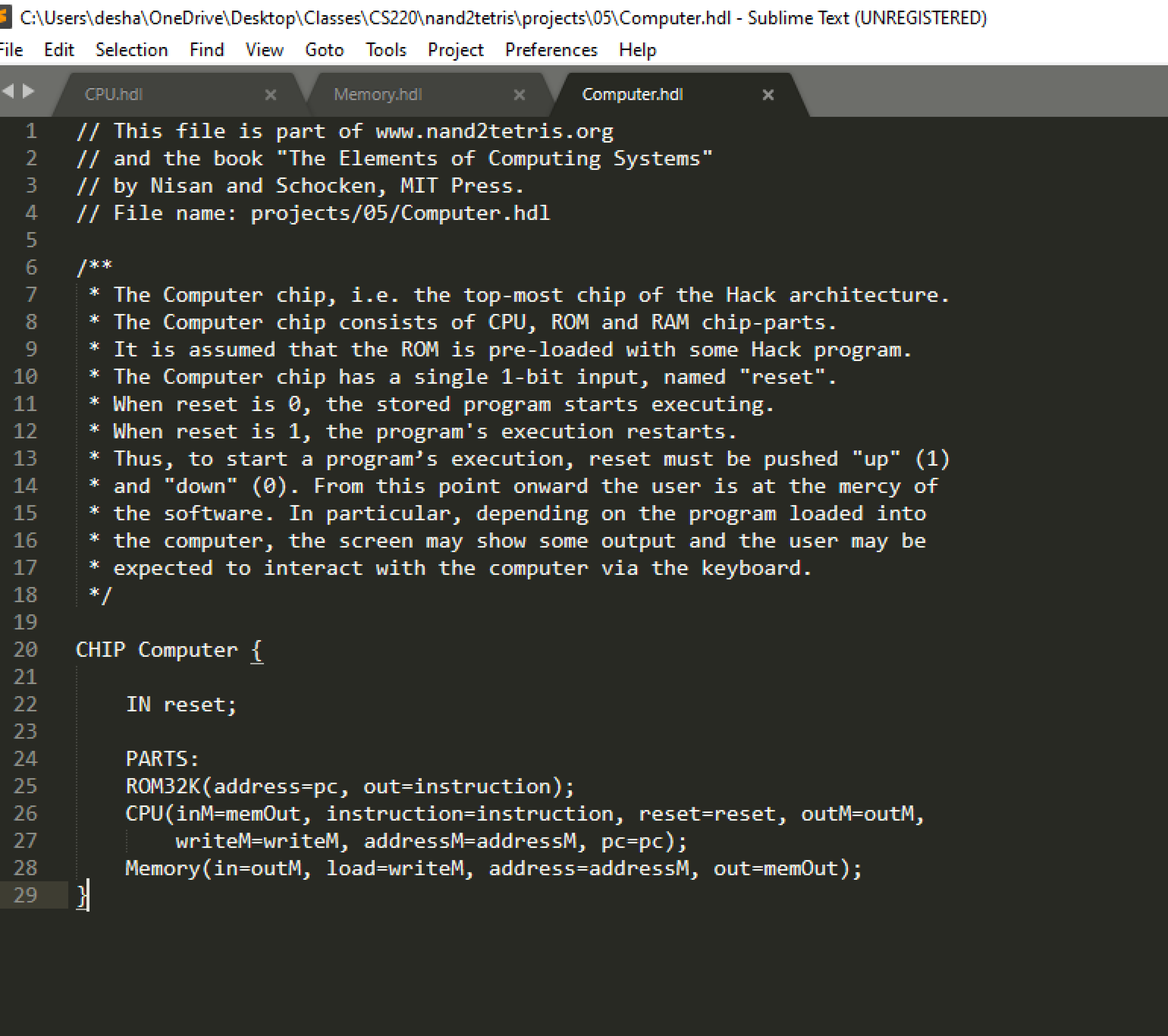
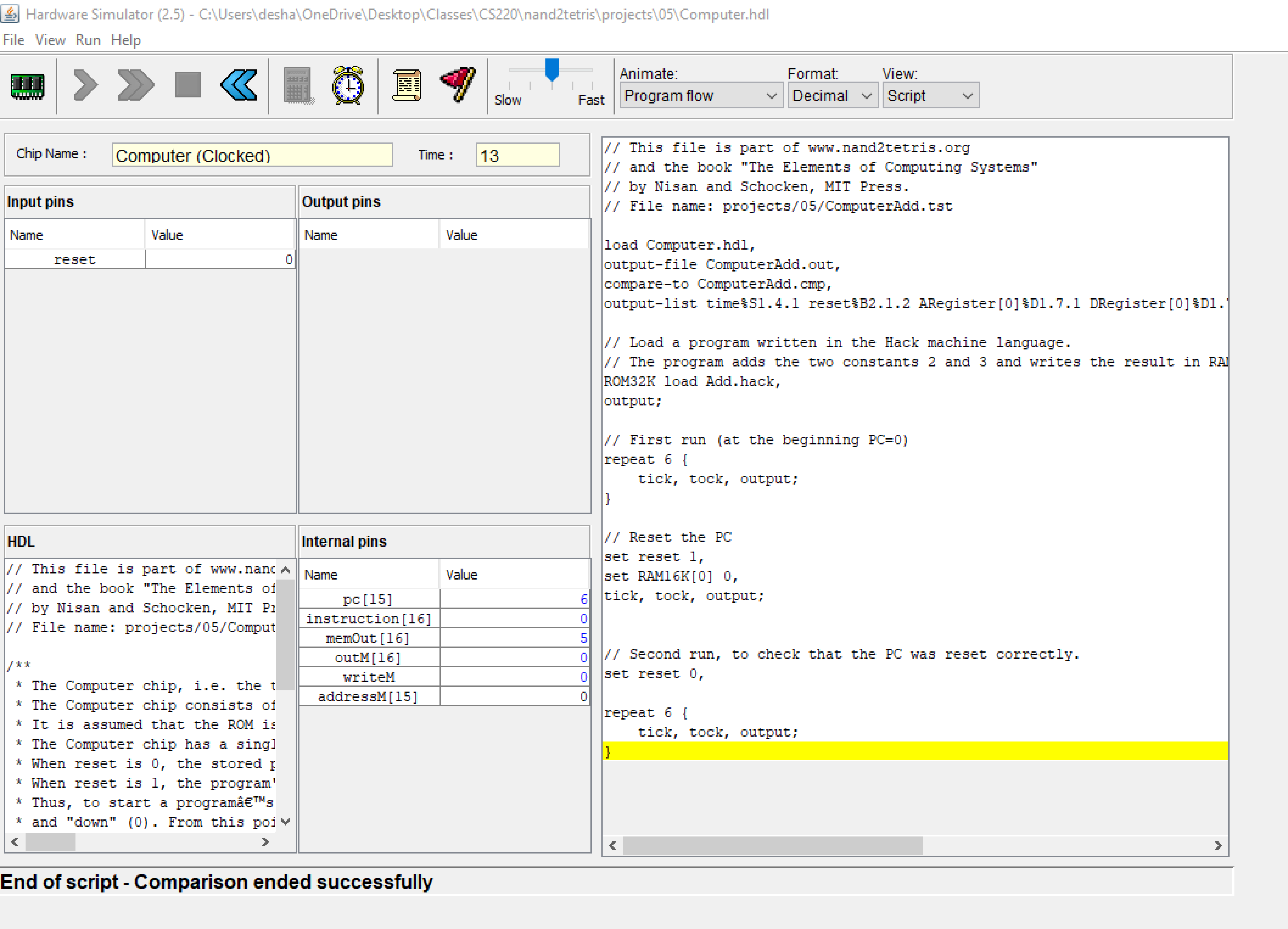


C:\Users\desha\OneDrive\Desktop\Classes\CS220\nand2tetris\projects\05\Computer.hdl - Sublime Text (UNREGISTERED)
File
Edit
Selection
Find
View
Goto
Tools
Project
Preferences
Help
CPU.hdl
Memory.hdl
Computer.hdl

```
// This file is part of www.nand2tetris.org
// and the book "The Elements of Computing Systems"
// by Nisan and Schocken, MIT Press.
// File name: projects/05/Computer.hdl

/**
 * The Computer chip, i.e. the top-most chip of the Hack architecture.
 * The Computer chip consists of CPU, ROM and RAM chip-parts.
 * It is assumed that the ROM is pre-loaded with some Hack program.
 * The Computer chip has a single 1-bit input, named "reset".
 * When reset is 0, the stored program starts executing.
 * When reset is 1, the program's execution restarts.
 * Thus, to start a program’s execution, reset must be pushed "up" (1)
 * and "down" (0). From this point onward the user is at the mercy of
 * the software. In particular, depending on the program loaded into
 * the computer, the screen may show some output and the user may be
 * expected to interact with the computer via the keyboard.
 */

CHIP Computer {

    IN reset;

    PARTS:
    ROM32K(address=pc, out=instruction);
    CPU(inM=memOut, instruction=instruction, reset=reset, outM=outM,
        writeM=writeM, addressM=addressM, pc=pc);
    Memory(in=outM, load=writeM, address=addressM, out=memOut);
}
```


