**计算机组成原理**

**课程设计报告**

**学 号\_\_\_\_\_ \_\_\_20020203 \_\_\_\_\_\_\_\_**

**姓 名\_\_\_\_\_\_\_\_\_\_\_王思哲\_\_\_\_\_\_\_\_\_\_\_**

**指导教师\_\_\_\_\_\_\_\_\_\_\_魏坚华\_\_\_\_\_\_\_\_\_\_\_**

**提交日期\_\_\_\_\_ \_2022年7月5日 \_\_\_**

**成绩评价表**

|  |  |  |
| --- | --- | --- |
| **报告内容** | **报告结构** | **报告最终成绩** |
| **□丰富正确**  **□基本正确**  **□有一些问题**  **□问题很大** | **□完全符合要求**  **□基本符合要求**  **□有比较多的缺陷**  **□完全不符合要求** |  |
| **报告与大作业功能一致性** | **报告图表** | **总体评价** |
| **□完全一致**  **□基本一致**  **□基本不一致** | **□符合规范**  **□基本符合规范**  **□有一些错误**  **□完全不正确** |  |

**教师签字:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

目录

[**Project1 VerilogHDL 完成多周期处理器开发** 6](#_Toc107950274)

[**一、总体数据通路结构设计图** 6](#_Toc107950275)

[**二、模块定义** 6](#_Toc107950276)

[2.1 ALU模块 6](#_Toc107950277)

[2.1.1 基本描述 6](#_Toc107950278)

[2.1.2 模块接口 6](#_Toc107950279)

[2.1.3 功能定义 7](#_Toc107950280)

[2.2 NPC模块 7](#_Toc107950281)

[2.2.1 基本描述 7](#_Toc107950282)

[2.2.2 模块接口 7](#_Toc107950283)

[2.2.3 功能定义 7](#_Toc107950284)

[2.3 PC模块 8](#_Toc107950285)

[2.3.1 基本描述 8](#_Toc107950286)

[2.3.2 模块接口 8](#_Toc107950287)

[2.3.3 功能定义 8](#_Toc107950288)

[2.4 IM模块 8](#_Toc107950289)

[2.4.1 基本描述 8](#_Toc107950290)

[2.4.2 模块接口 9](#_Toc107950291)

[2.4.3 功能定义 9](#_Toc107950292)

[2.5 IR模块 9](#_Toc107950293)

[2.5.1 基本描述 9](#_Toc107950294)

[2.5.2 模块接口 9](#_Toc107950295)

[2.5.3 功能定义 9](#_Toc107950296)

[2.6 GPR模块 10](#_Toc107950297)

[2.6.1 基本描述 10](#_Toc107950298)

[2.6.2 模块接口 10](#_Toc107950299)

[2.6.3 功能定义 10](#_Toc107950300)

[2.7 EXT模块 11](#_Toc107950301)

[2.7.1 基本描述 11](#_Toc107950302)

[2.7.2 模块接口 11](#_Toc107950303)

[2.7.3 功能定义 11](#_Toc107950304)

[2.8 aReg & bReg模块 11](#_Toc107950305)

[2.8.1 基本描述 11](#_Toc107950306)

[2.8.2 模块接口(以aReg模块为例) 11](#_Toc107950307)

[2.8.3 功能定义 12](#_Toc107950308)

[2.9 aluReg模块 12](#_Toc107950309)

[2.9.1 基本描述 12](#_Toc107950310)

[2.9.2 模块接口 12](#_Toc107950311)

[2.9.3 功能定义 12](#_Toc107950312)

[2.10 DM模块 12](#_Toc107950313)

[2.10.1 基本描述 12](#_Toc107950314)

[2.10.2 模块接口 13](#_Toc107950315)

[2.10.3 功能定义 13](#_Toc107950316)

[2.11 DR模块 13](#_Toc107950317)

[2.11.1 基本描述 13](#_Toc107950318)

[2.11.2 模块接口 13](#_Toc107950319)

[2.11.3 功能定义 13](#_Toc107950320)

[2.12 Controller模块 14](#_Toc107950321)

[2.12.1 基本描述 14](#_Toc107950322)

[2.12.2 模块接口 14](#_Toc107950323)

[2.12.3 功能定义 15](#_Toc107950324)

[2.13 Muxdmin模块 15](#_Toc107950325)

[2.13.1 基本描述 15](#_Toc107950326)

[2.13.2 模块接口 16](#_Toc107950327)

[2.13.3 功能定义 16](#_Toc107950328)

[**三、指令描述 16**](#_Toc107950329)

[**四、测试程序 17**](#_Toc107950330)

[4.1 MIPS-Lite2指令集的测试程序 17](#_Toc107950331)

[4.2 新增指令BLTZAL的测试程序 18](#_Toc107950332)

[**五、测试结果 18**](#_Toc107950333)

[5.1 MIPS-Lite2指令集测试结果 18](#_Toc107950334)

[5.1.1 MARS中结果 18](#_Toc107950335)

[5.1.2 Modelsim中结果 19](#_Toc107950336)

[5.2 BLTZAL指令测试结果 20](#_Toc107950337)

[5.2.1 MARS中结果 20](#_Toc107950338)

[5.2.2 Modelsim中结果 21](#_Toc107950339)

[**六、总结与心得体会 21**](#_Toc107950340)

[**Project2 VerilogHDL完成MIPS微系统开发(支持设备与中断) 22**](#_Toc107950341)

[**一、** **总体数据通路结构设计图** 22](#_Toc107950342)

[**二、** **模块定义** 23](#_Toc107950343)

[2.1 ALU模块 23](#_Toc107950344)

[2.1.1 基本描述 23](#_Toc107950345)

[2.1.2 模块接口 23](#_Toc107950346)

[2.1.3 功能定义 23](#_Toc107950347)

[2.2 NPC模块 23](#_Toc107950348)

[2.2.1 基本描述 23](#_Toc107950349)

[2.2.2 模块接口 24](#_Toc107950350)

[2.2.3 功能定义 24](#_Toc107950351)

[2.3 PC模块 25](#_Toc107950352)

[2.3.1 基本描述 25](#_Toc107950353)

[2.3.2 模块接口 25](#_Toc107950354)

[2.3.3 功能定义 25](#_Toc107950355)

[2.4 IM模块 25](#_Toc107950356)

[2.4.1 基本描述 25](#_Toc107950357)

[2.4.2 模块接口 25](#_Toc107950358)

[2.4.3 功能定义 26](#_Toc107950359)

[2.5 IR模块 26](#_Toc107950360)

[2.5.1 基本描述 26](#_Toc107950361)

[2.5.2 模块接口 26](#_Toc107950362)

[2.5.3 功能定义 26](#_Toc107950363)

[2.6 GPR模块 26](#_Toc107950364)

[2.6.1 基本描述 26](#_Toc107950365)

[2.6.2 模块接口 27](#_Toc107950366)

[2.6.3 功能定义 27](#_Toc107950367)

[2.7 EXT模块 27](#_Toc107950368)

[2.7.1 基本描述 27](#_Toc107950369)

[2.7.2 模块接口 28](#_Toc107950370)

[2.7.3 功能定义 28](#_Toc107950371)

[2.8 aReg & bReg模块 28](#_Toc107950372)

[2.8.1 基本描述 28](#_Toc107950373)

[2.8.2 模块接口(以aReg模块为例) 28](#_Toc107950374)

[2.8.3 功能定义 28](#_Toc107950375)

[2.9 aluReg模块 29](#_Toc107950376)

[2.9.1 基本描述 29](#_Toc107950377)

[2.9.2 模块接口 29](#_Toc107950378)

[2.9.3 功能定义 29](#_Toc107950379)

[2.10 DM模块 29](#_Toc107950380)

[2.10.1 基本描述 29](#_Toc107950381)

[2.10.2 模块接口 29](#_Toc107950382)

[2.10.3 功能定义 30](#_Toc107950383)

[2.11 DR模块 30](#_Toc107950384)

[2.11.1 基本描述 30](#_Toc107950385)

[2.11.2 模块接口 30](#_Toc107950386)

[2.11.3 功能定义 30](#_Toc107950387)

[2.12 Controller模块 31](#_Toc107950388)

[2.12.1 基本描述 31](#_Toc107950389)

[2.12.2 模块接口 32](#_Toc107950390)

[2.12.3 功能定义 33](#_Toc107950391)

[2.13 Muxdmin模块 33](#_Toc107950392)

[2.13.1 基本描述 33](#_Toc107950393)

[2.13.2 模块接口 33](#_Toc107950394)

[2.13.3 功能定义 33](#_Toc107950395)

[2.14 muxlw模块 34](#_Toc107950396)

[2.14.1 基本描述 34](#_Toc107950397)

[2.14.2 模块接口 34](#_Toc107950398)

[2.14.3 功能定义 34](#_Toc107950399)

[2.15 bridge模块 34](#_Toc107950400)

[2.15.1 基本描述 34](#_Toc107950401)

[2.15.2 模块接口 34](#_Toc107950402)

[2.15.3 功能定义 35](#_Toc107950403)

[2.16 cp0模块 35](#_Toc107950404)

[2.16.1 基本描述 35](#_Toc107950405)

[2.16.2 模块接口 35](#_Toc107950406)

[2.16.3 功能定义 36](#_Toc107950407)

[2.17 timer模块 36](#_Toc107950408)

[2.17.1 基本描述 36](#_Toc107950409)

[2.17.2 模块接口 36](#_Toc107950410)

[2.17.3 功能定义 37](#_Toc107950411)

[2.18 outputDev模块 37](#_Toc107950412)

[2.18.1 基本描述 37](#_Toc107950413)

[2.18.2 模块接口 37](#_Toc107950414)

[2.18.3 功能定义 37](#_Toc107950415)

[2.19 输入模块 38](#_Toc107950416)

[**三、** **指令描述** 38](#_Toc107950417)

[**四、** **测试程序** 38](#_Toc107950418)

[4.1 MIPS-Lite3指令集的测试主程序 38](#_Toc107950419)

[4.2 MIPS-Lite3指令集的测试中断子程序 39](#_Toc107950420)

[**五、** **测试结果** 39](#_Toc107950421)

[5.1 波形 39](#_Toc107950422)

[5.2 寄存器组结果 39](#_Toc107950423)

[5.2.1 gpr寄存器 39](#_Toc107950424)

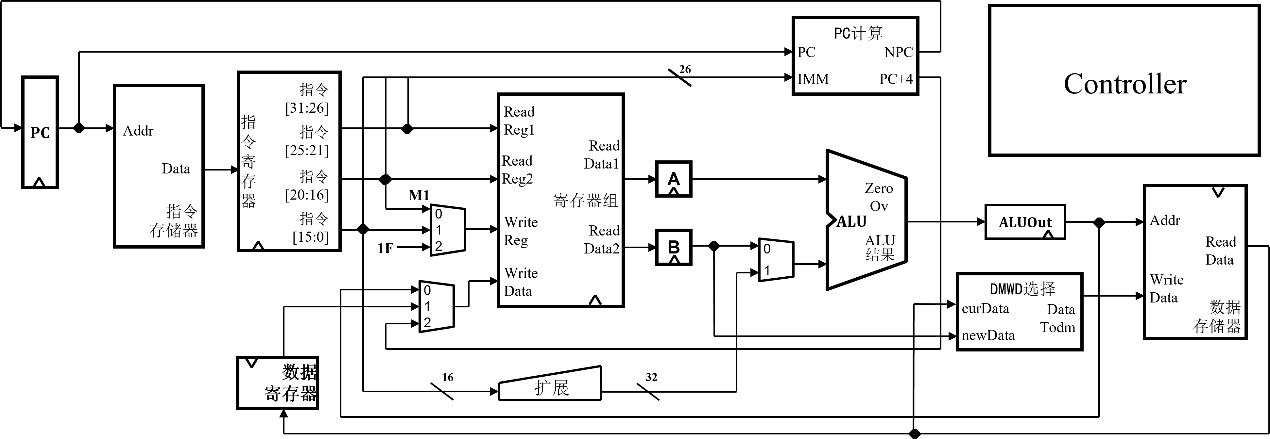
[5.2.2 cp0寄存器 40](#_Toc107950425)

