Verilog多周期CPU设计文档

1. CPU设计方案综述

使用Verilog开发一个简单的多周期CPU，总体概述如下：

1. 此CPU为32位CPU
2. 此CPU为多周期设计
3. 此CPU支持异常和中断
4. 此CPU支持的指令集为：{LB, LBU, LH, LHU, LW, SB, SH, SW, ADD, ADDU,SUB, SUBU, MULT, MULTU, DIV, DIVU, SLL, SRL, SRA, SLLV, SRLV, SRAV, AND, OR, XOR, NOR, ADDI, ADDIU, ANDI, ORI, XORI, LUI, SLT, SLTI, SLTIU, SLTU, BEQ, BNE, BLEZ, BGTZ, BLTZ, BGEZ, J, JAL, JALR, JR, MFHI, MFLO, MTHI, MTLO, MFC0, MTC0, ERET}
5. nop机器码为0x00000000
6. addu, subu, addiu不支持溢出
7. 关键模块定义
8. PC
9. 端口说明

表1-PC端口说明

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 信号名 | 方向 | 描述 |
| 1 | clk | I | 时钟信号 |
| 2 | reset | I | 复位信号 |
| 3 | NPC | I | 下一条指令所在IM地址 |
| 4 | PC | O | 当前指令所在IM地址 |

1. 功能定义

表2-PC功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能 | 描述 |
| 1 | 存储指令的地址 | 保存当前执行指令在IM中的地址 |

1. IM
2. 端口说明

表3-IM端口说明

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 信号名 | 方向 | 描述 |
| 1 | PC[31:0] | I | 时钟信号 |
| 2 | instr[31:0] | O | 指令 |

1. 功能定义

表4-IM功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能 | 描述 |
| 1 | 取指令 | 就是取指令 |

1. IFID
2. 端口说明

表5-IDIF功能定义

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 信号名 | 方向 | 描述 |
| 1 | clk | I | 时钟信号 |
| 2 | en | I | 使能信号 |
| 3 | reset | I | 同步复位信号 |
| 4 | PCF[31:0] | I | PC在F级的值 |
| 5 | InstrF[31:0] | I | instr在F级的值 |
| 6 | PCD[31:0] | O | PC在D级的值 |
| 7 | InstrD[31:0] | O | instr在D级的值 |

1. 功能定义

表6-IFID功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能 | 描述 |
| 1 | 存储流水线值 | 存储流水线值 |

1. NPC
2. 端口说明

表7-NPC端口说明

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 信号名 | 方向 | 描述 |
| 1 | branch | I | 分支信号 |
| 2 | JType | I | 跳转信号 |
| 3 | JReg | I | 判断指令是否需要跳转寄存器 |
| 4 | PCF[31:0] | I | F级PC值 |
| 5 | PCD[31:0] | I | D级PC值 |
| 6 | RegJump[31:0] | I | 跳转寄存器中地址值 |
| 7 | imm26D[25:0] | I | D级的26位立即数 |
| 8 | NPC[31:0] | O | 根据各种指令计算出的下一个PC值 |

1. 功能定义

表8-NPC功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能 | 描述 |
| 1 | 计算下一个PC的值 |  |

1. GRF
2. 端口说明

表9-GRF端口说明

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 信号名 | 方向 | 描述 |
| 1 | clk | I | 时钟信号 |
| 2 | reset | I | 同步复位信号，将32个寄存器中全部清零  1：清零  0：无效 |
| 3 | WE | I | 写使能信号  1：可向GRF中写入数据  0：不能向GRF中写入数据 |
| 4 | A1[4:0] | I | 5位地址输入信号，指定32个寄存器中的一个，将其中存储的数据读出到RD1 |
| 5 | A2[4:0] | I | 5位地址输入信号，指定32个寄存器中的一个，将其中存储的数据读出到RD2 |
| 6 | A3[4:0] | I | 5位地址输入信号，指定32个寄存器中的一个，作为RD的写入地址 |
| 7 | WD[31:0] | I | 32位写入数据 |
| 8 | RD1[31:0] | O | 输出A1指定的寄存器的32位数据 |
| 9 | RD2[31:0] | O | 输出A2指定的寄存器的32位数据 |

