验报告

实验名称（RISC-V 基本指令集模拟器设计与实现）

班级：计科 1605 学号：201608010631 姓名：刘梦函

实验目标

设计一个 CPU 模拟器，能模拟 CPU 指令集的功能。

实验要求

- 采用 C/C++编写程序
- 模拟器的输入是二进制的机器指令文件
- 模拟器的输出是 CPU 各个寄存器的状态和相关的存储器单元状态

实验内容

CPU 指令集

CPU 的指令集基本指令集共有 47 条指令。

模拟器程序框架

考虑到 CPU 执行指令的流程为：

1. 取指
2. 译码
3. 执行（包括运算和结果写回）

模拟器程序框架如下：

```
while(1) {
    inst = fetch(cpu.pc);
    cpu.pc = cpu.pc + 4;

    inst.decode();
```

```
	switch(inst.opcode) {
		case LUI:
			cout << "Do LUI" << endl;
			R[rd] = Imm31_12UtypeZeroFilled;
			break;
		case AUIPC:
			cout << "Do AUIPC" << endl;
			cout << "PC = " << PC << endl;
			cout << "Imm31_12UtypeZeroFilled = " << Imm31_12UtypeZeroFilled
<< enC + Imm31_12UtypeZeroFilled;
			break;
		case BRANCH:
			switch(funct3) {
				case BNE:
					cout << "Do BNE " << endl;
					if(src1!=src2){
						NextPC = PC + Imm12_1BtypeSignExtended;
					}
					break;
				default:
					cout << "ERROR: Unknown funct3 in BRANCH instruction " <<
IR << endl;
			}
			break;
		 case LOAD:
			switch(funct3) {
				case LH:
					cout << "Do LH " << endl;
					unsigned int temp_LH,temp_LH_UP;
```

```
                    temp_LH=readHalfWord(src1+Imm11_0ItypeSignExtended);
                    temp_LH_UP=temp_LH>>15;
                    if(temp_LH_UP==1){
                        temp_LH=0xffff0000 | temp_LH;
                    }else{
                        temp_LH=0x0000ffff & temp_LH;
                    }
                    R[rd]=temp_LH;
                    break;
                default:
                    cout << "ERROR: Unknown funct3 in LOAD instruction " << IR
<< endl;
            }
            break;
        case ALUIMM:
            switch(funct3) {
                case ADDI:
                    cout <<       "Do ADDI" << endl;
                    R[rd]=src1+Imm11_0ItypeSignExtended;
                    break;
                case SLLI:
                    cout << "Do SLLI " << endl;
                    R[rd]=src1<<shamt;
                    break；
                default:
                    cout << "ERROR: Unknown funct3 in ALUIMM instruction " <<
IR << endl;
            }
            break;
        case ALURRR:
```

```
                switch(funct3) {
                    case SLT:
                        cout << "Do SLT " << endl;
                        if((int)src1<(int)src2){
                            R[rd]=1;
                        }else{
                            R[rd]=0;
                        }
                        break;
                    default:
                        cout << "ERROR: Unknown funct3 in ALURRR instruction " <<
IR << endl;
                }
                break;
            default:
                cout << "无法识别的操作码：” << inst.opcode;
        }
}
```

测试

测试平台

部件	配置	备注
CPU	Intel(R) Core(TM) i5-6200U CPU (	
内存	DDR3 8GB	

操作系统	Windows 10	家庭中文版

模拟器在如下机器上进行了测试：

输入：

```
// Write memory with instructions to test
void m_progMem(){
    writeWord(0, (0x666 << 12) | (2 << 7) | (LUI));//指令功能在第2个寄存器写入0x666
    writeWord(4, (1 << 12) | (3 << 7) | (AUIPC));//指令功能在第3个寄存器中写入PC+0x1000
    writeWord(8, (0x66 << 12) | (5 << 7) | (LUI));//指令功能在第5个寄存器写入6
    writeWord(12, (0x0<<25) | (5<<20) | (0<<15) | (SW << 12) | (0x1a << 7) | (STORE));//向(0号寄存器的值加上0x1a)地址写入5号寄存器中的值
    writeWord(16, (0x10<<20) | (0<<15) | (LBU<<12) | (4<<7) | (LOAD));//读取0x10地址上的1byte取最后8位写入4号寄存器
    writeWord(20, (0x0<<25) | (2<<20) | (0<<15) | (BGE<<12) | (0x8<<7) | (BRANCH));//判断0号寄存器和2号寄存器值的大小，如果大于等于则修改NextPC为 PC
}
```

（最后一条指令的功能）//判断 0 号寄存器和 2 号寄存器值的大小，如果大于等于则修改 NextPC=PC +Imm12_1BtypeSignExtended;

