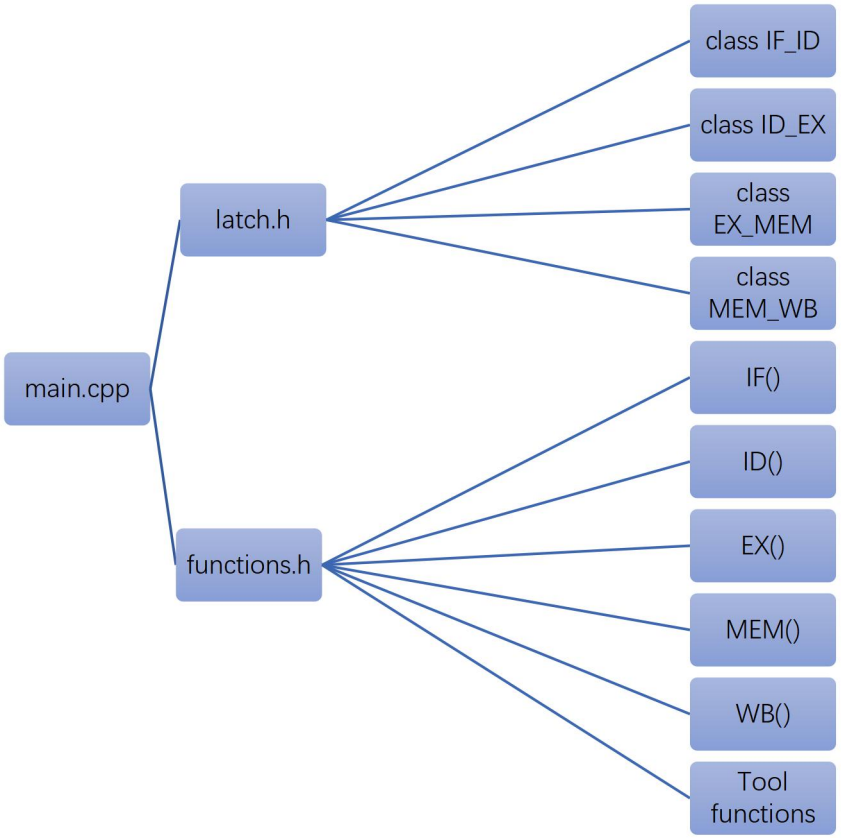
Project3 report

Xinyang Mao

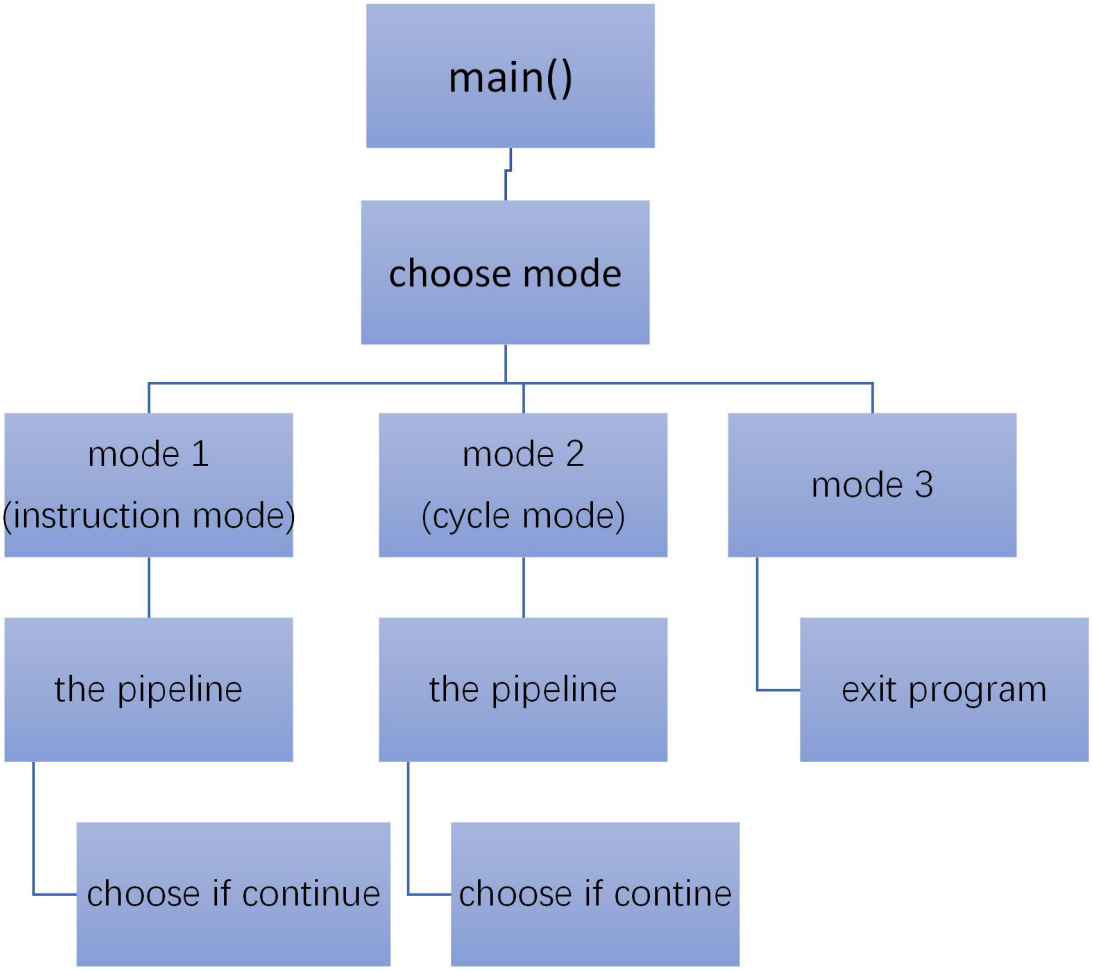
structure of this report

1) program structure
2) hazards detect
3) problems and summary

1. Program structure



1) Main.cpp



Main()

Initial instruction array

```
fill(ins, ins + 512, "");//initial ins[]

//choose mode
cout << "choose the mode" << endl;
int mode;//the choosed mode
cout << "1.instruction mode" << endl << "2.cycly mode" << endl << "3.exit" << endl;
cin >> mode;//input mode;

string filename;

string inputins;
ifstream fin;
int ins_counter = 0;
//vector<string> ins;
```

- Choose mode

1.If choose mode 1,input file and choose how many instructions want to execute.

```
//vector<string> ins;
switch (mode)
{
case 1:
    cout << "you choose mode 1:instruction" << endl;


    cout << "input filename:";
    cin >> filename;
    cout << "how many instruction you want to run:" << endl;
    cin >> instruction_number;

    //ifstream fin;//create stream object.
    fin.open(filename, ios::in);//open file
    if (!fin.is_open()) //judge if successful
    {
        cout << "fail to open file" << endl;

    }
    else
    {
        cout << "file opened successfully" << endl;
    }

    while (getline(fin, inputins))
    {


        //ins.push_back(inputins);

        ins[ins_counter] = inputins;

        cout << "ins[" << ins_counter << "]" << ins[ins_counter] << endl;
        ins_counter++;
    }

    break;
```

2.If choose mode 2(cycle mode),input file and input how many cycle you want to execute.

```
case 2:
    cout << "you choose mode 2:cycle" << endl;
    //readfile(filename);//read instructions from file;
    //string inputins;//


    cout << "input filename:" << endl;
    cin >> filename;
    cout << "how many cycle do you want to run:" << endl;
    cin >> cycle_number;
    //ifstream fin;//create stream object.
    fin.open(filename, ios::in);//open file
    if (!fin.is_open()) //judge if successful
    {
        cout << "fail to open file" << endl;

    }
    else
    {
        cout << "file opened successfully" << endl;
    }

    while (getline(fin, inputins))
    {


        //ins.push_back(inputins);

        ins[ins_counter] = inputins;

        cout << "ins[" << ins_counter << "]" << ins[ins_counter] << endl;
        ins_counter++;
    }

    break;
```

3.If choose mode 3(exit)

```
    case 3:
        cout << "you choose mode 3:exit;" << endl;
        cout<<"program finished"<<endl;
        break;
    default:
        cout << "invalid input" << endl;
        break;
```

- Pipeline

Call 5 stage functions reversely we needn't to define addition temporary variable to achieve the pipeline.

For instruction mode, input the number of how many instructions want to execute.

insnum is a variable to count how many time the WB() execute validly to represent how many instruction have executed. And the clock_WB is the variable to count how many cycles had execute.

```
if (mode == 1) //instruction mode
{
    while (instruction_number!= insnum)
    {
        WB();
        MEM();
        EX();
        ID();
        IF(ins);
    }
}
else if (mode == 2)//cycle mode
{
    while (cycle_number != clock_WB)
    {
        WB();
        MEM();
        EX();
        ID();
        IF(ins);

    }

}
else
{
    cout<<"exit"<<endl;
}
```

- Continue()

Input the Y/N to choose if continue to execute program. And if the first time choose mode 1,so will continue instruction mode, else will continue cycle mode.

```
//continue
string continue_answer;
cout << "continue?" << endl << "please input Y/N" << endl;
cin >> continue_answer;
if (continue_answer=="Y")
{
    if (mode==1)
    {
        cout <<"please input the number of instructions want to execute:"<< endl;
        cin >> instruction_number;
        while (instruction_number != insnum)
        {
            WB();
            MEM();
            EX();
            ID();
            IF(ins);
        }
    }
    else if (mode==2)
    {
        cout << "please input the number of cycle want to execute:" << endl;
        cin >> cycle_number;
        while (cycle_number != clock_WB)
        {
            WB();
            MEM();
            EX();
            ID();
            IF(ins);

        }
    }


    //  MEM_WB
    cout << "-------------------MEM_WB----------------------" << endl;
    cout << "IR=" << mem_wb.getIR() << endl;
    cout << "ALUoutput=" << mem_wb.getALUoutput() << endl;
    cout << "LMD=" << mem_wb.getLMD() << endl;
```

Display the result

After pipeline will display the result: the content of every latch to show the status of every stage, the clock number have execute and the utilization of every stage.

