MIPS模拟仿真实验

学号:2110951 姓名:梁晓储

前言

本次实验要求实现一个MIPS仿真软件，能够实现根据接收的MIPS的机器码，通过模拟寄存器/内存的读取操作实现基本指令的功能

实验流程：

- 将/inputs中的含有MIPS汇编的.s文件转换为含有机器码.x文件
- 实现/src/sim.c中的process_instruction函数
- 调用make得到的`sim`，测试其功能

使用方法：

```
$ cd src
$ make
$ cd ..
$ ./src/sim ./inputs/file.x
```

Sim.c

指令介绍

根据不同的功能，我们将指令分为三种类型：

- I型指令：I型指令通常用于实现立即数运算和数据传输等操作，其格式为op $t, $s, imm，其中op表示操作码，$t和$s表示目标寄存器和源寄存器，imm表示一个16位的立即数。常见的I型指令包括addi、lw、sw、andi、ori等。例如，addi $t0, $s0, 100的机器码为0x21080064
- R型指令：R型指令通常用于实现寄存器之间的运算和移位等操作，其格式为op $d, $s, $t，其中op表示操作码，$d、$s和$t表示目标寄存器、源寄存器和第二个源寄存器。常见的R型指令包括add、sub、and、or、sll、srl等。例如，add $t0, $s0, $s1的机器码为0x02118020
- J型指令：J型指令通常用于实现无条件跳转操作。常见的J型指令包括j、jal等。例如，j 0x00400014的机器码为0x0800000d

指令格式	字段	描述	作用和功能	位数范围
I型指令	op	操作码	指定该指令的具体操作类型	31-26
	rs	源寄存器号	指定源寄存器的编号	25-21
	rt	目标寄存器号	指定目标寄存器的编号	20-16
	imm	立即数	一个16位的立即数，用于运算或传输等操作	15-0

指令格式	字段	描述	作用和功能	位数范围
J型指令	op	操作码	指定该指令的具体操作类型	31-26
	addr	跳转地址	一个26位的跳转地址，用于实现无条件跳转等操作	25-0
R型指令	op	操作码	指定该指令的具体操作类型	31-26
	rs	源寄存器号	指定源寄存器的编号	25-21
	rt	第二个源寄存器号	指定第二个源寄存器的编号	20-16
	rd	目标寄存器号	指定目标寄存器的编号	15-11
	shamt	移位位数	一个5位的位移量，用于实现移位操作等	10-6
	funct	函数码	一个6位的函数码，用于指定具体的操作类型（如加法、逻辑运算等）	5-0

编写思路

1. 先对指令进行解析，得到`opcode`
2. 根据`opcode`将指令分为三类，并在每一类的分支进行switch操作，模拟对应指令
3. 完善switch语句，对其进行细化
4. 完善更新PC值操作

代码示例

首先进行指令解码，通过掩码和右移操作将指令中的 rs、rt 和 imm 字段提取出来，并存储到对应的变量中

```
uint32_t rs = (instruction & 0x03E00000) >> 21;
uint32_t rt = (instruction & 0x001F0000) >> 16;
int16_t imm = instruction & 0xFFFF;
uint32_t address = CURRENT_STATE.REGS[rs] + (int32_t)imm;
```

接下来，根据 opcode 的值进行 switch 语句的判断

```
switch (opcode) {
    case 0x1: // BGEZ or BLTZAL
        if(rt == 0x1) {
            if((int32_t)CURRENT_STATE.REGS[rs] >= 0) {
                NEXT_STATE.PC = CURRENT_STATE.PC + 4 + (imm << 2);
```

```
            }
        } else if(rt == 0x10) { // BLTZAL
            NEXT_STATE.REGS[31] = CURRENT_STATE.PC + 4;
            if((int32_t)CURRENT_STATE.REGS[rs] < 0) {
                NEXT_STATE.PC = CURRENT_STATE.PC + 4 + (imm << 2);
            }
        }
        break;

    ……

}
```

如果 opcode 为 0x1，则为 BGEZ 或 BLTZAL 类型的指令。其中，对于 BGEZ 类型的指令，如果 rs 寄存器的值大于等于 0，则根据偏移量 imm 进行跳转，并更新 NEXT_STATE.PC。对于 BLTZAL 类型的指令，除了进行跳转外，还会将当前指令的下一条指令地址保存到寄存器 REGS[31] 中，并更新 NEXT_STATE.PC。判断条件是 rs 寄存器的值小于 0

```
    default:
        printf("Invalid instruction at %08x\n", CURRENT_STATE.PC);
        break;
}
```

default 分支用于处理无效指令。当 opcode 不匹配已有的 case 时，程序会执行 default 分支中的代码块。

在这个 default 分支中，会打印一条错误信息，指示出现了无效指令，并将当前指令的地址（PC）以十六进制形式输出到控制台。

输入文件处理

运用了qtspim将MIPS汇编转换为二进制机器码

在软件中导入.x文件，随后文件会将每行命令生成为机器码指令，由此便实现了MIPS汇编的.s文件向机器码.x文件的转化