1. 功能定义

表10-GRF功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能 | 描述 |
| 1 | 异步复位 | reset为1时，将所有寄存器清零 |
| 2 | 读数据 | 将A1和A2地址对应的寄存器的值分别通过RD1和RD2读出 |
| 3 | 写数据 | 当WE为1且时钟上升沿来临时，将WD写入到A3对应的寄存器内部 |

1. CMP
2. 端口说明

表11-CMP端口说明

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 信号名 | 方向 | 描述 |
| 1 | A[31:0] | I | 操作数A |
| 2 | B[31:0] | I | 操作数B |
| 3 | eq | O | A==B？ |
| 4 | eqz | O | A==0？ |
| 5 | ltz | O | A<0? |

1. 功能描述

表12-CMP功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能 | 描述 |
| 1 | 判断A和B是否相等 | 若A等于B，eq置一，否则置零 |
| 2 | 判断A是否等于0 | 若A等于0，eqz置一，否则置零 |
| 3 | 判断A是否小于0 | 若A小于0，ltz置一，否则置零 |

1. EXT
2. 端口说明

表13-EXT端口说明

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 信号名 | 方向 | 描述 |
| 1 | imm16[15:0] | I | 代扩展的16位信号 |
| 2 | sign | I | 无符号或符号扩展选择信号  0：无符号扩展  1：符号扩展 |
| 3 | imm32[31:0] | O | 扩展后的32位的信号 |

1. 功能定义

表14-EXT功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能 | 描述 |
| 1 | 无符号扩展 | 当sign为0时，将imm16无符号扩展输出 |
| 2 | 符号扩展 | 当sign为1时，将imm16符号扩展输出 |

1. IDEX
2. 功能定义

表15-IDEX功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能 | 描述 |
| 1 | 流水线寄存器 | 保留ID/EX级流水线信息 |

1. ALU
2. 端口说明

表16-ALU端口说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 信号名 | 方向 | | 描述 | |
| 1 | A[31:0] | | I | | 参与运算的第一个数 |
| 2 | B[31:0] | | I | | 参与运算的第二个数 |
| 3 | ALUOp[2:0] | | I | | 决定ALU做何种操作  0000：无符号加  0001：无符号减  0010：与  0011：或  0100：将B[15:0]做为res[31:16],res[15:0]=0 |
| 5 | res[31:0] | | O | | A与B做运算后的结果 |

1. 功能定义

表17-ALU功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能 | 描述 |
| 1 | 加运算 | res = A + B |
| 2 | 减运算 | res = A - B |
| 3 | 与运算 | res = A & B |
| 4 | 或运算 | res = A | B |
| 5 | 加载高位运算 | res = {B[15:0], 16’h0} |

1. MDU
2. 端口说明

表18-MDU端口说明

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 信号 | 方向 | 描述 |
| 1 | clk | I | 时钟信号 |
| 2 | reset | I | 复位信号 |
| 3 | A[31:0] | I | 操作数A |
| 4 | B[31:0] | I | 操作数B |
| 5 | start | I | 启动信号 |
| 6 | MDUOp[2:0] | I | 操作选择信号 |
| 7 | HIWrite | I | HI寄存器写使能 |
| 8 | LOWrite | I | LO寄存器写使能 |
| 9 | HI[31:0] | O | HI寄存器的值 |
| 10 | LO[31:0] | O | LO寄存器的值 |
| 11 | busy | O | MDU计算ing |

1. 功能定义

表19-MDU功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能 | 描述 |
| 1 | 复位 | 同步复位 |
| 2 | 计算 | 乘除计算和写HILO |
| 3 | 输出 | 输出HI和LO的值 |

1. EXMEM
2. 功能定义

表20-EXMEM功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能 | 描述 |
| 1 | 流水线寄存器 | 保留EX/MEM级流水线信息 |