[**六、** **总结与心得** 40](#_Toc107950426)

# Project1 VerilogHDL 完成多周期处理器开发

**一、总体数据通路结构设计图**

下图为本次多周期处理器开发的数据通路设计图。本次的数据通路设计，是在ppt中给出的数据通路上进行了改进:为了适配sb指令，在dm模块之前新增DMWD选择模块。



**二、模块定义**

## 2.1 ALU模块

### 2.1.1 基本描述

ALU模块的功能有两个。一个是选择第二个操作数的来源，另一个是完成数据的运算并输出计算结果以及一些其他模块需要的标志位。

### 2.1.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [31:0]dataOut\_1 | I | GPR输出数据1的值，来自aReg模块 |
| [31:0]dataOut\_2 | I | GPR输出数据2的值，来自bReg模块 |
| [31:0]ext32 | I | 16位立即数扩展结果 |
| [1:0]ALUOp | I | 运算方式选择信号 00：+ 01：- 10：| 11：slt |
| ALUSrc | I | 参与运算的第二个操作数的选择信号  0：dataOut\_2 1:ext32 |
| write\_30 | I | 是否为addi标志 0：不是 1：是 |
| zero | O | 运算结果是否为0标志 0：不是 1：是 |
| overflow | O | addi运算是否产生溢出标志 0：不是 1：是 |
| [31:0]alu\_res | O | 运算结果 |
| nCondition | O | 第一个操作数是否小于0标志（bltzal指令）  0：否 1：是 |

### 2.1.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 数据选择 | 选择正确的数据作为第二个操作数的值 |
| 2 | 运算并输出 | 根据指定运算类型完成运算，并输出结果和一些标志位 |

## 2.2 NPC模块

### 2.2.1 基本描述

NPC模块主要完成的功能是下一条pc地址的生成，并根据选择信号选择正确的地址输出给pc模块。

### 2.2.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| rst | I | 复位信号 0：不复位 1：复位 |
| zero | I | 两操作数相等信号，用于选择下一条地址 |
| nCondition | I | 指定寄存器操作数小于0信号，用于选择下一条寄存器 |
| [1:0]npc\_sel | I | 选择输出的pc  00：pc\_ori+4  01：按照beq跳转方式计算地址  10：按照j跳转方式计算地址  11:选择jr地址 |
| [31:0]pcin | I | gpr中regFile[rs]值 |
| [31:0]pc\_ori | I | pc |
| [31:0]imm32 | I | 当前指令的32位值，只用到低26位 |
| [31:0]pcout | O | npc输出 |

### 2.2.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 复位 | 当复位信号有效时，pcout 🡨 32’h0000\_3000 |
| 2 | 计算指令地址，选择并输出pc | npc\_sel:  00: pcout🡨pc\_ori+4  01:如果zero或nCondition有效，则pcout按照beq跳转方式更新  10:按照j跳转方式更新  11:选择jr跳转地址并更新 |

## 2.3 PC模块

### 2.3.1 基本描述

PC模块的主要功能是，将NPC模块的输出（下一条指令地址），在时钟上升沿且写使能有效时，将输出的值更改为输入的值。

### 2.3.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| pcwr | I | 写使能信号 0：不写入 1：写入 |
| [31:0]pcin | I | 输入npc模块的输出，下一条指令的地址 |
| [31:0]pcout | O | 下一条指令的地址，PC模块寄存器存储的值 |

### 2.3.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 存下一条指令地址 | 在时钟沿上升沿且写使能有效时，将当前输出寄存器的值更改为当前输入值 |
| 2 | 输出下一条指令地址 | 输出当前寄存器的值 |

## 2.4 IM模块

### 2.4.1 基本描述

指令存储器，主要功能有两个。一个存储指令，另一个是根据输入地址的值，输出对应地址的指令。

### 2.4.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [9:0]addr | I | 输入地址（指令地址） |
| [31:0]dout | O | 地址对应的指令 |

### 2.4.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 存储指令 | 读取txt文件，存储指令 |
| 2 | 根据地址输出指令 | 根据模块的地址输入，输出对应的指令 |

## 2.5 IR模块

### 2.5.1 基本描述

IR模块的主要功能是切分数据通路。这个模块本质上是一个寄存器，用于存储IM输出的指令。IR模块的写入代表取指阶段的完成。

### 2.5.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| irwr | I | 写使能信号 0：不写入 1：写入 |
| [31:0]imin | I | 输入为IR模块的输出，是一条指令 |
| [31:0]irout | O | 输出寄存器中的指令 |

### 2.5.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 切分数据通路 | 在时钟沿上升沿且写使能有效时，将当前输出寄存器的值更改为当前输入值 |

## 2.6 GPR模块

### 2.6.1 基本描述

GPR模块的主要功能分三点。第一是数据选择器功能，选择正确的写入数据和正确的写入地址。第二个是数据的写入功能。第三个是数据的输出功能。

### 2.6.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| rst | I | 复位信号 0：不复位 1：复位 |
| gprwr | I | gpr写使能信号 0：不允许写入 1：允许写入 |
| [1:0]MemToReg | I | 选择信号，选择写入数据  00：alu运算结果  01：lw/lb数据  10：下一条指令地址 |
| [1:0]RegDst | I | 选择信号，选择写入地址 00:rt 01:rd 10:no.31 |
| [4:0]rs | I | 读出地址1 |
| [4:0]rt | I | 读出地址2 |
| [4:0]rd | I | 写入地址的一种选择 |
| write\_30 | I | addi指令标志 0：不是 1：是 |
| [31:0]pc\_p4 | I | 写入数据的一种选择，下一条指令地址 |
| [31:0]aluReg\_out | I | 写入数据的一种选择，alu的运算结果 |
| [31:0]dmReg\_out | I | 写入数据的一种选择，lw/lb数据 |
| overflow | I | 溢出写入数据 0：不溢出写入0 1：溢出写入1 |
| [31:0]dataOut\_1 | O | rs寄存器的值，数据输出 |
| [31:0]dataOut\_2 | O | rt寄存器的值，数据输出 |

### 2.6.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 数据选择 | 根据选择信号选择正确的写入数据和写入地址 |
| 2 | 寄存器存储数据 | 将数据写入对应寄存器，存储 |
| 3 | 输出数据 | 根据读出地址，读出对应的数据 |

## 2.7 EXT模块

### 2.7.1 基本描述

EXT模块的主要功能是，将16位立即数进行扩展。扩展的方式有三种，第一种是0扩展，第二种是符号扩展，第三种是将16位立即数加载到32位的高16位，低位补0。

### 2.7.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [15:0]imm16 | I | 16位立即数输入 |
| [1:0]ExtOp | I | 扩展方式选择信号  00：0扩展 01：符号扩展 10：lui指令方式扩展 |
| [31:0]ext32 | O | 扩展结果输出 |

### 2.7.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 扩展立即数 | 根据扩展方式的选择信号，扩展16位立即数至32位。 |

## 2.8 aReg & bReg模块

### 2.8.1 基本描述

aReg & bReg模块实现的是同样的功能，即切分数据通路。他们存储的是GPR模块的两个输出数据。当两个输出数据写入两个模块的寄存器时，代表译码阶段结束。

### 2.8.2 模块接口(以aReg模块为例)

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| [31:0]alu\_dataOut\_1 | I | GPR模块读出数据1 |
| [31:0]aReg\_out | O | 输出寄存器值 |

### 2.8.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 切分数据通路 | 在时钟沿上升沿时，将当前输出寄存器的值更改为当前输入值。 |

## 2.9 aluReg模块

### 2.9.1 基本描述

aluReg模块主要功能是切分数据通路。当时钟信号上升沿时，将输入数据写入模块中的寄存器。

### 2.9.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| [31:0]alu\_res | I | ALU模块运算结果 |
| overflow | I | ALU模块溢出标志 |
| [31:0]aluReg\_out | O | 当前模块输出的ALU模块运算结果 |
| overflowReg | O | 当前模块输出的ALU模块溢出标志 |

### 2.9.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 切分数据通路 | 在时钟沿上升沿时，将当前输出寄存器的值更改为当前输入值。 |

## 2.10 DM模块

### 2.10.1 基本描述

DM模块的功能有两个。一是在写使能有效且时钟上升沿时，将输入数据写入到输入地址的存储空间中。二是在读出输入地址在存储空间中的数据。

### 2.10.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [9:0]addr | I | 写入地址 |
| [31:0]din | I | 写入数据 |
| we | I | 写使能 0：不写入 1：允许写入 |
| clk | I | 时钟信号 |
| [31:0]dout | O | 读出数据 |

### 2.10.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 写入数据 | 在时钟沿上升沿且写使能有效时，写入数据 |
| 2 | 读出数据 | 读出指定地址的数据 |

## 2.11 DR模块

### 2.11.1 基本描述

DR模块的主要功能是切分数据通路，还有一个功能是选择输出数据（针对lb指令）。当时钟上升沿时，将输入数据写入到模块寄存器。

### 2.11.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| islb | I | lb指令标志 0：不是 1：是 |
| [31:0]dmout | I | DM模块读出的数据 |
| [31:0]drout | O | 当前模块的输出值 |