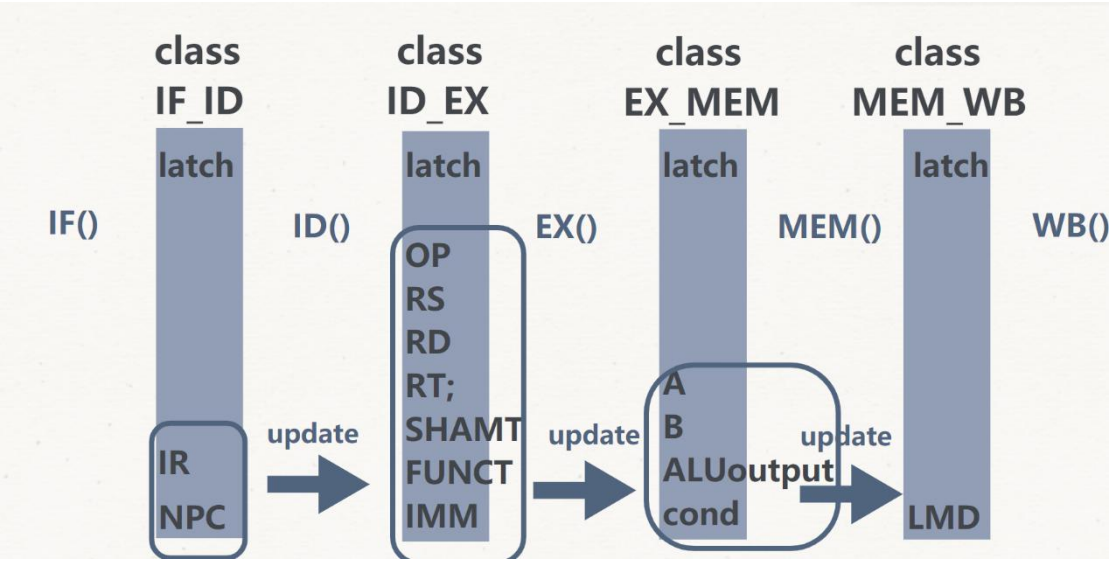
```
//  MEM_WB
cout << "------------------MEM_WB----------------------" << endl;
cout << "IR=" << mem_wb.getIR() << endl;
cout << "ALUoutput=" << mem_wb.getALUoutput() << endl;

cout << "LMD=" << mem_wb.getLMD() << endl;

cout << "rt=" << mem_wb.getrt() << endl;
cout << "RT=" << mem_wb.getRT() << endl;

cout << "------------------EX_MEM----------------------" << endl;
cout << endl;
cout << "IR=" << ex_mem.getIR() << endl;;
cout << "NPC=" << ex_mem.getNPC() << endl;
cout << "imm=" << ex_mem.getimm() << endl;
cout << "A=" << ex_mem.getA() << endl;//rs
cout << "B=" << ex_mem.getB() << endl;//rt
cout << "ALUoutput=" << ex_mem.getALUoutput() << endl;
cout << "cond=" << ex_mem.getcond() << endl;
cout << "rt=" << ex_mem.getrt() << endl;//unsigned long int rt;
cout << endl;
cout << "------------------ID_EX----------------------" << endl;

cout << "IR=" << id_ex.getIR() << endl;
cout << "NPC=" << id_ex.getNPC() << endl;
cout << "OP=" << id_ex.getOP() << endl;//string OP;
cout << "RS=" << id_ex.getRS() << endl;//string RS;
cout << "RD=" << id_ex.getRD() << endl;//string RD;
cout << "RT=" << id_ex.getRT() << endl;//string RT;
cout << "SHAMT=" << id_ex.getSHAMT() << endl;//string SHAMT;
cout << "FUNCT=" << id_ex.getFUNCT() << endl;//string FUNCT;
cout << "IMM=" << id_ex.getIMM() << endl;//string IMM;
cout << "rs=" << id_ex.getrs() << endl;//unsigned long int rs;//十进制
cout << "rt=" << id_ex.getrt() << endl;//unsigned long int rt;
cout << "rd=" << id_ex.getrd() << endl;//unsigned long int rd;
cout << "imm=" << id_ex.getimm() << endl;//unsigned long int imm;
cout << endl;
```

```
    cout << "------------------IF_ID----------------------" << endl;
    cout << "PC=" << IF_ID::PC << endl;//static unsigned long int PC;//PC
    cout << "IR=" << if_id.getIR() << endl;//string IR;
    //cout << "NPC=" << if_id.getNPC() << endl;//unsigned long int NPC;
    cout << endl;

    cout << "------------------Reg----------------------" << endl;
    cout << endl;
    //display Reg
    for (int i = 0; i < 32; i++)
    {
        cout << "R[" << i << "]=" << IF_ID::R[i] << endl;

    }
    cout << endl;

    cout << "------------------datamemory----------------------" << endl;
    //display datamemory
    cout << endl;
    for (int i = 0; i < 512; i++)
    {
        cout << "d[" << i << "]=" << datamemory[i] << " ";//循环输出一维数组
        if ((i + 1) % 16 == 0)//做判定条件
        {
            cout << endl;
            cout << endl;
        }
    }
    cout << endl;
```

```
//display utilization
cout << "invalid_WB=" << invalid_WB << endl;
cout << "invalid_MEM=" << invalid_MEM << endl;
cout << "invalid_EX=" << invalid_EX << endl;
cout << "invalid_ID=" << invalid_ID << endl;
cout << "invalid_IF=" << invalid_IF << endl;

cout << endl;
cout << "clock_WB=" << clock_WB << endl;
cout << "clock_MEM=" << clock_MEM << endl;
cout << "clock_EX=" << clock_EX << endl;
cout << "clock_ID=" << clock_ID << endl;
cout << "clock_IF=" << clock_IF << endl;

//cout << "nop_IF=" << nop_IF << endl;

cout << endl;
cout << "U_WB=" << ((float)(clock_WB-invalid_WB) / clock_WB) * 100 << "%" << endl;
cout << "U_MEM=" << ((float)(clock_WB - invalid_MEM) / clock_MEM) * 100 << "%" << endl;
cout << "U_EX=" << ((float)(clock_WB - invalid_EX) / clock_EX) * 100 << "%" << endl;
cout << "U_ID=" << ((float)(clock_WB - invalid_ID) / clock_ID) * 100 << "%" << endl;
cout << "U_IF=" << ((float)(clock_WB - invalid_IF) / clock_IF) * 100 << "%" << endl;
```

2) Latch.h

Created 4 latch classes to represent 4 latches between 5 stages and achieve the class functions both in latch.h file. Except R[32]and PC, all the latches were set as private member and define public member functions : get() and set() as the interface to set value and get value from latches to avoid the value in latches easy be changed.



class IF_ID (represent the latch between fetch stage and decode stage)

```
class IF_ID
{
private://latch of IF_ID,ID_EX,EX_MEM,MEM_WB.
    string IR;
    unsigned long int NPC;
public:
    static unsigned long int R[32];//32 Rrg
    static unsigned long int PC;//PC
    //long type:32 bits
    //latchReg() {};
    IF_ID() :
        IR(""),///*/*00000000000000000000000000000000*/*/
        NPC(0)
    {}
    ~IF_ID(){}

    void setIR(string ir)//IR
    {
        IR = ir;
    }
    string getIR()
    {
        return IR;
    }

    void setNPC(unsigned long int npc)//NPC
    {
        NPC = npc;

    }
    unsigned long int getNPC()
    {
        return NPC;
    }

};
//<静态变量的类型> <类名>::<类的静态成员变量名> <= 初始化值 >
unsigned long int IF_ID::R[32] = {};
unsigned long int IF_ID::PC = 0;
```

class ID_EX

```
class ID_EX
{
private://latch of IF_ID,ID_EX,EX_MEM,MEM_WB.
    string IR;
    unsigned long int NPC;
    unsigned long int imm;
    unsigned long int A;//rs
    unsigned long int B;//rt
    unsigned long int rs;//十进制
    unsigned long int rt;
    unsigned long int rd;
    string OP;
    string RS;
    string RD;
    string RT;
    string SHAMT;
    string FUNCT;
    string IMM;
public:

    ID_EX() :
        IR(""),///*/*00000000000000000000000000000000*/*/
        OP(""),
        RS(""),
        RD(""),
        RT(""),
        SHAMT(""),
        FUNCT(""),
        IMM(""),
        NPC(0),
        imm(0),
        A(0),
        B(0),
        rs(0),//十进制
        rt(0),
        rd(0)
    {
    }
    ~ID_EX() {}

    void setIR(string ir)//IR
    {
        IR = ir;
```

Class EX_MEM

```
class EX_MEM
{
private://latch of IF_ID,ID_EX,EX_MEM,MEM_WB.
    string IR;
    unsigned long int NPC;
    unsigned long int imm;
    unsigned long int A;//rs
    unsigned long int B;//rt
    unsigned long int ALUoutput;
    unsigned long int cond;
    unsigned long int rs;//十进制
    unsigned long int rt;
    unsigned long int rd;
    string OP;
    string RS;
    string RD;
    string RT;
    string SHAMT;
    string FUNCT;
    string IMM;
public:

    EX_MEM() :
        IR(""),///*/*00000000000000000000000000000000*/*/
        OP(""),
        RS(""),
        RD(""),
        RT(""),
        SHAMT(""),
        FUNCT(""),
        IMM(""),
        NPC(0),
        imm(0),
        A(0),
        B(0),
        ALUoutput(0),
        cond(0),
        rs(0),//十进制
        rt(0),
        rd(0)
    {
    }
    ~EX_MEM() {}

    void setIR(string ir)//IR
    {
```

Class MEM_WB

```
class MEM_WB
{
private://latch of IF_ID, ID_EX, EX_MEM, MEM_WB.
    string IR;
    unsigned long int NPC;
    unsigned long int imm;
    unsigned long int A;//rs
    unsigned long int B;//rt
    unsigned long int ALUoutput;
    unsigned long int cond;
    unsigned long int LMD;
    unsigned long int rs;//十进制
    unsigned long int rt;
    unsigned long int rd;
    string OP;
    string RS;
    string RD;
    string RT;
    string SHAMT;
    string FUNCT;
    string IMM;
public:
    MEM_WB() :
        IR(""),///*/*00000000000000000000000000000000*/*/
        OP(""),
        RS(""),
        RD(""),
        RT(""),
        SHAMT(""),
        FUNCT(""),
        IMM(""),
        NPC(0),
        imm(0),
        A(0),
        B(0),
        ALUoutput(0),
        cond(0),
        LMD(0),
        rs(0),//十进制
        rt(0),
        rd(0)
    {
    }

    ~MEM_WB() {}

    void setIR(string ir)//IR
```

3) Functions.h

5 functions to simulate 5 pipeline stages and some tool functions will be use in stages were Stated in functions.h file.

