## 1 单周期处理器设计整体方案

### 指令选取

选取指令如表格1所示（其实能实现的指令比这些多，可以按自己的需求调整。ADDI，ORI，ANDI，ADD，SUB，SLL，SLTU，XOR，SRL，OR，AND，JAL，BEQ，LW，、SW，SLT，XORI，SLTI，SRLI，BGE，BNE）。

表1 指令功能与数目

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 操作码 | 助记符 | 功能 | 描述 |
| 1 | 0010011 | ADDI | R[rd] ← R[rs] + imm; PC ← PC + 4 | 立即数加法 |
| 2 | 0010011 | ORI | R[rd] ← R[rs] | imm; PC ← PC + 4 | 立即数按位或 |
| 3 | 0010011 | ANDI | R[rd] ← R[rs] & imm; PC ← PC + 4 | 立即数按位与 |
| 4 | 0110011 | ADD | R[rd] ← R[rs] + R[rt]; PC ← PC + 4 | 寄存器加法 |
| 5 | 0110011 | SUB | R[rd] ← R[rs] - R[rt]; PC ← PC + 4 | 寄存器减法 |
| 6 | 0110011 | SLL | R[rd] ← R[rs] << R[rt]; PC ← PC + 4 | 逻辑左移 |
| 7 | 0110011 | OR | R[rd] ← R[rs] | R[rt]; PC ← PC + 4 | 寄存器按位或 |
| 8 | 0110011 | AND | R[rd] ← R[rs] & R[rt]; PC ← PC + 4 | 寄存器按位与 |
| 9 | 0000011 | LW | R[rd] ← Mem[R[rs] + imm]; PC ← PC + 4 | 加载字 |
| 10 | 0100011 | SW | Mem[R[rs] + imm] ← R[rd]; PC ← PC + 4 | 存储字 |
| 11 | 0010011 | XORI | R[rd] ← R[rs] ^ imm; PC ← PC + 4 | 立即数按位异或 |
| 12 | 1100011 | BEQ | if (R[rs] == R[rt]) PC ← PC + imm else PC ← PC + 4 | 分支等于 |
| 13 | 1100011 | BGE | if (R[rs] >= R[rt]) PC ← PC + imm else PC ← PC + 4 | 分支大于等于 |
| 14 | 1100011 | BNE | if (R[rs] != R[rt]) PC ← PC + imm else PC ← PC + 4 | 分支不等于 |
| 指令总数目 | | | 14 | |

### 处理器电路整体结构

### 

图1 单周期处理器的结构图

表2 CPU模型控制信号列表

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 操作码op | 信号/指令 | aluscr | memtoreg | wr\_en | memwrite | op | aluop | |
|  | LW | 0 | 1 | 1 | 0 | 0000011 | ADD | 0001 |
|  | SW | 0 | 0 | 0 | 1 | 0100011 | ADD | 0001 |
|  | ADD | 1 | 0 | 1 | 0 | 0110011 | ADD | 0001 |
|  | SUB | 1 | 0 | 1 | 0 | 0110011 | SUB | 0011 |
|  | BGE | 1 | 0 | 0 | 0 | 1100011 | SUB | 0011 |
|  | BNE | 1 | 0 | 0 | 0 | 1100011 | SUB | 0011 |
|  | ORI | 0 | 0 | 1 | 0 | 0010011 | OR | 0101 |
|  | BEQ | 1 | 0 | 0 | 0 | 1100011 | SUB | 0011 |
|  | ADDI | 0 | 0 | 1 | 0 | 0010011 | ADD | 0001 |
|  | ORI | 0 | 0 | 1 | 0 | 0010011 | OR | 0101 |
|  | ANDI | 0 | 0 | 1 | 0 | 0010011 | AND | 0100 |
|  | SLL | 1 | 0 | 1 | 0 | 0110011 | SLL | 1100 |
|  | OR | 1 | 0 | 1 | 0 | 0110011 | OR | 0101 |
|  | AND | 1 | 0 | 1 | 0 | 0110011 | AND | 0100 |
|  | XORI | 0 | 0 | 1 | 0 | 0010011 | XOR | 0101 |

表3 排序汇编代码

|  |
| --- |
| RISC-V汇编源代码粘贴处： |
| addi x1 x0 72  addi x2 x0 99  addi x3 x0 45  addi x4 x0 48  addi x5 x0 32  addi x8 x0 1  addi x9 x0 5  bge x2 x1 20  add x6 x0 x1  add x7 x0 x2  add x1 x7 x0  add x2 x6 x0  bge x3 x2 20  add x6 x0 x2  add x7 x0 x3  add x2 x7 x0  add x3 x6 x0  bge x4 x3 20  add x6 x0 x3  add x7 x0 x4  add x3 x7 x0  add x4 x6 x0  bge x5 x4 20  add x6 x0 x4  add x7 x0 x5  add x4 x7 x0  add x5 x6 x0  sub x9 x9 x8  bne x9 x0 -84 |

