LOGISIM单周期CPU实验报告（完成版）

一、顶层设计视图

单周期CPU的顶层设计视图

IFU

CTR

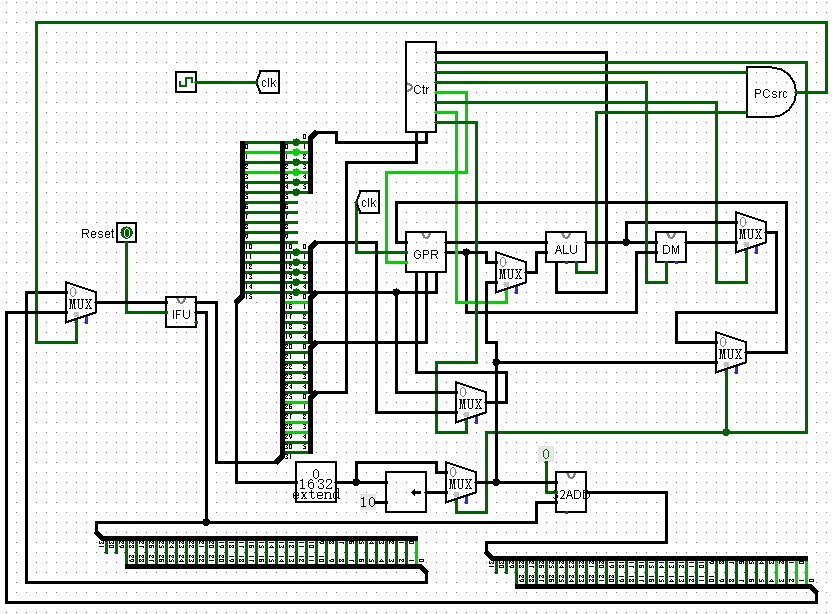
RF

DM

ALU

ALU

EXT



二、模块定义

1. IFU模块

（1） 基本描述

内部包括PC和IM级相关逻辑；PC以30位的寄存器实现，设置复位信号，IM中存有32位的指令，PC中是IM中指令的地址，选择低五位作为IM中的选择信号；

（2） 模块定义

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| PCnext | I | 下一次的PC地址 |
| Reset | I | 复位信号 |
| Clk | I | 时钟信号 |
| PC4 | O | 当前地址+4后的地址 |
| OPcode | O | 从IM选择的代码 |

2. ALU模块

（1）基本描述

算术逻辑运算单元ALU用32加法器加上逻辑门实现了加减与或的逻辑算术运算功能，有两个32位操作数，一个32位输出，和一个3位的控制ALUcontrol和zero单位。

（2）模块定义

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| SrcA | I | A操作数32位 |
| SrcB | I | B操作数32位 |
| ALUcontrol | I | ALU运算控制代码  001 ： A and B  001 : A or B  010 : A + B  100 : A and /B  101 : A or /B  110 : A – B |
| zero | O | 零标志位 |
| Result | O | 运算结果32位 |

3. DM模块

（1） 基本描述

数据存储器从ＡＬＵ中计算出结果得出存储的地址，将Wdata的数据写入ＤＭ中，并把数据写出，有时钟和MemWrite和Memread作为控制信号，用32位的RAM实现，将ＤＭ做成利用地址将选定寄存器实现数据存储。

（2） 模块定义

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| Ａ | I | 指令存储器中寄存器的地址 |
| Wdata | I | 写入ＤＭ中的３２位数据 |
| Clk | I | 时钟信号 |
| MemWrite | I | 写使能信号 |
| ReadData | O | 从DM读出的数据 |

4. GPR模块

（1） 基本描述

寄存器堆GPR有32个存储32位数据的寄存器，从IFU中获得的32位指令，从32位代码的编码中可以知道即将要进行操作的寄存器的地址，在GPR中选择相应的寄存器，读出代码之后，从RD1和RD2中读出操作数。

