计算机组成原理实验报告参考模板

一、CPU设计方案综述

（一）总体设计概述

本CPU为Verilog实现的单周期MIPS-COU,支持的指令集包含{addu,subu,ori, lw, sw, beq, lui, nop,sll, j, jr, jal, addiu}。CPU主要包含了PC,NPC,IM,GRF,ALU,DM,EXT等模块，这些模块按照数据通路、控制信号的设计逐级展开。

（二）关键模块定义

1. GRF

信号名	方向	描述
clk	I	CPU时钟
reset	I	同步复位信号
WE	I	写使能
A1[4:0]	I	rs输入
A2[4:0]	I	rt输入
A3[4:0]	I	rd输入
WD[31:0]	I	寄存器文件写入数据
PC[31:0]	I	current pc
RD1[31:0]	O	读寄存器文件时寄存器rs的输出
RD2[31:0]	O	读寄存器文件时寄存器rt的输出

2. NPC

信号名	方向	描述
PC[31:0]	I	当前PC值
IMM26[25:0]	I	j型指令的26位立即数
IMM16[15:0]	I	b型指令的16位立即数
IMM32[31:0]	I	jr型指令的寄存器数据
NPCOP[2:0]	I	npc控制指令
Zero[4:0]	I	第0位beq信号，1为跳转，0为pc+4；第一位bgez信号，1为跳转，0为pc+4
NPC[31:0]	O	下一个PC的值
PCA4[31:0]	O	当前PC+4

功能定义

OP值	功能
000	pc+4

OP值	功能
001	beq
010	jal
011	jr
100	bgez

3. ALU

信号名	方向	描述
A[31:0]	I	第一个操作数
B[31:0]	I	第二个操作数
SHAMT[4:0]	I	移位信号
ALUOP[3:0]	I	ALU功能控制
Zero[4:0]	O	第0位beq信号，1为跳转，0为pc+4；第一位bgez信号，1为跳转，0为pc+4
ALUOUT[31:0]	O	ALU计算结果

功能定义

OP值	功能
0000	A+B
0001	A-B
0010	A|B
0011	A&B
0100(lui)	B<<16
0101(SLL)	B<< shamt
0110(SLTI)	若A小于B，则输出1，否则0

4.Controller

信号名	方向	描述
OPCODE[5:0]	I	指令opcode
FUNC[31:0]	I	指令func
NPCOP[3:0]	O	npc功能控制
RFWE	O	grf写使能
EXTOP[1:0]	O	扩展控制
DMWE	O	数据文件写使能
RFA3MUX[2:0]	O	grfA3控制
RFWDMUX[2:0]	O	grf写入控制
ALUBMUX[2:0]	O	ALUB端控制

信号名	方向	描述
ALUAMUX[2:0]	O	ALUA端控制

信号控制矩阵

指令	指令类型	指令码	NPCOP	RFWE	ALUOP	EXTOP	DMWE	RFA3MUX	RFWDMUX	ALUBMUX
ADDU	R	100001	PC+4	1	ADD(0000)	XX	0	000	000	000
SUBU	R	100011	PC+4	1	SUB(0001)	XX	0	000	000	000
ORI	I	001101	PC+4	1	ori(0010)	01	0	001	000	001
LW	I	100011	PC+4	1	ADD	00	0	001	001	001
SW	I	101011	PC+4	0	ADD	00	1	XXX	XXX	001
BEQ	I	000100	BEQ(001)	0	SUB	XX	0	XXX	XXX	000
LUI	I	001111	PC+4	1	LUI	01	0	001	000	001
SLL	R	000000	PC+4	1	SLL(0101)	XX	0	000	000	000
J	J	000010	010	0	XXXX	XX	0	XXX	XXX	XXX
JAL	J	000011	010	1	XXXX	XX	0	010	010	XXX
JR	R	001000	011	0	XXXX	XX	0	XXX	XXX	XXX
ADDIU	I	001001	PC+4	1	ADD	00	0	001	000	001
BGEZ	I	000001	100	0	XXXX	00	0	XXX	XXX	XXX
JALR	R	001001	011	1	XXXX	XX	0	000	010	XXX
SLTI	I	001010	PC+4	1	0110	00	0	001	000	001

5.MUX

RFA3MUX

信号名	方向	描述
RFA3OP[2:0]	I	控制信号
rt[4:0]	I	指令的rt
rd[4:0]	I	指令的rd
A3[4:0]	O	grfA3的输入

选择	功能
000	rd
001	rt
010	$31

RFWDMUX

信号名	方向	描述
RFWDOP[2:0]	I	控制信号

信号名	方向	描述
ALUOUT[31:0]	I	ALU计算结果
DMOUT[31:0]	I	数据存储器的输出数据
PCA4[31:0]	O	PC+4
RFWD[31:0]	O	写入GRF的数据

选择	功能
000	ALUOUT
001	DMOUT
010	PC+4

ALUBMUX

信号名	方向	描述
ALUBOP[2:0]	I	控制信号
rt[31:0]	I	RT寄存器的值
IMM16[31:0]	I	经过EXT扩展后的16位立即数
ALUB[31:0]	O	ALUB的输入

选择	功能
000	rt
001	imm16(I型指令)

6. DM

信号名	方向	描述
Address[31:0]	I	数据存储地址
Input[31:0]	I	输入数据
clk	I	全局时钟
Reset	I	同步清零
DMWE	I	写使能
PC[31:0]	I	pc
Data[31:0]	O	输出数据