- IF()

In this stage, will fetch a instruction from instruction memory than store it in IF_ID class. Except that will update the PC and NPC.

```
//IF()
void IF(string ins[])
{
    if_id.setIR(ins[IF_ID::PC]);
    string s= if_id.getIR();
    if (s != "")
    {
        if (beq_flag == 1)//last instruction is beq
        {
            //stall
        }
        else if(beq_flag==0)
        {
            if (ex_mem.getcond()==1)
            {
                ex_mem.setcond(0);
                IF_ID::PC = ex_mem.getNPC();
                if_id.setIR(ins[IF_ID::PC]);
            }
            else//cond=0
            {
                IF_ID::PC++;
                if_id.setNPC(IF_ID::PC);

            }
        }
    }
    else
    {
        invalid_IF++;
    }

    clock_IF++;
}
```

- ID()

  This stage will use substr() to cut a string type instruction into several part and store them into the latches and use function to change string type data to int type decimal number. And do hazards detect operate.

```
//ID()
void ID()
{

    //
    id_ex.setNPC(if_id.getNPC());
    id_ex.setIR(if_id.getIR());
        string s = "0";
        string s1 = "0";
        s = if_id.getIR();//current instruction
        if ((s!="")&&(data_flag==0))
        {
            s1 = s.substr(0, 6);
            id_ex.setOP(s1);//OP
            if (id_ex.getOP() == "000000")//R type
            {
                string s2 = "0";
                s2 = s.substr(6, 5);//RS
                id_ex.setRS(s2);
                id_ex.setrs(bin_to_dec(id_ex.getRS()));//把RS转为rs
                //id_ex.setA(IF_ID::R[id_ex.getrs()]);//将R[rs]存入A

                s2 = s.substr(11, 5);//RT
                id_ex.setRT(s2);
                id_ex.setrt(bin_to_dec(id_ex.getRT()));//把RT转为rt
                //id_ex.setB(IF_ID::R[id_ex.getrt()]);//将R[rt]存入B


                s2 = s.substr(16, 5);//RD
                id_ex.setRD(s2);
                id_ex.setrd(bin_to_dec(id_ex.getRD()));//把RD转为rd


                s2 = s.substr(21, 5);//SHAMT
                id_ex.setSHAMT(s2);

                s2 = s.substr(26, 6);//FUNCT
                id_ex.setFUNCT(s2);

                s2 = "0";//IMM寄存器清零
                id_ex.setIMM(s2);
            }
            else//I type
```

```
            s2 = "0";//IMM寄存器清零
            id_ex.setIMM(s2);
        }
        else//I type
        {
            string s3 = "0";
            s3 = s.substr(6, 5);//RS
            id_ex.setRS(s3);
            id_ex.setrs(bin_to_dec(id_ex.getRS()));//把RS转为rs
            //id_ex.setA(IF_ID::R[id_ex.getrs()]);//将R[rs]存入A

            s3 = s.substr(11, 5);//RT
            id_ex.setRT(s3);
            id_ex.setrt(bin_to_dec(id_ex.getRT()));

            s3 = s.substr(16, 16);//IMM
            id_ex.setIMM(s3);
            id_ex.setimm(bin_to_dec(id_ex.getIMM()));//IMM转dec imm

            s3 = "0";//其他寄存器清零
            id_ex.setRD(s3);
            id_ex.setSHAMT(s3);
            id_ex.setFUNCT(s3);

        }

        //hazard detect


            if (id_ex.getOP() == "000100")//beq
            {
                beq_flag = 1;//control hazard

            }
            else
            {
                if (id_ex.getOP() == "000000")//R type
                {
                    if ((id_ex.getRS() == ex_mem.getRD() || id_ex.getRT() == ex_mem.getRD()))
                        //||((id_ex.getRS() == mem_wb.getRD() || id_ex.getRT() == mem_wb.getRD()))
                        //)

                    {

                        if ((id_ex.getRS() == ex_mem.getRD() || id_ex.getRT() == ex_mem.getRD()))
                            //||((id_ex.getRS() == mem_wb.getRD() || id_ex.getRT() == mem_wb.getRD()))
                            //)

                        {
                            //execute stall
                            id_ex.setIR("");
                            IF_ID::PC--;
                            data_flag = 1;
                        }

                    }
                    else//I type
                    {
                        if ((id_ex.getOP() != "101011"))//I type except sw
                        {

                            if ((id_ex.getRS() == ex_mem.getRT()))
                                //||((id_ex.getRS() == mem_wb.getRT()))(no)


                            {
                                id_ex.setIR("");
                                IF_ID::PC--;
                                data_flag = 1;
                            }

                        }

                    }
                }
    }
    else
    {
    invalid_ID++;
    }



clock_ID++;
}
```

- EX()

Execute stage main will doing add, sub, addi, mul, lw, sw, beq, lui, and, andi, or, ori, sll, srl, slti, and sltiu operates.

Firstly, judge the instruction will be execute which type is, than doing different type calculate operates and store the result in EX_MEM class as ALUoutput.

```
//EX()
void EX()
{

        unsigned long int a = 0;//&
        unsigned long int b = 0;
        ex_mem.setIR(id_ex.getIR());
        ex_mem.setA(id_ex.getA());//unsigned long int A;//rs
        ex_mem.setB(id_ex.getB());//rt
        ex_mem.setrs(id_ex.getrs());//十进制
        ex_mem.setrt(id_ex.getrt());
        ex_mem.setrd(id_ex.getrd());
        ex_mem.setimm(id_ex.getimm());
        ex_mem.setOP(id_ex.getOP());
        ex_mem.setRS(id_ex.getRS());
        ex_mem.setRT(id_ex.getRT());
        ex_mem.setRD(id_ex.getRD());
        ex_mem.setSHAMT(id_ex.getSHAMT());
        ex_mem.setFUNCT(id_ex.getFUNCT());
        ex_mem.setIMM(id_ex.getIMM());
        string s = id_ex.getIR();
        if (s != "")
        {
            if (id_ex.getOP() == "000000")//R type
            {

                if (id_ex.getFUNCT() == "100000")//add
                {
                    id_ex.setA(IF_ID::R[id_ex.getrs()]);
                    id_ex.setB(IF_ID::R[id_ex.getrt()]);
                    ex_mem.setALUoutput(id_ex.getA() + id_ex.getB());//int ALUoutput=A+B
                }
                else if (id_ex.getFUNCT() == "100010")//sub
                {
                    id_ex.setA(IF_ID::R[id_ex.getrs()]);
                    id_ex.setB(IF_ID::R[id_ex.getrt()]);
                    ex_mem.setALUoutput(id_ex.getA() - id_ex.getB());//int ALUoutput=A-B
                }
                else if (id_ex.getFUNCT() == "100101")//or
```

```
        else if (id_ex.getFUNCT() == "100101")//or
        {
            id_ex.setA(IF_ID::R[id_ex.getrs()]);
            id_ex.setB(IF_ID::R[id_ex.getrt()]);
            a = id_ex.getA();
            b = id_ex.getB();
            ex_mem.setALUoutput(OrOperator(a, b));//or
        }

        else if (id_ex.getFUNCT() == "011000")//mul
        {
            id_ex.setA(IF_ID::R[id_ex.getrs()]);
            id_ex.setB(IF_ID::R[id_ex.getrt()]);
            unsigned long int c = 0;
            //id_ex.setA(40960);//test
            //id_ex.setB(40960);
            c = id_ex.getA() * id_ex.getB();
            ex_mem.setALUoutput(c);

        }

    }
    //I type
    else if (id_ex.getOP() == "000100")//beq
    {
        id_ex.setA(IF_ID::R[id_ex.getrs()]);
        id_ex.setB(IF_ID::R[id_ex.getrt()]);
        beq_flag = 0;//stall a instruction
        //bool(R[rs]==R[rt])
        ex_mem.setALUoutput(0);//beq not WB()
        ex_mem.setcond(bool(id_ex.getA() == id_ex.getB()));//相等 cond=1;


        ex_mem.setNPC(id_ex.getNPC() + id_ex.getimm());//NPC ->branch instruction number
    }
    else if (id_ex.getOP() == "001100")//andi
    {
        id_ex.setA(IF_ID::R[id_ex.getrs()]);
        a = id_ex.getA();
        //latch.setimm(bin_to_dec(latch.getIMM()));//IMM转dec imm
```