Hardware Simulator (2.5) - C:\Users\desha\OneDrive\Desktop\Classes\CS220\nand2tetris\projects\05\Computer.hdl
File
View
Run
Help
Slow
Fast

Animate: Program flow
Format: Decimal
View: Script

Chip Name : Computer (Clocked)
Time : 13

Input pins

Name	Value
reset	0

Output pins

Name	Value

HDL

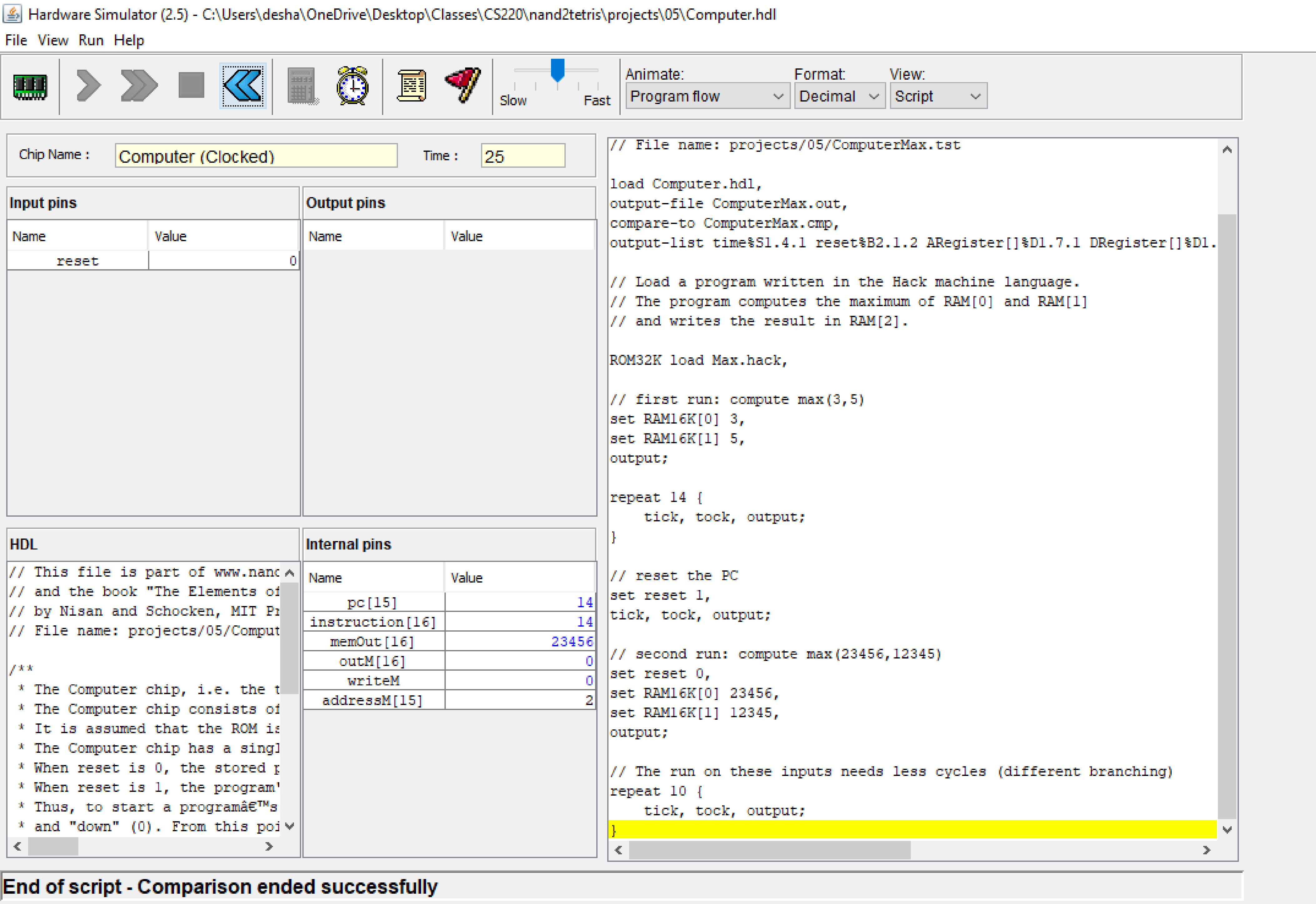
```
// This file is part of www.nanc
// and the book "The Elements of
// by Nisan and Schocken, MIT Pr
// File name: projects/05/Comput

/**
 * The Computer chip, i.e. the t
 * The Computer chip consists of
 * It is assumed that the ROM is
 * The Computer chip has a singl
 * When reset is 0, the stored p
 * When reset is 1, the program'
 * Thus, to start a programâ€™s
 * and "down" (0). From this poi
```

Internal pins

Name	Value
pc[15]	6
instruction[16]	0
memOut[16]	5
outM[16]	0
writeM	0
addressM[15]	0

```
// This file is part of www.nand2tetris.org
// and the book "The Elements of Computing Systems"
// by Nisan and Schocken, MIT Press.
// File name: projects/05/ComputerAdd.tst

load Computer.hdl,
output-file ComputerAdd.out,
compare-to ComputerAdd.cmp,
output-list time%S1.4.1 reset%B2.1.2 ARegister[0]%D1.7.1 DRegister[0]%D1.

// Load a program written in the Hack machine language.
// The program adds the two constants 2 and 3 and writes the result in RAI
ROM32K load Add.hack,
output;

// First run (at the beginning PC=0)
repeat 6 {
    tick, tock, output;
}

// Reset the PC
set reset 1,
set RAM16K[0] 0,
tick, tock, output;


// Second run, to check that the PC was reset correctly.
set reset 0,

repeat 6 {
    tick, tock, output;
}
```

End of script - Comparison ended successfully



Hardware Simulator (2.5) - C:\Users\desha\OneDrive\Desktop\Classes\CS220\nand2tetris\projects\05\Computer.hdl

Animate: Program flow

Format: Decimal

View: Script

Chip Name : Computer (Clocked)