7. EXT

信号名	方向	描述
EXTOP[2:0]	I	选择信号
imm16[15:0]	I	16位立即数
shamt[4:0]	I	5位立即数
extout[31:0]	O	输出

选择	功能
00	符号扩展16-32
01	无符号扩展16-32

8. IFU(分为PC和IM)

- PC

信号名	方向	描述
npc[31:0]	I	下个PC值
reset	I	同步复位至0x0000_3000
clk	I	时钟
pc[31:0]	O	当前PC

- IM

信号名	方向	描述
Addr[23:0]	I	当前指令地址
instr[31:0]	O	当前指令

（三）重要机制实现方法

1. 跳转
   NPC模块和ALU模块协同工作支持指令beq的跳转机制。
   NPC模块内置了判定单元和计算单元来独立支持指令J、JAL、JR的跳转机制。
2. nop
   nop即为sll $0, $0, 0,采用实现sll的方式实现nop

二、测试方案

（一）典型测试样例

1. ALU功能测试

```
lui $16, 0x2486
ori $16 0xff54
lui $12 0xffff
ori $12 0xffff
brach:
addu $16, $16, $12
subu $12, $12, $16
beq $0, $0, brach
```

2. DM功能测试

```
lui $16, 0x2486
ori $16 0xff54
lui $12 0xffff
ori $12 0xffff
sw $16, 0($0)
lw $12, 0($0)
```

3. 跳转测试

```
lui $16, 0x2486
ori $16 0xff54
lui $12 0xffff
ori $12 0xffff
li $a0, 1
li $a1, -100
jal loop
move $s6, $v0
li $t9, 3114
j no


loop:
move $t0, $a1
l1:
bgez $t0, end
addiu $t0, $t0, 1
j l1
end:
addiu $v0, $t0, 26
addiu $t0, $t0, 11
jr $ra
no:
```

（二）自动测试工具

讨论区白嫖

三、思考题

（一）根据你的理解，在下面给出的DM的输入示例中，地址信号addr位数为什么是[11:2]而不是[9:0]？这个addr信号又是从哪里来的?

答：因为DM的寄存器堆是按字来寻址，因此需要addr>>2,故应取addr的[11:2]位，输入地址信号addr为[11:2]而不是[9:0]能更好的体现这一特点。addr信号从ALU输出中来。

（二）思考Verilog语言设计控制器的译码方式，给出代码示例，并尝试对比各方式的优劣

答：第一种:根据指令对应的控制信号取值

```
    //NPCOP
    assign NPCOP[2] = 0|BGEZ;
    assign NPCOP[1] = 0|J|JAL|JR;
    assign NPCOP[0] = 0|BEQ|JR;

    //RFWE
    assign RFWE = 0|ADDU|SUBU|ORI|LW|LUI|SLL|JAL|ADDIU;

    //EXTOP
    assign EXTOP[1] = 0;
    assign EXTOP[0] = 0|ORI|LUI;

    //DMWE
    assign DMWE = 0|SW;

    //RFA3MUX
    assign RFA3MUX[2] = 0;
    assign RFA3MUX[1] = 0|JAL;
    assign RFA3MUX[0] = 0|ORI|LW|LUI|ADDIU;

    //RFWDMUX
    assign RFWDMUX[2] = 0;
    assign RFWDMUX[1] = 0|JAL;
    assign RFWDMUX[0] = 0|LW;

    //ALUBMUX
    assign ALUBMUX[2] = 0;
    assign ALUBMUX[1] = 0;
    assign ALUBMUX[0] = 0|ORI|LW|SW|LUI|ADDIU;

    //ALUOP
    assign ALUOP[3] = 0;
    assign ALUOP[2] = 0|LUI|SLL;
    assign ALUOP[1] = 0|ORI;
    assign ALUOP[0] = 0|SUBU|SLL;
```

第二种:控制信号每种取值所对应的指令

```
if(opcode==`ADDU) begin
    NPCOP <= xxx;
    RFWE<=X;
    EXTOP<=XX;
    DMWE<=X;
    RFA3MUX<=XXX;
    ALUBMUX<=XXX;
    ALUOP<=XXX;
    RFWD<=XXX;
end
else if (opcode==`SUBU) begin
   ................
   ................
end
```

（三）在相应的部件中，reset的优先级比其他控制信号（不包括clk信号）都要高，且相应的设计都是同步复位。清零信号reset所驱动的部件具有什么共同特点?

答：1.都是时序电路，存储的CPU当前的状态。
2.初始状态不确定，需要reset至规定初始状态。

（四）C语言是一种弱类型程序设计语言。C语言中不对计算结果溢出进行处理，这意味着C语言要求程序员必须很清楚计算结果是否会导致溢出。因此，如果仅仅支持C语言，MIPS指令的所有计算指令均可以忽略溢出。 请说明为什么在忽略溢出的前提下，addi与addiu是等价的，add与addu是等价的。提示：阅读《MIPS32® Architecture For Programmers Volume II: The MIPS32® Instruction Set》中相关指令的Operation部分 。

答：addiu和addi指令在数据通路中的行为几乎一样，区别只是在发生溢出时addi会抛出Overflow异常，而addiu不会。如果忽略溢出，则addi不会抛出异常，因此addiu和addi在忽略溢出的情况下等价。addu和add同理。

（五）根据自己的设计说明单周期处理器的优缺点。

答:优点:数据通路和控制单元设计简单,一条指令只需要考虑一条指令的行为。
缺点:速度慢，时钟周期取决于关键路径延迟最慢的那条指令。