### 2.11.3 功能定义

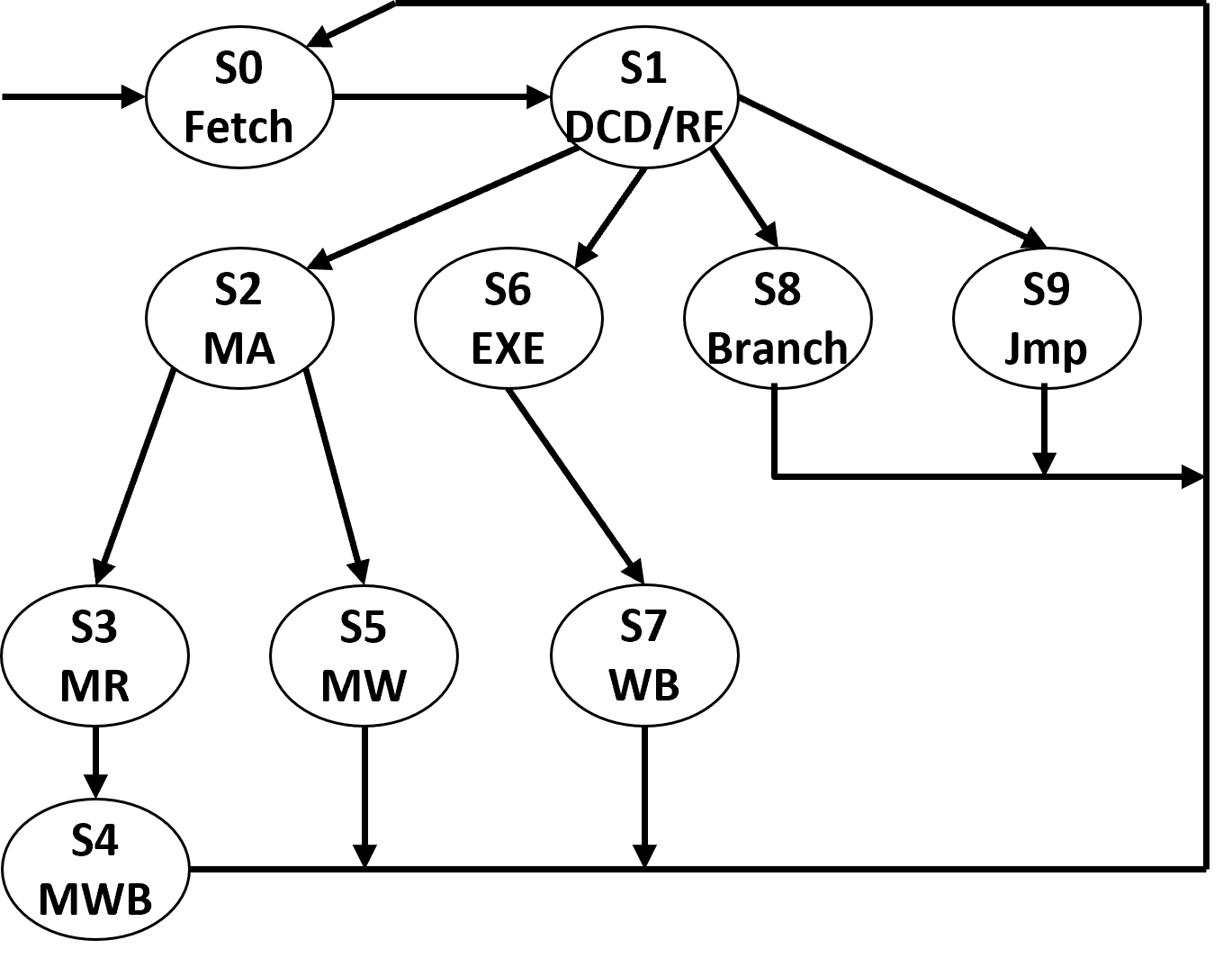
|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 切分数据通路 | 在时钟沿上升沿时，将当前输出寄存器的值更改为当前输入值。 |
| 2 | 选择输出数据 | 根据islb信号，选择或生成正确的数据并输出。 |

## 2.12 Controller模块

### 2.12.1 基本描述

Controller模块是根据输入的opcode和funct值（还有zero信号），生成对应的指令。同时构建状态机，在对应的状态时，产生对应的控制信号。

状态机共有10个状态，为S0至S9，状态机如下图所示：



S0和S1是所有指令都要经历的取指和译码阶段。lw/sw/lb/sb走S2，beq/jr/bltzal走S8，j/jal走S9，其余指令走S6。

lw/lb指令由于访存后有回写阶段，故走S3读dm，然后S4回写。sw/sb指令则走S5访存（写入）阶段。

S6阶段是执行阶段，进行alu的运算，为R型和I型指令。然后走到S7，进行回写阶段。

j/jal指令是直接跳转，故分到同一个状态S9。需要注意的是，S9阶段，如果是jal指令，是需要执行回写的，此时的形成跳转地址与回写是在同一个状态下完成的。

bltzal指令与beq指令类似，先考察条件。满足判断条件形成跳转地址，故被分到同一个状态S8。

### 2.12.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| rst | I | 复位信号 |
| [5:0]opcode | I | 当前指令opcode |
| [5:0]funct | I | 当前指令funct |
| zero | I | ALU模块产生的zero标志 |
| nCondition | I | ALU模块产生的nCondition标志（bltzal指令） |
| [1:0]RegDst | O | GPR模块写入地址选择 00:rt 01:rd 10:no.31 |
| [1:0]MemToReg | O | GPR模块写入数据选择  00：alu运算结果  01：lw/lb数据  10：下一条指令地址 |
| [1:0]npc\_sel | O | NPC模块输出选择  00：pc\_ori+4  01：按照beq跳转方式计算地址  10：按照j跳转方式计算地址  11:选择jr地址 |
| [1:0]ALUOp | O | ALU模块运算功能选择  00：+ 01：- 10：| 11：slt |
| [1:0]EXTOp | O | EXT模块扩展类型选择  00：0扩展  01：符号扩展  10：lui指令方式扩展 |
| RegWrite | O | GPR模块写使能 |
| ALUSrc | O | ALU模块操作数2的选择  0：GPR模块读出数据2 1：16位立即数扩展结果 |
| MemWrite | O | DM模块写使能 |
| write\_30 | O | addi指令标志 |
| pcwr | O | PC模块写使能 |
| irwr | O | IR模块写使能 |
| islb | O | LB指令标志 |
| issb | O | SB指令标志 |

### 2.12.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 构建状态机，确定当前状态 | 按设计好的状态机构建状态，并根据不同的指令确定下一个状态。 |
| 2 | 产生控制信号 | 在不同的状态下，产生对应的控制信号和标志。 |

## 2.13 Muxdmin模块

### 2.13.1 基本描述

Muxdmin模块设计的初衷是为了支持sb指令。主要功能是根据issb信号，选择或生成正确的写入数据，传送给DM模块的写入数据端口。

这里的逻辑是，由于DM模块在时钟上升沿来临时才进行写入，所以可以在来临时将需要写入的地址的值取出。如果是sb指令，则将需要写入的低8位拼上原来的24位输出即可。如果不是，则直接将取出的值输出。这一部分是组合逻辑电路，在时钟沿来临时，DM模块将数据写入即可。

### 2.13.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [31:0]bReg\_out | I | bReg寄存器输出 |
| [31:0]dmout\_data | I | DM模块读出数据 |
| issb | I | sb指令标志 |
| [31:0]dm\_datain | O | 当前模块数据输出，输出给DM模块输入 |

### 2.13.3 功能定义

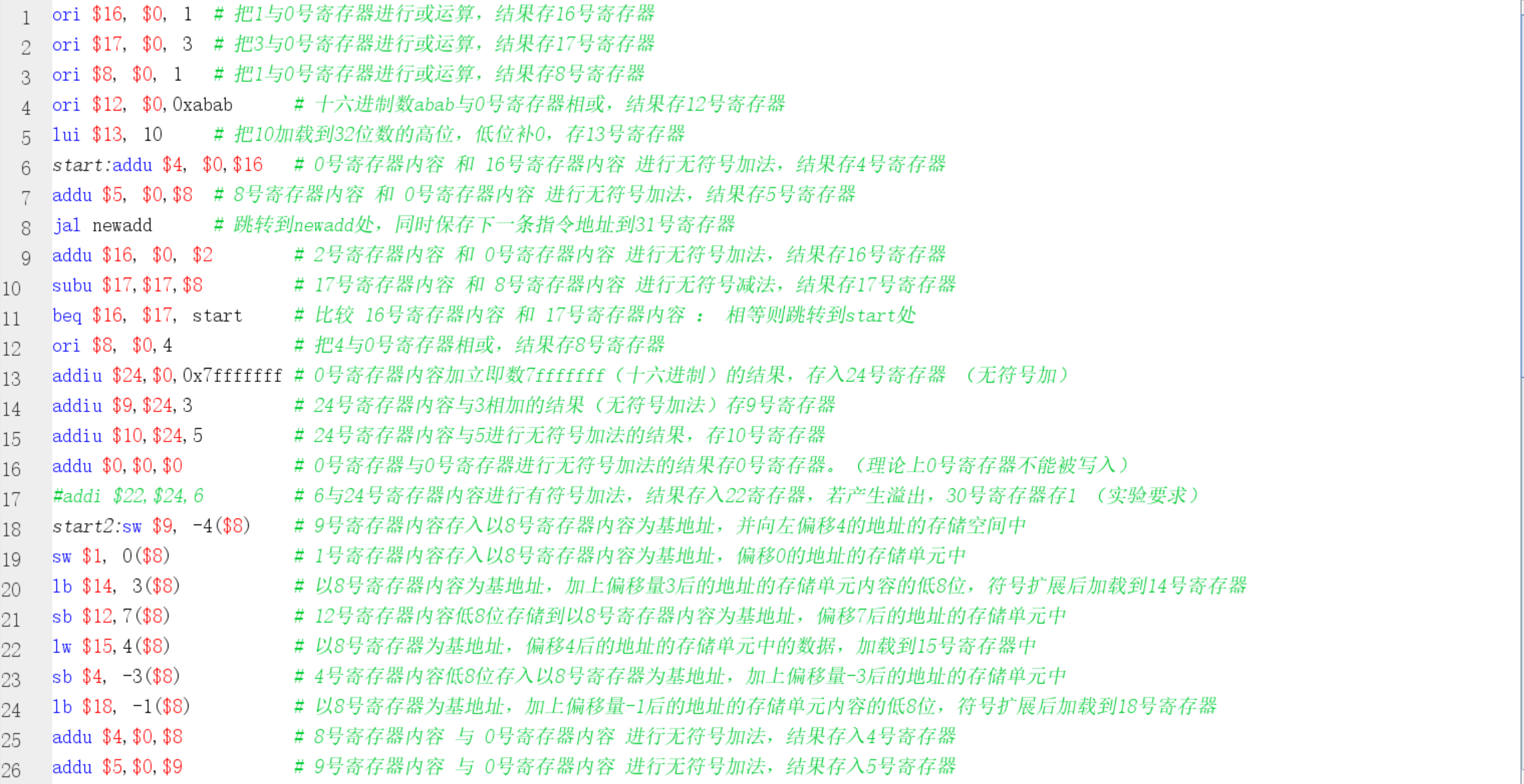
|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 选择或生成DM模块写入数据 | 根据issb控制信号，生成或选择DM模块的写入数据。 |