Time : 25

Name	Value
reset	0

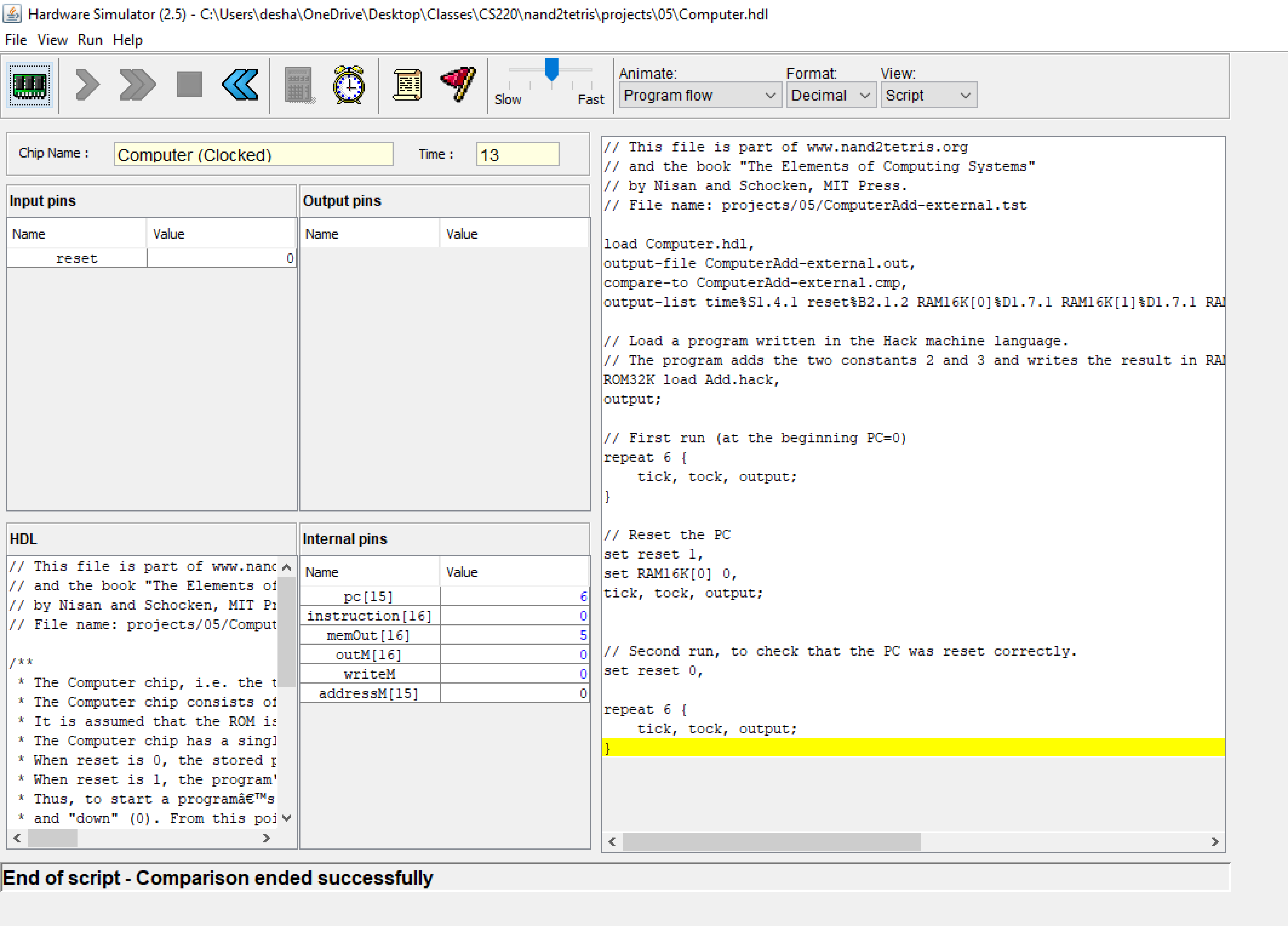
Name	Value

```
// This file is part of www.nanc
// and the book "The Elements of
// by Nisan and Schocken, MIT Pr
// File name: projects/05/Comput

/**
 * The Computer chip, i.e. the t
 * The Computer chip consists of
 * It is assumed that the ROM is
 * The Computer chip has a singl
 * When reset is 0, the stored p
 * When reset is 1, the program'
 * Thus, to start a programâ€™s
 * and "down" (0). From this poi
```

Name	Value
pc[15]	14
instruction[16]	14
memOut[16]	23456
outM[16]	0
writeM	0
addressM[15]	2

```
// File name: projects/05/ComputerMax.tst

load Computer.hdl,
output-file ComputerMax.out,
compare-to ComputerMax.cmp,
output-list time%S1.4.1 reset%B2.1.2 ARegister[]%D1.7.1 DRegister[]%D1.

// Load a program written in the Hack machine language.
// The program computes the maximum of RAM[0] and RAM[1]
// and writes the result in RAM[2].

ROM32K load Max.hack,

// first run: compute max(3,5)
set RAM16K[0] 3,
set RAM16K[1] 5,
output;

repeat 14 {
    tick, tock, output;
}

// reset the PC
set reset 1,
tick, tock, output;

// second run: compute max(23456,12345)
set reset 0,
set RAM16K[0] 23456,
set RAM16K[1] 12345,
output;

// The run on these inputs needs less cycles (different branching)
repeat 10 {
    tick, tock, output;
}
```

End of script - Comparison ended successfully


Hardware Simulator (2.5) - C:\Users\desha\OneDrive\Desktop\Classes\CS220\nand2tetris\projects\05\Computer.hdl
File
View
Run
Help
Slow
Fast

Animate: Program flow

Format: Decimal

View: Script

Chip Name : Computer (Clocked)

Time : 13

Input pins

Name	Value
reset	0

Output pins

Name	Value

HDL

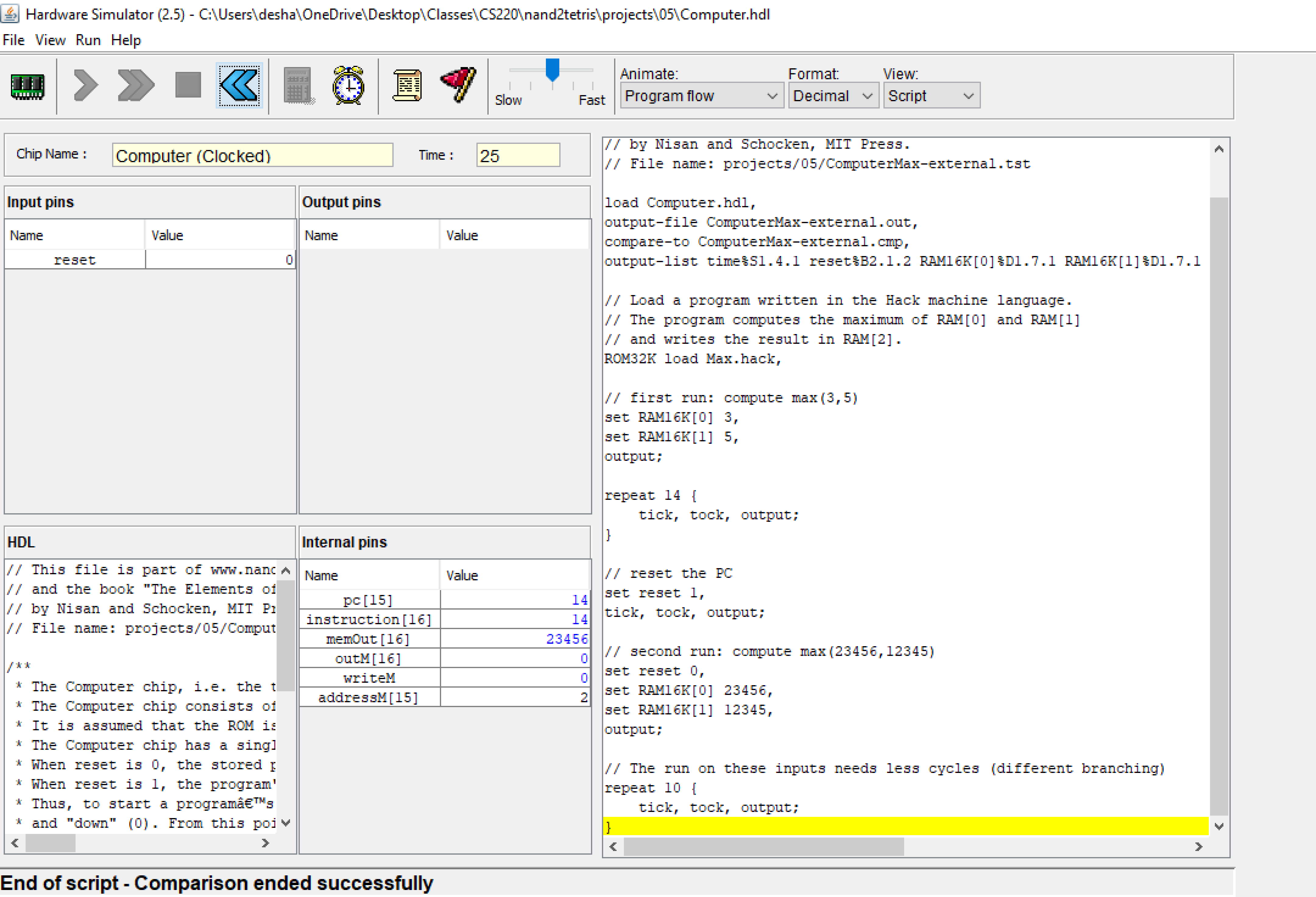
```
// This file is part of www.nanc
// and the book "The Elements of
// by Nisan and Schocken, MIT Pr
// File name: projects/05/Comput

/**
 * The Computer chip, i.e. the t
 * The Computer chip consists of
 * It is assumed that the ROM is
 * The Computer chip has a singl
 * When reset is 0, the stored p
 * When reset is 1, the program'
 * Thus, to start a programâ€™s
 * and "down" (0). From this poi
```

