**To be implemented**

| Code | Meaning | Opcodes | rs | rt | rd | shift/sa | Function | Type | Description |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Offset/Immediate | | |  |
| Address | | | | | |
| ADDIU | Add immediate unsigned (no overflow) | 001001 |  |  | immediate | | | I | rd = rs + immediate (sign extend) |
| ADDU | Add unsigned (no overflow) | 000000 |  |  |  | 00000 | 100001 | R | rd = rs + rt |
| AND | Bitwise and | 000000 |  |  |  | 00000 | 100100 | R | rd = rs AND rt |
| ANDI | Bitwise and immediate | 001100 |  |  | immediate | | | I | rt = rs AND immediate (zero extend) |
| BEQ | Branch on equal | 000100 |  |  | offset | | | I | If (rs=rt) branch, PC= (offset sl2 signex) +PCn |
| BGEZ | Branch on greater than or equal to zero | 000001 |  | 00001 | offset | | | I | If (rs>=0) branch, PC= (offset sl2 signex) +PCn |
| BGEZAL | Branch on non-negative (>=0) and link | 000001 |  | 10001 | offset | | | I | If (rs>=0) branch, PC= (offset sl2 signex) +PCn, GPR [31] = PCnn |
| BGTZ | Branch on greater than zero | 000111 |  | 00000 | offset | | | I | If (rs>0) branch, PC= (offset sl2 signex) +PCn |
| BLEZ | Branch on less than or equal to zero | 000110 |  | 00000 | offset | | | I | If (rs<=0) branch, PC= (offset sl2 signex) +PCn |
| BLTZ | Branch on less than zero | 000001 |  | 00000 | offset | | | I | If (rs<0) branch, PC= (offset sl2 signex) +PCn |
| BLTZAL | Branch on less than zero and link | 000001 |  | 10000 | offset | | | I | If (rs<0) branch, PC= (offset sl2 signex) +PCn, GPR [31] = PCnn |
| BNE | Branch on not equal | 000101 |  |  | offset | | | I | If (rs≠rt) branch, PC= (offset sl2 signex) +PCn |
| DIV | Divide | 000000 |  |  | 00000 | 00000 | 011010 | R | (LO, HI) = rs/rt, treat operands as signed |
| DIVU | Divide unsigned | 000000 |  |  | 00000 | 00000 | 011011 | R | (LO, HI) = rs/rt, treat operands as unsigned |
| J | Jump | 000010 | Instr\_index | | | | | J | PC=PCn(31:28),instr\_index sl2 |
| JALR | Jump and link register | 000000 |  | 00000 |  | 00000 | 001001 | R | Jump to rs, PC=rs, rd=PCnn |
| JAL | Jump and link | 000011 |  |  |  |  | - | J | PC=PCn(31:28),instr\_index sl2  GPR[31]=PCnn |
| JR | Jump register | 000000 |  | 00000 | 00000 | 00000 | 001000 | R | Jump to rs, PC=rs |
| LB | Load byte | 100000 | base |  | offset | | | I | 8 bit , rt = mem[base + offset signex] signex |
| LBU | Load byte unsigned | 100100 | base |  | offset | | | I | 8 bit, rt = mem[base + offset signex] zeroex |
| LH | Load half-word | 100001 | base |  | offset | | | I | 16 bit, rt = mem[base + offset signex] signex |
| LHU | Load half-word unsigned | 100101 | base |  | offset | | | I | 16 bit, rt = mem[base + offset signex] zeroex |
| LUI | Load upper immediate | 001111 | 00000 |  | immediate | | | I | rt = immediate sl16 |
| LW | Load word | 100011 | base |  | offset | | | I | 32 bit, rt = mem[base + offset signex] |
| LWL | Load word left | 100010 | base |  | offset | | | I | rt = rt MERGE memory[base + offset signex] |
| LWR | Load word right | 100110 | base |  | offset | | | I | rt = rt MERGE memory[base + offset signex] |
| MTHI | Move to HI | 000000 |  | 00000 | 00000 | 00000 | 010001 | R | HI = rs |
| MTLO | Move to LO | 000000 |  | 00000 | 00000 | 00000 | 010011 | R | LO = rs |
| MULT | Multiply | 000000 |  |  | 00000 | 00000 | 011000 | R | (LO, HI) = rs x rt, treat as signed |
| MULTU | Multiply unsigned | 000000 |  |  | 00000 | 00000 | 011001 | R | (LO, HI) = rs x rt, treat as unsigned |
| OR | Bitwise or | 000000 |  |  |  | 00000 | 100101 | R | rd = rs OR rt, bitwise |
| ORI | Bitwise or immediate | 001101 |  |  | immediate | | | I | rd = rs OR (immediate zeroex), bitwise |
| SB | Store byte | 101000 | base |  | offset | | | I | 8 bit, memory[base + offset signex] = rt[7:0] |
| SH | Store half-word | 101001 | base |  | offset | | | I | 16 bit, memory[base + offset signex] = rt[15:0] |
| SLL | Shift left logical | 000000 | 00000 |  |  |  | 000000 | R | rd = rt << sa, insert zeroes right |
| SLLV | Shift left logical variable | 000000 |  |  |  | 00000 | 000100 | R | rd = rt << rs, insert zeroes right |
| SLT | Set on less than (signed) | 000000 |  |  |  | 00000 | 101010 | R | rd = (rs < rt), either 0 or 1, treat as signed |
| SLTI | Set on less than immediate (signed) | 001010 |  |  | immediate | | | I | rd = (rs < im signex), either 0 or 1, treat as signed |
| SLTIU | Set on less than immediate unsigned | 001011 |  |  | immediate | | | I | rd = (rs < im signex), either 0 or 1, treat as unsigned |
| SLTU | Set on less than unsigned | 000000 |  |  |  | 00000 | 101011 | R | rd = (rs < rt), either 0 or 1, treat as unsigned |
| SRA | Shift right arithmetic | 000000 | 00000 |  |  |  | 000011 | R | rd = rt >> sa, duplicate sign bit 31 left |
| SRAV | Shift right arithmetic variable | 000000 |  |  |  | 00000 | 000111 | R | rd = rt >> rs, duplicate sign bit 31 left |
| SRL | Shift right logical | 000000 | 00000 |  |  |  | 000010 | R | rd = rt >> sa, insert zeroes right |
| SRLV | Shift right logical variable | 000000 |  |  |  | 00000 | 000110 | R | rd = rt >> rs, insert zeroes right |
| SUBU | Subtract unsigned | 000000 |  |  |  | 00000 | 100011 | R | rd = rs - rt |
| SW | Store word | 101011 | base |  | offset | | | I | 32 bit, memory[base + offset signex]=rt |
| XOR | Bitwise exclusive or | 000000 |  |  |  | 00000 | 100110 | R | rd = rs XOR rt, bitwise |
| XORI | Bitwise exclusive or immediate | 001110 |  |  | immediate | | | I | rd = rs XOR (immediate zeroex), bitwise |

*From <*[*https://github.com/m8pple/elec50010-2021-cpu-cw*](https://github.com/m8pple/elec50010-2021-cpu-cw)*>*