表格4 指令与其对应的机械码

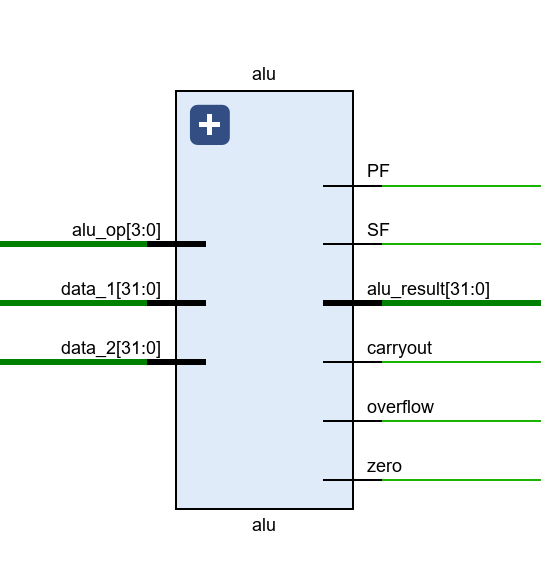
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 指令格式 | 具体指令 | Instr(31:12) | | | | Rd (11-7) | Op (6-0) |
| 1 | addi rd, rs1, imm | addi x1, x0, 72 | 0000000 | 00000 | 00001 | 000 | 01001 | 0010011 |
| 2 | addi rd, rs1, imm | addi x2, x0, 99 | 0000000 | 00000 | 00010 | 000 | 01010 | 0010011 |
| 3 | addi rd, rs1, imm | addi x3, x0, 45 | 0000000 | 00000 | 00011 | 000 | 01100 | 0010011 |
| 4 | addi rd, rs1, imm | addi x4, x0, 48 | 0000000 | 00000 | 00100 | 000 | 01101 | 0010011 |
| 5 | addi rd, rs1, imm | addi x5, x0, 32 | 0000000 | 00000 | 00101 | 000 | 01110 | 0010011 |
| 6 | addi rd, rs1, imm | addi x8, x0, 1 | 0000000 | 00000 | 01000 | 000 | 01000 | 0010011 |
| 7 | addi rd, rs1, imm | addi x9, x0, 5 | 0000000 | 00000 | 01001 | 000 | 01001 | 0010011 |
| 8 | bge rs1, rs2, imm | bge x2, x1, 20 | 0000000 | 01010 | 00001 | 101 | 00000 | 1100011 |
| 9 | add rd, rs1, rs2 | add x6, x0, x1 | 0000000 | 00001 | 00110 | 000 | 00110 | 0110011 |
| 10 | add rd, rs1, rs2 | add x7, x0, x2 | 0000000 | 00010 | 00111 | 000 | 00111 | 0110011 |
| 11 | add rd, rs1, rs2 | add x1, x7, x0 | 0000000 | 00000 | 00001 | 000 | 01001 | 0110011 |
| 12 | add rd, rs1, rs2 | add x2, x6, x0 | 0000000 | 00000 | 00010 | 000 | 01010 | 0110011 |
| 13 | bge rs1, rs2, imm | bge x3, x2, 20 | 0000000 | 00011 | 00100 | 101 | 00000 | 1100011 |
| 14 | add rd, rs1, rs2 | add x6, x0, x2 | 0000000 | 00010 | 00110 | 000 | 00110 | 0110011 |
| 15 | add rd, rs1, rs2 | add x7, x0, x3 | 0000000 | 00011 | 00111 | 000 | 00111 | 0110011 |
| 16 | add rd, rs1, rs2 | add x2, x7, x0 | 0000000 | 00000 | 00010 | 000 | 01010 | 0110011 |
| 17 | add rd, rs1, rs2 | add x3, x6, x0 | 0000000 | 00000 | 00011 | 000 | 01100 | 0110011 |
| 18 | bge rs1, rs2, imm | bge x4, x3, 20 | 0000000 | 00100 | 00101 | 101 | 00000 | 1100011 |
| 19 | add rd, rs1, rs2 | add x6, x0, x3 | 0000000 | 00011 | 00110 | 000 | 00110 | 0110011 |
| 20 | add rd, rs1, rs2 | add x7, x0, x4 | 0000000 | 00100 | 00111 | 000 | 00111 | 0110011 |
| 21 | add rd, rs1, rs2 | add x3, x7, x0 | 0000000 | 00000 | 00011 | 000 | 01100 | 0110011 |
| 22 | add rd, rs1, rs2 | add x4, x6, x0 | 0000000 | 00000 | 00100 | 000 | 01101 | 0110011 |
| 23 | bge rs1, rs2, imm | bge x5, x4, 20 | 0000000 | 00101 | 00101 | 101 | 00000 | 1100011 |
| 24 | add rd, rs1, rs2 | add x6, x0, x4 | 0000000 | 00100 | 00110 | 000 | 00110 | 0110011 |
| 25 | add rd, rs1, rs2 | add x7, x0, x5 | 0000000 | 00101 | 00111 | 000 | 00111 | 0110011 |
| 26 | add rd, rs1, rs2 | add x4, x7, x0 | 0000000 | 00000 | 00100 | 000 | 01101 | 0110011 |
| 27 | add rd, rs1, rs2 | add x5, x6, x0 | 0000000 | 00000 | 00101 | 000 | 01110 | 0110011 |
| 28 | sub rd, rs1, rs2 | sub x9, x9, x8 | 0100000 | 01000 | 01001 | 000 | 01001 | 0110011 |
| 29 | bne rs1, rs2, imm | bne x9, x0, -84 | 0000000 | 00000 | 01001 | 001 | 01001 | 1100011 |

表格5

|  |  |  |  |
| --- | --- | --- | --- |
| 指令序列 | 单周期处理器结果 | | 仿真结果 |
| addi x1 x0 72  addi x2 x0 99  addi x3 x0 45  addi x4 x0 48  addi x5 x0 32  addi x8 x0 1  addi x9 x0 5  bge x2 x1 20  add x6 x0 x1  add x7 x0 x2  add x1 x7 x0  add x2 x6 x0  bge x3 x2 20  add x6 x0 x2  add x7 x0 x3  add x2 x7 x0  add x3 x6 x0  bge x4 x3 20  add x6 x0 x3  add x7 x0 x4  add x3 x7 x0  add x4 x6 x0  bge x5 x4 20  add x6 x0 x4  add x7 x0 x5  add x4 x7 x0  add x5 x6 x0  sub x9 x9 x8  bne x9 x0 -84 | |  |  |