Internal pins

Name	Value
pc[15]	6
instruction[16]	0
memOut[16]	5
outM[16]	0
writeM	0
addressM[15]	0

```
// This file is part of www.nand2tetris.org
// and the book "The Elements of Computing Systems"
// by Nisan and Schocken, MIT Press.
// File name: projects/05/ComputerAdd-external.tst

load Computer.hdl,
output-file ComputerAdd-external.out,
compare-to ComputerAdd-external.cmp,
output-list time%S1.4.1 reset%B2.1.2 RAM16K[0]%D1.7.1 RAM16K[1]%D1.7.1 RAI

// Load a program written in the Hack machine language.
// The program adds the two constants 2 and 3 and writes the result in RAI
ROM32K load Add.hack,
output;

// First run (at the beginning PC=0)
repeat 6 {
    tick, tock, output;
}

// Reset the PC
set reset 1,
set RAM16K[0] 0,
tick, tock, output;


// Second run, to check that the PC was reset correctly.
set reset 0,

repeat 6 {
    tick, tock, output;
}
```

End of script - Comparison ended successfully


Hardware Simulator (2.5) - C:\Users\desha\OneDrive\Desktop\Classes\CS220\nand2tetris\projects\05\Computer.hdl
File
View
Run
Help
Slow
Fast

Animate: Program flow
Format: Decimal
View: Script

Chip Name : Computer (Clocked)
Time : 25

Input pins

Name	Value
reset	0

Output pins

Name	Value

HDL

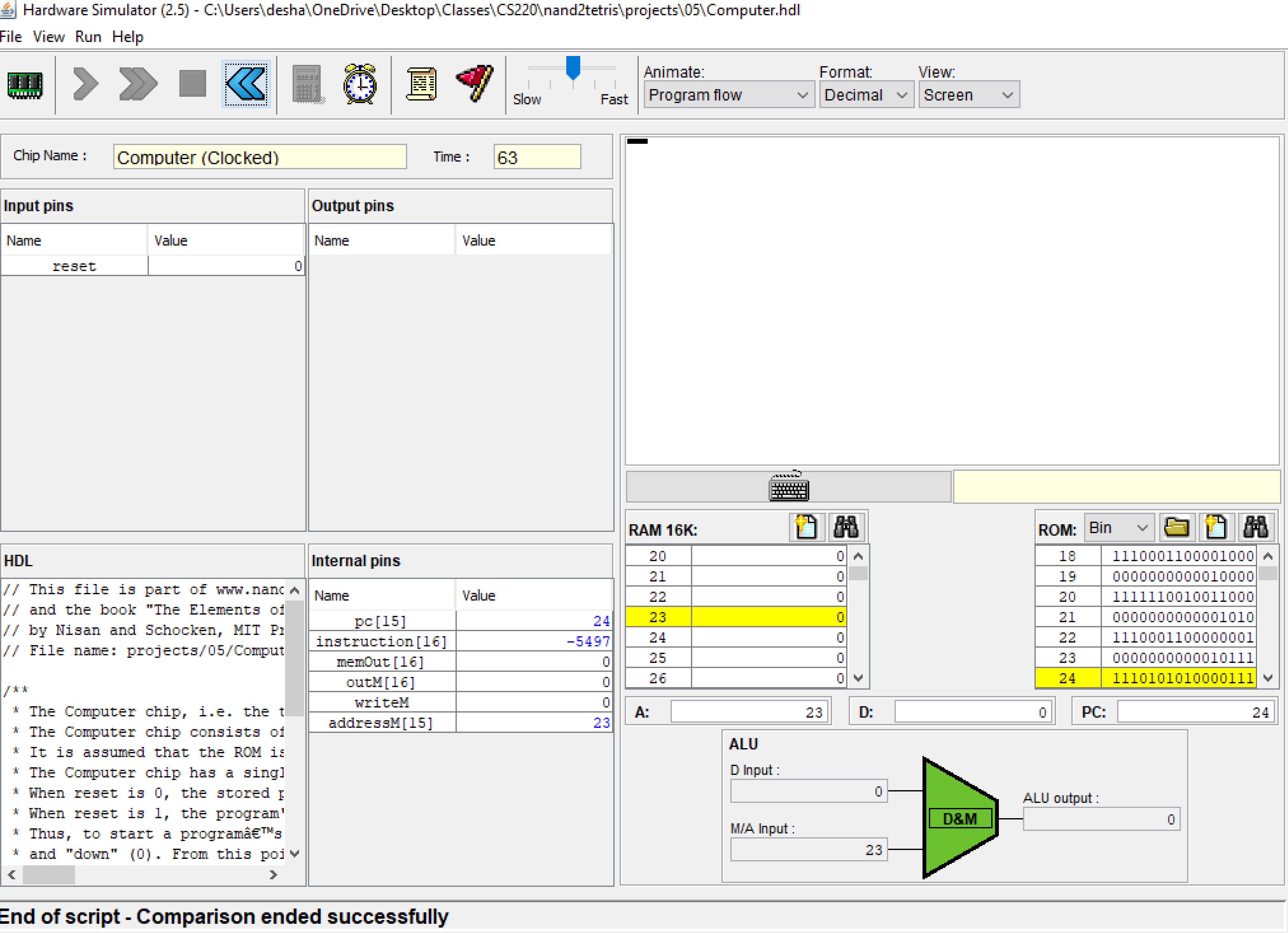
```
// This file is part of www.nanc
// and the book "The Elements of
// by Nisan and Schocken, MIT Pr
// File name: projects/05/Comput

/**
 * The Computer chip, i.e. the t
 * The Computer chip consists of
 * It is assumed that the ROM is
 * The Computer chip has a singl
 * When reset is 0, the stored p
 * When reset is 1, the program'
 * Thus, to start a programâ€™s
 * and "down" (0). From this poi
```

Internal pins

Name	Value
pc[15]	14
instruction[16]	14
memOut[16]	23456
outM[16]	0
writeM	0
addressM[15]	2

```
// by Nisan and Schocken, MIT Press.
// File name: projects/05/ComputerMax-external.tst

load Computer.hdl,
output-file ComputerMax-external.out,
compare-to ComputerMax-external.cmp,
output-list time%S1.4.1 reset%B2.1.2 RAM16K[0]%D1.7.1 RAM16K[1]%D1.7.1

// Load a program written in the Hack machine language.
// The program computes the maximum of RAM[0] and RAM[1]
// and writes the result in RAM[2].
ROM32K load Max.hack,

// first run: compute max(3,5)
set RAM16K[0] 3,
set RAM16K[1] 5,
output;

repeat 14 {
    tick, tock, output;
}

// reset the PC
set reset 1,
tick, tock, output;

// second run: compute max(23456,12345)
set reset 0,
set RAM16K[0] 23456,
set RAM16K[1] 12345,
output;

// The run on these inputs needs less cycles (different branching)
repeat 10 {
    tick, tock, output;
}
```

End of script - Comparison ended successfully



Hardware Simulator (2.5) - C:\Users\desha\OneDrive\Desktop\Classes\CS220\nand2tetris\projects\05\Computer.hdl

Animate: Program flow

Format: Decimal

View: Screen

Chip Name : Computer (Clocked)

Time : 63

Input pins

Name	Value
reset	0

Output pins

Name	Value

```
// This file is part of www.nanc
// and the book "The Elements of
// by Nisan and Schocken, MIT Pr
// File name: projects/05/Comput

/**
 * The Computer chip, i.e. the t
 * The Computer chip consists of
 * It is assumed that the ROM is
 * The Computer chip has a singl
 * When reset is 0, the stored p
 * When reset is 1, the program'
 * Thus, to start a programâ€™s
 * and "down" (0). From this poi
```

Internal pins

Name	Value
pc[15]	24
instruction[16]	-5497
memOut[16]	0
outM[16]	0
writeM	0
addressM[15]	23

RAM 16K:

20	0
21	0
22	0
23	0
24	0
25	0
26	0

ROM: Bin

18	1110001100001000
19	0000000000010000
20	1111110010011000
21	0000000000001010
22	1110001100000001
23	0000000000010111
24	1110101010000111

