实验五——单周期CPU设计

实验内容

使用Verilog设计实现单周期CPU。

文件内容

- ./实验报告.pdf：该实验的实验报告。
- ./code文件夹：储存实验相关代码的文件夹。
    - ./code/SingleCycle.v：单周期CPU。
    - ./code/alu.v：算术/逻辑运算模块。
    - ./code/ControllerUnit.v：控制器模块。
    - ./code/DataMemory.v：内存模块。
    - ./code/ProgramCounter.v：指令寄存器模块。
    - ./code/GeneralPurposeRegisters.v：寄存器模块。

实验分析

ArithmeticLogicUnit

该模块使用实验二设计的ALU模块。能够处理有（无）符号加减法、移位操作、位操作。

ControllerUnit

	31 30 29 28 27 26	25 24 23 22 21	20 19 18 17 16	15 14 13 12 11	10 9 8 7 6	5 4 3 2 1 0
ADDU / SUBU / JR	op	rs	rt	rd		ALUop
BEQ / LW / SW	op	rs	rt	imm		
LUI	op		rt	imm		
JAL / J	op	instr				

在控制器中输入一个二进制指令，然后输出各种信号，包括
$op, rs, rt, rd, ALUop, imm, instr$。各个信号的含义已经通过上图进行展示。

对于一个32位的指令，先取出其最高位作为操作类型。然后根据各个指令的形式进行归类，然后取出对应的部分进行作为控制器的输出。

ProgramCounter

该模块使用实验四设计的指令寄存器模块。对于指令寄存器（PC），在一般情况下每个时钟周期自加4个字节，在jumpEnabled标识设置位1的情况下进行跳转操作。

GeneralPurposeRegisters

对于寄存器的读取采用组合逻辑，即任意时刻都能进行读取。对寄存器的重置与写入仅在时钟周期的上升沿有效。

使用32个reg模拟寄存器，每个寄存器可以储存32个二进制位。读取的时候，按照输入的两个寄存器的编号，将对应寄存器的内容返回到对应的输出接口上。写入时，设置always模块对时钟上升沿敏感，此时将需要写入的数据写入到对应的寄存器中。

DataMomory

该模块使用实验四设计的内存模块。对于内存，支持读内存、写内存和重置。由于测试程序使用的内存略微超过$4KB$，故使用$8KB$内存，即2048×4。

InstructionMemory

使用initial在TopLevelUnit的开头读入MIPS汇编程序的机器码。

TopLevelUnit

- ADDU

  从寄存器模块中读出所需要的内容，然后通过ALU计算结果，再次使用寄存器模块将数据写回寄存器。

  需要允许寄存器写入、禁止内存写入、禁止PC进行地址跳转。

- SUBU

  从寄存器模块中读出所需要的内容，然后通过ALU计算结果，再次使用寄存器模块将数据写回寄存器。

  需要允许寄存器写入、禁止内存写入、禁止PC进行地址跳转。

- ORI

  从寄存器模块读取需要的数据，然后将16位立即数扩展成32位，然后使用ALU进行按位或运算，然后将结果储存到寄存器中。

  需要允许寄存器写入、禁止内存写入、禁止PC进行地址跳转。

- LW

从寄存器模块中获取base地址，然后使用ALU计算需要读取的内存地址，最后将从内存读到的内容存到寄存器中。需要注意的是，要进行读取的相对地址（16位地址）需要进行符号扩展，扩展为32位。

需要允许寄存器写入、禁止内存写入、禁止PC进行地址跳转。

- SW

  从寄存器模块中获取base地址，然后使用ALU计算需要读取的内存地址，最后将从寄存器读到的内容存到内存中。需要注意的是，要进行写入的相对地址（16位地址）需要进行符号扩展，扩展为32位。

  需要禁止寄存器写入、允许内存写入、禁止PC进行地址跳转。

- BEQ

  从寄存器模块中取出需要的两个值，然后使用if语句判断其是否相等。如果相等，使用ALU计算需要跳转到的地址，然后允许PC进行地址跳转。否则禁止PC进行地址跳转。

  需要禁止寄存器写入、禁止内存写入。

- LUI

  使用ALU进行左移操作，将指令中的16位二进制数左移16位得到一个32位整数，使用寄存器模块进行写入。

  需要允许寄存器写入、禁止内存写入、禁止PC进行地址跳转。

- JAL

  将第31号寄存器的值标记为PC+4（返回地址），然后使用ALU对读取到的需要跳转的地址进行左移（4对齐），然后扩展成32位，进行跳转。

  需要禁止寄存器写入、禁止内存写入、允许PC进行地址跳转。

- JR

  从寄存器模块读取需要跳转的地址，然后进行跳转。

  需要禁止寄存器写入、禁止内存写入、允许PC进行地址跳转。

- J

  使用ALU对读取到的需要跳转的地址进行左移（4对齐），然后扩展成32位，进行跳转。

  需要禁止寄存器写入、禁止内存写入、允许PC进行地址跳转。

实验结果

（略）

运行结果将在上机课上进行检查。