1. DM
2. 端口说明

表21-DM端口说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 信号名 | 方向 | 描述 | |
| 1 | clk | I | 时钟信号 | |
| 2 | reset | I | 异步复位信号  0：无效  1：内存值全部清零 | |
| 3 | WE | I | 写使能信号  0：禁止写入  1：允许写入 | |
| 4 | witdth | I | 读写位宽 | |
| 5 | LoadSign | I | 读写时是否带符号 | |
| 6 | addr[31:0] | I | 读取或写入信号地址 |
| 7 | WD[31:0] | I | 写入Mem的值 |
| 6 | RD[31:0] | O | 32位读出数据 | |

1. 功能定义

表22-DM功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能 | 描述 |
| 1 | 异步复位 | 当reset为1时，DM中所有数据清零 |
| 2 | 写入数据 | 当WE有效时，时钟上升沿来临时，WD中数据写入A对应的DM地址中 |
| 3 | 读出数据 | RD永远读出A对应的DM地址中的值 |

1. Controller
2. 端口说明

表23-Controller端口说明

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 信号名 | 方向 | 描述 |
| 1 | instr[31:0] | I | instr[31:26],6位控制信号 |
| 2 | eq | I | RegRead1和RegRead2是否相等 |
| 3 | sign | O | Ext是否进行符号扩展 |
| 4 | branch | O | 是否进行分支 |
| 5 | JType | O | J型信号 |
| 6 | JReg | O | 是否读取寄存器值作为NPC |
| 7 | WDSelD | O | 若D级生成写入寄存器的值的选择信号。 |
| 8 | A3DE | O | 写入寄存器的地址 |
| 9 | ALUOp | O | ALU怎么算 |
| 10 | ALUASel | O | ALU第一个操作数的选择信号 |
| 11 | ALUBSel | O | ALU第二个操作数的选择信号 |
| 12 | WDSelE | O | E级写入寄存器的值的选择信号 |
| 13 | WEMem | O | DM写使能信号  0：禁止写入  1：允许写入 |
| 14 | width | O | 写入Mem的位宽 |
| 15 | LoadSign | O | 是否load带符号的值 |
| 16 | WDSelM | O | M级生成写入寄存器的值的选择信号 |
| 17 | D1Use | O | D级是否读取rs |
| 18 | D2Use | O | D级是否读取rt |
| 19 | MD | O | 是否将使用MDU |
| 20 | E1Use | O | E级是否读取rs |
| 21 | D2Use | O | E级是否读取rt |
| 22 | M2Use | O | M级是否读取rt |