A: 23

D: 0

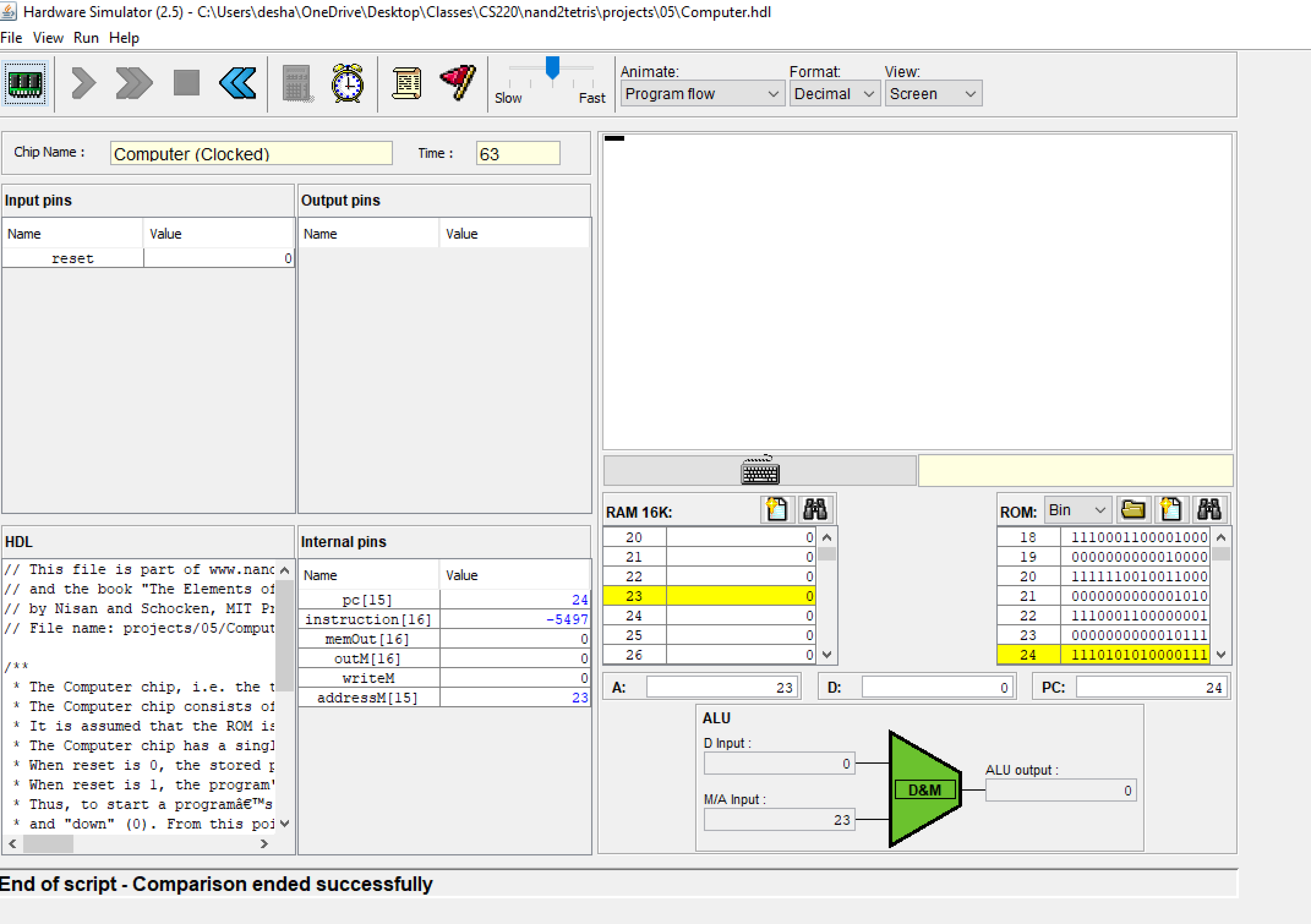
PC: 24

D Input : 0

M/A Input : 23

ALU output : 0

End of script - Comparison ended successfully



Hardware Simulator (2.5) - C:\Users\desha\OneDrive\Desktop\Classes\CS220\nand2tetris\projects\05\Computer.hdl
File View Run Help
Slow
Fast

Animate: Program flow

Format: Decimal

View: Screen

Chip Name : Computer (Clocked)

Time : 63

Input pins

Name	Value
reset	0

Output pins

Name	Value

HDL

```
// This file is part of www.nanc
// and the book "The Elements of
// by Nisan and Schocken, MIT Pr
// File name: projects/05/Comput

/**
 * The Computer chip, i.e. the t
 * The Computer chip consists of
 * It is assumed that the ROM is
 * The Computer chip has a singl
 * When reset is 0, the stored p
 * When reset is 1, the program'
 * Thus, to start a programâ€™s
 * and "down" (0). From this poi
```

Internal pins

Name	Value
pc[15]	24
instruction[16]	-5497
memOut[16]	0
outM[16]	0
writeM	0
addressM[15]	23

RAM 16K:

20	0
21	0
22	0
23	0
24	0
25	0
26	0

ROM: Bin

18	1110001100001000
19	0000000000010000
20	1111110010011000
21	0000000000001010
22	1110001100000001
23	0000000000010111
24	1110101010000111

A: 23

D: 0

PC: 24

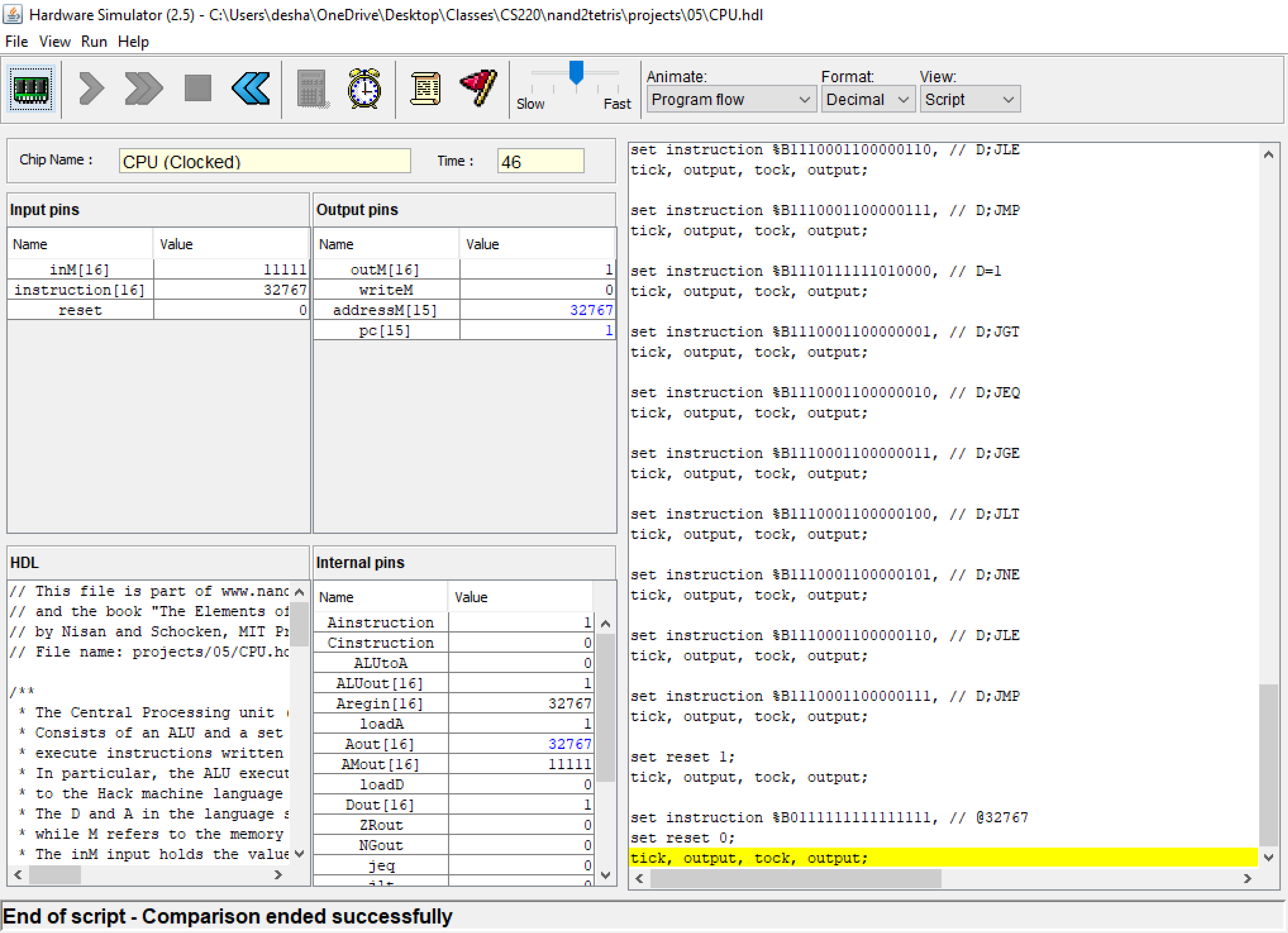
ALU

D Input : 0

M/A Input : 23

ALU output : 0

End of script - Comparison ended successfully


Hardware Simulator (2.5) - C:\Users\desha\OneDrive\Desktop\Classes\CS220\nand2tetris\projects\05\CPU.hdl
File
View
Run
Help
Slow
Fast

