指令的control信号

阶段一：运算指令

SSLI、SRLI、SRAI

- 偏移立即数指令，三条指令分别为逻辑左移、逻辑右移、算数右移
- **opcode**为0010011，**SLLI**、**SRLI**、**SRAI**的**func3**字段分别为001、101、101
- 指令具体格式如下，按inst[31:0]的顺序（**src**和**dest**各占5位，**opcode**占7位，**fun3**占3位）
  - *000000 位移次数[4:0] src SLLI dest opcode*
  - *000000 位移次数[4:0] src SRLI dest opcode*
  - *010000 位移次数[4:0] src SRAI dest opcode*

ADD、SUB、SLL、SRL、SRA、SLT、SLTU、XOR、OR、AND

- 加法、减法、逻辑左移、逻辑右移、算数右移、符号数比较（*rs1 < rs2* ***?*** *rd = 1* ***:*** *rd = 0*）、无符号数比较（*rs1 < rs2* ***?*** *rd = 1* ***:*** *rd = 0*）、亦或、或、与
- **opcode**为0110011，不同指令的**func3**字段分别为：**ADD/SUB**（000）、**SLL**（001）、**SLT**（010）、**SLTU**（011）、**XOR**（100）、**SRL /SRA**（101）、**OR**（110）、**AND**（111）
- 指令具体格式如下，按inst[31:0]的顺序（**src**和**dest**各占5位，**opcode**占7位，**fun3**占3位）
  - *000000 src2 src1 ADD/SLT/SLTU dest opcode*
  - *000000 src2 src1 XOR/OR/AND dest opcode*
  - *000000 src2 src1 SLL/SRL dest opcode*
  - *010000 src2 src1 SUB/SRA dest opcode*

ADDI、SLTI、SLTIU、XORI、ORI、ANDI

- 加法、有符号数比较（*rs1 < imm* ***?*** *rd = 1* ***:*** *rd = 0*）、无符号数比较（*rs1 < imm* ***?*** *rd = 1* ***:*** *rd = 0*）、亦或、或、与
- **opcode**为0010011，不同指令的**func3**字段分别为：**ADDI**（000）、**SLTI**（010）、**SLTIU**（011）、**XORI**（100）、**ORI**（110）、**ANDI**（111）
- 指令具体格式如下，按inst[31:0]的顺序（**src**和**dest**各占5位，**opcode**占7位，**fun3**占3位）
  - *I立即数[11:0] src ADDI/SLTI/SLTIU dest opcode*
  - *I立即数[11:0] src ANDI/ORI/XORI dest opcode*

LUI、AUIPC

- 立即数加载指令（将20位的U立即数放到目标寄存器rd的31-12位，rd的低12位填0，32位结果将符号拓展到64位）、PC相对地址计算指令（将20位的U立即数低位添加12个0，符号位扩展成64位，将其加到PC上，结果写入rd）
- **LUI**的**opcode**为0110111，**AUIPC**的**opcode**为0010111
- 指令具体格式如下，按inst[31:0]的顺序（**rd**占5位，**opcode**占7位）
  - imm[31:12] rd opcode

二：装载指令、跳转分支指令

JALR

- 间接跳转指令（将12位有符号I类立即数加上**rs1**，将结果的最低位设置位0，作为目标地址，原始的**pc+4**保存到寄存器**rd**中）
- **JALR**的**opcode**为1100111
- 指令具体格式如下，按inst[31:0]的顺序（**rd**和**rs1**占5位，**opcode**占7位）
  - *imm[11:0] rs1 000 rd opcode*

JAL

- 跳转并连接指令（将J立即数编码的2的倍数的有符号偏移量符号扩展，加到PC上，形成目标跳转地址，原始的**PC+4**保存到寄存器rd中）
- **JAL**的**opcode**为1100111
- 指令具体格式如下，按inst[31:0]的顺序（**rd**占5位，**opcode**占7位）
  - *imm[20] imm[10:1] imm[11] imm[19:12] rd opcode*

BEQ、BNE、BLT、BGE、BLTU、BGEU

- 分支指令，**BEQ**：**rs1 == rs2** 则跳转；**BNE**：**rs1 != rs2 则跳转；BLT：rs1 <= rs2 则跳转；BGT：rs1 >= rs2 则跳转；BLTU和BGEU分别是BLT和BGE**的无符号数版本。所有分支指令采用SB类指令格式，12位B立即数编码了以2字节倍数的有符号偏移量，并被加到当前**PC**上，生成目标地址
- **opcode**为1100011，不同指令的**func3**字段分别为：**BEQ**（000）、**BNE**（001）、**BLT**（100）、**BGE**（101）、**BLTU**（110）、**BGEU**（111）
- 指令具体格式如下，按inst[31:0]的顺序（**rs1**和**rs2**占5位，**func3**占5位，**opcode**占7位）
  - *imm[12, 10:5] rs2 rs1 func3 imm[4:1, 11] opcode*

LB、LH、LW、LBU、LHU

- **load**指令，将寄存器**rs1**与符号扩展的12位偏移量下相加得到有效地址，将存储器的一个值复制到寄存器**rd**中。这些**load**指令的区别在于，加载的数值位数不同。**LW**指令读取一个32位数值；**LH**指令读取一个16位数值后，符号扩展到32位；**LHU**指令读取一个16位无符号数值，零扩展到32位；**LB**指令读取一个8位数值，符号扩展到32位；**LBU**指令读取一个8位无符号数值，零扩展到32位
- **opcode**为0000011，不同指令的**func3**字段分别为：**LB**（000）、**LH**（001）、**LW**（010）、**LBU**（100）、**LHU**（101）
- 指令具体格式如下，按inst[31:0]的顺序（**rs1**和**rd**占5位，**func3**占5位，**opcode**占7位）
  - *imm[11:0] rs1 func3 rd opcode*

SB、SH、SW

- **store**指令，将寄存器**rs1**与符号扩展的12位偏移量下相加得到有效地址，将寄存器**rs2**的值复制到存储器中。这些**store**指令的区别在于，存储的数值位数不同。**SW**指令存储一个32位数值；**SH**指令存储寄存器**rs2**中的低16位数值；**SB**指令存储寄存器**rs2**中的低8位数值
- **opcode**为0100011，不同指令的**func3**字段分别为：**SB**（000）、**SH**（001）、**SW**（010）
- 指令具体格式如下，按inst[31:0]的顺序（**rs1**和**rs2**占5位，**func3**占5位，**opcode**占7位）
  - *imm[11:5] rs2 rs1 func3 imm[4:0] opcode*