计算机组成原理P3实验报告

1. 模块定义
2. IFU模块定义

（1）基本描述

IFU主要功能是完成取指令功能。IFU内部包括了PC、IM(指令存储器)以及其他相关逻辑。IFU除了能执行顺序取指令外，还能根据PCSrc反映的beq指令的执行情况来决定是顺序取指令还是转移取指令。

（2）模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| PCSrc | I | 当前指令是否是beq指令  1:当前指令是beq指令  0:当前指令不是beq指令 |
| Imm[31:0] | I | 若为beq指令，需要进行移位计算的立即数 |
| Reset | I | 复位信号，1:有效，0:无效 |
| Clk | I | 时钟信号 |
| Instr[31:0] | O | 当前指令输出 |

（3）功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 复位 | 复位信号有效时，PC设置为0x00000000 |
| 2 | 取指令 | 根据PC当前值从IM中取指令输出 |
| 3 | 计算下一条指令地址 | beq == 0 时，PC 🡨 PC + 4；  beq == 1 时，PC 🡨 PC + 4 + Imm||0^2 |

1. GRF模块定义

（1）基本描述

GRF模块为通用寄存器堆，主要由32个具有写使能端的32位寄存器组成，能够同时根据由R1和R2输入的地址从其中两个寄存器中读出数据，并根据WR中输入的地址向其中一个寄存器写入数据。

（2）模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| R1[4:0] | I | rs寄存器的地址 |
| R2[4:0] | I | rt寄存器的地址 |
| WR[4:0] | I | 要写入的寄存器的地址 |
| WriteData[31:0] | I | 要写入的数据 |
| Clk | I | 时钟信号 |
| RegWrite | I | 写使能信号，1:有效，0:无效 |
| RD1[31:0] | O | rs寄存器的值 |
| RD2[31:0] | O | rt寄存器的值 |

（3）功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 读数据 | RD1🡨(R1)  RD2🡨(R2) |
| 2 | 写数据 | RegWrite有效时，(WR)🡨WriteData |

1. ALU模块定义

（1）基本描述

ALU为算数逻辑单元，可以对输入的两个数据进行加、减、按位与和按位或操作，并能够判断输入数据是否相等。

（2）模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| A1[31:0] | I | 第一个运算数 |
| A2[31:0] | I | 第二个运算数 |
| ALUCtr[1:0] | I | ALU控制信号  2’b00:加法运算  2’b01:减法运算  2’b10:按位与运算  2’b11:按位或运算 |
| Zero | O | 输入数据是否相等  1:相等  2:不相等 |
| ALUResult[31:0] | O | ALU运算结果 |

（3）功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 加法运算 | ALUResult 🡨 A1+A2 |
| 2 | 减法运算 | ALUResult 🡨 A1-A2 |
| 3 | 按位与运算 | ALUResult 🡨 A1&A2 |
| 4 | 按位或运算 | ALUResult 🡨 A1|A2 |
| 5 | 等于判断 | Zero 🡨 (A1-A2)==0?1:0 |

1. DM模块定义

（1）基本描述

DM模块为数据存储器，由一个32bit \* 32字的RAM构成，起始地址为0x00000000用于存储数据。

（2）模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| Addr[4:0] | I | 读/写DM的地址 |
| WriteData[31:0] | I | 要写入DM的数据 |
| MemWrite | I | 写DM的控制信号 |
| MemRead | I | 读DM的控制信号 |
| ReadData[31:0] | O | 从DM读出的数据 |

（3）功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 读数据 | 当MemRead为1时，ReadData 🡨 RAM(Addr) |
| 2 | 写数据 | 当MemWrite为1时，RAM(Addr) 🡨 WriteData |

1. EXT模块定义

（1）基本描述

EXT模块的作用是将16位立即数扩展为32位。

（2）模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| Imm\_16[15:0] | I | 要扩展的16位立即数 |
| ExtOp[1:0] | I | 扩展方式选择信号  2’b00:符号扩展  2’b01:后接16位0  2’b10:无符号扩展 |
| Imm\_32[31:0] | O | 扩展后的32位立即数 |