Animate: Program flow

Format: Decimal

View: Script

Chip Name : CPU (Clocked)

Time : 46

Input pins

Name	Value
inM[16]	11111
instruction[16]	32767
reset	0

Output pins

Name	Value
outM[16]	1
writeM	0
addressM[15]	32767
pc[15]	1

HDL

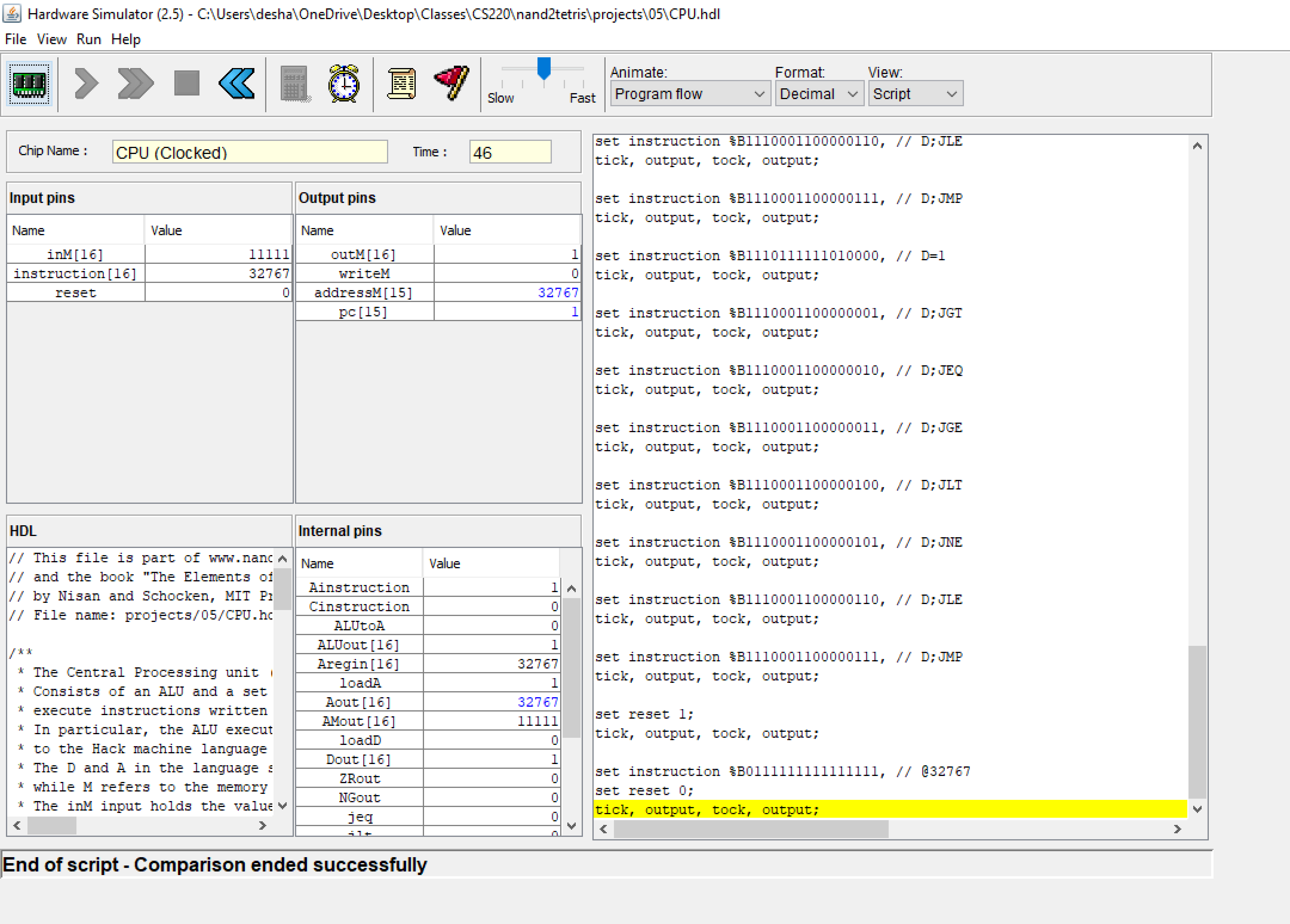
```
// This file is part of www.nanc
// and the book "The Elements of
// by Nisan and Schocken, MIT Pr
// File name: projects/05/CPU.hc

/**
 * The Central Processing unit (
 * Consists of an ALU and a set
 * execute instructions written
 * In particular, the ALU execut
 * to the Hack machine language
 * The D and A in the language s
 * while M refers to the memory
 * The inM input holds the value
```

Internal pins

Name	Value
Ainstruction	1
Cinstruction	0
ALUtoA	0
ALUout[16]	1
Aregin[16]	32767
loadA	1
Aout[16]	32767
AMout[16]	11111
loadD	0
Dout[16]	1
ZRout	0
NGout	0
jeq	0
jlt	0

```
set instruction %B1110001100000110, // D;JLE
tick, output, tock, output;

set instruction %B1110001100000111, // D;JMP
tick, output, tock, output;

set instruction %B1110111111010000, // D=1
tick, output, tock, output;

set instruction %B1110001100000001, // D;JGT
tick, output, tock, output;

set instruction %B1110001100000010, // D;JEQ
tick, output, tock, output;

set instruction %B1110001100000011, // D;JGE
tick, output, tock, output;

set instruction %B1110001100000100, // D;JLT
tick, output, tock, output;

set instruction %B1110001100000101, // D;JNE
tick, output, tock, output;

set instruction %B1110001100000110, // D;JLE
tick, output, tock, output;

set instruction %B1110001100000111, // D;JMP
tick, output, tock, output;

set reset 1;
tick, output, tock, output;

set instruction %B0111111111111111, // @32767
set reset 0;
tick, output, tock, output;
```

End of script - Comparison ended successfully


Hardware Simulator (2.5) - C:\Users\desha\OneDrive\Desktop\Classes\CS220\nand2tetris\projects\05\CPU.hdl
File
View
Run
Help
Slow
Fast
Input pins
Output pins
HDL
Internal pins

Animate: Program flow

Format: Decimal

View: Script

Chip Name : CPU (Clocked)

