P4实验报告

14061075 修闽珂

1. 模块定义
2. GPR模块定义
   1. 基本描述

GPR是一个32bit \* 32寄存器堆，寄存器带有写使能功能。能够根据读入的地址寻找相应寄存器的位置以及读出内部的值。当写使能有效时，能够在写入地址写入需要被储存的值。

* 1. 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [4:0]RS1 | I | 读取的第一个数据的寄存器地址 |
| [4:0]RS2 | I | 读取的第二个数据的寄存器地址 |
| RegWrite | I | 写使能 |
| [4:0]RD\_WA | I | 写入地址 |
| WData | I | 写入数据 |
| ResetReg | I | 寄存器重置 |
| Clk | I | 时钟上升沿信号到来执行操作 |
| [31:0]PC\_4 | I | 当前PC加上四之后的结果，用于jr的跳回 |
| Ifjal | I | Jal指令的一个参数  1：jal有效，将返回计数器的值存到31号寄存器  0：jal无效。 |
| [31:0]RData1, RData2 | O | 寄存器输出的第一、二个值 |

* 1. 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 读数据 | 地址信号到来，寄存器输出这个地址储存的值 |
| 2 | 写数据 | 将数据写入到写入地址所示的寄存器中 |
| 3 | 清零 | 当这个信号不为零的时候，将所有寄存器中的值清零 |

1. ALU模块定义
   1. 基本描述

ALU模块是算数逻辑单元。在这个单元里面，能够进行加、减、按位或以及lui指令。数据输入到ALU后，同步进行这四种运算，由ALUCtrl来控制哪一个结果被输出。

* 1. 模块接口

|  |  |  |
| --- | --- | --- |
| c | 方向 | 描述 |
| [31:0]A | I | 第一个运算数 |
| [31:0]B | I | 第二个运算数 |
|  | I |  |
| [3:0]ALUCtrl | I | ALU控制器 |
| Result | O | 输出运算结果 |
| Zero | O | 运算结果是否为零的标志 |

1. 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 加法 | 将第一个运算数和第二个运算数相加  ALUCtrl：0001 |
| 2 | 减法 | 第一个和第二个运算数相减  ALUCtrl：0010 |
| 3 | 按位或 | 将第一个数和第二个数按位或  ALUCtrl：0100 |
| 4 | Lui | 将指令的低16位作为输入，放置到高位，作为输出数据  ALUCtrl：1000 |

1. EXT模块定义
   1. 基本描述

通过EXTOp来决定信号的扩展是在高位还是在低位执行。

* 1. 模块接口

|  |  |  |
| --- | --- | --- |
| c | 方向 | 描述 |
| [15:0]inComing | I | 指令的低十六位 |
| EXTOp | I | 来自控制单元的信号  1：低位加零  0：高位加零 |
| [31:0]Extresult | O | 输出的信号扩展结果 |

* 1. 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 信号位宽扩增 | 在高位加零（lui） |
| 2 | 信号位宽扩增 | 在低位加零 |

1. dm\_4k模块定义
   1. 基本描述

用作数据存储，以及从内存中取数的来源。

* 1. 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [11:2]addr | I | 访问存储器的地址 |
| [31:0]din | I | 写入的数据 |
| we | I | 存入内存相应地址中  1：是  0：否 |
| clk | I | 时钟信号 |
| [31:0]dout | O | 从寄存器中读出的数据 |

* 1. 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 写入到存储器中 | 写使能为是的时候将写入的数据放入相应地址中 |
| 2 | 读出数据 | 将数据从相应地址中读出 |

1. PC模块定义
   1. 基本描述

程序计数器本体，完成本次指令，保存下一次指令地址，复位后指向第一条指令。

1. 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| Clk | I | 时钟信号 |
| Reset | I | 复位信号  1： 复位  0：无效 |
| [31:0]PC | O | 指令寄存器地址 |
| [31:0]NPC | I | 下条指令地址 |

1. 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 复位 | 复位，将地址设为第一条指令的地址 |
| 2 | 保存NPC并输出 | 每个时钟上升沿保存NPC并输出 |

