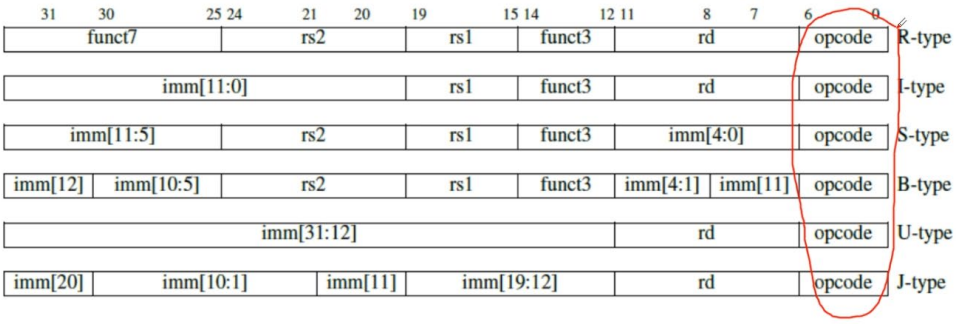
funct7

0000000	rs2	rs1	000	rd	0110011	R add
0100000	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	Rsltu
0000000	rs2	rs1	100	rd	0110011	R xor
0000000	rs2	rs1	101	rd	0110011	R srl
0100000	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
0100000	shamt	rs1	101	rd	0010011	I srai



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

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(funct3)
                3'b000:
                    begin
                        if(funct7==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(funct7==0) aluOpCode=4'b0110;//srl
                        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;
```

```
                bIs20bImm=1'b0;
                regDataIsFromMem=1'b0;
                regDataIsFromPC4=1'b0;
                case(funct3)
                    3'b000:aluOpCode=4'b0000; //addi
                    3'b111:aluOpCode=4'b0010; //andi
                    3'b110:aluOpCode=4'b0011; // ori
                    3'b100:aluOpCode=4'b0100; // xori
                    3'b001:aluOpCode=4'b0101; // slli
                    3'b101:
                        begin
                            if(funct7==0) aluOpCode=4'b0110; // srli
                            else aluOpCode=4'b0111; //srai
                        end
                endcase
            end
            7'b0000011: //lw
            begin
                regWe=1'b1;
                pcSourceCode=2'b00;
                memWe=1'b0;
                bIsImm=1'b1;
                bIs20bImm=1'b0;
                regDataIsFromMem=1'b1;
                regDataIsFromPC4=1'b0;
                aluOpCode=4'b1000; //lw
            end
            7'b0100011://sw
                begin
                    regWe=1'b0;
                    pcSourceCode=2'b00;
                    memWe=1'b1;
                    bIsImm=1'b1;
                    bIs20bImm=1'b0;
                    regDataIsFromMem=1'b0;
                    regDataIsFromPC4=1'b0;
                    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;
```

```
127                     bIsImm=1'b0;
128                     bIs20bImm=1'b0;
129                     regDataIsFromMem=1'b0;
130                     regDataIsFromPC4=1'b0;
131                     case(funct3)
132                         3'b000:aluOpCode=4'b1011; //beq
133                         3'b100:aluOpCode=4'b1100; //blt
134                         3'b110:aluOpCode=4'b1101; //bltu
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;
```

```
                    regDataIsFromMem=1'b0;
                    regDataIsFromPC4=1'b1;
                    aluOpCode=4'b1110; //jal
                end
        endcase
    end
//    assign pcSourceCode=2'b00;
//    assign regWe=1'b1;
//    assign memWe=1'b0;
//    assign bIsImm=1'b1;
//    assign bIs20bImm=1'b0;
//    assign regDataIsFromMem=1'b0;
//    assign regDataIsFromPC4=1'b0;
//    assign aluOpCode=4'b0011;


endmodule
```