## 2 关键部件设计

### 2.1 alu模块



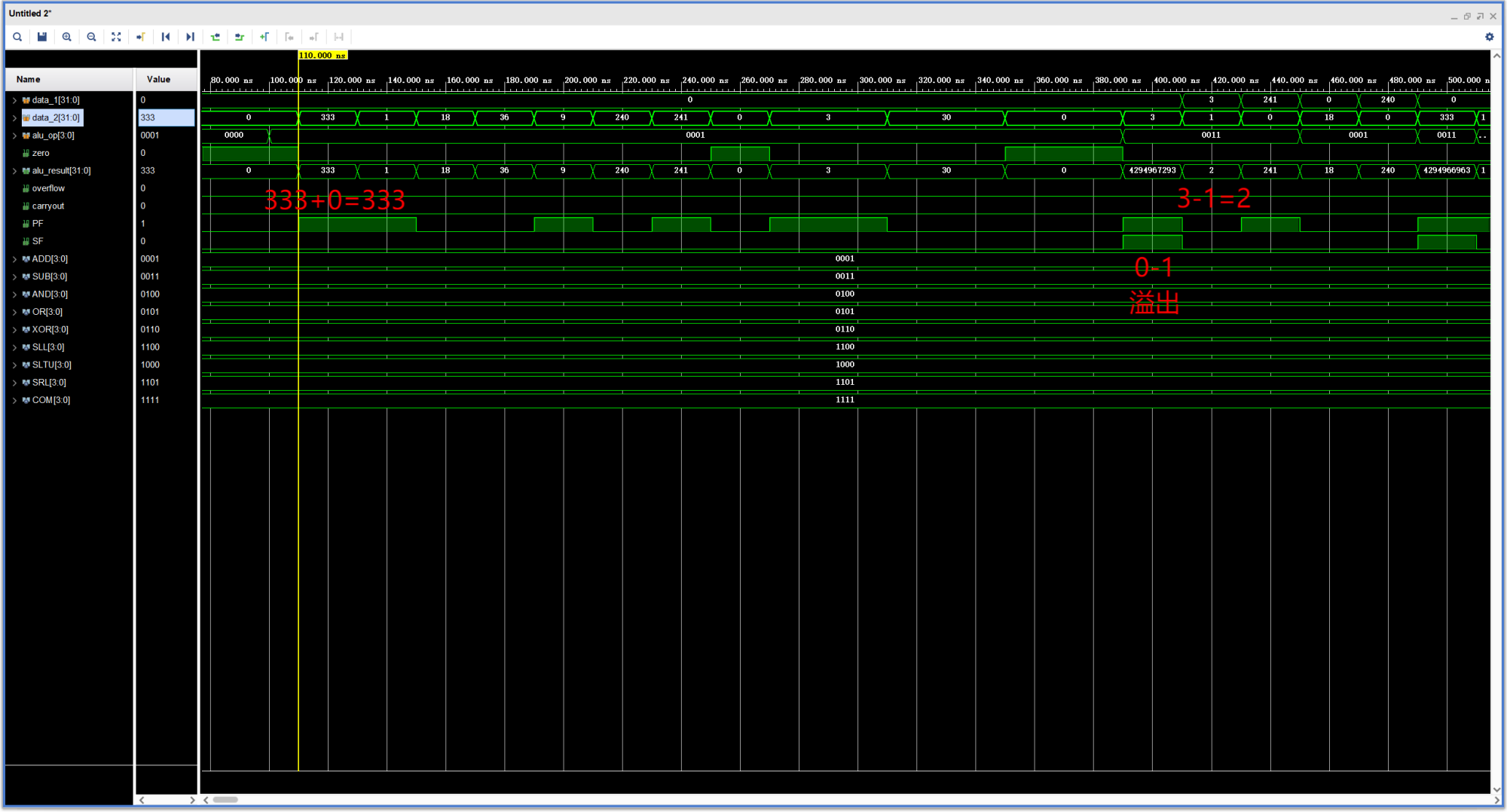
#### 2.1.1 功能描述

将输入的数据按输入的操作类型进行处理，将运算结果输出。

#### 2.1.2 接口信号

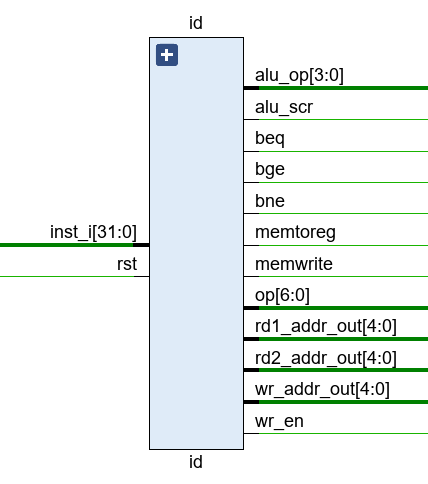
|  |  |  |  |
| --- | --- | --- | --- |
| 信号名称 | I/O | 说明 | 位宽 |
| alu\_op | I | 运算类型 | 4 |
| data\_1 | I | 运算数据 | 32 |
| data\_2 | I | 运算数据 | 32 |
| zero | O | 零标志 | 1 |
| alu\_result | O | 运算结果 | 32 |
| overflow | O | 溢出标志 | 1 |
| carryout | O | 进位标志 | 1 |
| PF | O | 奇偶标志 | 1 |
| SF | O | 符号位 | 1 |