未找到相关问题

```
        //latch.setimm(bin_to_dec(latch.getIMM()));//IMM转dec imm
        b = id_ex.getimm();
        ex_mem.setALUoutput(AndOperator(a, b));//and
    }
    else if (id_ex.getOP() == "001101")//ori
    {
        id_ex.setA(IF_ID::R[id_ex.getrs()]);
        a = id_ex.getA();
        //latch.setimm(bin_to_dec(latch.getIMM()));//IMM转dec imm
        b = id_ex.getimm();
        ex_mem.setALUoutput(OrOperator(a, b));
    }
    else if (id_ex.getOP() == "001111")//lui
    {

        unsigned long int a = 0;
        a = 65536;
        //latch.setimm(bin_to_dec(latch.getIMM()));//IMM转dec imm
        ex_mem.setALUoutput(a * id_ex.getimm());

    }
    else if (id_ex.getOP() == "100011")//lw
    {
        //R[rs]*16
        //imm/4
        id_ex.setA(IF_ID::R[id_ex.getrs()]);
        //latch.setimm(bin_to_dec(latch.getIMM()));//IMM转dec imm
        a = id_ex.getA() * 16 + id_ex.getimm() / 4;//address=R[rs]*16+imm/4
        ex_mem.setALUoutput(a);//ALUoutput=address
        //datamemory[latch.getALUoutput()]=66;//for test
    }
    else if (id_ex.getOP() == "101011")//sw
    {
        //R[rs]*16

        //imm/4
        id_ex.setA(IF_ID::R[id_ex.getrs()]);
        //latch.setimm(bin_to_dec(latch.getIMM()));//IMM转dec imm
        a = (id_ex.getA() * 16) + (id_ex.getimm() / 4);//address=R[rs]*16+imm/4
        ex_mem.setALUoutput(a);//ALUoutput=address
```



```
            a = id_ex.getA() * 16 + id_ex.getimm() / 4;//address=R[rs]*16+imm/4
            ex_mem.setALUoutput(a);//ALUoutput=address
            //datamemory[latch.getALUoutput()]=66;//for test
        }
        else if (id_ex.getOP() == "101011")//sw
        {
            //R[rs]*16

            //imm/4
            id_ex.setA(IF_ID::R[id_ex.getrs()]);
            //latch.setimm(bin_to_dec(latch.getIMM()));//IMM转dec imm
            a = (id_ex.getA() * 16) + (id_ex.getimm() / 4);//address=R[rs]*16+imm/4
            ex_mem.setALUoutput(a);//ALUoutput=address

        }
        else if (id_ex.getOP() == "001010" || id_ex.getOP() == "001011")//
        {
        int result;
        if (IF_ID::R[id_ex.getrt()] < id_ex.getimm())
        {
            result = 1;
        }
        else
        {
            result = 0;
        }
        ex_mem.setALUoutput(result);
        }

        else
        {
            ex_mem.setALUoutput(0);

        }
}
    else {
        invalid_EX++;
    }
clock_EX++;
}
```

- MEM()

  Only sw and lw instruction will use this stage.

  For sw,fetch the value of R[rt],than store into datamemory[ALUoutput],the address were calculated in EX().

  For lw,fetch the data from datamemory[ALUoutput] and store it in LMD latch of MEM_WB class,waiting to write back to Register.

```
void MEM()
{
    mem_wb.setIR(ex_mem.getIR());
    mem_wb.setALUoutput(ex_mem.getALUoutput());
    //cond;
    mem_wb.setimm(ex_mem.getimm());//ex_mem.setimm(id_ex.getimm());
    mem_wb.setA(ex_mem.getA());//unsigned long int A;//rs
    mem_wb.setB(ex_mem.getB());//ex_mem.setB(id_ex.getB());//rt
    mem_wb.setrs(ex_mem.getrs());//十进制
    mem_wb.setrt(ex_mem.getrt());
    mem_wb.setrd(ex_mem.getrd());
    mem_wb.setOP(ex_mem.getOP());
    mem_wb.setRS(ex_mem.getRS());

    mem_wb.setRT(ex_mem.getRT());
    mem_wb.setRD(ex_mem.getRD());
    mem_wb.setSHAMT(ex_mem.getSHAMT());
    mem_wb.setFUNCT(ex_mem.getFUNCT());
    mem_wb.setIMM(ex_mem.getIMM());
    string s = ex_mem.getIR();
    if (s != "")
    {
        //sw M[ALUoutput]=R[rt]

        if (ex_mem.getOP() == "101011")//sw
        {
            datamemory[ex_mem.getALUoutput()] = ex_mem.getB();
            ex_mem.setALUoutput(0);//no WB

        }

        //lw LMD=M[address]
        else if (ex_mem.getOP() == "100011")//lw
        {

            mem_wb.setLMD(datamemory[ex_mem.getALUoutput()]);

        }
        else
        {
            invalid_MEM++;
        }

    }
```

- WB()

Except sw and beq instructions,other instruction all need to write back the result of calculate stage to the destination register.

```
//WB()
void WB()
{

    string s = mem_wb.getIR();
    if (s != "")
    {
        if (data_flag ==1)
        {
            data_flag = 0;
        }

        if (mem_wb.getOP() == "100011")//lw
        {
            IF_ID::R[mem_wb.getrt()] = mem_wb.getLMD();
        }

        else
        {
            if (mem_wb.getOP() == "101011")//sw
            {
                //no WB
            }
            else
            {
                if (mem_wb.getOP() == "000100")//beq
                {
                    //no WB

                }
                else
                {

                    if (mem_wb.getOP() == "000010")//mul
                    {
                        bitset<64> b1(mem_wb.getALUoutput());
                        //cout << b1 << endl;
                        string s;
                        string s1;
                        string s2;
                        s = b1.to_string();
```

```
                    string s2;
                    s = b1.to_string();
                    s1 = s.substr(0, 32);
                    s2 = s.substr(32, 32);
                    //cout << "s1=" << s1 << endl;
                    //cout << "s2=" << s2 << endl;

                    IF_ID::R[mem_wb.getrd()] = bin_to_dec(s2);//低32位存rd
                    IF_ID::R[mem_wb.getrd() + 1] = bin_to_dec(s1);//高32位存rd+1

                }
                else
                {

                    if (mem_wb.getOP() == "000000")//R type
                    {

                        IF_ID::R[mem_wb.getrd()] = mem_wb.getALUoutput();

                    }
                    else //I type
                    {

                        IF_ID::R[mem_wb.getrt()] = mem_wb.getALUoutput();

                    }

                }
            }
        }
    }

        insnum++;

}
else
{
    invalid_WB++;
}
clock_WB++;
}
```

The tool functions

Bin_to_dec() was used to change string type binary number to int type decimal number.

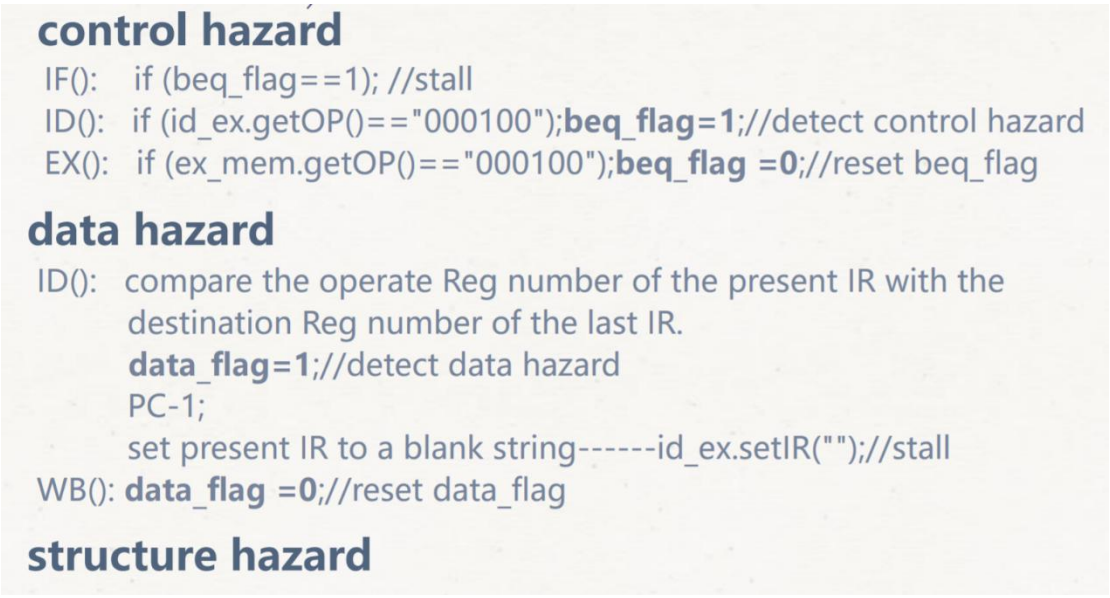
ROOperator() was used to execute or operation for or and ori instruction.

ANDOperator() was used to execute and operation for add and andi instruction.

signExtend() and zeroExtend were used to extend immediate number.

```cpp
//tool functions
unsigned long int bin_to_dec(string bin)//change string type binary number to int type decimal
{
    //将传入的二进制字符串转换成十进制，并返回十进制数字。
    //在此处调用判断一个字符串有多长的函数
    unsigned long int size = bin.length();

    //将二进制数字转换为十进制
    unsigned long int parseBinary = 0;
    for (unsigned long int i = 0; i < size; ++i) {
        if (bin[i] == '1') {
            parseBinary += pow(2.0, size - i - 1);
        }
    }

    return parseBinary;
}

unsigned long int OrOperator(unsigned long int& a, unsigned long int& b) //or
{
    return a | b;
}

unsigned long int AndOperator(unsigned long int& a, unsigned long int& b) //&
{
    return a & b;
}


//sign extend
void signExtend(string imm)
{
    string s=imm;
    string imme;
    imme = s.substr(0, 1);
    if (imme == "0")
    {
        id_ex.setIMM("0000000000000000" + s);
    }
    else if (imme == "1")
    {
        id_ex.setIMM("1111111111111111" + s);
    }

}

//zero extend
void zeroExtend(string imm)
{
    string s=imm;
    string imme;
    id_ex.setIMM("0000000000000000" + s);
}
```

2. hazards detect



For control hazard:

Judge if the instruction is beq type in decode stage,if yes, set the beq_flag=1,means control hazard happened. Because I call the 5 stage functions reversely, so after decode stage,will into fetch stage, so before fetch next instruction need judge if beq_flag equal to 1. If beq_flag equal to 1,stop fetch next instruction. Untill next execute stage, means cond result is come out after calculate, reset the beq_flag to 0;

In this this program can achieve stall when control hazard happened.