he

1. 转发暂停控制

不同于教程所采用的方法，我使用的是一种方法。这种方法并不来源于我，而是来kxh学长。

面对不同的情况，分别将指令暂停在D级或E级。

转发和暂停之间有优先级，

暂停在D级和E级有优先级。

每一级都将要写入到寄存器的数据流水，当一级要使用的数据冲突时，发现要写入的数据为高阻态，则暂停。

表24-HZD端口说明

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 信号名 | 方向 | 描述 |
| 1 | clk | I | 时钟信号 |
| 2 | reset | I | 重置信号 |
| 3 | A1D[4:0] | I | D级读取rs的序号 |
| 4 | A2D[4:0] | I | D级读取rt的序号 |
| 5 | RD1D[31:0] | I | D级读取rs的值 |
| 6 | RD2D[31:0] | I | D级读取rt的值 |
| 7 | D1Use | I | D级是否在使用rs |
| 8 | D2Use | I | D级是否在使用rt |
| 9 | A1E[4:0] | I | E级读取rs的序号 |
| 10 | A2E[4:0] | I | E级读取rt的序号 |
| 11 | RD1E[31:0] | I | E级读取rs的值 |
| 12 | RD2E[31:0] | I | E级读取rt的值 |
| 13 | E1Use | I | E级是否使用rs |
| 14 | E2Use | I | E级是否使用rt |
| 15 | A3E[4:0] | I | 写入寄存器的序号 |
| 16 | WDE[31:0] | I | E级写入寄存器的值 |
| 17 | A2M[4:0] | I | M级读取rt的序号 |
| 18 | RD2M[31:0] | I | M级读取到寄存器的值 |
| 19 | M2Use | I | M级是否在使用rt |
| 20 | A3M[4:0] | I | M级写入寄存器的信号 |
| 21 | WDM[31:0] | I | M级写入寄存器的值 |
| 22 | A3W[4:0] | I | W级写入寄存器的序号 |
| 23 | WDW[31:0] | I | W级写入寄存器的值 |
| 24 | FwdD1[31:0] | O | 转发给D级的值 |
| 25 | FwdD2[31:0] | O | 转发给D级的值 |
| 26 | FwdE1[31:0] | O | 转发给E级的值 |
| 27 | FwdE2[31:0] | O | 转发给E级的值 |
| 28 | FwdM2[31:0] | O | 转发给M级的值 |
| 29 | EnPC | O | PC的使能信号 |
| 30 | EnIFID | O | IFID的使能信号 |
| 31 | EnIDEX | O | IDEX的使能信号 |
| 32 | FlushIDEX | O | IDEX的清零信号 |
| 33 | FlushEXMEM | O | EXMEM的清零信号 |

1. 异常与IO
2. 桥
3. 端口定义

表25-Bridge端口说明

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 端口名 | 方向 | 描述 |
| 1 | PrAddr[31:2] | I | Processer要读写的DEV地址 |
| 2 | PrWD[31:0] | I | Processer要写入的值 |
| 3 | PrWE | I | 写入使能 |
| 4 | DEV0RD[31:0] | I | 从DEV0读取的数据 |
| 5 | DEV1RD[31:0] | I | 从DEV1读取的数据 |
| 6 | DEVAddr[31:2] | O | 要读写DEV的地址 |
| 7 | DEVWD[31:0] | O | 要写入DEV的数据 |
| 8 | DEV0WE | O | DEV0的写入使能 |
| 9 | DEV1WE | O | DEV1的写入使能 |
| 10 | PrRD[31:0] | O | 从DEV读取进入Processer的数据 |

1. 功能描述

表26-Bridge功能描述

|  |  |  |
| --- | --- | --- |
| 序号 | 功能 | 描述 |
| 1 | 从DEV读取数据 | 根据从Processer计算出的地址，从DEV读取数据 |
| 2 | 将数据写入DEV | 当写使能为1时，将数据写入 |

1. CP0

内部有SR、Cause、EPC、PrIDf四个寄存器

1. 端口说明

表27-CP0端口说明

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 端口名 | 方向 | 描述 |
| 1 | clk | I | 时钟信号 |
| 2 | reset | I | 复位信号 |
| 3 | WE | I | 写使能信号 |
| 4 | addr[4:0] | I | 写入或读取数据的地址 |
| 5 | DIn[31:2] | I | 写入数据 |
| 6 | BD | I | 是否分支延迟 |
| 7 | PC[31:2] | I | 受害或中断指令 |
| 8 | ExcCode[6:2] | I | 异常指令码 |
| 9 | HWInt[7:2] | I | 外部中断 |
| 10 | eret | I | eret信号 |
| 11 | IntReq | O | 对Processer的中断请求 |
| 12 | EPC[31:0] | O | 受害指令 |
| 13 | Dout[31:0] | O | 根据addr输出读取数据 |

1. 功能描述

表28-CP0功能描述

|  |  |  |
| --- | --- | --- |
| 序号 | 功能 | 描述 |
| 1 | 复位 | 同步复位信号 |
| 2 | 写入数据 | WE有效，在时钟上升沿，将外部数据写入 |
| 3 | 存储中断状态 | 当中断发生时，更新CP0寄存器的值，即更新状态 |
| 4 | 从异常状态中返回 | 当eret有效时，EXL回落 |
| 5 | 读取数据 | DOut据addr返回寄存器的数据 |