#### 2.1.3 仿真结果



输入数据data\_1和data\_2根据alu\_op进行运算后得到的结果正确。

### 2.2 译码模块



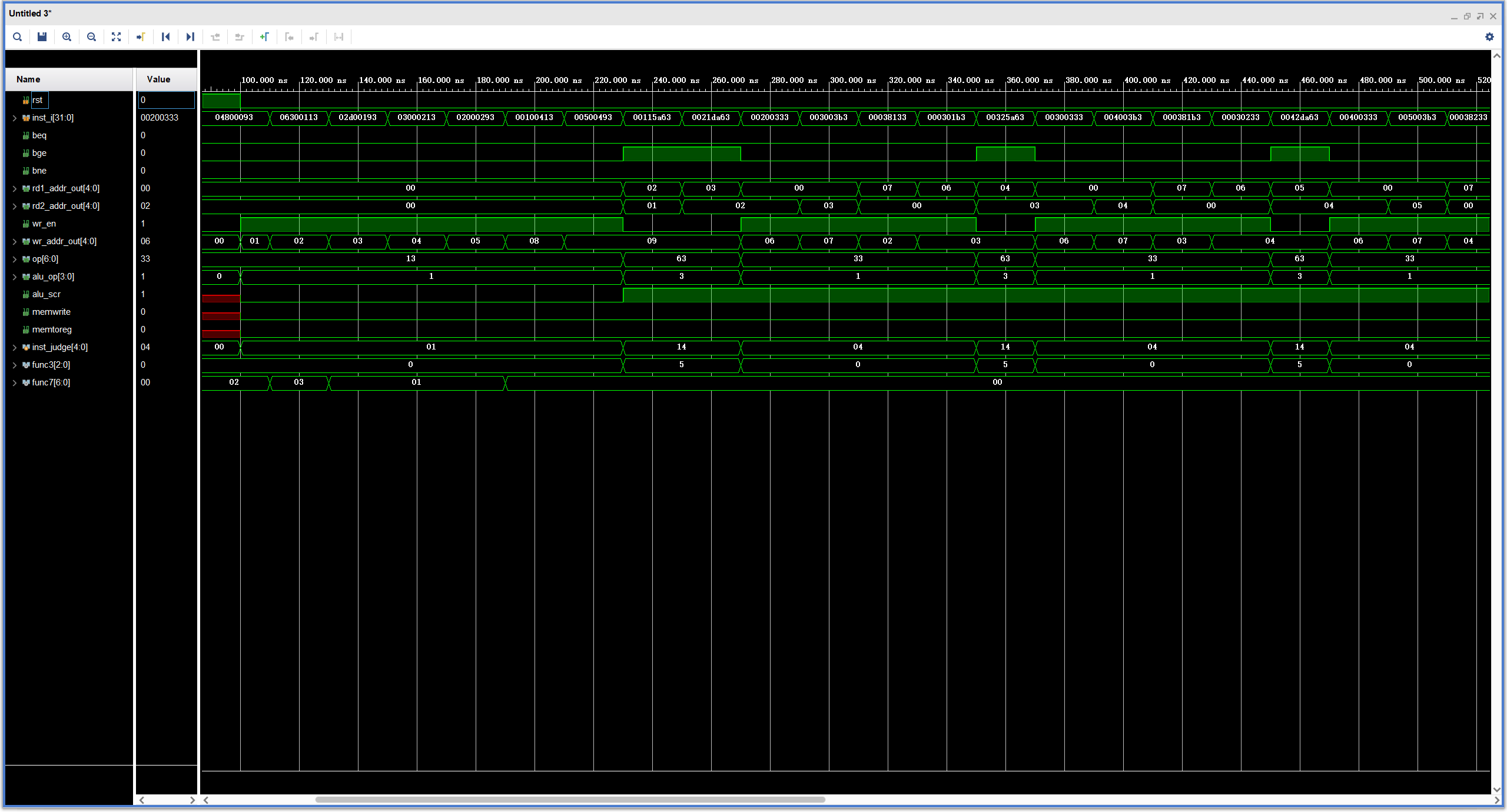
#### 2.2.1 功能描述

译码模块负责判断指令类型，并生成相应的控制信号，用于控制器件的使能和多路选择器的选择，根据不同指令的功能分析其所需要的路径，即可得到信号所对应的值。

#### 2.2.2 接口信号

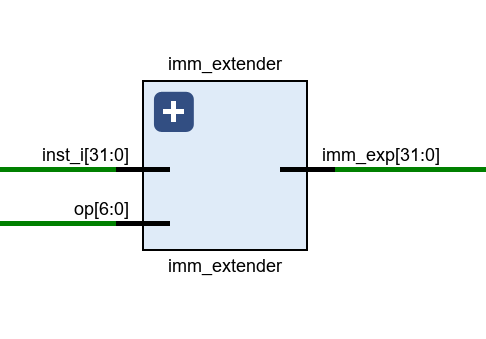
|  |  |  |  |
| --- | --- | --- | --- |
| 信号名称 | I/O | 说明 | 位宽 |
| inst\_i | I | 指令 | 32 |
| rst | I | 系统复位 | 1 |
| alu\_op | O | alu运算类型 | 4 |
| alu\_scr | O | 送入 ALU B 端口的值是立即数的 32 位扩展/寄存器堆读取的值 | 1 |
| beq | O | beq指令标志 | 1 |
| bge | O | bge指令标志 | 1 |
| bne | O | ben指令标志 | 1 |
| memtoreg | O | 回写的数据来自于 ALU 计算的结果/存储器读取的数据 | 1 |
| memwrite | O | 是否需要写数据存储器 | 1 |
| op | O | 指令类型（opcode） | 7 |
| rd1\_addr | O | rs1 | 5 |
| rd2\_addr | O | rs2 | 5 |
| wr\_addr | O | 寄存器堆写入的地址 | 5 |
| wr\_en | O | 寄存器堆写入使能 | 1 |

