

ps75cgvv

| | | | | | | |
|---------|-------|-----|-----|----|---------|--------|
| 0000000 | rs2 | rs1 | 000 | rd | 0110011 | R add |
| 0000000 | rs2 | rs1 | 000 | rd | 0110011 | R sub |
| 0000000 | rs2 | rs1 | 001 | rd | 0110011 | R sll |
| 0000000 | rs2 | rs1 | 010 | rd | 0110011 | R slt |
| 0000000 | rs2 | rs1 | 011 | rd | 0110011 | R sltu |
| 0000000 | rs2 | rs1 | 100 | rd | 0110011 | R xor |
| 0000000 | rs2 | rs1 | 101 | rd | 0110011 | R srl |
| 0000000 | rs2 | rs1 | 101 | rd | 0110011 | R sra |
| 0000000 | rs2 | rs1 | 110 | rd | 0110011 | R or |
| 0000000 | rs2 | rs1 | 111 | rd | 0110011 | R and |
| 0000000 | shamt | rs1 | 001 | rd | 0010011 | I slli |
| 0000000 | shamt | rs1 | 101 | rd | 0010011 | I srli |
| 0000000 | shamt | rs1 | 101 | rd | 0010011 | I srai |

ps75cgvv

| | | | | | | | | | | | | | | | |
|------------|----|-----------|----|-----|---------|-----|------------|--------|----|----------|---|---------|---|--------|--|
| 31 | 30 | 25 | 24 | 21 | 20 | 19 | 15 | 14 | 12 | 11 | 8 | 7 | 6 | 0 | |
| funct7 | | | | rs2 | | rs1 | | funct3 | | rd | | opcode | | R-type | |
| imm[11:0] | | | | | | rs1 | | funct3 | | rd | | opcode | | I-type | |
| imm[11:5] | | | | rs2 | | rs1 | | funct3 | | imm[4:0] | | opcode | | S-type | |
| imm[12] | | imm[10:5] | | rs2 | | rs1 | | funct3 | | imm[4:1] | | imm[11] | | B-type | |
| imm[31:12] | | | | | | | | | | rd | | opcode | | U-type | |
| imm[20] | | imm[10:1] | | | imm[11] | | imm[19:12] | | | rd | | opcode | | J-type | |

指令包含两类信息：

1. 身份信息：opcode、funct3、funct7 → CU
2. 参数信息：rs1、rs2、rd → REGFILES
imm12、imm20 → ALU

立即数相关指令

ps75cgvv

| | | | | | | |
|-----------|-------|-----|-----|----|---------|---------|
| imm[11:0] | | rs1 | 000 | rd | 0010011 | I addi |
| imm[11:0] | | rs1 | 010 | rd | 0010011 | I slti |
| imm[11:0] | | rs1 | 011 | rd | 0010011 | I sltiu |
| imm[11:0] | | rs1 | 100 | rd | 0010011 | I xori |
| imm[11:0] | | rs1 | 110 | rd | 0010011 | I ori |
| imm[11:0] | | rs1 | 111 | rd | 0010011 | I andi |
| 0000000 | shamt | rs1 | 001 | rd | 0010011 | I slli |
| 0000000 | shamt | rs1 | 101 | rd | 0010011 | I srli |
| 0100000 | shamt | rs1 | 101 | rd | 0010011 | I srai |
| imm[11:0] | | rs1 | 010 | rd | 0000011 | I lw |

```

begin
    case(opcode)
        7'b0110011:    //r
        begin
            regWe=1'b1;
            pcSourceCode=2'b00;
            memWe=1'b0;
            bIsImm=1'b0;
            bIs20bImm=1'b0;
            regDataIsFromMem=1'b0;
            regDataIsFromPC4=1'b0;
            case(func3)
                3'b000:
                begin
                    if(func7==0) aluOpCode=4'b0000; //add
                    else aluOpCode=4'b0001; //sub
                end
                3'b111: aluOpCode=4'b0010; //and
                3'b110: aluOpCode=4'b0011; //or
                3'b100: aluOpCode=4'b0100; //xor
                3'b001: aluOpCode=4'b0101;    //sll
                3'b101:
                begin
                    if(func7==0) aluOpCode=4'b0110; //sr1
                    else aluOpCode=4'b0111;    //sra
                end
            endcase
        end
        7'b0010011: //i
        begin
            regWe=1'b1;
            pcSourceCode=2'b00;
            memWe=1'b0;
            bIsImm=1'b1;
            bIs20bImm=1'b0;

```

```

73         bIs20bImm=1'b0;
74         regDataIsFromMem=1'b0;
75         regDataIsFromPC4=1'b0;
76     case(func3)
77         3'b000:aluOpCode=4'b0000; //addi
78         3'b111:aluOpCode=4'b0010; //andi
79         3'b110:aluOpCode=4'b0011; // ori
80         3'b100:aluOpCode=4'b0100; // xori
81         3'b001:aluOpCode=4'b0101; // slli
82     3'b101:
83         begin
84             if(func7==0) aluOpCode=4'b0110; // srli
85             else aluOpCode=4'b0111; //srai
86         end
87     endcase
88 end
89 7'b0000011: //lw
90 begin
91     regWe=1'b1;
92     pcSourceCode=2'b00;
93     memWe=1'b0;
94     bIsImm=1'b1;
95     bIs20bImm=1'b0;
96     regDataIsFromMem=1'b1;
97     regDataIsFromPC4=1'b0;
98     aluOpCode=4'b1000; //lw
99 end
100 7'b0100011://sw
101 begin
102     regWe=1'b0;
103     pcSourceCode=2'b00;
104     memWe=1'b1;
105     bIsImm=1'b1;
106     bIs20bImm=1'b0;
107     regDataIsFromMem=1'b0;
108     regDataIsFromPC4=1'b0;
109     aluOpCode=4'b0000; //sw

```

```

109         aluOpCode=4'b0000; //sw
110     end
111 7'b0110111://lui
112     begin
113         regWe=1'b1;
114         pcSourceCode=2'b00;
115         memWe=1'b0;
116         bIsImm=1'b1;
117         bIs20bImm=1'b1;
118         regDataIsFromMem=1'b0;
119         regDataIsFromPC4=1'b0;
120         aluOpCode=4'b1010; //lui
121     end
122 7'b1100011://b
123     begin
124         regWe=1'b0;
125         pcSourceCode=condition?2'b01:2'b00;
126         memWe=1'b0;
127         bIsImm=1'b0;
128         bIs20bImm=1'b0;
129         regDataIsFromMem=1'b0;
130         regDataIsFromPC4=1'b0;
131     case(func3)
132         3'b000:aluOpCode=4'b1011; //beq
133         3'b100:aluOpCode=4'b1100; //b1t
134         3'b110:aluOpCode=4'b1101; //b1tu
135     endcase
136     end
137 7'b1101111:
138     begin
139         regWe=1'b1;
140         pcSourceCode=2'b10;
141         memWe=1'b0;
142         bIsImm=1'b1;
143         bIs20bImm=1'b1;
144         regDataIsFromMem=1'b0;

```

```
144         regDataIsFromMem=1'b0;
145         regDataIsFromPC4=1'b1;
146         aluOpCode=4'b1110; //jal
147     end
148 endcase
149 end
150 // assign pcSourceCode=2'b00;
151 // assign regWe=1'b1;
152 // assign memWe=1'b0;
153 // assign bIsImm=1'b1;
154 // assign bIs20bImm=1'b0;
155 // assign regDataIsFromMem=1'b0;
156 // assign regDataIsFromPC4=1'b0;
157 // assign aluOpCode=4'b0011;
158
159
160 endmodule
```