（3）功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 位数扩展 | ExtOp为2’b00时，16位立即数正常符号扩展为32位  ExtOp为2’b01时，16为立即数后接16位0扩展为32位  ExtOp为2’b10时，16为立即数无符号扩展为32位 |

1. 控制器设计
2. Controller模块定义

（1）基本描述

Controller模块为CPU控制器，可以根据输入指令的opcode和funct值输出各种控制信号。

（2）模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| opcode[5:0] | I | 指令中的opcode,即[31:26]位 |
| Funct[5:0] | I | 指令中的funct,即[5:0]位 |
| RegDst | O | 寄存器写入端地址控制  1:选择rd字段  0:选择rt字段 |
| ALUSrc | O | ALU输入端A2选择  1:选择Imm\_32  0:选择RD2 |
| MemtoReg | O | 寄存器堆写入端WD选择  0:来自ALU输出  1:来自DM输出 |
| RegWrite | O | 写寄存器控制信号 |
| MemWrite | O | 写DM控制信号 |
| MemRead | O | 读DM控制信号 |
| ExtOp[1:0] | O | EXT扩展方式控制信号 |
| nPC\_Sel | O | 判断是否为beq指令 |
| ALUCtr[1:0] | O | ALU控制信号 |

（3）功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | addu指令 | 当前指令为addu时，RegDst、RegWrite信号为1，其他全为0 |
| 2 | subu指令 | 当前指令为subu时，RegDst、RegWrite、ALUCtr[1]信号为1，其他全为0 |
| 3 | ori指令 | 当前指令为ori时，ALUSrc、RegWrite、ALUCtr[0]、ALUCtr[1]、ExtOp[1]信号为1，其他全为0 |
| 4 | lw指令 | 当前指令为lw时，ALUSrc、MemtoReg、RegWrite、MemRead信号为1，其他全为0 |
| 5 | sw指令 | 当前指令为sw时，ALUSrc、MemWrite信号为1，其他全为0 |
| 6 | beq指令 | 当前指令为beq时，nPC\_Sel、ALUCtr[1]信号为1，其他全为0 |
| 7 | lui指令 | 当前指令为lui时，ALUSrc、RegWrite、ExtOp[0]信号为1，其他全为0 |

1. Controller真值表

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instr | | addu | subu | ori | lw | sw | beq | lui |
| opcode | I | 000000 | 000000 | 001101 | 100011 | 101011 | 000100 | 001111 |
| funct | 100001 | 100011 | N/A | | | | |
| RegDst | O | 1 | 1 | 0 | 0 | X | X | 0 |
| ALUSrc | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| MemtoReg | 0 | 0 | 0 | 1 | X | X | 0 |
| RegWrite | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| nPC\_Sel | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| ExtOp[1] | X | X | 1 | 0 | 0 | 0 | 0 |
| ExtOp[0] | X | X | 0 | 0 | 0 | 0 | 1 |
| MemRead | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| MemWrite | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| ALUOp[1] | | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| ALUOp[0] | | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| ALUCtr[1] | O | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| ALUCtr[0] | 0 | 1 | 1 | 0 | 0 | 1 | 0 |

1. ALU控制器设计

ALU控制器是控制器的一部分，它利用funct[5:0]的值和过渡信号ALUOp[1:0]得到控制ALU运算的ALU控制信号ALUCtr[1:0]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ALUOp | Funct | ALUCtr | 运算 | 描述 |
| 00 | X | 00 | 加法 | 针对lw、sw、lui指令 |
| 01 | X | 01 | 减法 | 针对beq指令 |
| 11 | X | 11 | 按位或 | 针对ori指令 |
| 10 | 100001(addu) | 00 | 加法 | 针对addu指令 |
| 10 | 100011(subu) | 01 | 减法 | 针对subu指令 |

1. 测试程序设计