#### 2.2.3 仿真结果



将仿真结果与表2和表4进行对比，仿真结果正确。

### 2.3 立即数扩展模块



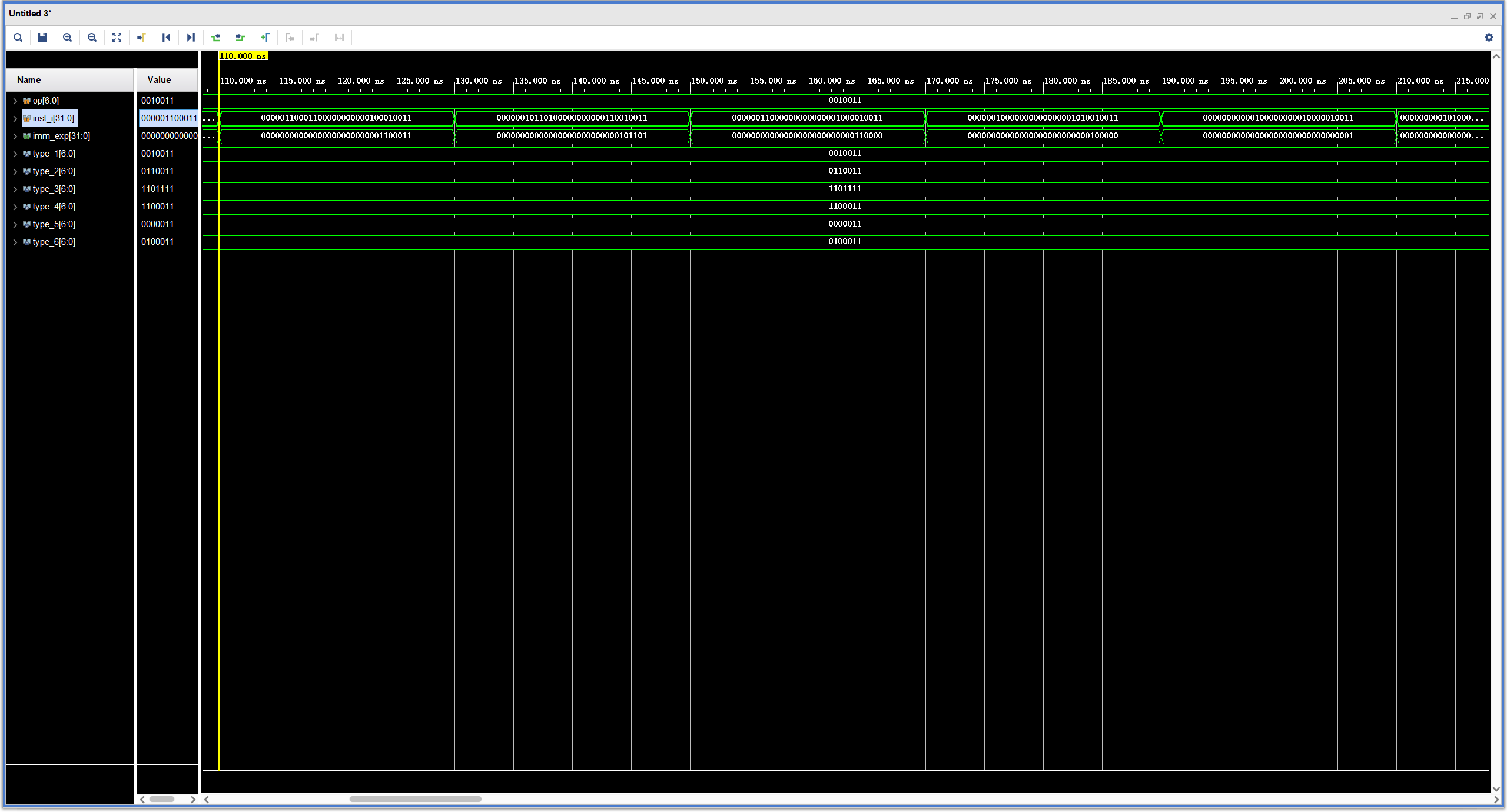
#### 2.3.1 功能描述

根据指令类型进行立即数扩展。

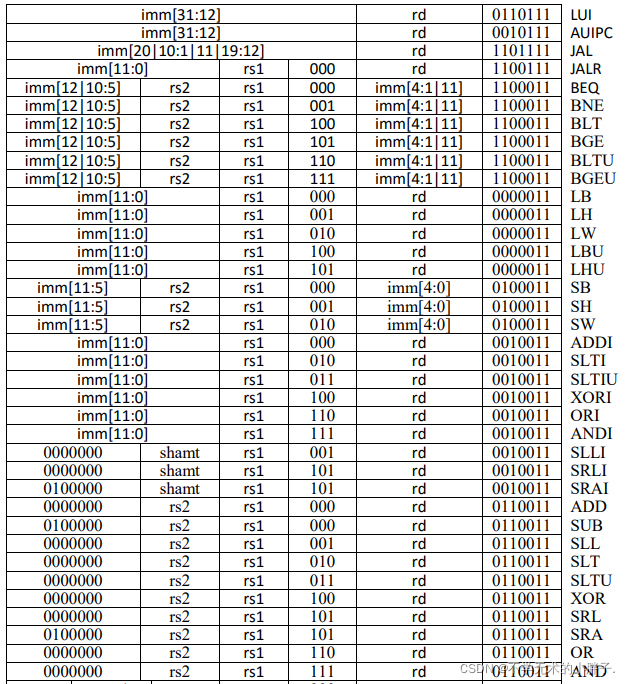
#### 2.3.2 接口信号

|  |  |  |  |
| --- | --- | --- | --- |
| 信号名称 | I/O | 说明 | 位宽 |
| inst\_i | I | 指令 | 32 |
| op | I | 指令类型 | 7 |
| imm\_exp | O | 进行扩展后的立即数 | 32 |