1. NPC模块定义
   1. 基本描述

下条指令地址的生成器

1. 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| PC | I | 当前指令地址 |
| [31:0]Instruction | I | 上条指令 |
| Ifbeq | I | 是否是beq转跳指令 |
| Zero | I | 是否是零的判断  1:是  0：不是 |
| Ifjal | I | 是否是jal指令  1：是  0：不是 |
| Ifjr | I | 是否是jr指令  1：是  0：不是 |
| Ifj | I | 是否是j指令  1：是  0：不是 |
| [31:0]GPR31 | I | 来自GPR的31号寄存器的值 |
| [31:0]nPC | O | 输出的下条指令的地址 |
| [31:0]PC\_4 | O | 当前指令加四的结果，可在GPR中存储 |

1. 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 计算下条指令地址 | 如果有jal、j指令，就转跳到相应指令  如果有beq指令，就判断Zero是否成立，成立就转跳否则不转跳。  如果以上指令都没有就在当前pc上加四作为下一条指令地址 |

1. Im\_4k模块定义
   1. 基本描述

指令寄存器，储存指令代码。

1. 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [11:2]addr | I | 指令地址 |
| [31:0]dout | O | 输出的指令 |

1. 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 输出指令 | 按照PC输入的地址输出指令 |

1. Controller模块定义
   1. 基本描述

控制器，产生各种指令对应的控制代码

1. 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [5:0]op | I | 指令的高五位信号 |
| [5:0]func | I | 低五位信号 |
| RegDst | O | 判断是否有RD  1：有  0：无 |
| ALUSrc | O | ALU的运算数据的来源选择 |
| MemtoReg | O | 是否要输入到寄存器中 |
| RegWrite | O | 寄存器的写使能 |
| BEQ | O | BEQ指令的指示 |
| JAL | O | JAL指令的指示 |
| JR | O | JR指令的指示 |
| ExtOp | O | Lui指令的指示 |
| J | O | J指令的指示 |
| [3:0]ALUCtrl | O | ALU控制器  0001：加  0010：减  0100：或  1000：lui |

1. 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 输出控制信号 | 根据指令输出控制信号控制个单各个单元 |

1. 多路选择器定义
   1. 基本描述

指令寄存器，储存指令代码。

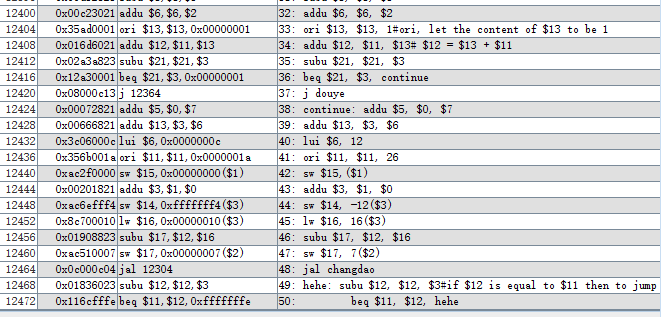
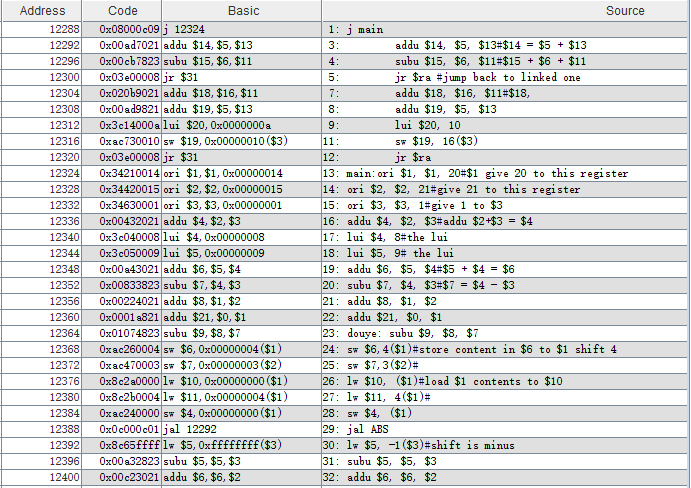
1. 模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [4:0]RT  [4:0]RD  RegDst  [4:0]RD\_WA | I  I  I  O | RT地址  RD地址  是否要用到RD  1：是  0 : 否  数据写入寄存器的地址 |
| [31：0]RData2  [31:0]Extended  ALUSrc  [31:0]ALUB | I  I  I  O | 选择的数据  选择的数据  ALU计算的来源  1：Extended作为ALU的一个计算数  0：RData2作为ALU的一个计算数  选择后的结果 |
| [31：0]dout  [31:0]ALUResult  MemtoReg  [31:0]WData | I  I  I  O | DM的输出  ALU的计算结果  哪个输入到寄存器中  1：dout  0：ALUResult  输出的数据 |