1. 测试方案

代码见附件

1. 新增指令

先保证新增的指令eret、mfc0和mtc0正确，

再保证新增指令的暂停转发正确

1. IO控制

对DEV进行读写操作，测试IO

1. 异常处理

对所有的异常进行测试，看看能否进入handle再返回

1. 中断处理

外界施以interrupt，测试程序能否进入中断处理

1. 特殊情况下的处理

包括但不限于：

eret后紧跟interrupt

使用乘除模块前中断

数据冲突的暂停中，在nop里中断

在eret前使用乘除模块

1. 思考题
2. **我们计组课程一本参考书目标题中有“硬件/软件接口”接口字样，那么到底什么是“硬件/软件接口”？（Tips：什么是接口？和我们到现在为止所学的有什么联系？）**

软硬件接口，我觉得指的是软件运行以及异常处理部分在软件、硬件中的约定。沟通起了软件和硬件。

1. **在我们设计的流水线中，DM 处于 CPU 内部，请你考虑现代计算机中它的位置应该在何处。**

我觉得会在CPU外部 😊

比如内存条啥的。

1. **BE 部件对所有的外设都是必要的吗？**

不是。像我们这次的project读写都是按照word的，就没有用到BE。

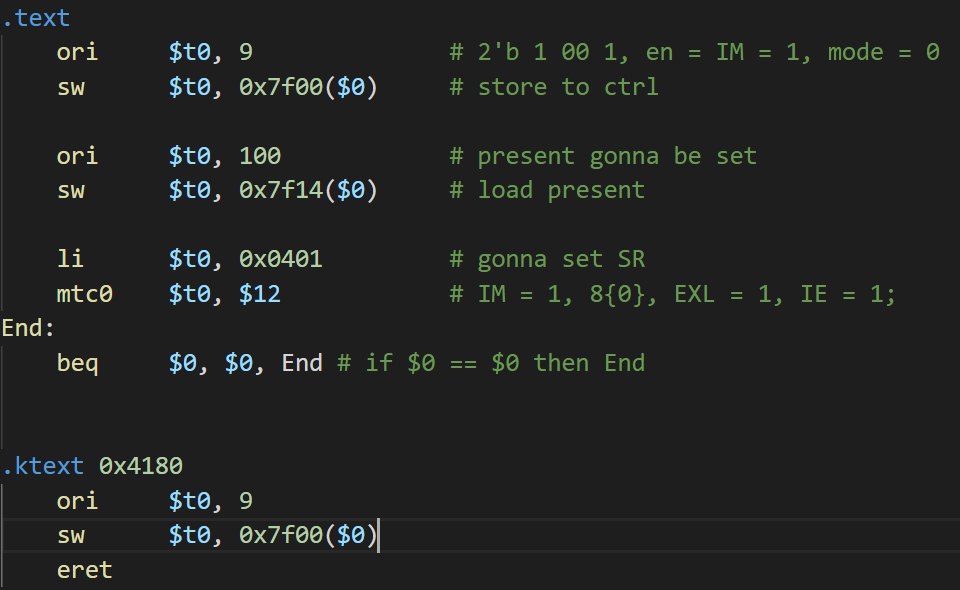
1. **请开发一个主程序以及定时器的exception handler。整个系统完成如下功能：**

**定时器在主程序中被初始化为模式0；**

**定时器倒计数至0产生中断；**

**handler设置使能Enable为1从而再次启动定时器的计数器。2及3被无限重复。**

**主程序在初始化时将定时器初始化为模式0，设定初值寄存器的初值为某个值，如100或1000。（注意，主程序可能需要涉及对CP0.SR的编程，推荐阅读过后文后再进行。）**



1. **请查阅相关资料，说明鼠标和键盘的输入信号是如何被CPU知晓的？**

鼠标点击、位置移动或按下键盘时，将产生中断信号，从而使 CPU 进入相应的中断处理程序。