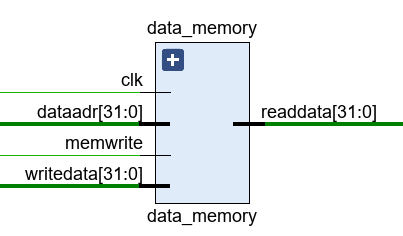
#### 2.3.3 仿真结果

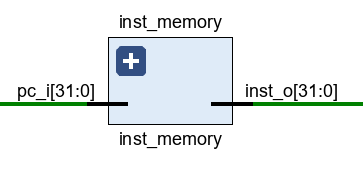


结合下图，立即数扩展正确



### 2.4 指令存储器和数据存储器模块





#### 2.4.1 功能描述

数据存储器：将处理后得到的数据存放在数据存储器处，根据指令类型将对应数据送出。

指令存储器：指令存储器用来存储指令，依据当前PC的输入，读取指令存储器中 相对应地址的指令，并输出。

#### 2.4.2 接口信号

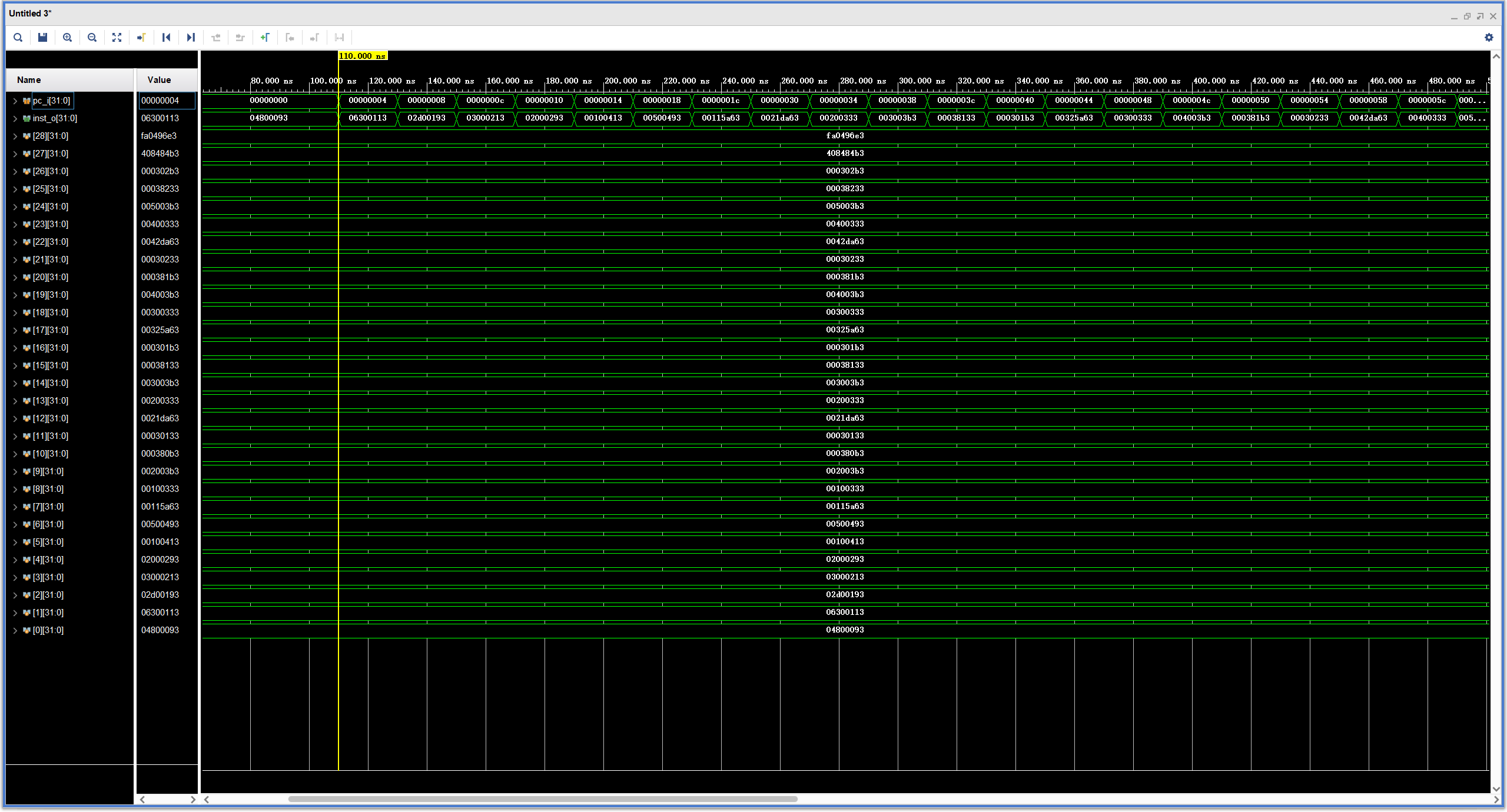
|  |  |  |  |
| --- | --- | --- | --- |
| 信号名称 | I/O | 说明 | 位宽 |
| clk | I | 系统时钟 | 1 |
| dataaddr | I | 地址 | 32 |
| memwrite | I | 写使能 | 1 |
| writedata | I | 写入的数据 | 32 |
| readdata | O | 读取的数据 | 32 |

数据存储器

|  |  |  |  |
| --- | --- | --- | --- |
| 信号名称 | I/O | 说明 | 位宽 |
| pc\_i | I | 指令地址 | 32 |
| inst\_o | O | 读取的指令 | 32 |