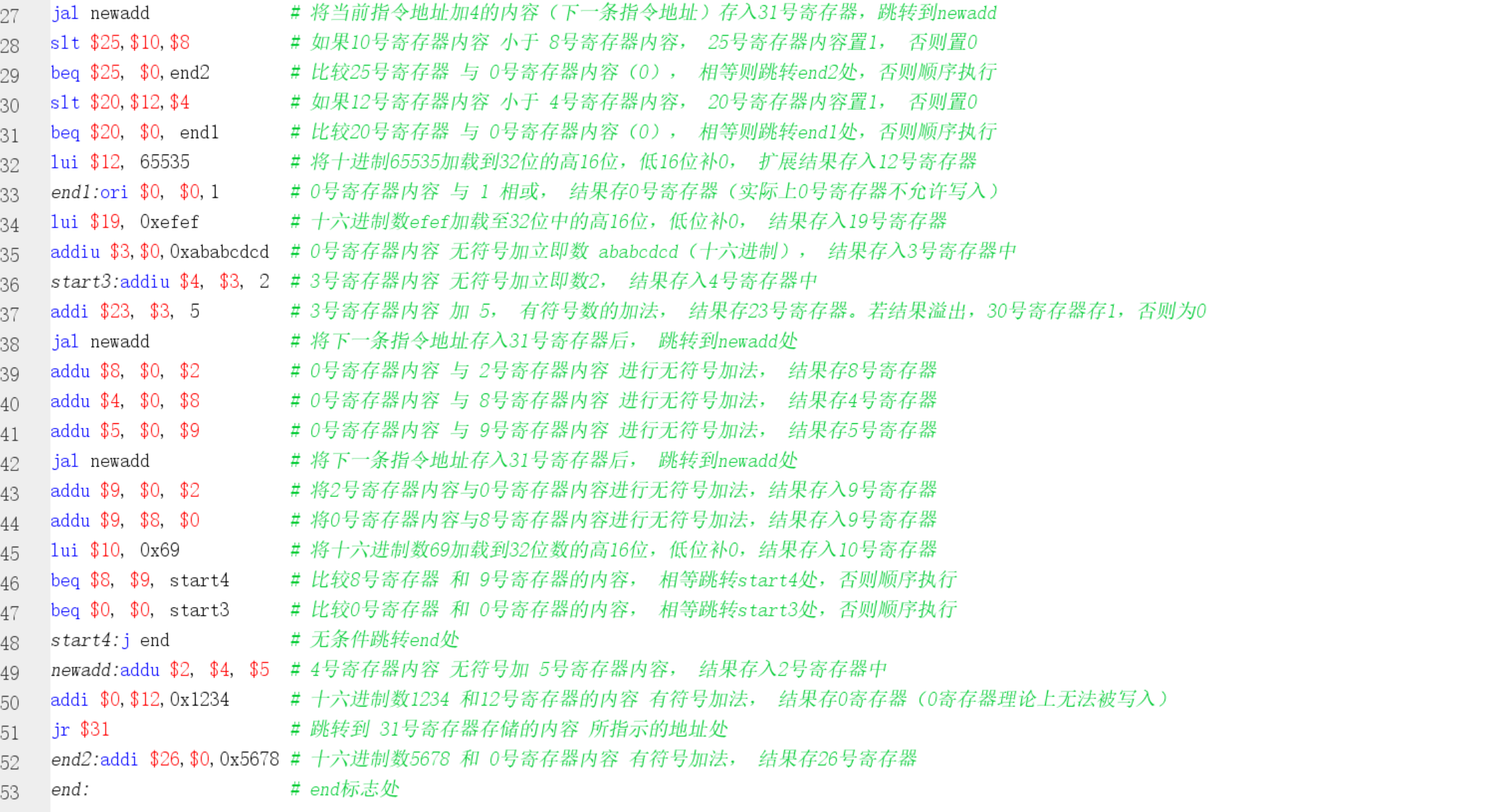
# 三、指令描述



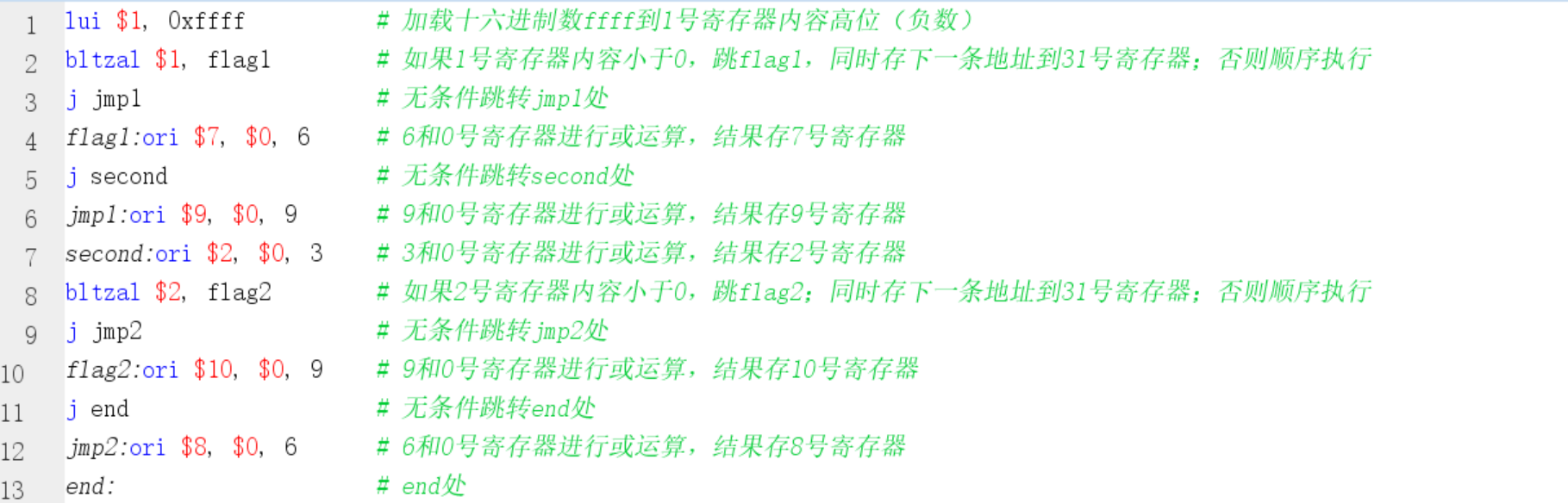
# 四、测试程序

## 4.1 MIPS-Lite2指令集的测试程序





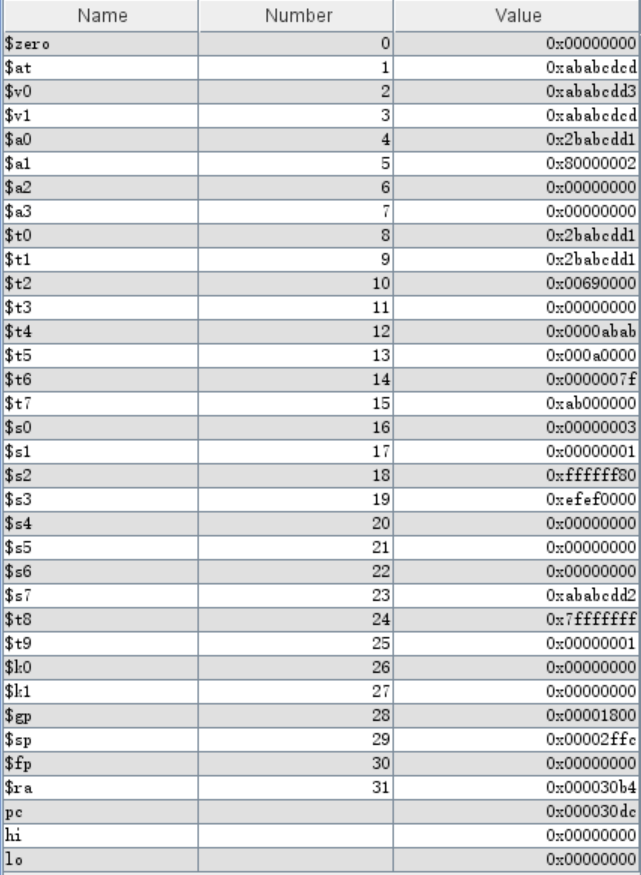
## 4.2 新增指令BLTZAL的测试程序



# 五、测试结果

## 5.1 MIPS-Lite2指令集测试结果

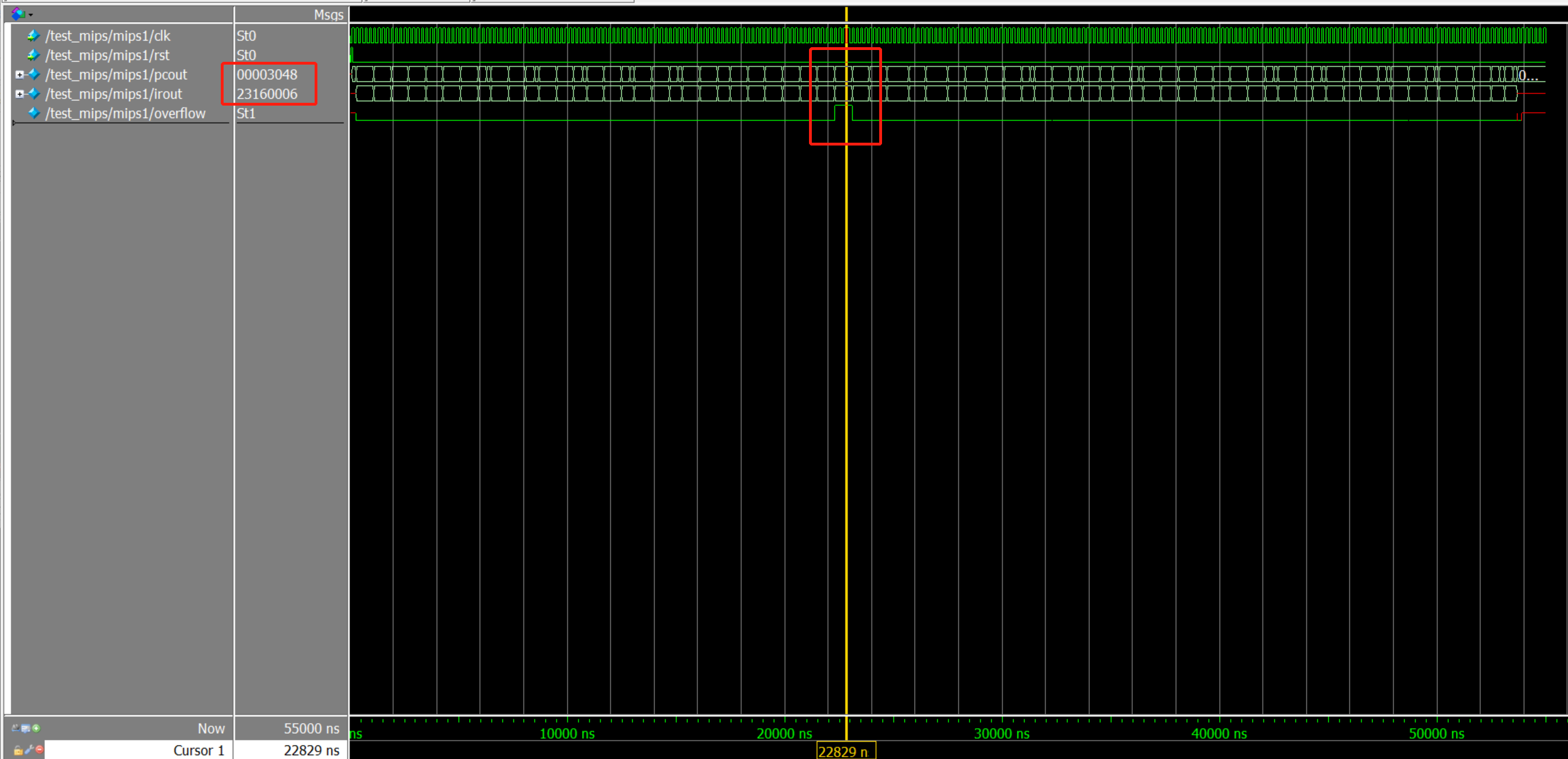
### 5.1.1 MARS中结果





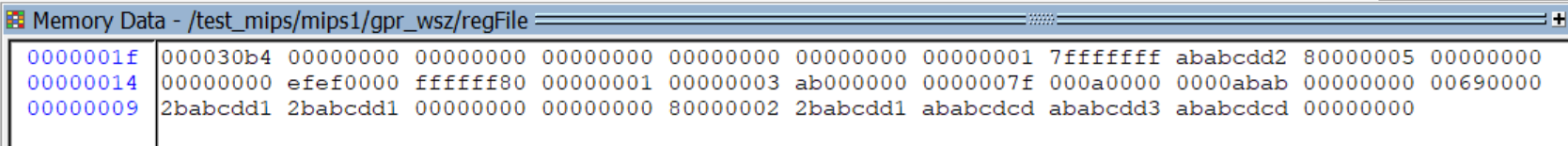
### 5.1.2 Modelsim中结果

#### （1）溢出波形

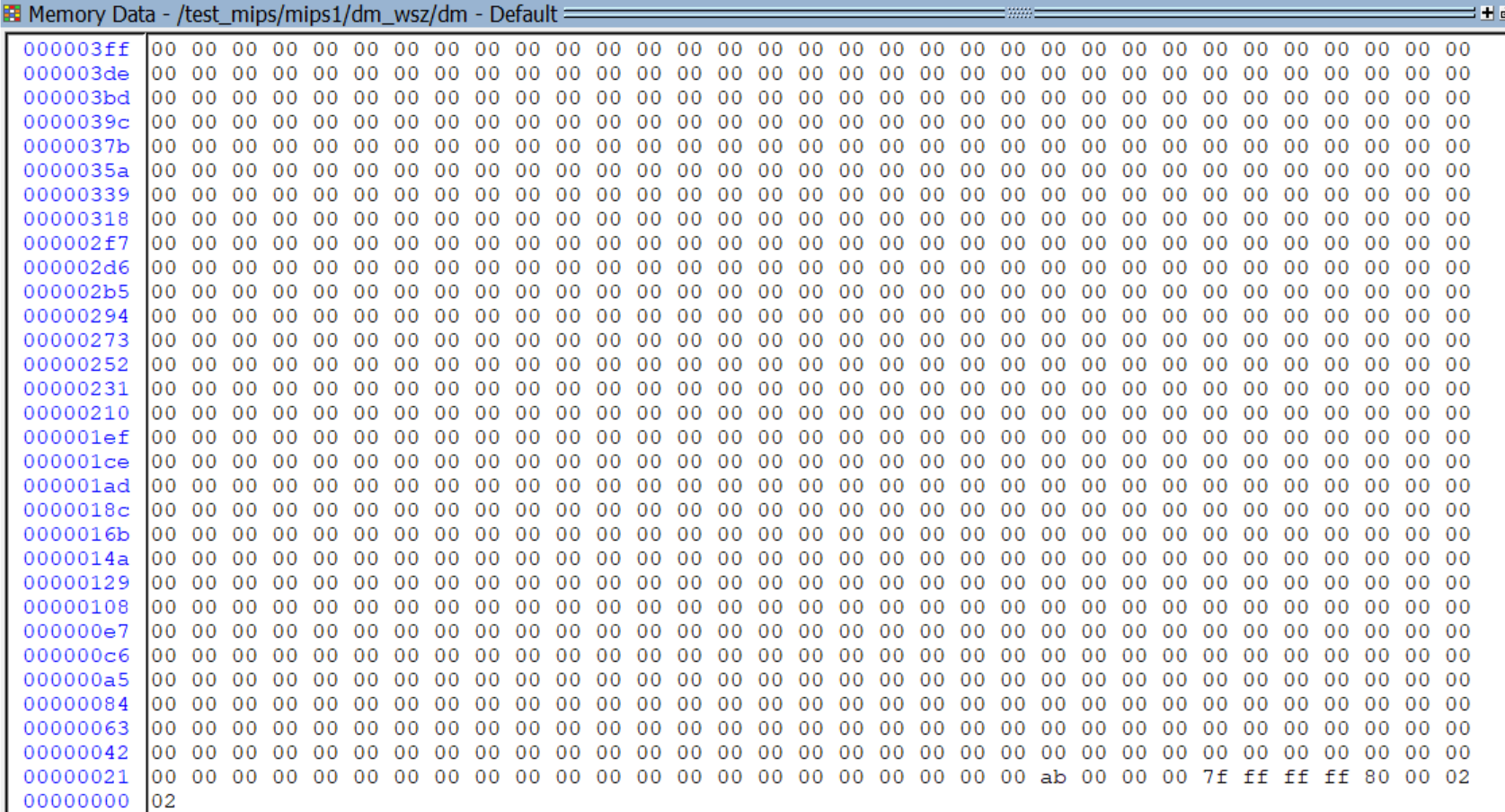


#### （2）寄存器 & 数据存储器内容

##### a. 寄存器

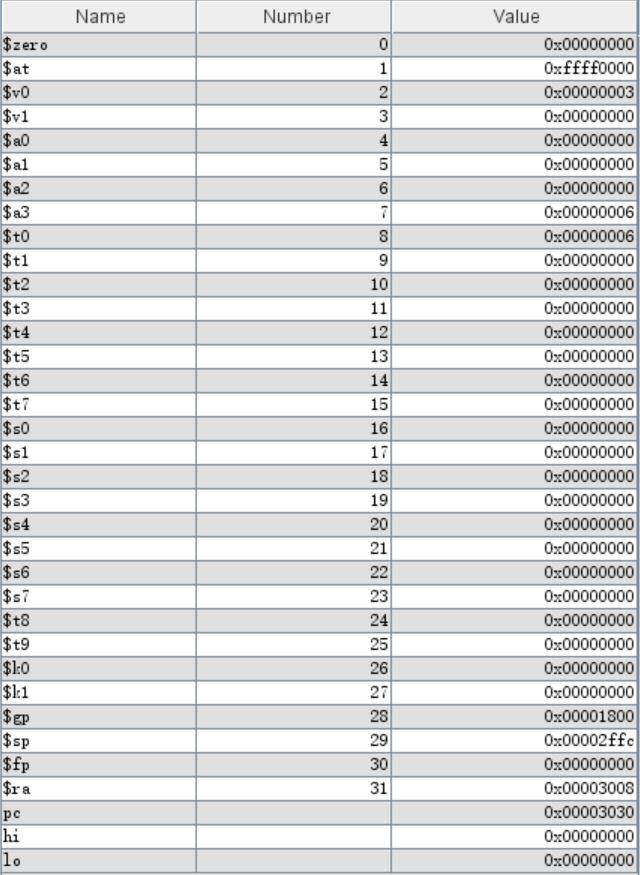


##### b. 数据存储器

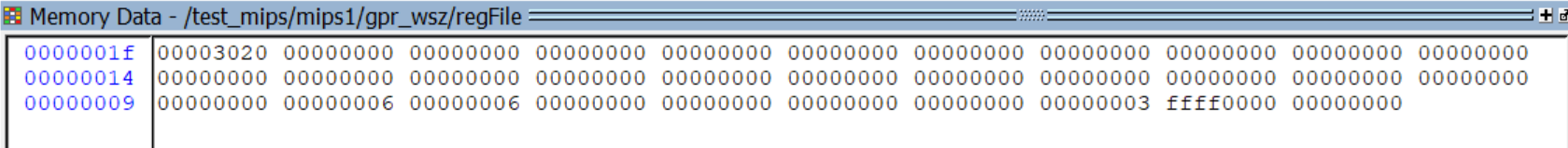


## 5.2 BLTZAL指令测试结果

### 5.2.1 MARS中结果



### 5.2.2 Modelsim中结果



# 六、总结与心得体会

本次课设完成的是多周期处理器的开发，是在单周期处理器上的优化与改进，也是MIPS微系统的cpu内部部分，意义重大。

多周期处理器，相较于单周期来说，主要区别在于，将一条指令分为多个阶段来进行处理。原因是，每一条指令的指令周期不同，有的指令需要的时间短，有的指令需要的时间长，通过分为多个阶段进行处理，cpu的主频将不再受限于最慢的那条指令的指令周期时间，可以大幅提高主频，加快cpu的处理速度。

多周期处理器的不同阶段，是通过构建状态机控制的。在不同的状态产生不同的控制信号，以达到合理控制的效果。

通过完成这次课设，我对于将指令分成多个阶段进行处理这一思想有了更为深刻的认识，也正是因为这一次课设，我把每一条指令需要的指令阶段以及控制信号的何时产生，都理的更加顺畅了，在自己脑子里有了更加清晰的构建。课本上的知识得到了运用，做到了理论与实践相结合。

# Project2 VerilogHDL完成MIPS微系统开发(支持设备与中断)

1. **总体数据通路结构设计图**

MIPS微系统的数据通路主要分为两个部分，一个是多周期处理器的数据通路，另一个是CPU外部的bridge以及和外设的连接。

多周期处理器的数据通路，在P1的基础上增加了CP0协处理器及其相应数据通路，同时增加了一个lwsel模块，用于选择在MemToWrite信号为01时，写入dm的数据到寄存器，还是写入从外设读的数据到寄存器。

CPU外部设备主要有三个：输入设备、输出设备以及定时器。

系统桥将外设与CPU内部连接起来，起到联系的作用。

通路中画出了数据通路以及中断信号的传输路径。



1. **模块定义**