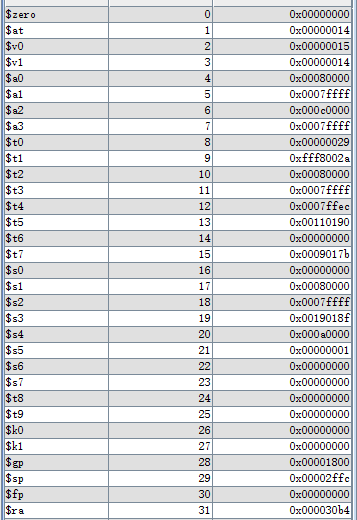
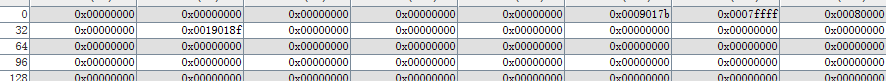
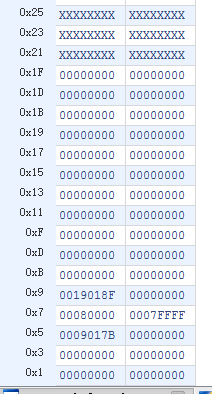
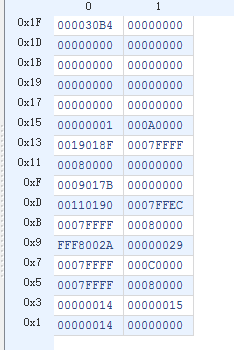
1. 功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 选择 | 根据控制信号选择数据 |

1. 测试数据



期望结果



08000c09#j 跳转

00ad7021#addu指令

00cb7823#subu指令

03e00008#jr指令

020b9021#addu指令

00ad9821#addu指令

3c14000a#lui指令

ac730010#sw， 存到19号寄存器

03e00008#jr

34210014#ori， 对$1自身

34420015#ori

34630001#ori

00432021#addu, 6 = 5 + 4

3c040008#7 = 4 + 3

3c050009#8 = 1 + 2

00a43021#21 = 0 + 1

00833823#9 = 8 - 7

00224021 #addu， 8 = 1 + 2

0001a821#addu 21 = 0 + 1

01074823#subu 9 = 8 - 7

ac260004##sw $6 到 1所指向的位置的偏移4位

ac470003#sw

8c2a0000#lw将1 中的所指向的地址的值给予6

8c2b0004#lw

ac240000#sw

0c000c01#jal, 跳到12292

8c65ffff#lw，3 的偏移-1

00a32823#subu

00c23021#addu

35ad0001#ori

016d6021#addu

02a3a823#subu

12a30001#beq，判断￥21与￥3是否相等，相等跳出循环

08000c13#循环的控制

00072821#addu

00666821#sw

3c06000c#lw

356b001a#subu

ac2f0000#sw

00201821#addu

ac6efff4#sw

8c700010#lw

01908823#subu

ac510007#sw

0c000c04#jal

01836023#subu

116cfffe#beq（注释中的数字是寄存器的序号）。

1. 问答
   1. 有溢出的操作在溢出发生时会有overflow exception，无符号运算在溢出发生不会报告溢出，对于溢出的处理没有要求。可以用两个GPR表示溢出的结果，如果对溢出结果不做要求，则可以忽略。