指令存储器

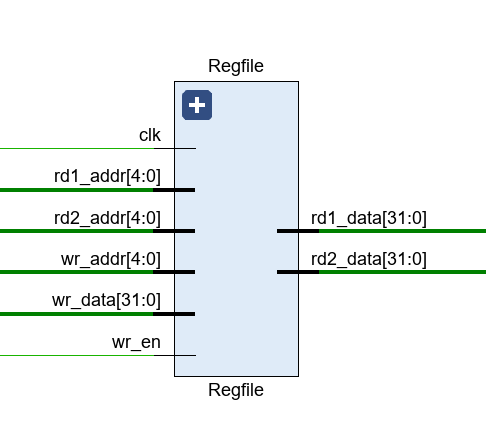
#### 2.4.3 仿真结果



指令存储器仿真

输入的地址和读取的指令一一对应，仿真结果正确。

### 2.5 寄存器堆模块



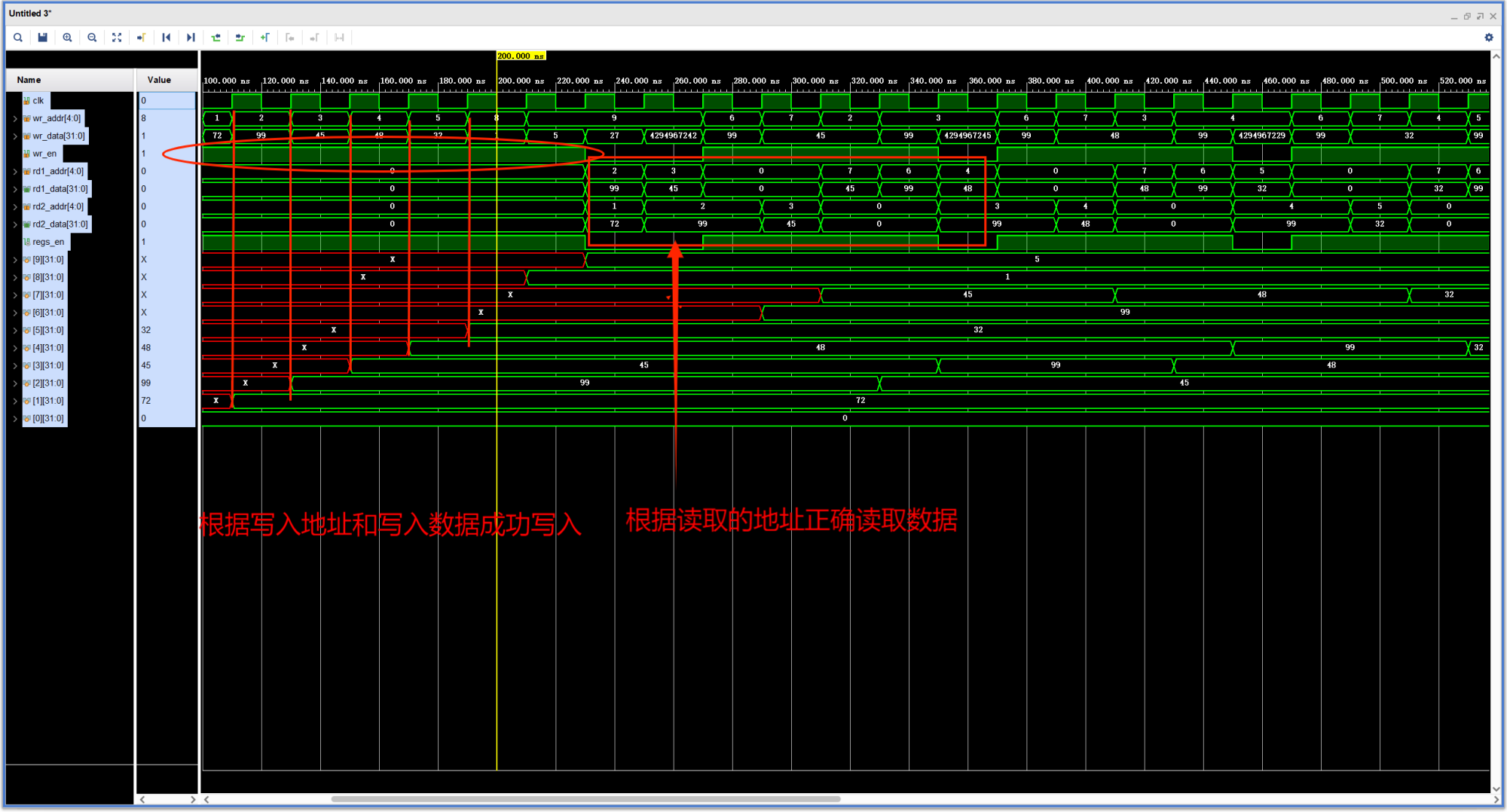
#### 2.5.1 功能描述

寄存器堆设计单元的功能是接收译码模块的控制信号和地址信号，对相应寄存器进行操作。通过一个32大小的32位寄存器数组来模拟寄存器，开始时全部置0。通过访问寄存器的地址，来获取寄存器里面的值，并进行操作。

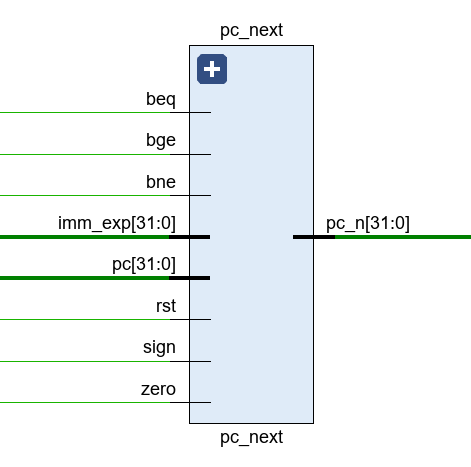
#### 2.5.2 接口信号

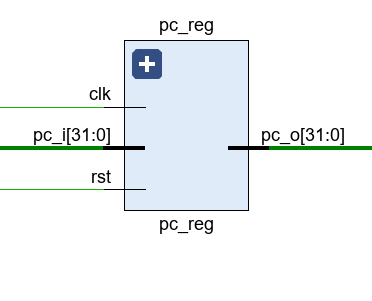
|  |  |  |  |
| --- | --- | --- | --- |
| 信号名称 | I/O | 说明 | 位宽 |
| clk | I | 系统时钟 | 1 |
| rd1\_addr | I | 读取地址1 | 5 |
| rd2\_addr | I | 读取地址2 | 5 |
| wr\_addr | I | 写入地址 | 5 |
| wr\_data | I | 写入数据 | 32 |
| wr\_en | I | 写入使能 | 1 |
| rd1\_data | O | 读取数据1 | 32 |
| rd2\_data | O | 读取数据2 | 32 |