## 2.1 ALU模块

### 2.1.1 基本描述

ALU模块的功能有两个。一个是选择第二个操作数的来源，另一个是完成数据的运算并输出计算结果以及一些其他模块需要的标志位。模块设计与p1相同。

### 2.1.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [31:0]dataOut\_1 | I | GPR输出数据1的值，来自aReg模块 |
| [31:0]dataOut\_2 | I | GPR输出数据2的值，来自bReg模块 |
| [31:0]ext32 | I | 16位立即数扩展结果 |
| [1:0]ALUOp | I | 运算方式选择信号 00：+ 01：- 10：| 11：slt |
| ALUSrc | I | 参与运算的第二个操作数的选择信号  0：dataOut\_2 1:ext32 |
| write\_30 | I | 是否为addi标志 0：不是 1：是 |
| zero | O | 运算结果是否为0标志 0：不是 1：是 |
| overflow | O | addi运算是否产生溢出标志 0：不是 1：是 |
| [31:0]alu\_res | O | 运算结果 |
| nCondition | O | 第一个操作数是否小于0标志（bltzal指令）  0：否 1：是 |

### 2.1.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 数据选择 | 选择正确的数据作为第二个操作数的值 |
| 2 | 运算并输出 | 根据指定运算类型完成运算，并输出结果和一些标志位 |

## 2.2 NPC模块

### 2.2.1 基本描述

NPC模块主要完成的功能是下一条pc地址的生成，并根据选择信号选择正确的地址输出给pc模块。

与p1的区别在于，增加了ERET, EPC, intreq等信号。ERET和intreq是中断返回信号和中断跳转信号，EPC是下一条指令地址输入。

### 2.2.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| rst | I | 复位信号 0：不复位 1：复位 |
| zero | I | 两操作数相等信号，用于选择下一条地址 |
| nCondition | I | 指定寄存器操作数小于0信号，用于选择下一条寄存器 |
| [1:0]npc\_sel | I | 选择输出的pc  00：pc\_ori+4  01：按照beq跳转方式计算地址  10：按照j跳转方式计算地址  11:选择jr地址 |
| [31:0]pcin | I | gpr中regFile[rs]值 |
| [31:0]pc\_ori | I | pc |
| [31:0]imm32 | I | 当前指令的32位值，只用到低26位 |
| ERET | I | ERET中断返回指令标志 |
| [31:0]EPC | I | 下一条指令地址 |
| intpc | I | 中断信号 |
| [31:0]pcout | O | npc输出 |