Time : 46

Name	Value
inM[16]	11111
instruction[16]	32767
reset	0

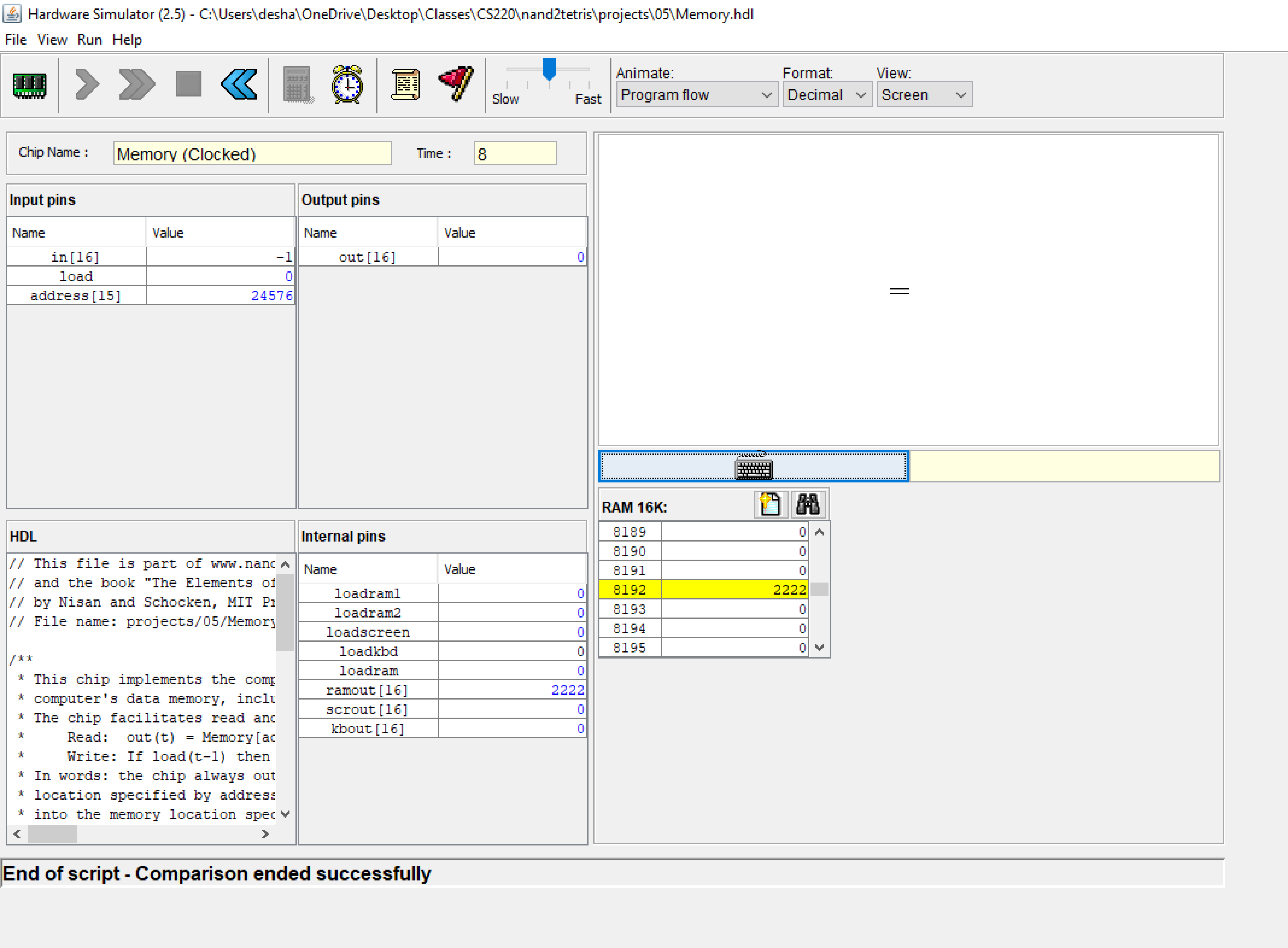
Name	Value
outM[16]	1
writeM	0
addressM[15]	32767
pc[15]	1

```
// This file is part of www.nanc
// and the book "The Elements of
// by Nisan and Schocken, MIT Pr
// File name: projects/05/CPU.hc

/**
 * The Central Processing unit (
 * Consists of an ALU and a set
 * execute instructions written
 * In particular, the ALU execut
 * to the Hack machine language
 * The D and A in the language s
 * while M refers to the memory
 * The inM input holds the value
```

Name	Value
Ainstruction	1
Cinstruction	0
ALUtoA	0
ALUout[16]	1
Aregin[16]	32767
loadA	1
Aout[16]	32767
AMout[16]	11111
loadD	0
Dout[16]	1
ZRout	0
NGout	0
jeq	0

```
set instruction %B1110001100000110, // D;JLE
tick, output, tock, output;

set instruction %B1110001100000111, // D;JMP
tick, output, tock, output;

set instruction %B1110111111010000, // D=1
tick, output, tock, output;

set instruction %B1110001100000001, // D;JGT
tick, output, tock, output;

set instruction %B1110001100000010, // D;JEQ
tick, output, tock, output;

set instruction %B1110001100000011, // D;JGE
tick, output, tock, output;

set instruction %B1110001100000100, // D;JLT
tick, output, tock, output;

set instruction %B1110001100000101, // D;JNE
tick, output, tock, output;

set instruction %B1110001100000110, // D;JLE
tick, output, tock, output;

set instruction %B1110001100000111, // D;JMP
tick, output, tock, output;

set reset 1;
tick, output, tock, output;

set instruction %B0111111111111111, // @32767
set reset 0;
tick, output, tock, output;
```

End of script - Comparison ended successfully



Hardware Simulator (2.5) - C:\Users\desha\OneDrive\Desktop\Classes\CS220\nand2tetris\projects\05\Memory.hdl

Animate: Program flow

Format: Decimal

View: Screen

Chip Name : Memory (Clocked)

Time : 8

Input pins

Name	Value
in[16]	-1
load	0
address[15]	24576

Output pins

Name	Value
out[16]	0

HDL

```
// This file is part of www.nanc
// and the book "The Elements of
// by Nisan and Schocken, MIT Pr
// File name: projects/05/Memory

/**
 * This chip implements the comp
 * computer's data memory, inclu
 * The chip facilitates read anc
 *     Read:  out(t) = Memory[ac
 *     Write: If load(t-1) then
 * In words: the chip always out
 * location specified by address
 * into the memory location spec
```

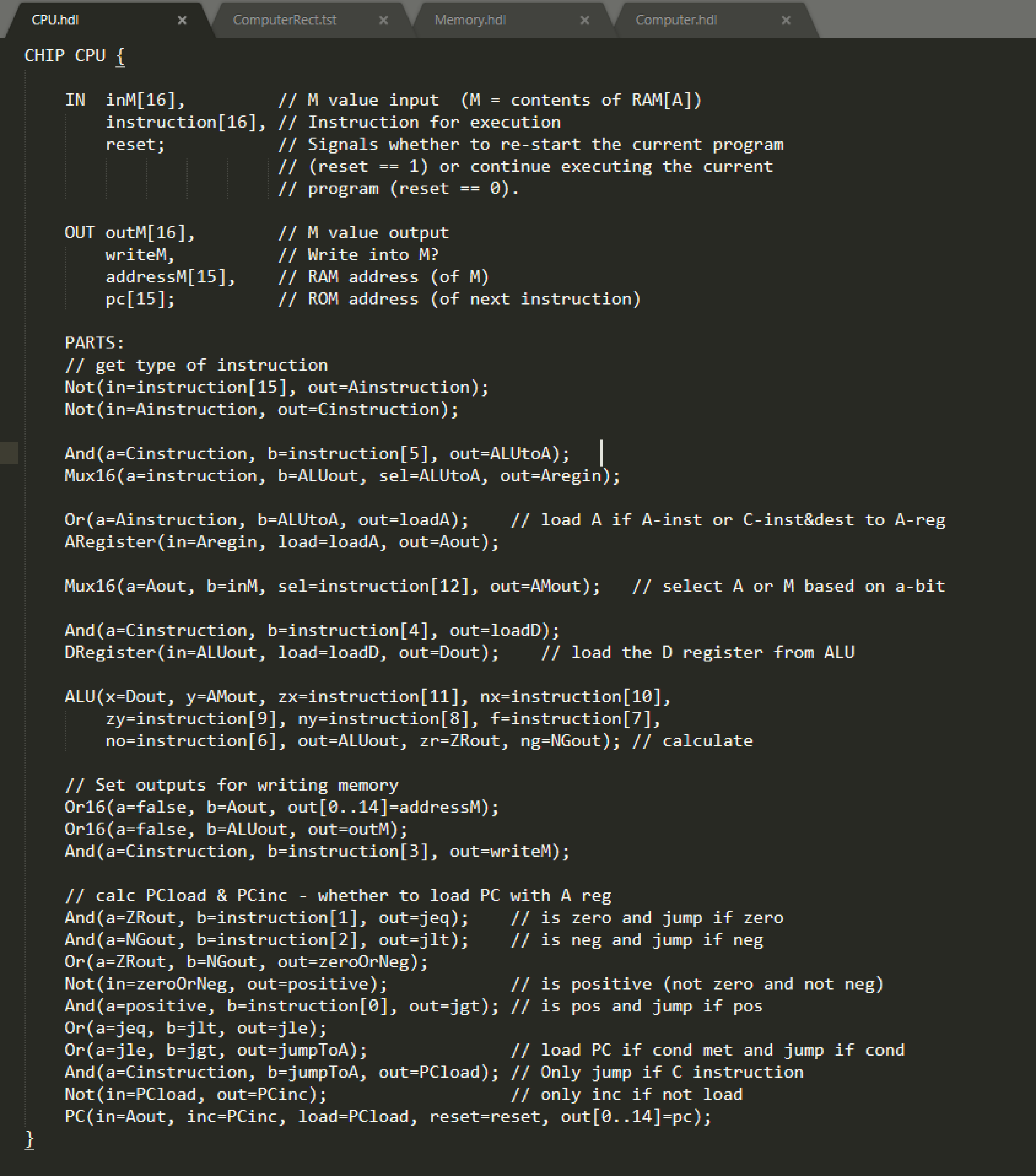
Internal pins

Name	Value
loadram1	0
loadram2	0
loadscreen	0
loadkbd	0
loadram	0
ramout[16]	2222
scrout[16]	0
kbout[16]	0