#### 2.6.3 仿真结果



### 2.6 指令计数器模块





#### 2.6.1 功能描述

pc\_next根据译码模块的信号产生指令计数器的下一个地址。

pc\_next更新pc的值。

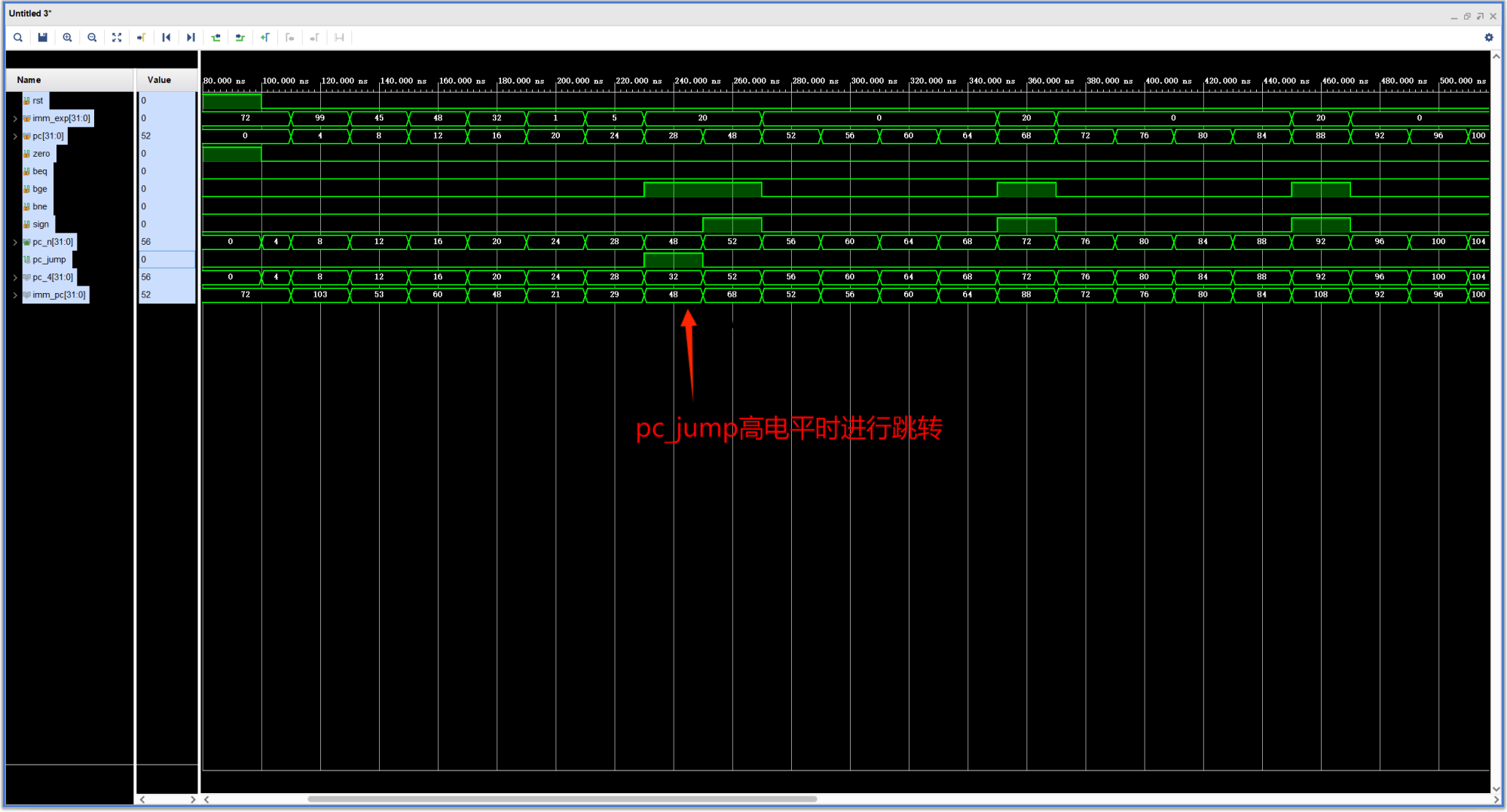
#### 2.6.2 接口信号

|  |  |  |  |
| --- | --- | --- | --- |
| 信号名称 | I/O | 说明 | 位宽 |
| beq | I | beq指令标志 | 1 |
| bge | I | bge指令标志 | 1 |
| bne | I | bne指令标志 | 1 |
| imm\_exp | I | 扩展后的立即数 | 32 |
| pc | I | 当前指令计数器的值 | 32 |
| rst | I | 系统复位 | 1 |
| sign | I | alu运算结果符号位 | 1 |
| zero | I | alu零标志 | 1 |
| pc\_n | O | 要跳转的地址 | 32 |

pc\_next模块

|  |  |  |  |
| --- | --- | --- | --- |
| 信号名称 | I/O | 说明 | 位宽 |
| clk | I | 系统始终 | 1 |
| pc\_i | I | 要跳转的之地 | 32 |
| rst | I | 系统复位 | 1 |
| pc\_o | O | 当前地址 | 32 |

#### 2.6.3 仿真结果



pc\_next