### 2.2.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 复位 | 当复位信号有效时，pc\_new 🡨 32’h0000\_3000 |
| 2 | 计算指令地址，更新pcnew | 如果intpc信号有效，则pc\_new 🡨 32’h0000\_4180  否则根据npc\_sel信号选择:  00: pcnew🡨pc\_ori+4  01:如果zero或nCondition有效，则pcnew按照beq跳转方式更新  10:按照j跳转方式更新  11:选择jr跳转地址并更新 |
| 3 | 选择输出数据 | 根据ERET指令标志选择pcout为pcnew还是epc的值 |

## 2.3 PC模块

### 2.3.1 基本描述

PC模块的主要功能是，将NPC模块的输出（下一条指令地址），在时钟上升沿且写使能有效时，将输出的值更改为输入的值。模块设计与p1相同。

### 2.3.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| pcwr | I | 写使能信号 0：不写入 1：写入 |
| [31:0]pcin | I | 输入npc模块的输出，下一条指令的地址 |
| [31:0]pcout | O | 下一条指令的地址，PC模块寄存器存储的值 |

### 2.3.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 存下一条指令地址 | 在时钟沿上升沿且写使能有效时，将当前输出寄存器的值更改为当前输入值 |
| 2 | 输出下一条指令地址 | 输出当前寄存器的值 |

## 2.4 IM模块

### 2.4.1 基本描述

指令存储器，主要功能有两个。一个存储指令，另一个是根据输入地址的值，输出对应地址的指令。

注意p2中IM寄存器大小变为了8kb，则输入地址需要13位。

### 2.4.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [12:0]addr | I | 输入地址（指令地址） |
| [31:0]dout | O | 地址对应的指令 |

### 2.4.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 存储指令 | 读取txt文件，存储指令 |
| 2 | 根据地址输出指令 | 根据模块的地址输入，输出对应的指令 |

## 2.5 IR模块

### 2.5.1 基本描述

IR模块的主要功能是切分数据通路。这个模块本质上是一个寄存器，用于存储IM输出的指令。IR模块的写入代表取指阶段的完成。IR的写入在时钟上升沿且写使能有效时完成。

模块设计与p1相同。

### 2.5.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| irwr | I | 写使能信号 0：不写入 1：写入 |
| [31:0]imin | I | 输入为IR模块的输出，是一条指令 |
| [31:0]irout | O | 输出寄存器中的指令 |

### 2.5.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 切分数据通路 | 在时钟沿上升沿且写使能有效时，将当前输出寄存器的值更改为当前输入值 |

## 2.6 GPR模块

### 2.6.1 基本描述

GPR模块的主要功能分三点。第一是数据选择器功能，选择正确的写入数据和正确的写入地址。第二个是数据的写入功能。第三个是数据的输出功能。

这里与p1的不同在于，写入数据的数据选择端加了一路输入，为cp0in，即将cp0内指定寄存器写入到gpr的寄存器中。

### 2.6.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| rst | I | 复位信号 0：不复位 1：复位 |
| gprwr | I | gpr写使能信号 0：不允许写入 1：允许写入 |
| [1:0]MemToReg | I | 选择信号，选择写入数据  00：alu运算结果  01：lw/lb数据  10：下一条指令地址  11：cp0中的指定寄存器信息 |
| [1:0]RegDst | I | 选择信号，选择写入地址 00:rt 01:rd 10:no.31 |
| [4:0]rs | I | 读出地址1 |
| [4:0]rt | I | 读出地址2 |
| [4:0]rd | I | 写入地址的一种选择 |
| write\_30 | I | addi指令标志 0：不是 1：是 |
| [31:0]pc\_p4 | I | 写入数据的一种选择，下一条指令地址 |
| [31:0]aluReg\_out | I | 写入数据的一种选择，alu的运算结果 |
| [31:0]dmReg\_out | I | 写入数据的一种选择，lw/lb数据 |
| overflow | I | 溢出写入数据 0：不溢出写入0 1：溢出写入1 |
| cp0in | I | 读取的cp0中寄存器的信息 |
| [31:0]dataOut\_1 | O | rs寄存器的值，数据输出 |
| [31:0]dataOut\_2 | O | rt寄存器的值，数据输出 |

### 2.6.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 数据选择 | 根据选择信号选择正确的写入数据和写入地址 |
| 2 | 寄存器存储数据 | 将数据写入对应寄存器，存储 |
| 3 | 输出数据 | 根据读出地址，读出对应的数据 |

## 2.7 EXT模块

### 2.7.1 基本描述

EXT模块的主要功能是，将16位立即数进行扩展。扩展的方式有三种，第一种是0扩展，第二种是符号扩展，第三种是将16位立即数加载到32位的高16位，低位补0。

模块设计与p1相同。

### 2.7.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [15:0]imm16 | I | 16位立即数输入 |
| [1:0]ExtOp | I | 扩展方式选择信号  00：0扩展 01：符号扩展 10：lui指令方式扩展 |
| [31:0]ext32 | O | 扩展结果输出 |

### 2.7.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 扩展立即数 | 根据扩展方式的选择信号，扩展16位立即数至32位。 |

## 2.8 aReg & bReg模块

### 2.8.1 基本描述

aReg & bReg模块实现的是同样的功能，即切分数据通路。他们存储的是GPR模块的两个输出数据。当两个输出数据写入两个模块的寄存器时，代表译码阶段结束。

模块设计与p1相同。

### 2.8.2 模块接口(以aReg模块为例)

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| [31:0]alu\_dataOut\_1 | I | GPR模块读出数据1 |
| [31:0]aReg\_out | O | 输出寄存器值 |

### 2.8.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 切分数据通路 | 在时钟沿上升沿时，将当前输出寄存器的值更改为当前输入值。 |

## 2.9 aluReg模块

### 2.9.1 基本描述

aluReg模块主要功能是切分数据通路。当时钟信号上升沿时，将输入数据写入模块中的寄存器。

模块设计与p1相同。

### 2.9.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| [31:0]alu\_res | I | ALU模块运算结果 |
| overflow | I | ALU模块溢出标志 |
| [31:0]aluReg\_out | O | 当前模块输出的ALU模块运算结果 |
| overflowReg | O | 当前模块输出的ALU模块溢出标志 |

### 2.9.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 切分数据通路 | 在时钟沿上升沿时，将当前输出寄存器的值更改为当前输入值。 |

## 2.10 DM模块

### 2.10.1 基本描述

DM模块的功能有两个。一是在写使能有效且时钟上升沿时，将输入数据写入到输入地址的存储空间中。二是在读出输入地址在存储空间中的数据。

与p1的区别在于，存储容量变成了12KB，故写入地址addr需要14位。

### 2.10.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [13:0]addr | I | 写入地址 |
| [31:0]din | I | 写入数据 |
| we | I | 写使能 0：不写入 1：允许写入 |
| clk | I | 时钟信号 |
| [31:0]dout | O | 读出数据 |

### 2.10.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 写入数据 | 在时钟沿上升沿且写使能有效时，写入数据 |
| 2 | 读出数据 | 读出指定地址的数据 |

## 2.11 DR模块

### 2.11.1 基本描述

DR模块的主要功能是切分数据通路，还有一个功能是选择输出数据（针对lb指令）。当时钟上升沿时，将输入数据写入到模块寄存器。

模块设计与p1相同。

### 2.11.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| islb | I | lb指令标志 0：不是 1：是 |
| [31:0]dmout | I | DM模块读出的数据 |
| [31:0]drout | O | 当前模块的输出值 |

### 2.11.3 功能定义

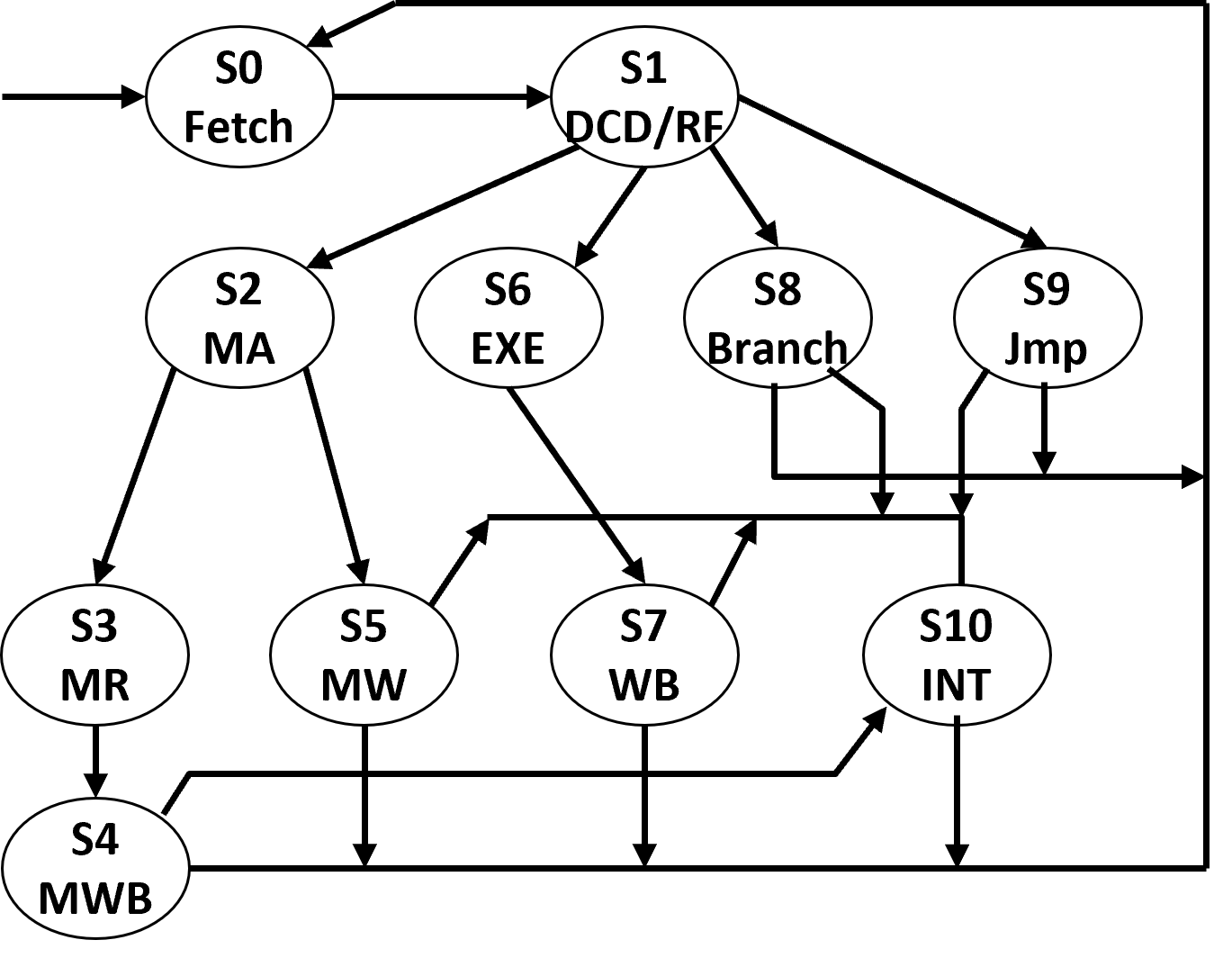
|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 切分数据通路 | 在时钟沿上升沿时，将当前输出寄存器的值更改为当前输入值。 |
| 2 | 选择输出数据 | 根据islb信号，选择或生成正确的数据并输出。 |