第一条指令结果截图；

```
Registers bofore executing the instruction @0x0
PC=0x0 IR=0x0


M[0]=0x37 M[1]=0x61 M[2]=0x66 M[3]=0x0 M[4]=0x97 M[5]=0x11 M[6]=0x0 M[7]=0x0 M[8]=0xb7 M[9]=0x62 M[a]=0x6 M[b]
=0x23 M[d]=0x2d M[e]=0x50 M[f]=0x0 M[10]=0x3 M[11]=0x42 M[12]=0x0 M[13]=0x1 M[14]=0x63 M[15]=0x54 M[16]=0x20 M
M[18]=0x0 M[19]=0x0 M[1a]=0x0 M[1b]=0x0 M[1c]=0x0 M[1d]=0x0 M[1e]=0x0 M[1f]=0x0

R[0]=0x0 R[1]=0x0 R[2]=0x0 R[3]=0x0 R[4]=0x0 R[5]=0x0 R[6]=0x0 R[7]=0x0 R[8]=0x0 R[9]=0x0 R[a]=0x0 R[b]=0x0 R
d]=0x0 R[e]=0x0 R[f]=0x0 R[10]=0x0 R[11]=0x0 R[12]=0x0 R[13]=0x0 R[14]=0x0 R[15]=0x0 R[16]=0x0 R[17]=0x0 R[18
]=0x0 R[1a]=0x0 R[1b]=0x0 R[1c]=0x0 R[1d]=0x0 R[1e]=0x0 R[1f]=0x0

Do LUI
Registers after executing the instruction
PC=0x4 IR=0x666137


M[0]=0x37 M[1]=0x61 M[2]=0x66 M[3]=0x0 M[4]=0x97 M[5]=0x11 M[6]=0x0 M[7]=0x0 M[8]=0xb7 M[9]=0x62 M[a]=0x6 M[b]
=0x23 M[d]=0x2d M[e]=0x50 M[f]=0x0 M[10]=0x3 M[11]=0x42 M[12]=0x0 M[13]=0x1 M[14]=0x63 M[15]=0x54 M[16]=0x20 M
M[18]=0x0 M[19]=0x0 M[1a]=0x0 M[1b]=0x0 M[1c]=0x0 M[1d]=0x0 M[1e]=0x0 M[1f]=0x0

R[0]=0x0 R[1]=0x0 R[2]=0x666000 R[3]=0x0 R[4]=0x0 R[5]=0x0 R[6]=0x0 R[7]=0x0 R[8]=0x0 R[9]=0x0 R[a]=0x0 R[b]=0
x0 R[d]=0x0 R[e]=0x0 R[f]=0x0 R[10]=0x0 R[11]=0x0 R[12]=0x0 R[13]=0x0 R[14]=0x0 R[15]=0x0 R[16]=0x0 R[17]=0x0
 R[19]=0x0 R[1a]=0x0 R[1b]=0x0 R[1c]=0x0 R[1d]=0x0 R[1e]=0x0 R[1f]=0x0
```