For data hazard:

Compare the operate Reg number of the present IR with the destination Reg number of the last IR in decode stage. If the

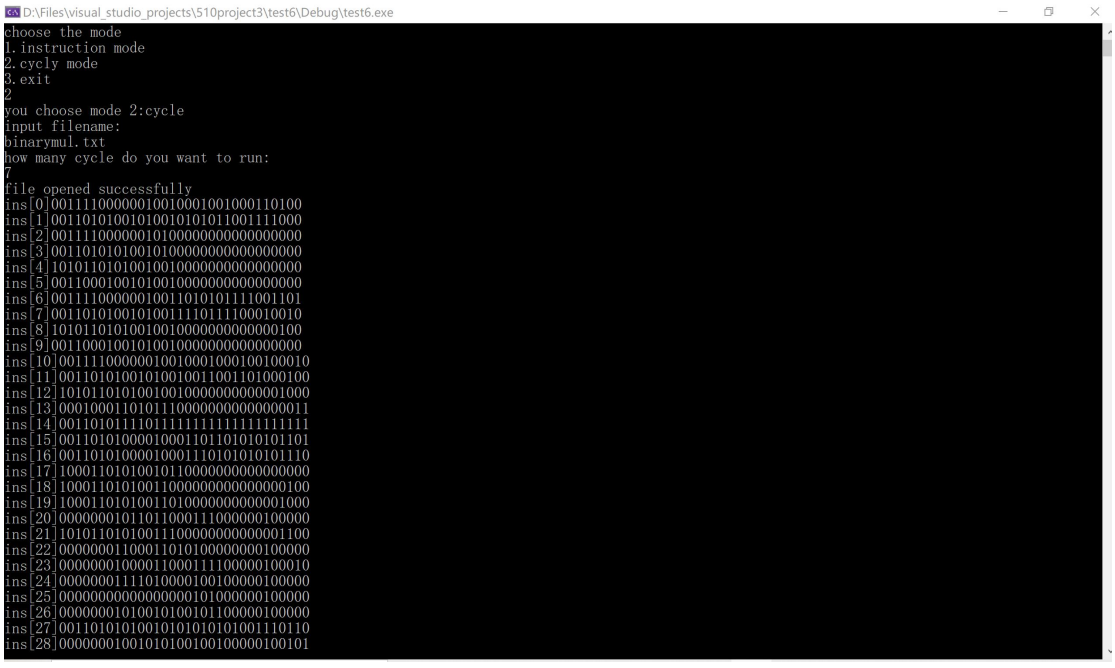
result is they will read and write a same register,it means data hazard happened. Set data_flag=1,and set PC-1 to let PC go back to last instruction and set present IR to a blank string.Finally,reset data_flag =0 in write back stage.

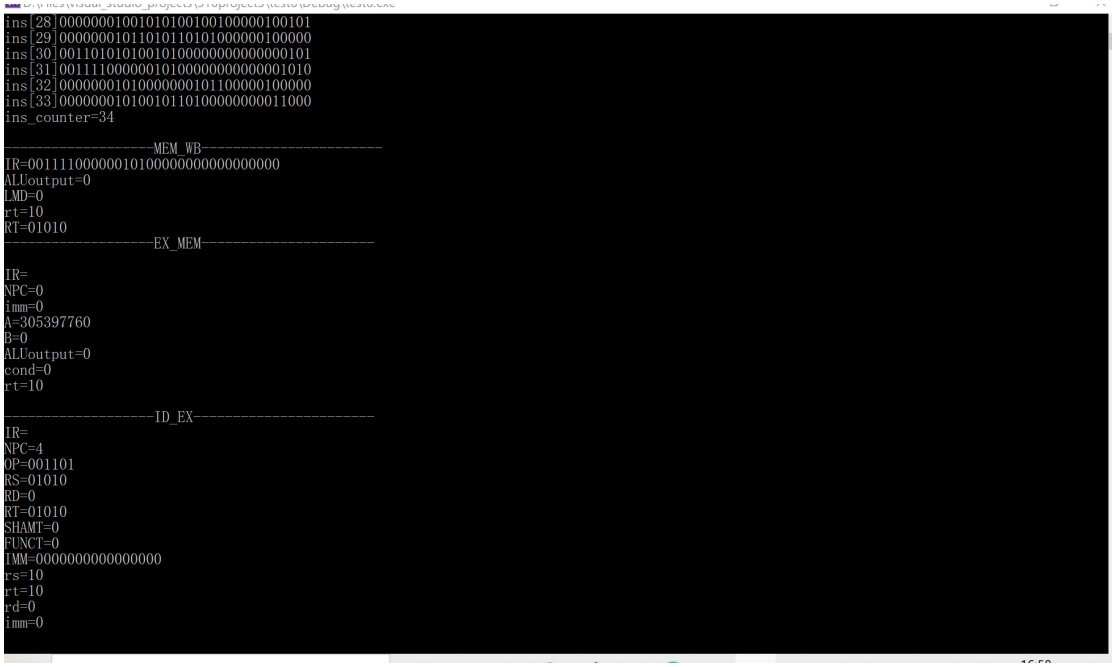
For structure hazard:

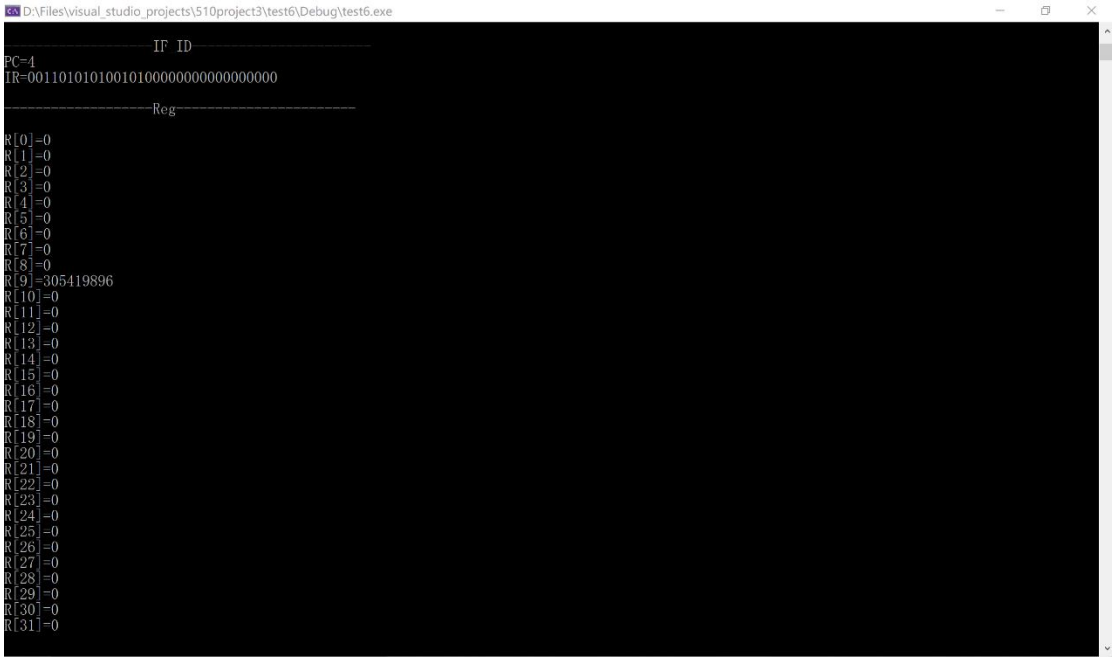
Because I call 5 stage function reversely,so in every cycle,will doing writing register first and later read from register,so structure will not happen structure in this program.

3.problems and summary

Simulate result



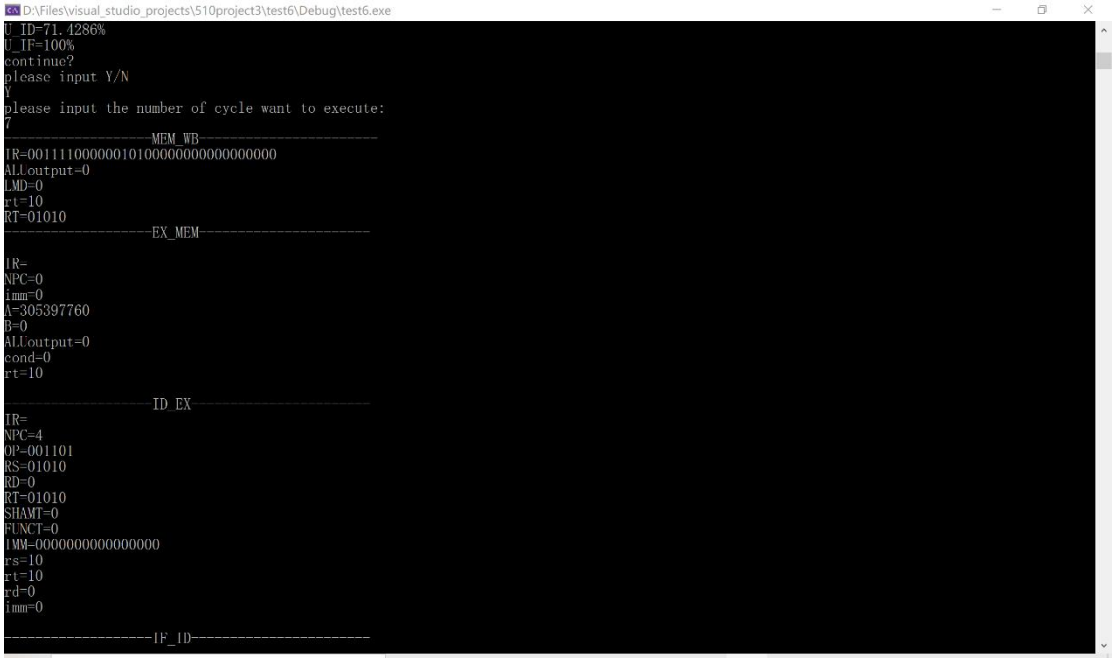

ins[28]00000001001010100100100000100101
ins[29]00000001011010110101000000100000
ins[30]00110101010010100000000000000101
ins[31]00111100000010100000000000001010
ins[32]00000001010000000101100000100000
ins[33]00000001010010110100000000011000
ins_counter=34
----------MEM_WB----------
IR=00111100000010100000000000000000
ALUoutput=0
LMD=0
rt=10
RT=01010
----------EX_MEM----------
IR=
NPC=0
imm=0
A=305397760
B=0
ALUoutput=0
cond=0
rt=10
----------ID_EX----------
IR=
NPC=4
OP=001101
RS=01010
RD=0
RT=01010
SHAMT=0
FUNCT=0
IMM=0000000000000000
rs=10
rt=10
rd=0
imm=0