## 2.12 Controller模块

### 2.12.1 基本描述

Controller模块是根据输入的opcode、funct、rt值（用于MTC0&MFC0指令）、zero信号、intreq信号，生成对应的指令。同时构建状态机，在对应的状态时，产生对应的控制信号。

状态机共有11个状态，为S0至S10，状态机如下图所示：



S0和S1是所有指令都要经历的取指和译码阶段。lw/sw/lb/sb走S2，beq/jr/bltzal走S8，j/jal走S9，其余指令走S6。

lw/lb指令由于访存后有回写阶段，故走S3读dm，然后S4回写。sw/sb指令则走S5访存（写入）阶段。

S6阶段是执行阶段，进行alu的运算，为R型和I型指令。然后走到S7，进行回写阶段。

j/jal指令是直接跳转，故分到同一个状态S9。需要注意的是，S9阶段，如果是jal指令，是需要执行回写的，此时的形成跳转地址与回写是在同一个状态下完成的。

上述阶段均与p1相同，下面阐述p2新增的状态S10以及新增的指令MTC0，MFC0和ERET。

首先阐述状态机。这里新增加了一个S10状态，他是附于每一条指令的最后一个阶段之后的。如果在指令的最后一个阶段，并且intreq信号有效的话，就会进入S10状态，中断状态；否则就忽略掉S10状态，直接跳回S0即可。

对于MTC0和MFC0两条指令，他们的指令功能同sw/sb/lw/lb是相类似的，都是读/写功能，故将他们两个与上述四个指令分在同一个周期S2。到了S2阶段后，MFC0是读cp0寄存器并写入gpr寄存器中，与lw/lb指令类似，有回写阶段，故分到S3读阶段（MFC0是读cp0的寄存器）然后S4回写阶段。MTC0是写cp0寄存器，这与sw/sb指令功能类似，故分到S5阶段进行写入（这里写的是cp0的寄存器）。

对于ERET指令，其功能与j/jal类似，进行无条件跳转（形成地址），故分到S9阶段，在S9阶段完成跳转地址的更新。

### 2.12.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| rst | I | 复位信号 |
| [5:0]opcode | I | 当前指令opcode |
| [5:0]funct | I | 当前指令funct |
| zero | I | ALU模块产生的zero标志 |
| nCondition | I | ALU模块产生的nCondition标志（bltzal指令） |
| [4:0]M | I | 指令的rt部分的内容，用于译码MTC0和MFC0指令 |
| intreq | I | 中断请求信号 |
| [1:0]RegDst | O | GPR模块写入地址选择 00:rt 01:rd 10:no.31 |
| [1:0]MemToReg | O | GPR模块写入数据选择  00：alu运算结果  01：lw/lb数据  10：下一条指令地址 |
| [1:0]npc\_sel | O | NPC模块输出选择  00：pc\_ori+4  01：按照beq跳转方式计算地址  10：按照j跳转方式计算地址  11:选择jr地址 |
| [1:0]ALUOp | O | ALU模块运算功能选择  00：+ 01：- 10：| 11：slt |
| [1:0]EXTOp | O | EXT模块扩展类型选择  00：0扩展  01：符号扩展  10：lui指令方式扩展 |
| RegWrite | O | GPR模块写使能 |
| ALUSrc | O | ALU模块操作数2的选择  0：GPR模块读出数据2 1：16位立即数扩展结果 |
| MemWrite | O | DM模块写使能 |
| write\_30 | O | addi指令标志 |
| pcwr | O | PC模块写使能 |
| irwr | O | IR模块写使能 |
| islb | O | LB指令标志 |
| issb | O | SB指令标志 |
| cp0\_wen | O | cp0写使能信号 |
| bridge\_wen | O | 系统桥写使能信号，用于对外设的写使能，与选择信号配合使用 |
| exlset | O | 标志信号：存epc，exl中断防再入， |
| exlclr | O | 标志信号：pc🡨epc，exl恢复可中断 |
| intpc | O | 标志信号：pc跳转中断地址 |

### 2.12.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 构建状态机，确定当前状态 | 按设计好的状态机构建状态，并根据不同的指令确定下一个状态。 |
| 2 | 产生控制信号 | 在不同的状态下，产生对应的控制信号和标志。 |

## 2.13 Muxdmin模块

### 2.13.1 基本描述

Muxdmin模块设计的初衷是为了支持sb指令。主要功能是根据issb信号，选择或生成正确的写入数据，传送给DM模块的写入数据端口。

这里的逻辑是，由于DM模块在时钟上升沿来临时才进行写入，所以可以在来临时将需要写入的地址的值取出。如果是sb指令，则将需要写入的低8位拼上原来的24位输出即可。如果不是，则直接将取出的值输出。这一部分是组合逻辑电路，在时钟沿来临时，DM模块将数据写入即可。

模块设计与p1相同。

### 2.13.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [31:0]bReg\_out | I | bReg寄存器输出 |
| [31:0]dmout\_data | I | DM模块读出数据 |
| issb | I | sb指令标志 |
| [31:0]dm\_datain | O | 当前模块数据输出，输出给DM模块输入 |

### 2.13.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 选择或生成DM模块写入数据 | 根据issb控制信号，生成或选择DM模块的写入数据。 |

## 2.14 muxlw模块

### 2.14.1 基本描述

muxlw模块主要功能是在npc\_sel为01时，选择写入选择端的数据是dm的输出数据，还是从外设读的数据。这么做的原因是，gpr的写入数据的数据选择器只有两位位宽，而2’b11要给cp0读出信息使用，故在数据选择器前另加模块，选择npc\_sel为2’b01时的写入数据。

### 2.14.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [31:0]addr | I | bReg寄存器输出 |
| [31:0]drData | I | DM模块读出数据 |
| [31:0]bridgeData | I | sb指令标志 |
| [31:0]muxlwOut | O | 当前模块数据输出，输出给DM模块输入 |

### 2.14.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 选择数据 | 根据地址，判断并选择数据 |

## 2.15 bridge模块

### 2.15.1 基本描述

Bridge模块的基本功能是连接cpu与外设的通讯桥梁，传递彼此之间的地址、数据以及自身生成的使能信号。三大功能为，地址转换、读数据、写数据。同时还是定时器产生的IRQ信号的传递桥梁。

外设1输入设备，不需要写使能，只读。外设0是timer，外设2是输出设备。

### 2.15.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [31:0]praddr | I | 外设地址 |
| [31:0]prwd | I | 外设写数据 |
| [31:0]dev0\_rd | I | 外设0读数据 |
| [31:0]dev1\_rd | I | 外设1读数据 |
| [31:0]dev2\_rd | I | 外设2读数据 |
| wecpu | I | 外设写使能，需要搭配选择信号使用 |
| IRQ | I | 计时器产生的中断请求信号 |
| [31:0]prrd | O | 外设读的数据（选择后的） |
| [31:0]dev\_wd | O | 外设写数据 |
| [31:0]dev\_addr | O | 外设地址 |
| wedev0 | O | 外设0写使能 |
| wedev2 | O | 外设2写使能 |
| [5:0]HWInt | O | 6路中断信号 |

### 2.15.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 传递读出的外设数据 | 将从外设读的数据传给gpr进行写入 |
| 2 | 传递要写给外设的数据 | 将dm读出数据传给外设进行写入 |
| 3 | 外设选择，使能信号生成 | 根据地址和译码信号，产生使能信号 |
| 4 | 汇总中断信号并传递 | 计时器IRQ传给桥，桥汇总到6位中断信号，传给cpu内部 |

## 2.16 cp0模块

### 2.16.1 基本描述

cp0模块是协处理器，主要用来处理异常。本次cp0模块中包含4个寄存器：sr、cause、epc和prid。sr寄存器的功能是存储屏蔽位im，全局中断使能信号ie以及防止再中断的标志位exl；cause寄存器的内容是不断的锁存中断产生的原因；epc寄存器就是存储下一条指令地址的；prid寄存器是存储个性化签名等等。需要注意的是，cause寄存器只读。

### 2.16.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| rst | I | 复位信号 |
| wen | I | 寄存器写使能信号 |
| exlset | I | 中断开始信号，完成epc寄存器写入以及exl置位 |
| exlclr | I | ERET信号，开exl中断 |
| [4:0]sel | I | 读/写地址 |
| [5:0]HWInt | I | 6路中断信号 |
| [31:0]din | I | 写入数据 |
| [31:0]pc\_p4 | I | 下一条指令地址，用于写入npc寄存器 |
| [31:0]pcout | O | epc的输出 |
| [31:0]dout | O | 读出数据的输出 |
| intreq | O | 根据屏蔽位im以及标志位ie、exl产生的中断信号 |

### 2.16.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 写数据 | 根据数据输入和写入地址，写入数据（使能有效时） |
| 2 | 读数据 | 根据地址输入，读数据并输出 |
| 3 | 产生中断信号 | 根据6路hwint中断信号产生1位中断信号intreq |
| 4 | 记录异常原因 | cause寄存器不断锁存HWInt信号 |

## 2.17 timer模块

### 2.17.1 基本描述

Timer模块主要由三个寄存器组成：控制寄存器、初值寄存器以及计数器。主要的功能是，当计数器为0时，产生中断请求信号IRQ。

接下来阐述每个寄存器的功能。控制寄存器主要负责的是管理是否允许中断、计数模式选择以及是否允许计数。初值寄存器用于存放初值，保持不变。计数器用于在时钟沿来临时进行自减，完成计数。

当计数器值为0时，根据不同的模式执行不同的操作。当为模式0时，则一直保持0不变，直到初值寄存器再次被外部写入后，初值寄存器值再次被加载至计数器，计数器重新启动倒计数；当为模式1时，则自动加载初值寄存器的值到计数器，开始新一轮的倒计数。

### 2.17.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| CLK\_I | I | 时钟信号 |
| RST\_I | I | 复位信号 |
| [3:2]ADDR\_I | I | 读/写地址,三个寄存器，两位地址信号即可。且低两位永远为0，故不用考虑。 |
| WE\_I | I | 写使能信号 |
| [31:0]DAT\_I | I | 写入数据 |
| [31:0]DAT\_O | O | 读出数据 |
| IRQ | O | 中断请求信号，count为0时产生 |

### 2.17.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 计数 | 根据控制寄存器内容，在不同的模式下完成计数（允许计数时）。 |
| 2 | 产生中断请求信号 | 当计数器值为0时，产生中断请求信号IRQ。 |

## 2.18 outputDev模块

### 2.18.1 基本描述

outputDev为外设输出模块，包含两个寄存器：初值寄存器以及当前值寄存器。模块具有输入输出功能：输入时，在时钟上升沿且写使能有效时，将数据写入地址指示的寄存器中；输出时，根据地址，选择正确的寄存器的值进行输出。

两个寄存器的目的是，便于在后续秒计数时，判断当前输入与初值寄存器的值是否相同。如果相同，即输入没变化，则自加1。否则将两个寄存器更新为新的输入值，然后在下一次中断时再进行自加1。

