 0 | 0 | 1 | 01(+) | 001 | 001 |
| ori | 011 | 0 | 0 | 1 | 0 | 0 | 0 |  | 0 | 0 | 1 | 11(|) | 111 | 111 |
| sw | 101 | 0 | 0 | 1 | 0 | 0 | 0 |  | 1 | 0 | 0 | 01(+) | 001 | 001 |
| lw | 010 | 0 | 0 | 1 | 0 | 0 | 0 |  | 0 | 1 | 1 | 01(+) | 001 | 001 |
| beq | 111 | 0 | 1 | 0 | 1 | 0 | 0 |  | 0 | 0 | 0 | 10(-) | 000 | 000 |
| bgez | 110 | 0 | 1 | 0 | 0 | 1 | 0 |  | 0 | 0 | 0 | 10(-) | 000 | 000 |
| j | 100 | 0 | 0 | 0 | 0 | 0 | 1 |  | 0 | 0 | 0 | 10(-) | 000 | 000 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

URL: <https://drive.google.com/file/d/1SI7x2Gp_2m3SEkwnXuGt4ns4voYzpGBH/view?usp=sharing>