（2）模块定义

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| Wreg | I | 写入数据时候写入的地址 |
| Wdata | I | 写入Wreg指向的地址的寄存器中写入32位数据 |
| Clk | I | 时钟信号 |
| RegWrite | I | 写使能信号 |
| Reg1 | I | 存储在RF中的第一个数据的地址 |
| Reg2 | I | 存储在RF中的第二个数据的地址 |
| Rdata1 | O | 从RF中读出的第一个数据 |
| Rdata2 | O | 从RF中读出的第二个数据 |

5. Control模块

（1） 基本描述

将32位指令按规定格式转化成各个单元的控制信号，达到利用指令编码控制硬件和执行程序的目的。

（2） 模块定义

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| funct | I | 特定操作说明位 |
| op | I | Opcode操作码 |
| RegDst | O | 写使能信号 |
| ALUSrc | O | 控制ALU操作  0为R类型运算 从SrcB中取出数据运算  1为Lw/sw运算 从SignImm选择数据 |
| MemtoReg | O | 控制数据从ALU读出还是DM中读出，是Sw和R类型的区别  0 选择ALUresult 作为result R类型  1 选择ReadData 支持lw |
| RegWrite | O | 寄存器堆的写使能信号  1 写入数据到寄存器堆  0 不写入寄存器文件 |
| MemWrite | O | DM的写使能信号  1 向DM写入数据 |
| Branch | O | Beq分支指令的控制端 |
| ALUctr<2:0> | O | ALU的三位控制信号，控制ALU的操作  000 and 001 or 010 + 110 - |

（3） 支持指令集

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | op | Rs | Rt | Rd | Shamt | Func |  |  |
| Addu | 000000 | Rs | Rt | Rd | 00000 | 100001 | Addu $1,$2,$3 |  |
| Subu | 000000 | Rs | Rt | Rd | 00000 | 100011 | Subu $1,$2,$3 |  |
| Ori | 001101 | Rs | Rt | immediate | | |  |  |
| Lw | 100011 | Rs | Rt | immediate | | |  |  |
| Sw | 101011 | Rs | Rt | immediate | | |  |  |
| lui | 001111 | 00000 | Rt | immediate | | |  |  |
| beq | 000100 | Rs | Rt | immediate | | |  |  |

（4） 控制器的控制真值表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| funct | 10 0000 | 10 0010 |  |  |  |  |
| op | 00 0000 | 00 0000 | 00 1101 | 10 0011 | 10 1011 | 00 0100 |
|  | Add | Sub | Ori | Lw | Sw | Beq |
| RegDst | 1 | 1 | 0 | 0 | X | X |
| ALUSrc | 0 | 0 | 1 | 1 | 1 | 0 |
| MemtoReg | 0 | 0 | 0 | 1 | X | X |
| RegWrite | 1 | 1 | 1 | 1 | 0 | 0 |
| MemWrite | 0 | 0 | 0 | 0 | 1 | 0 |
| Branch | 0 | 0 | 0 | 0 | 0 | 1 |
| ALUctr<2:0> | Add 010 | Sub 110 | Or 001 | Add 010 | Add 010 | Sub 110 |

三、 测试程序原理

答：在MARS里面编写MIPS的汇编测试指令，包含将要测试的代码addu,subu,ori,lw,sw,beq,lui指令，将汇编代码转化成16进制表示的二进制汇编代码，然后加载到logisim中的IFU中，运行程序，看一下GPR和DM中的数据是否和程序预期的结果一样，如果运行的结果符合预期，则说明该条测试程序通过，改设计对于这条程序而言，设计是正确的。

四、 问答题

17、答：控制信号的逻辑表达式

Regdst=add+sub+~ori+~lw

ALUsrc=~add+~sub+ori+lw+sw

Memtoreg=~add+~sub+~ori+lw

Regwrite=add+sub+ori+lw+~sw+~beq

Branch=beq+~add+~sub+~ori+~lw+~sw

18、答：下列信号需要2输入与门、2输入或门、非门的数量如下：采用（输出信号名=（2输入与门、2输入或门、非门）

RegDst=（17,1,11） ALUSrc=(15,2,8) MemtoReg=（5,1,3） RegWrite=（22,3,21） MemWrite=（5,1,2） Branch=（5,1,5） ALUctr<2:0>=（22,4,18）