### 2.18.2 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| en | I | 写使能信号 |
| [3:2]addr | I | 读/写地址，也可以理解为寄存器选择信号。 |
| [31:0]din | I | 写入数据 |
| [31:0]dout | O | 读出数据 |

### 2.18.3 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 读数据 | 根据addr读出指定寄存器数据。 |
| 2 | 写数据 | 根据addr，在写使能有效且时钟上升沿时，写入数据到寄存器中。 |

## 2.19 输入模块

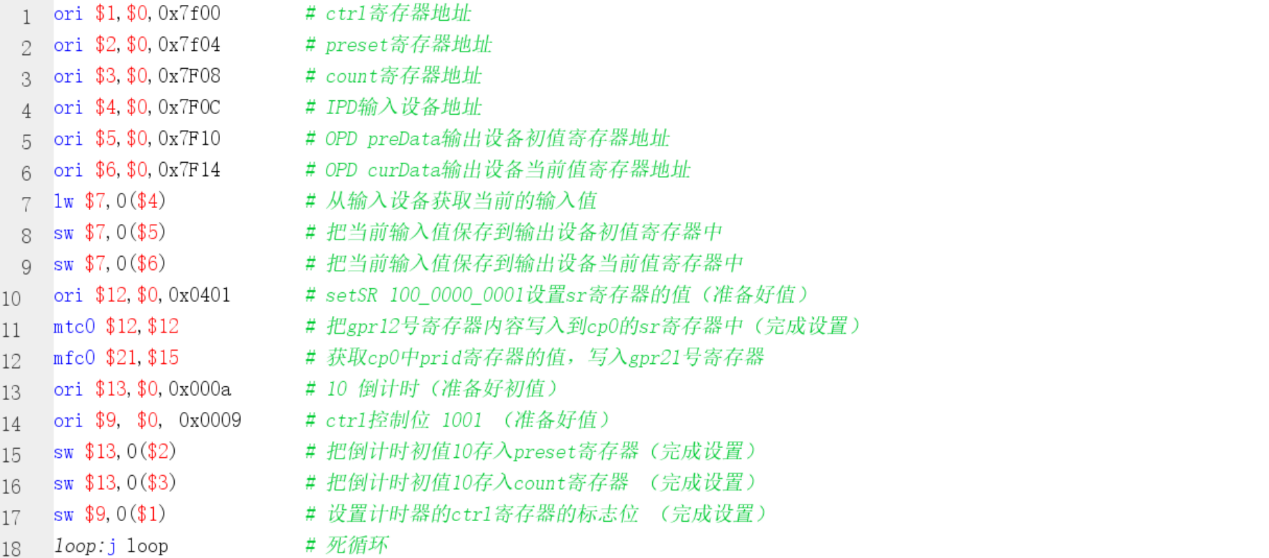
本次MIPS微系统的输入模块，是在顶层文件上直接定义了一个32位输入作为输入模块。因为输入模块只涉及数据的输入，不涉及其他功能，故只定义一个输入端口就够用了。

1. **指令描述**

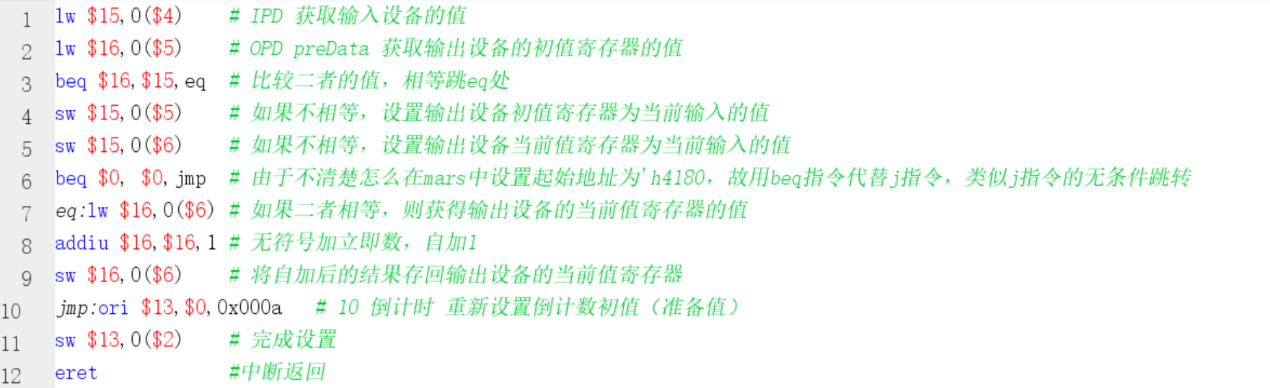


1. **测试程序**

## 4.1 MIPS-Lite3指令集的测试主程序

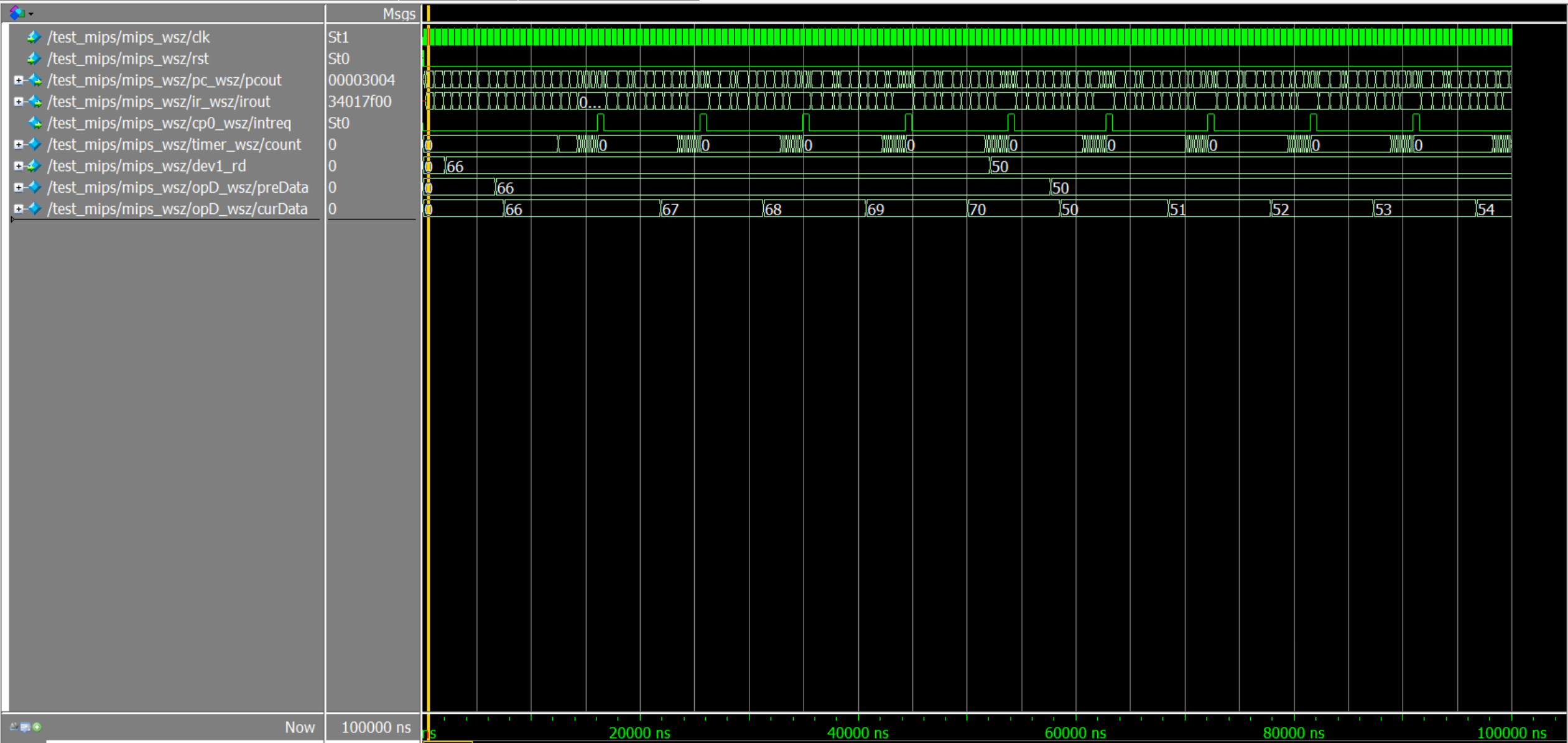


## 4.2 MIPS-Lite3指令集的测试中断子程序



1. **测试结果**

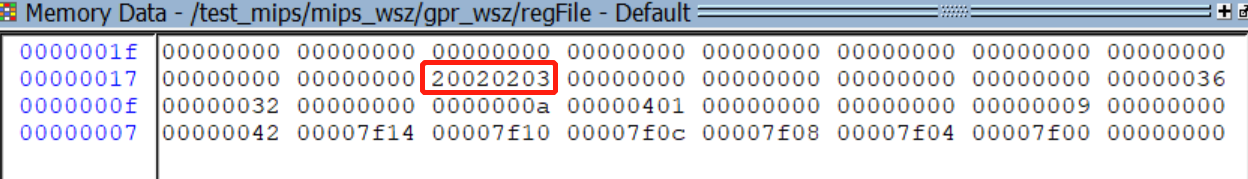
## 5.1 波形



## 5.2 寄存器组结果

### 5.2.1 gpr寄存器

mfc0指令，加载cp0的prid寄存器的值到gpr寄存器中。

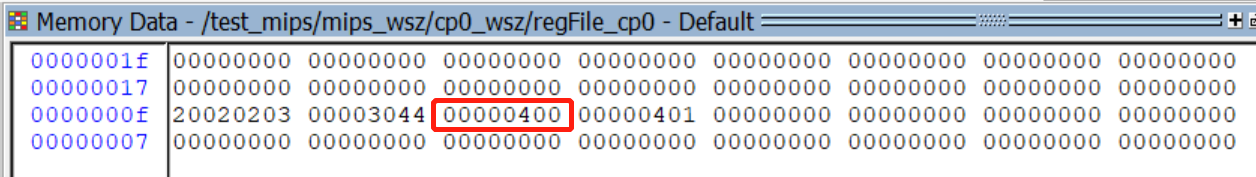


### 5.2.2 cp0寄存器

sr寄存器由mtc0指令设置；epc寄存器存返回地址；prid的签名是在初始化时设置的。



Cause寄存器不断锁存hwint的值，下图是波形最后时刻为产生中断请求时，锁存的结果。



1. **总结与心得**

通过完成MIPS微系统的开发，我不仅仅对cpu内部的数据通路、信号控制有了更上一层的了解，更关键的是，我对于cpu内部是如何与cpu外部进行通讯的有了很全面的了解。

在开发过程中，印象比较深刻的就是中断请求信号的传输过程：从外设定时器发出，经历系统桥，到协处理器，再到控制器，最后再到npc模块。其中比较关键的是intreq信号与状态机的配合使用，以及在多个中断需要处理时，使用屏蔽位来管理他们的优先级这一思想，都使我受益匪浅。

这次的MIPS微系统开发，主要解决的问题是与外设的通讯、交互功能。解决问题的主要思想是，合理设计控制信号的通路、合理设计状态机、合理设计系统桥，并且需要给外设分配符合存储空间规定的外设地址，以此来达到设计要求。

至此，计算机组成原理的课设结束了，但是我对于硬件课程的学习还没有停止，并且相信在今后的课程、课设中，一定会加倍努力学习，将知识融会贯通。