

实验报告

软件部分 RISC-V 基本指令集模拟器设计与实现

智能 1601 201508010326 永青达杰

实验任务

完成一个模拟 RISC-V 的基本整数指令集 RV32I 的模拟器设计

实验要求

软件设计采用 C/C++或 SystemC 语言

实验报告采用 markdown 语言，或者直接上传 PDF 文档

实验最终提交所有代码和文档

实验过程

模拟器的整体框架是首先设定好主要的参数的定义

包括操作码标识的基本指令，还有常用操作码标签加载，操作码

标签的存储, 及时的 alu 操作, 所有寄存器操作数的运算

其他主要的操作的话还有取指

```
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)写入0x1a
    writeWord(16, (0x10<<20) | (0<<15) | (LBU<<12) | (4<<7) | (LOAD)); //读取0x10地址上的1byte数据
    writeWord(20, (0x0<<25) | (2<<20) | (0<<15) | (BGE<<12) | (0x8<<7) | (BRANCH)); //判断0号寄存器是否大于等于8号寄存器
}
```

这段主要就是我们要使用的指令了，取指操作也就对应他完成了

```
void decode(uint32_t instruction) { //decode是译码的意思, RV32I指令4个字节
```

还有这个部分，译码的操作也是必不可少的部分了

最后的任务肯定就是打印了

```

void show32Mess(){
    cout << endl << endl;
    for(int i=0; i<32; i++) {
        char tp = M[i];
        cout << "M[" << i << "]=0x" << ((unsigned int)tp&0x000000ff)<<" ";
    }
    cout << endl << endl;
}

void showRegs() {
    cout << "PC=0x" << std::hex << PC << " " << "IR=0x" << std::hex << IR << endl;
    show32Mess();
    for(int i=0; i<32; i++) {
        cout << "R[" << i << "]=0x" << std::hex << R[i] << " ";
    }
    cout << endl<<endl;
}

```

这两个函数，前者打印 32 个寄存器的值，后者打印 pc 的值，当然也是会打印寄存器的值的

```

while(c != 'n') {
    cout << "Registers before executing the instruction @0x" << std::hex << PC << endl;
    showRegs();
    IR = readWord(PC);
    NextPC = PC + WORDSIZE;
    decode(IR);
}

```

然后就是在主函数里面，上面的判断这个 while 说明了主要是靠循环来完成操作，如果执行命令按 n 表示否定的话，就跳出循环，模拟结束

实验测试

接下来就是测试环节了

测试平台

如果

部件	配置
CPU	Core i5
内存	4G
操作系统	Windows 10

测试的指令

```
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
    writeWord(16, (0x10 << 20) | (0 << 15) | (LBU << 12) | (4 << 7) | (LOAD)); //读取0x10地址上的1byte取最后8位
    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
}
```

第一条指令 在第 2 个寄存器写入 0x666

```
writeWord(0, (0x666 << 12) | (2 << 7) | (LUI));
```

```
Registers before 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]=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]=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[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

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]=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]=0x
0 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
```

第二条指令 在第 3 个寄存器中写入 PC+0x1000

```
Registers before 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]=0x
0 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]=0x
0 R[19]=0x0 R[1a]=0x0 R[1b]=0x0 R[1c]=0x0 R[1d]=0x0 R[1e]=0x0 R[1f]=0x0
```

第三条指令 在第 5 个寄存器写入 6

```

Registers before executing the instruction @0x8
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]=0x
0 R[19]=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=0xc IR=0x662b7

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]=0x66000 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

```

第四条指令 向(0号寄存器的值加上 0x1a)地址写入 5 号寄存器中的值

```

Registers before executing the instruction @0xc
PC=0xc IR=0x662b7

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]=0x66000 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

DO SW
SW Addr and Data are: 1a, 66000
Registers after executing the instruction
PC=0x10 IR=0x502d23

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]=0x60 M[1c]=0x6 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]=0x66000 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

```

第五条指令 读取 0x10 地址上的 1byte 取最后 8 位写入 4 号寄存器

```

Registers before executing the instruction @0x10
PC=0x10 IR=0x502d23

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]=0x60 M[1c]=0x6 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]=0x66000 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

Do LBU
Registers after executing the instruction
PC=0x14 IR=0x1004203

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]=0x60 M[1c]=0x6 M[1d]=0x0 M[1e]=0x0 M[1f]=0x0

R[0]=0x0 R[1]=0x0 R[2]=0x666000 R[3]=0x1004 R[4]=0x3 R[5]=0x66000 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

```

第六条指令 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 + Imm12_1BtypeSignExtended;


```

Registers before executing the instruction @0x14
PC=0x14 IR=0x1004203

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]=0x60 M[1c]=0x6 M[1d]=0x0 M[1e]=0x0 M[1f]=0x0

R[0]=0x0 R[1]=0x0 R[2]=0x666000 R[3]=0x1004 R[4]=0x3 R[5]=0x66000 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

Do BGE
src1为 0
src2为 666000
imm为 8
Registers after executing the instruction
PC=0x18 IR=0x205463

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]=0x60 M[1c]=0x6 M[1d]=0x0 M[1e]=0x0 M[1f]=0x0

R[0]=0x0 R[1]=0x0 R[2]=0x666000 R[3]=0x1004 R[4]=0x3 R[5]=0x66000 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

```

实验总结

归根究底，还是要明白取值后译码的部分的最后呈现，还有寄存器的运算操作的结果之类的，主要还是理解好程序，然后不断测试就容易搞懂了