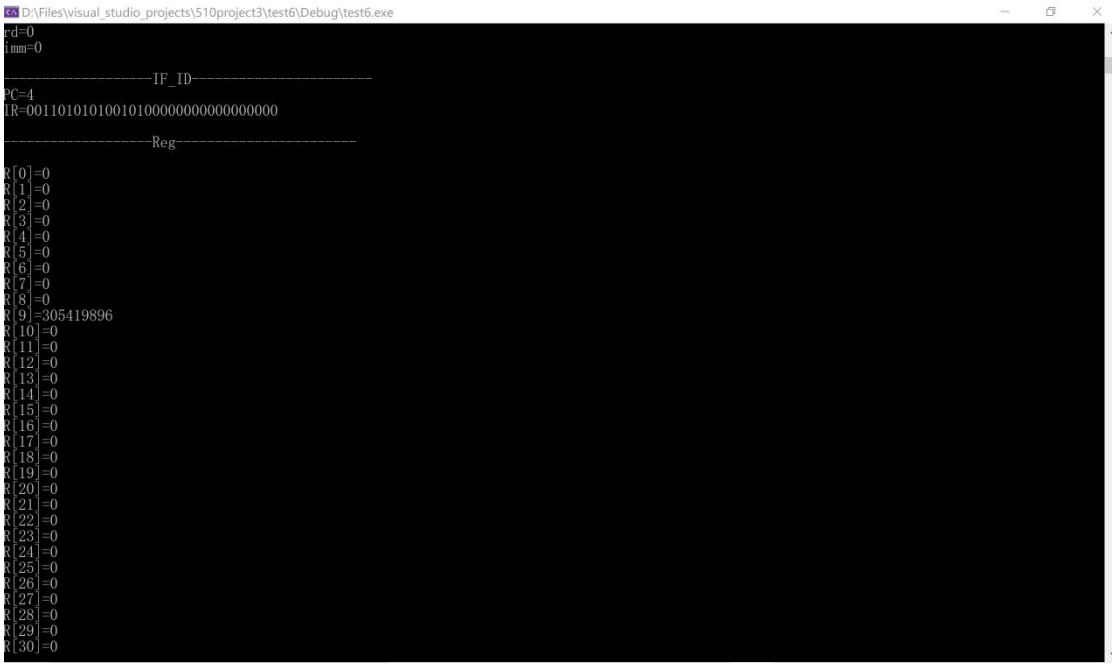

D:\Files\visual_studio_projects\510project3\test6\Debug\test6.exe
----------IF ID----------
PC=4
IR=00110101010010100000000000000000
----------Reg----------
R[0]=0
R[1]=0
R[2]=0
R[3]=0
R[4]=0
R[5]=0
R[6]=0
R[7]=0
R[8]=0
R[9]=305419896
R[10]=0
R[11]=0
R[12]=0
R[13]=0
R[14]=0
R[15]=0
R[16]=0
R[17]=0
R[18]=0
R[19]=0
R[20]=0
R[21]=0
R[22]=0
R[23]=0
R[24]=0
R[25]=0
R[26]=0
R[27]=0
R[28]=0
R[29]=0
R[30]=0
R[31]=0


D:\Files\visual_studio_projects\510project3\test6\Debug\test6.exe
----------datamemory----------
d[0]=0 d[1]=0 d[2]=0 d[3]=0 d[4]=0 d[5]=0 d[6]=0 d[7]=0 d[8]=0 d[9]=0 d[10]=0 d[11]=0 d[12]=0 d[13]=0 d[14]=0 d[15]=0
d[16]=0 d[17]=0 d[18]=0 d[19]=0 d[20]=0 d[21]=0 d[22]=0 d[23]=0 d[24]=0 d[25]=0 d[26]=0 d[27]=0 d[28]=0 d[29]=0 d[30]=0 d[31]=0
d[32]=0 d[33]=0 d[34]=0 d[35]=0 d[36]=0 d[37]=0 d[38]=0 d[39]=0 d[40]=0 d[41]=0 d[42]=0 d[43]=0 d[44]=0 d[45]=0 d[46]=0 d[47]=0
d[48]=0 d[49]=0 d[50]=0 d[51]=0 d[52]=0 d[53]=0 d[54]=0 d[55]=0 d[56]=0 d[57]=0 d[58]=0 d[59]=0 d[60]=0 d[61]=0 d[62]=0 d[63]=0
d[64]=0 d[65]=0 d[66]=0 d[67]=0 d[68]=0 d[69]=0 d[70]=0 d[71]=0 d[72]=0 d[73]=0 d[74]=0 d[75]=0 d[76]=0 d[77]=0 d[78]=0 d[79]=0
d[80]=0 d[81]=0 d[82]=0 d[83]=0 d[84]=0 d[85]=0 d[86]=0 d[87]=0 d[88]=0 d[89]=0 d[90]=0 d[91]=0 d[92]=0 d[93]=0 d[94]=0 d[95]=0
d[96]=0 d[97]=0 d[98]=0 d[99]=0 d[100]=0 d[101]=0 d[102]=0 d[103]=0 d[104]=0 d[105]=0 d[106]=0 d[107]=0 d[108]=0 d[109]=0 d[110]=0 d[111]=0
d[112]=0 d[113]=0 d[114]=0 d[115]=0 d[116]=0 d[117]=0 d[118]=0 d[119]=0 d[120]=0 d[121]=0 d[122]=0 d[123]=0 d[124]=0 d[125]=0 d[126]=0 d[127]=0
d[128]=0 d[129]=0 d[130]=0 d[131]=0 d[132]=0 d[133]=0 d[134]=0 d[135]=0 d[136]=0 d[137]=0 d[138]=0 d[139]=0 d[140]=0 d[141]=0 d[142]=0 d[143]=0
d[144]=0 d[145]=0 d[146]=0 d[147]=0 d[148]=0 d[149]=0 d[150]=0 d[151]=0 d[152]=0 d[153]=0 d[154]=0 d[155]=0 d[156]=0 d[157]=0 d[158]=0 d[159]=0
d[160]=0 d[161]=0 d[162]=0 d[163]=0 d[164]=0 d[165]=0 d[166]=0 d[167]=0 d[168]=0 d[169]=0 d[170]=0 d[171]=0 d[172]=0 d[173]=0 d[174]=0 d[175]=0
d[176]=0 d[177]=0 d[178]=0 d[179]=0 d[180]=0 d[181]=0 d[182]=0 d[183]=0 d[184]=0 d[185]=0 d[186]=0 d[187]=0 d[188]=0 d[189]=0 d[190]=0 d[191]=0
d[192]=0 d[193]=0 d[194]=0 d[195]=0 d[196]=0 d[197]=0 d[198]=0 d[199]=0 d[200]=0 d[201]=0 d[202]=0 d[203]=0 d[204]=0 d[205]=0 d[206]=0 d[207]=0
d[208]=0 d[209]=0 d[210]=0 d[211]=0 d[212]=0 d[213]=0 d[214]=0 d[215]=0 d[216]=0 d[217]=0 d[218]=0 d[219]=0 d[220]=0 d[221]=0 d[222]=0 d[223]=0
d[224]=0 d[225]=0 d[226]=0 d[227]=0 d[228]=0 d[229]=0 d[230]=0 d[231]=0 d[232]=0 d[233]=0 d[234]=0 d[235]=0 d[236]=0 d[237]=0 d[238]=0 d[239]=0


D:\Files\visual_studio_projects\510project3\test6\Debug\test6.exe
]=0
d[224]=0 d[225]=0 d[226]=0 d[227]=0 d[228]=0 d[229]=0 d[230]=0 d[231]=0 d[232]=0 d[233]=0 d[234]=0 d[235]=0 d[236]=0 d[237]=0 d[238]=0 d[239]=0
d[240]=0 d[241]=0 d[242]=0 d[243]=0 d[244]=0 d[245]=0 d[246]=0 d[247]=0 d[248]=0 d[249]=0 d[250]=0 d[251]=0 d[252]=0 d[253]=0 d[254]=0 d[255]=0
d[256]=0 d[257]=0 d[258]=0 d[259]=0 d[260]=0 d[261]=0 d[262]=0 d[263]=0 d[264]=0 d[265]=0 d[266]=0 d[267]=0 d[268]=0 d[269]=0 d[270]=0 d[271]=0
d[272]=0 d[273]=0 d[274]=0 d[275]=0 d[276]=0 d[277]=0 d[278]=0 d[279]=0 d[280]=0 d[281]=0 d[282]=0 d[283]=0 d[284]=0 d[285]=0 d[286]=0 d[287]=0
d[288]=0 d[289]=0 d[290]=0 d[291]=0 d[292]=0 d[293]=0 d[294]=0 d[295]=0 d[296]=0 d[297]=0 d[298]=0 d[299]=0 d[300]=0 d[301]=0 d[302]=0 d[303]=0
d[304]=0 d[305]=0 d[306]=0 d[307]=0 d[308]=0 d[309]=0 d[310]=0 d[311]=0 d[312]=0 d[313]=0 d[314]=0 d[315]=0 d[316]=0 d[317]=0 d[318]=0 d[319]=0
d[320]=0 d[321]=0 d[322]=0 d[323]=0 d[324]=0 d[325]=0 d[326]=0 d[327]=0 d[328]=0 d[329]=0 d[330]=0 d[331]=0 d[332]=0 d[333]=0 d[334]=0 d[335]=0
d[336]=0 d[337]=0 d[338]=0 d[339]=0 d[340]=0 d[341]=0 d[342]=0 d[343]=0 d[344]=0 d[345]=0 d[346]=0 d[347]=0 d[348]=0 d[349]=0 d[350]=0 d[351]=0
d[352]=0 d[353]=0 d[354]=0 d[355]=0 d[356]=0 d[357]=0 d[358]=0 d[359]=0 d[360]=0 d[361]=0 d[362]=0 d[363]=0 d[364]=0 d[365]=0 d[366]=0 d[367]=0
d[368]=0 d[369]=0 d[370]=0 d[371]=0 d[372]=0 d[373]=0 d[374]=0 d[375]=0 d[376]=0 d[377]=0 d[378]=0 d[379]=0 d[380]=0 d[381]=0 d[382]=0 d[383]=0
d[384]=0 d[385]=0 d[386]=0 d[387]=0 d[388]=0 d[389]=0 d[390]=0 d[391]=0 d[392]=0 d[393]=0 d[394]=0 d[395]=0 d[396]=0 d[397]=0 d[398]=0 d[399]=0
d[400]=0 d[401]=0 d[402]=0 d[403]=0 d[404]=0 d[405]=0 d[406]=0 d[407]=0 d[408]=0 d[409]=0 d[410]=0 d[411]=0 d[412]=0 d[413]=0 d[414]=0 d[415]=0
d[416]=0 d[417]=0 d[418]=0 d[419]=0 d[420]=0 d[421]=0 d[422]=0 d[423]=0 d[424]=0 d[425]=0 d[426]=0 d[427]=0 d[428]=0 d[429]=0 d[430]=0 d[431]=0