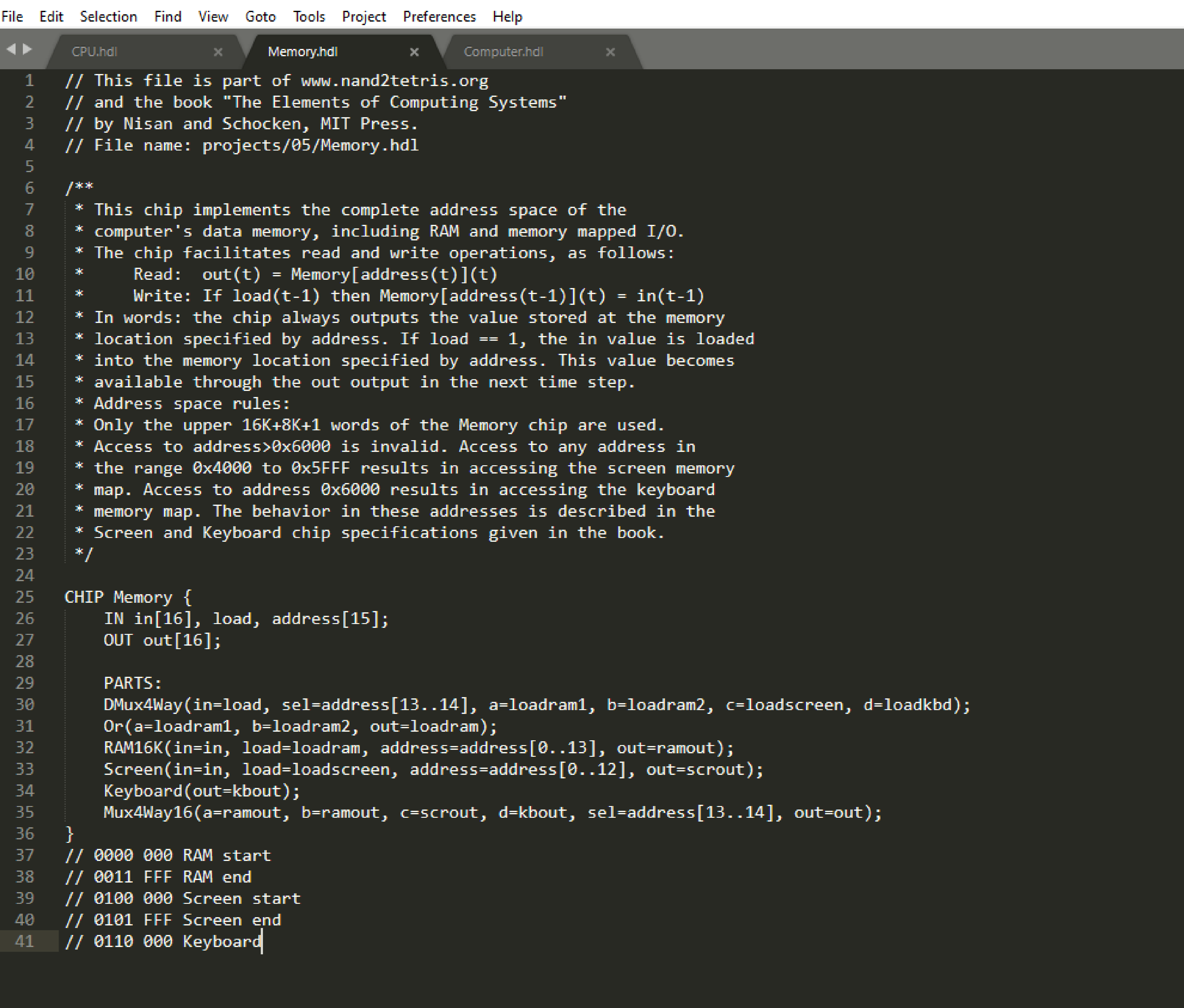
RAM 16K:

8189	0
8190	0
8191	0
8192	2222
8193	0
8194	0
8195	0

End of script - Comparison ended successfully



```
CHIP CPU {

    IN  inM[16],         // M value input  (M = contents of RAM[A])
        instruction[16], // Instruction for execution
        reset;           // Signals whether to re-start the current program
                         // (reset == 1) or continue executing the current
                         // program (reset == 0).

    OUT outM[16],        // M value output
        writeM,          // Write into M?
        addressM[15],    // RAM address (of M)
        pc[15];          // ROM address (of next instruction)

    PARTS:
    // get type of instruction
    Not(in=instruction[15], out=Ainstruction);
    Not(in=Ainstruction, out=Cinstruction);

    And(a=Cinstruction, b=instruction[5], out=ALUtoA);
    Mux16(a=instruction, b=ALUout, sel=ALUtoA, out=Aregin);

    Or(a=Ainstruction, b=ALUtoA, out=loadA);    // load A if A-inst or C-inst&dest to A-reg
    ARegister(in=Aregin, load=loadA, out=Aout);

    Mux16(a=Aout, b=inM, sel=instruction[12], out=AMout);   // select A or M based on a-bit

    And(a=Cinstruction, b=instruction[4], out=loadD);
    DRegister(in=ALUout, load=loadD, out=Dout);    // load the D register from ALU

    ALU(x=Dout, y=AMout, zx=instruction[11], nx=instruction[10],
        zy=instruction[9], ny=instruction[8], f=instruction[7],
        no=instruction[6], out=ALUout, zr=ZRout, ng=NGout); // calculate

    // Set outputs for writing memory
    Or16(a=false, b=Aout, out[0..14]=addressM);
    Or16(a=false, b=ALUout, out=outM);
    And(a=Cinstruction, b=instruction[3], out=writeM);

    // calc PCload & PCinc - whether to load PC with A reg
    And(a=ZRout, b=instruction[1], out=jeq);    // is zero and jump if zero
    And(a=NGout, b=instruction[2], out=jlt);    // is neg and jump if neg
    Or(a=ZRout, b=NGout, out=zeroOrNeg);
    Not(in=zeroOrNeg, out=positive);            // is positive (not zero and not neg)
    And(a=positive, b=instruction[0], out=jgt); // is pos and jump if pos
    Or(a=jeq, b=jlt, out=jle);
    Or(a=jle, b=jgt, out=jumpToA);              // load PC if cond met and jump if cond
    And(a=Cinstruction, b=jumpToA, out=PCload); // Only jump if C instruction
    Not(in=PCload, out=PCinc);                  // only inc if not load
    PC(in=Aout, inc=PCinc, load=PCload, reset=reset, out[0..14]=pc);
}
```



```
// This file is part of www.nand2tetris.org
// and the book "The Elements of Computing Systems"
// by Nisan and Schocken, MIT Press.
// File name: projects/05/Memory.hdl

/**
 * This chip implements the complete address space of the
 * computer's data memory, including RAM and memory mapped I/O.
 * The chip facilitates read and write operations, as follows:
 *     Read:  out(t) = Memory[address(t)](t)
 *     Write: If load(t-1) then Memory[address(t-1)](t) = in(t-1)
 * In words: the chip always outputs the value stored at the memory
 * location specified by address. If load == 1, the in value is loaded
 * into the memory location specified by address. This value becomes
 * available through the out output in the next time step.
 * Address space rules:
 * Only the upper 16K+8K+1 words of the Memory chip are used.
 * Access to address>0x6000 is invalid. Access to any address in
 * the range 0x4000 to 0x5FFF results in accessing the screen memory
 * map. Access to address 0x6000 results in accessing the keyboard
 * memory map. The behavior in these addresses is described in the
 * Screen and Keyboard chip specifications given in the book.
 */

CHIP Memory {
    IN in[16], load, address[15];
    OUT out[16];

    PARTS:
    DMux4Way(in=load, sel=address[13..14], a=loadram1, b=loadram2, c=loadscreen, d=loadkbd);
    Or(a=loadram1, b=loadram2, out=loadram);
    RAM16K(in=in, load=loadram, address=address[0..13], out=ramout);
    Screen(in=in, load=loadscreen, address=address[0..12], out=scrout);
    Keyboard(out=kbout);
    Mux4Way16(a=ramout, b=ramout, c=scrout, d=kbout, sel=address[13..14], out=out);
}
// 0000 000 RAM start
// 0011 FFF RAM end
// 0100 000 Screen start
// 0101 FFF Screen end
// 0110 000 Keyboard
```