指令集规格

指令格式：[ 操作码4位 | 操作数4位 ]  
寄存器：R0（00）、R1（01）、R2（10）、R3（11）  
立即数：4位（符号扩展为4位运算）

指令表

| 指令格式 | 操作码（4位） | 操作数（4位） | 功能描述 | 二进制示例 | 执行动作说明 |
| --- | --- | --- | --- | --- | --- |
| ADD Rd, Rs | 0000 | Rd(2位) Rs(2位) | 寄存器加法：Rd ← Rd + Rs | 0000 0110 | ALU加法，结果写回Rd |
| SUB Rd, Rs | 0001 | Rd(2位) Rs(2位) | 寄存器减法：Rd ← Rd - Rs | 0001 1001 | ALU减法， |
| LOAD Rd, [A] | 0010 | Rd(2位) A(2位) | 加载内存：Rd ← MEM[A] | 0010 1011 | 地址总线输出A，数据总线输入 |
| STORE Rs, [A] | 0011 | A(2位) Rs(2位) | 存储内存：MEM[A] ← Rs | 0011 1101 | 地址总线输出A，数据总线输出 |
| JZ Rd, offset | 0100 | Rd(2位) offset(2位) | 条件跳转：若Rd=0则PC+=offset | 0100 0101 | 符号扩展offset，修改PC |

| 指令助记符 | 操作数 | 功能描述 | 建议的16位二进制格式 (字段名[位宽]) | 主操作码(示例) |
| --- | --- | --- | --- | --- |
| LDI Rd, imm8 | Rd， imm8 | Rd ← imm8 (将8位立即数 imm8 加载到目标寄存器 Rd) | Op[3:0] Rd[3:0] imm8[7:0] | 0000 |
| ADD Rd, Rs1, Rs2 | Rd， Rs1， Rs2 | Rd ← Rs1 + Rs2 (寄存器 Rs1 和 Rs2 的内容相加，结果存入 Rd) | Op[3:0] Rd[3:0] Rs1[3:0] Rs2[3:0] | 0001 |
| SUB Rd, Rs1, Rs2 | Rd， Rs1， Rs2 | Rd ← Rs1 - Rs2 (寄存器 Rs1 的内容减去 Rs2 的内容，结果存入 Rd) | Op[3:0] Rd[3:0] Rs1[3:0] Rs2[3:0] | 0010 |
| LOAD Rd, [A] | Rd， A | Rd ← MEM[A] (将数据存储器中地址 A (8位) 的内容加载到寄存器 Rd) | Op[3:0] Rd[3:0] A[7:0] | 0011 |
| STORE Rs, [A] | Rs， A | MEM[A] ← Rs (将寄存器 Rs 的内容存储到数据存储器中地址 A (8位)) | Op[3:0] Rs[3:0] A[7:0] | 0100 |
| JRZ Rd, offset | Rd， offset | IF Rd == 0 THEN PC ← PC + sign\_extend(offset) (若寄存器Rd为0, 则相对跳转。offset为字偏移量，相对于PC+1) | Op[3:0] Rd[3:0] offset[7:0] | 0101 |

Test2

| 地址 | 助记符 | 十六进制<br>机器码 | 二进制机器码 (Op Rd Rs1 Rs2 / Op Rd Imm / Op Rs Addr / Op Rd Offset) | 注释 (指令地址: 十六进制机器码 ; 行为) |
| --- | --- | --- | --- | --- |
| 0x00 | LDI R1, 0x05 | 0105 | 0000 0001 0000 0101 | 00: 0105 ; R1 ← 0x05 (将立即数5加载到寄存器R1) |
| 0x01 | LDI R2, 0x0A | 020A | 0000 0010 0000 1010 | 01: 020A ; R2 ← 0x0A (将立即数10加载到寄存器R2) |
| 0x02 | ADD R3, R1, R2 | 1312 | 0001 0011 0001 0010 | 02: 1312 ; R3 ← R1 + R2 (0x05 + 0x0A = 0x0F) |
| 0x03 | STORE R3, [0x10] | 4310 | 0100 0011 0001 0000 | 03: 4310 ; Mem[0x10] ← R3 (将R3的内容0x0F存储到内存地址0x10) |
| 0x04 | LDI R4, 0x00 | 0400 | 0000 0100 0000 0000 | 04: 0400 ; R4 ← 0x00 (将寄存器R4清零) |
| 0x05 | LOAD R4, [0x10] | 3410 | 0011 0100 0001 0000 | 05: 3410 ; R4 ← Mem[0x10] (从内存地址0x10加载数据到R4, R4应为0x0F) |
| 0x06 | SUB R5, R4, R1 | 2541 | 0010 0101 0100 0001 | 06: 2541 ; R5 ← R4 - R1 (0x0F - 0x05 = 0x0A) |
| 0x07 | LDI R6, 0x00 | 0600 | 0000 0110 0000 0000 | 07: 0600 ; R6 ← 0x00 (将寄存器R6清零，用于JRZ测试) |
| 0x08 | JRZ R6, +2 | 5602 | 0101 0110 0000 0010 | 08: 5602 ; 若R6==0则跳转。PC=0x08, PC+1=0x09。目标地址 = 0x09 + 2 = 0x0B。跳转发生。 |
| 0x09 | LDI R7, 0x7F | 077F | 0000 0111 0111 1111 | 09: 077F ; 此指令应被跳过 (LDI R7, 127) |
| 0x0A | LDI R8, 0x55 | 0855 | 0000 1000 0101 0101 | 0A: 0855 ; 此指令应被跳过 (LDI R8, 85) |
| 0x0B | LDI R9, 0x3C | 093C | 0000 1001 0011 1100 | 0B: 093C ; R9 ← 0x3C (将立即数60加载到R9, JRZ跳转到此) |
| 0x0C | LDI R0, 0x00 | 0000 | 0000 0000 0000 0000 | 0C: 0000 ; R0 ← 0x00 (将寄存器R0清零，用于无限循环) |
| 0x0D | JRZ R0, -1 | 50FF | 0101 0000 1111 1111 | 0D: 50FF ; 若R0==0则跳转。PC=0x0D, PC+1=0x0E。目标地址 = 0x0E - 1 = 0x0D。无限循环。 (offset -1 is 0xFF) |