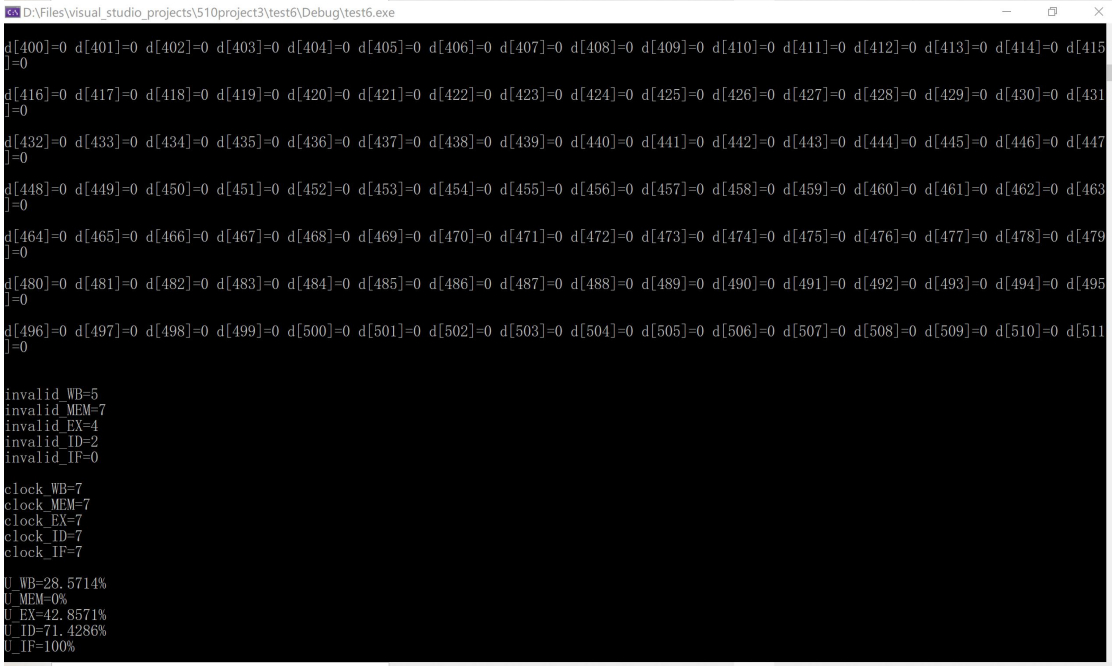

D:\Files\visual_studio_projects\510project3\test6\Debug\test6.exe
]=0
d[416]=0 d[417]=0 d[418]=0 d[419]=0 d[420]=0 d[421]=0 d[422]=0 d[423]=0 d[424]=0 d[425]=0 d[426]=0 d[427]=0 d[428]=0 d[429]=0 d[430]=0 d[431
]=0
d[432]=0 d[433]=0 d[434]=0 d[435]=0 d[436]=0 d[437]=0 d[438]=0 d[439]=0 d[440]=0 d[441]=0 d[442]=0 d[443]=0 d[444]=0 d[445]=0 d[446]=0 d[447
]=0
d[448]=0 d[449]=0 d[450]=0 d[451]=0 d[452]=0 d[453]=0 d[454]=0 d[455]=0 d[456]=0 d[457]=0 d[458]=0 d[459]=0 d[460]=0 d[461]=0 d[462]=0 d[463
]=0
d[464]=0 d[465]=0 d[466]=0 d[467]=0 d[468]=0 d[469]=0 d[470]=0 d[471]=0 d[472]=0 d[473]=0 d[474]=0 d[475]=0 d[476]=0 d[477]=0 d[478]=0 d[479
]=0
d[480]=0 d[481]=0 d[482]=0 d[483]=0 d[484]=0 d[485]=0 d[486]=0 d[487]=0 d[488]=0 d[489]=0 d[490]=0 d[491]=0 d[492]=0 d[493]=0 d[494]=0 d[495
]=0
d[496]=0 d[497]=0 d[498]=0 d[499]=0 d[500]=0 d[501]=0 d[502]=0 d[503]=0 d[504]=0 d[505]=0 d[506]=0 d[507]=0 d[508]=0 d[509]=0 d[510]=0 d[511
]=0
invalid_WB=5
invalid_MEM=7
invalid_EX=4
invalid_ID=2
invalid_IF=0
clock_WB=7
clock_MEM=7
clock_EX=7
clock_ID=7
clock_IF=7
U_WB=28.5714%
U_MEM=0%
U_EX=42.8571%
U_ID=71.4286%
U_IF=100%
continue?
please input Y/N

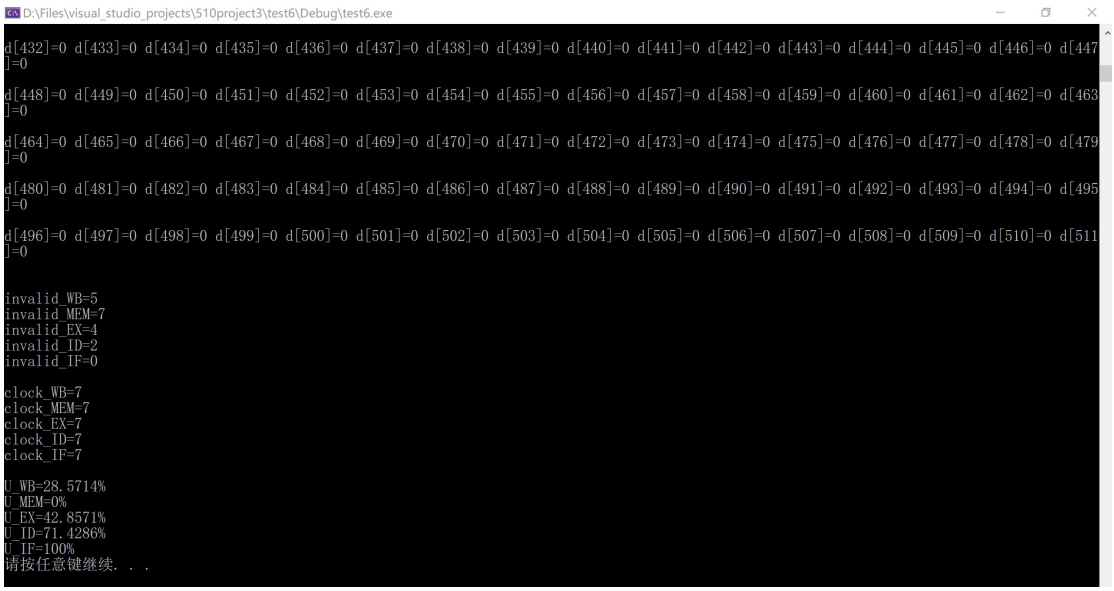

D:\Files\visual_studio_projects\510project3\test6\Debug\test6.exe
U_ID=71.4286%
U_IF=100%
continue?
please input Y/N
Y
please input the number of cycle want to execute:
7
----------MEM_WB----------
IR=00111100000010100000000000000000
ALUoutput=0
LMD=0
rt=10
RT=01010
----------EX_MEM----------
IR=
NPC=0
imm=0
A=305397760
B=0
ALUoutput=0
cond=0
rt=10
----------ID_EX----------
IR=
NPC=4
OP=001101
RS=01010
RD=0
RT=01010
SHAMT=0
FUNCT=0
IMM=0000000000000000
rs=10
rt=10
rd=0
imm=0
----------IF_ID----------


D:\Files\visual_studio_projects\510project3\test6\Debug\test6.exe
rd=0
imm=0
----------IF_ID----------
PC=4
IR=00110101010010100000000000000000
----------Reg----------
R[0]=0
R[1]=0
R[2]=0
R[3]=0
R[4]=0
R[5]=0
R[6]=0
R[7]=0
R[8]=0
R[9]=305419896
R[10]=0
R[11]=0
R[12]=0
R[13]=0
R[14]=0
R[15]=0
R[16]=0
R[17]=0
R[18]=0
R[19]=0
R[20]=0
R[21]=0
R[22]=0
R[23]=0
R[24]=0
R[25]=0
R[26]=0
R[27]=0
R[28]=0
R[29]=0
R[30]=0