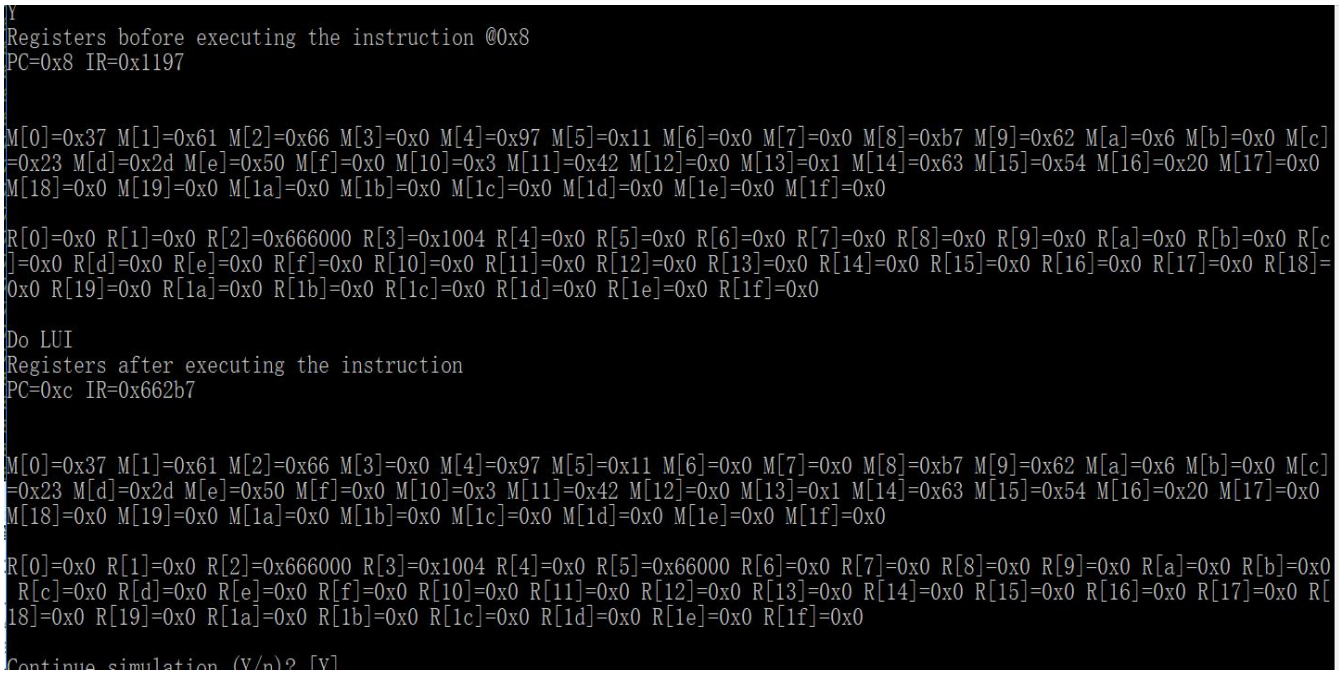
指令功能在第 2 个寄存器写入 0x666

第二条指令结果截图：

```
Registers bofore executing the instruction @0x4
PC=0x4 IR=0x666137


M[0]=0x37 M[1]=0x61 M[2]=0x66 M[3]=0x0 M[4]=0x97 M[5]=0x11 M[6]=0x0 M[7]=0x0 M[8]=0xb7 M[9]=0x62 M[a]=0x6 M[b]=0x0 M[c]
=0x23 M[d]=0x2d M[e]=0x50 M[f]=0x0 M[10]=0x3 M[11]=0x42 M[12]=0x0 M[13]=0x1 M[14]=0x63 M[15]=0x54 M[16]=0x20 M[17]=0x0
M[18]=0x0 M[19]=0x0 M[1a]=0x0 M[1b]=0x0 M[1c]=0x0 M[1d]=0x0 M[1e]=0x0 M[1f]=0x0

R[0]=0x0 R[1]=0x0 R[2]=0x666000 R[3]=0x0 R[4]=0x0 R[5]=0x0 R[6]=0x0 R[7]=0x0 R[8]=0x0 R[9]=0x0 R[a]=0x0 R[b]=0x0 R[c]=0
x0 R[d]=0x0 R[e]=0x0 R[f]=0x0 R[10]=0x0 R[11]=0x0 R[12]=0x0 R[13]=0x0 R[14]=0x0 R[15]=0x0 R[16]=0x0 R[17]=0x0 R[18]=0x0
 R[19]=0x0 R[1a]=0x0 R[1b]=0x0 R[1c]=0x0 R[1d]=0x0 R[1e]=0x0 R[1f]=0x0

Do AUIPC
PC = 4
Imm31_12UtypeZeroFilled = 1000
Registers after executing the instruction
PC=0x8 IR=0x1197


M[0]=0x37 M[1]=0x61 M[2]=0x66 M[3]=0x0 M[4]=0x97 M[5]=0x11 M[6]=0x0 M[7]=0x0 M[8]=0xb7 M[9]=0x62 M[a]=0x6 M[b]=0x0 M[c]
=0x23 M[d]=0x2d M[e]=0x50 M[f]=0x0 M[10]=0x3 M[11]=0x42 M[12]=0x0 M[13]=0x1 M[14]=0x63 M[15]=0x54 M[16]=0x20 M[17]=0x0
M[18]=0x0 M[19]=0x0 M[1a]=0x0 M[1b]=0x0 M[1c]=0x0 M[1d]=0x0 M[1e]=0x0 M[1f]=0x0

R[0]=0x0 R[1]=0x0 R[2]=0x666000 R[3]=0x1004 R[4]=0x0 R[5]=0x0 R[6]=0x0 R[7]=0x0 R[8]=0x0 R[9]=0x0 R[a]=0x0 R[b]=0x0 R[c
]=0x0 R[d]=0x0 R[e]=0x0 R[f]=0x0 R[10]=0x0 R[11]=0x0 R[12]=0x0 R[13]=0x0 R[14]=0x0 R[15]=0x0 R[16]=0x0 R[17]=0x0 R[18]=
0x0 R[19]=0x0 R[1a]=0x0 R[1b]=0x0 R[1c]=0x0 R[1d]=0x0 R[1e]=0x0 R[1f]=0x0

Continue simulation (Y/n)? [Y]
```

指令功能在第 3 个寄存器中写入 PC+0x1000

第三条指令结果截图：



指令功能在第 5 个寄存器写入 6

分析和结论

从测试记录来看，模拟器实现了对二进制指令文件的读入、指令功能的模拟，CPU 和存储器状态的输出。

根据分析结果，可以认为编写的模拟器实现了所要求的功能，完成了实验目标。