D:\Files\visual_studio_projects\510project3\test6\Debug\test6.exe
R[30]=0
R[31]=0
----------datamemory----------
d[0]=0 d[1]=0 d[2]=0 d[3]=0 d[4]=0 d[5]=0 d[6]=0 d[7]=0 d[8]=0 d[9]=0 d[10]=0 d[11]=0 d[12]=0 d[13]=0 d[14]=0 d[15]=0
d[16]=0 d[17]=0 d[18]=0 d[19]=0 d[20]=0 d[21]=0 d[22]=0 d[23]=0 d[24]=0 d[25]=0 d[26]=0 d[27]=0 d[28]=0 d[29]=0 d[30]=0 d[31]=0
d[32]=0 d[33]=0 d[34]=0 d[35]=0 d[36]=0 d[37]=0 d[38]=0 d[39]=0 d[40]=0 d[41]=0 d[42]=0 d[43]=0 d[44]=0 d[45]=0 d[46]=0 d[47]=0
d[48]=0 d[49]=0 d[50]=0 d[51]=0 d[52]=0 d[53]=0 d[54]=0 d[55]=0 d[56]=0 d[57]=0 d[58]=0 d[59]=0 d[60]=0 d[61]=0 d[62]=0 d[63]=0
d[64]=0 d[65]=0 d[66]=0 d[67]=0 d[68]=0 d[69]=0 d[70]=0 d[71]=0 d[72]=0 d[73]=0 d[74]=0 d[75]=0 d[76]=0 d[77]=0 d[78]=0 d[79]=0
d[80]=0 d[81]=0 d[82]=0 d[83]=0 d[84]=0 d[85]=0 d[86]=0 d[87]=0 d[88]=0 d[89]=0 d[90]=0 d[91]=0 d[92]=0 d[93]=0 d[94]=0 d[95]=0
d[96]=0 d[97]=0 d[98]=0 d[99]=0 d[100]=0 d[101]=0 d[102]=0 d[103]=0 d[104]=0 d[105]=0 d[106]=0 d[107]=0 d[108]=0 d[109]=0 d[110]=0 d[111]=0
d[112]=0 d[113]=0 d[114]=0 d[115]=0 d[116]=0 d[117]=0 d[118]=0 d[119]=0 d[120]=0 d[121]=0 d[122]=0 d[123]=0 d[124]=0 d[125]=0 d[126]=0 d[127]=0
d[128]=0 d[129]=0 d[130]=0 d[131]=0 d[132]=0 d[133]=0 d[134]=0 d[135]=0 d[136]=0 d[137]=0 d[138]=0 d[139]=0 d[140]=0 d[141]=0 d[142]=0 d[143]=0
d[144]=0 d[145]=0 d[146]=0 d[147]=0 d[148]=0 d[149]=0 d[150]=0 d[151]=0 d[152]=0 d[153]=0 d[154]=0 d[155]=0 d[156]=0 d[157]=0 d[158]=0 d[159]=0
d[160]=0 d[161]=0 d[162]=0 d[163]=0 d[164]=0 d[165]=0 d[166]=0 d[167]=0 d[168]=0 d[169]=0 d[170]=0 d[171]=0 d[172]=0 d[173]=0 d[174]=0 d[175]=0
d[176]=0 d[177]=0 d[178]=0 d[179]=0 d[180]=0 d[181]=0 d[182]=0 d[183]=0 d[184]=0 d[185]=0 d[186]=0 d[187]=0 d[188]=0 d[189]=0 d[190]=0 d[191]=0
d[192]=0 d[193]=0 d[194]=0 d[195]=0 d[196]=0 d[197]=0 d[198]=0 d[199]=0 d[200]=0 d[201]=0 d[202]=0 d[203]=0 d[204]=0 d[205]=0 d[206]=0 d[207]=0
d[208]=0 d[209]=0 d[210]=0 d[211]=0 d[212]=0 d[213]=0 d[214]=0 d[215]=0 d[216]=0 d[217]=0 d[218]=0 d[219]=0 d[220]=0 d[221]=0 d[222]=0 d[223]=0


D:\Files\visual_studio_projects\510project3\test6\Debug\test6.exe
d[208]=0 d[209]=0 d[210]=0 d[211]=0 d[212]=0 d[213]=0 d[214]=0 d[215]=0 d[216]=0 d[217]=0 d[218]=0 d[219]=0 d[220]=0 d[221]=0 d[222]=0 d[223]=0
d[224]=0 d[225]=0 d[226]=0 d[227]=0 d[228]=0 d[229]=0 d[230]=0 d[231]=0 d[232]=0 d[233]=0 d[234]=0 d[235]=0 d[236]=0 d[237]=0 d[238]=0 d[239]=0
d[240]=0 d[241]=0 d[242]=0 d[243]=0 d[244]=0 d[245]=0 d[246]=0 d[247]=0 d[248]=0 d[249]=0 d[250]=0 d[251]=0 d[252]=0 d[253]=0 d[254]=0 d[255]=0
d[256]=0 d[257]=0 d[258]=0 d[259]=0 d[260]=0 d[261]=0 d[262]=0 d[263]=0 d[264]=0 d[265]=0 d[266]=0 d[267]=0 d[268]=0 d[269]=0 d[270]=0 d[271]=0
d[272]=0 d[273]=0 d[274]=0 d[275]=0 d[276]=0 d[277]=0 d[278]=0 d[279]=0 d[280]=0 d[281]=0 d[282]=0 d[283]=0 d[284]=0 d[285]=0 d[286]=0 d[287]=0
d[288]=0 d[289]=0 d[290]=0 d[291]=0 d[292]=0 d[293]=0 d[294]=0 d[295]=0 d[296]=0 d[297]=0 d[298]=0 d[299]=0 d[300]=0 d[301]=0 d[302]=0 d[303]=0
d[304]=0 d[305]=0 d[306]=0 d[307]=0 d[308]=0 d[309]=0 d[310]=0 d[311]=0 d[312]=0 d[313]=0 d[314]=0 d[315]=0 d[316]=0 d[317]=0 d[318]=0 d[319]=0
d[320]=0 d[321]=0 d[322]=0 d[323]=0 d[324]=0 d[325]=0 d[326]=0 d[327]=0 d[328]=0 d[329]=0 d[330]=0 d[331]=0 d[332]=0 d[333]=0 d[334]=0 d[335]=0
d[336]=0 d[337]=0 d[338]=0 d[339]=0 d[340]=0 d[341]=0 d[342]=0 d[343]=0 d[344]=0 d[345]=0 d[346]=0 d[347]=0 d[348]=0 d[349]=0 d[350]=0 d[351]=0
d[352]=0 d[353]=0 d[354]=0 d[355]=0 d[356]=0 d[357]=0 d[358]=0 d[359]=0 d[360]=0 d[361]=0 d[362]=0 d[363]=0 d[364]=0 d[365]=0 d[366]=0 d[367]=0
d[368]=0 d[369]=0 d[370]=0 d[371]=0 d[372]=0 d[373]=0 d[374]=0 d[375]=0 d[376]=0 d[377]=0 d[378]=0 d[379]=0 d[380]=0 d[381]=0 d[382]=0 d[383]=0
d[384]=0 d[385]=0 d[386]=0 d[387]=0 d[388]=0 d[389]=0 d[390]=0 d[391]=0 d[392]=0 d[393]=0 d[394]=0 d[395]=0 d[396]=0 d[397]=0 d[398]=0 d[399]=0
d[400]=0 d[401]=0 d[402]=0 d[403]=0 d[404]=0 d[405]=0 d[406]=0 d[407]=0 d[408]=0 d[409]=0 d[410]=0 d[411]=0 d[412]=0 d[413]=0 d[414]=0 d[415]=0


D:\Files\visual_studio_projects\510project3\test6\Debug\test6.exe
d[400]=0 d[401]=0 d[402]=0 d[403]=0 d[404]=0 d[405]=0 d[406]=0 d[407]=0 d[408]=0 d[409]=0 d[410]=0 d[411]=0 d[412]=0 d[413]=0 d[414]=0 d[415]=0
d[416]=0 d[417]=0 d[418]=0 d[419]=0 d[420]=0 d[421]=0 d[422]=0 d[423]=0 d[424]=0 d[425]=0 d[426]=0 d[427]=0 d[428]=0 d[429]=0 d[430]=0 d[431]=0
d[432]=0 d[433]=0 d[434]=0 d[435]=0 d[436]=0 d[437]=0 d[438]=0 d[439]=0 d[440]=0 d[441]=0 d[442]=0 d[443]=0 d[444]=0 d[445]=0 d[446]=0 d[447]=0
d[448]=0 d[449]=0 d[450]=0 d[451]=0 d[452]=0 d[453]=0 d[454]=0 d[455]=0 d[456]=0 d[457]=0 d[458]=0 d[459]=0 d[460]=0 d[461]=0 d[462]=0 d[463]=0
d[464]=0 d[465]=0 d[466]=0 d[467]=0 d[468]=0 d[469]=0 d[470]=0 d[471]=0 d[472]=0 d[473]=0 d[474]=0 d[475]=0 d[476]=0 d[477]=0 d[478]=0 d[479]=0
d[480]=0 d[481]=0 d[482]=0 d[483]=0 d[484]=0 d[485]=0 d[486]=0 d[487]=0 d[488]=0 d[489]=0 d[490]=0 d[491]=0 d[492]=0 d[493]=0 d[494]=0 d[495]=0
d[496]=0 d[497]=0 d[498]=0 d[499]=0 d[500]=0 d[501]=0 d[502]=0 d[503]=0 d[504]=0 d[505]=0 d[506]=0 d[507]=0 d[508]=0 d[509]=0 d[510]=0 d[511]=0
invalid_WB=5
invalid_MEM=7
invalid_EX=4
invalid_ID=2
invalid_IF=0
clock_WB=7
clock_MEM=7
clock_EX=7
clock_ID=7
clock_IF=7
U_WB=28.5714%
U_MEM=0%
U_EX=42.8571%
U_ID=71.4286%
U_IF=100%



During this project, I encountered a lot of problems, various errors, warnings, and bugs, and even rewritten the program 6 times. At the beginning, I didn't understand the principle of this cpu very well, but at the beginning of the project, I spent a lot of time to understand and learn the structure and principle of cpu, and I quickly learned the syntax of c++ two weeks in advance. As my first C++ program, this project is indeed very difficult and challenging, but I also learned a lot of knowledge in the process, in addition to the principle of cpu, the use of c++, the usage of classes and so on. I spent a lot of time to complete this project. Although it is not perfect until now, it is the best result I can achieve.