



Cal Poly Pomona

Advanced Computer Architecture (CS 5250.01)

Project # 1

Talha Ahmed

018960110

Submitted to: Professor Nima Davarpanah

Dated: Friday, 30 January 2026

Table of Contents

Solution	4
NOT Gate	4
Code	4
Result	4
AND Gate	4
Code	4
Result	5
OR Gate	5
Code	5
Result	6
XOR Gate	6
Code	6
Result	7
MUX Gate	7
Code	7
Result	8
DMUX Gate	8
Code	8
Result	9
NOT16 Gate	9
Code	9
Result	10
AND16 Gate	10
Code	10
Result	11
OR16 Gate	11
Code	11
Result	12
MUX16 Gate	13
Code	13
Result	14
OR8WAY Gate	14
Code	14
Result	15
MUX4WAY16 Gate	15
Code	15
Result	16
MUX8WAY16 Gate	16
Code	16

Result	17
DMUX4WAY Gate	17
Code	17
Result	18
DMUX8WAY Gate	18
Code	18
Result	19

Solution

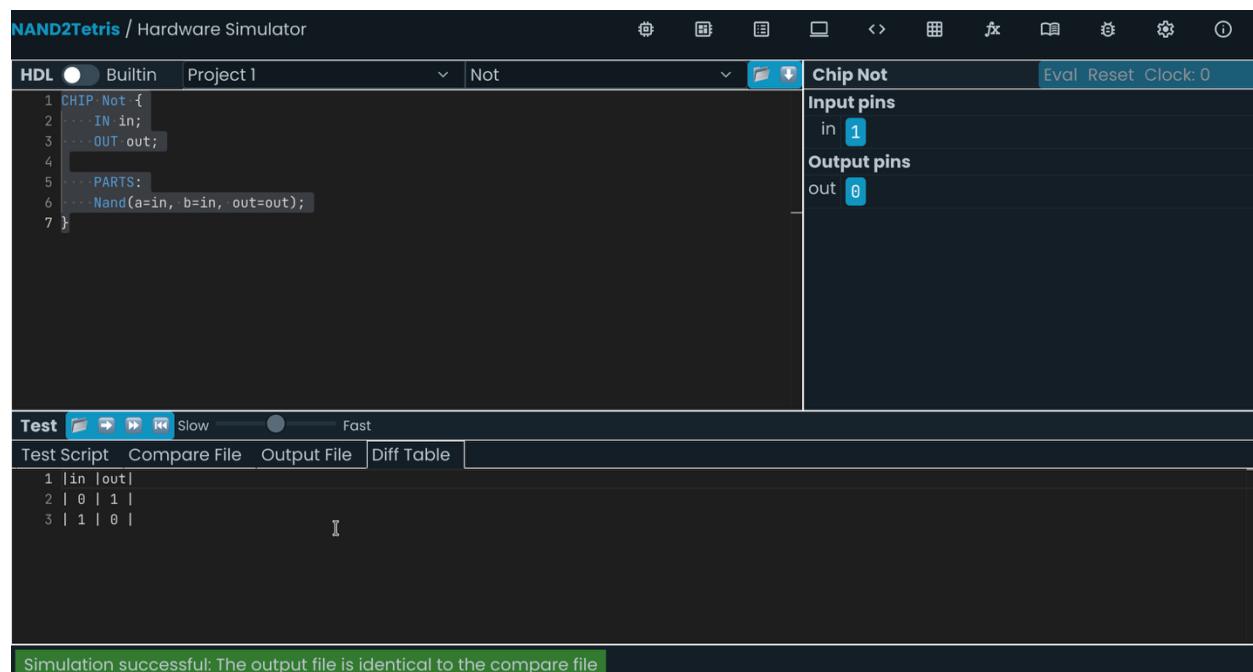
NOT Gate

Code

```
CHIP Not {
    IN in;
    OUT out;

    PARTS:
    Nand(a=in, b=in, out=out);
}
```

Result



AND Gate

Code

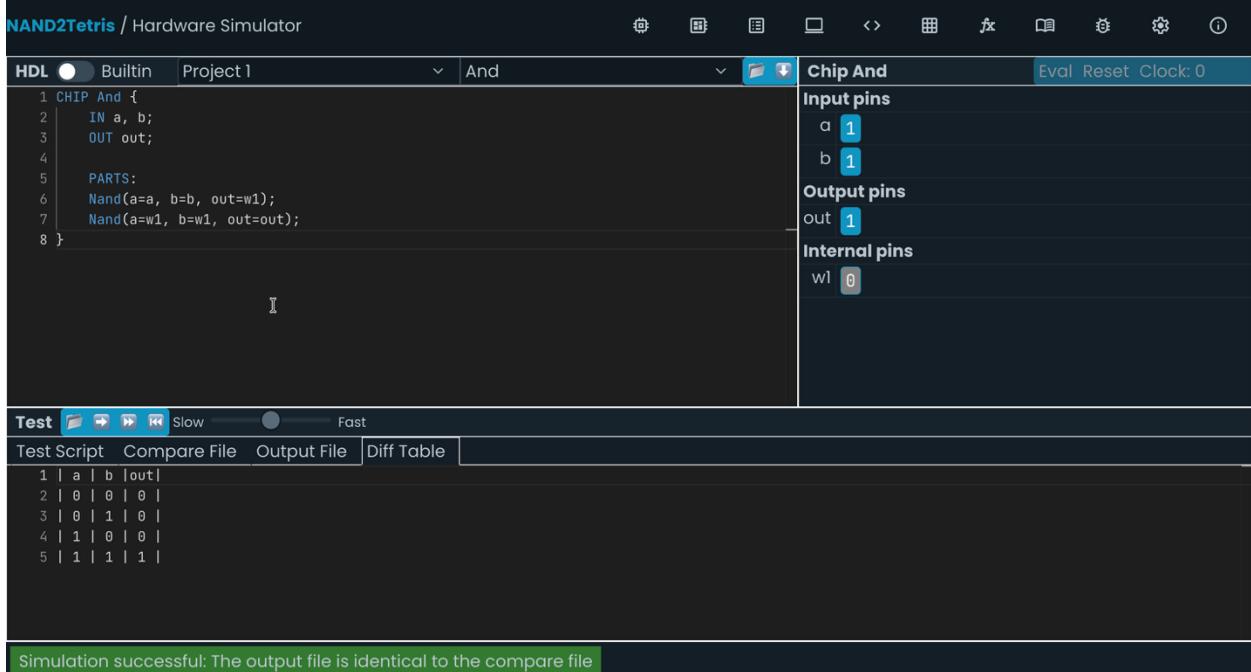
```
CHIP And {
    IN a, b;
    OUT out;
```

```

PARTS:
Nand(a=a, b=b, out=w1);
Nand(a=w1, b=w1, out=out);
}

```

Result



OR Gate

Code

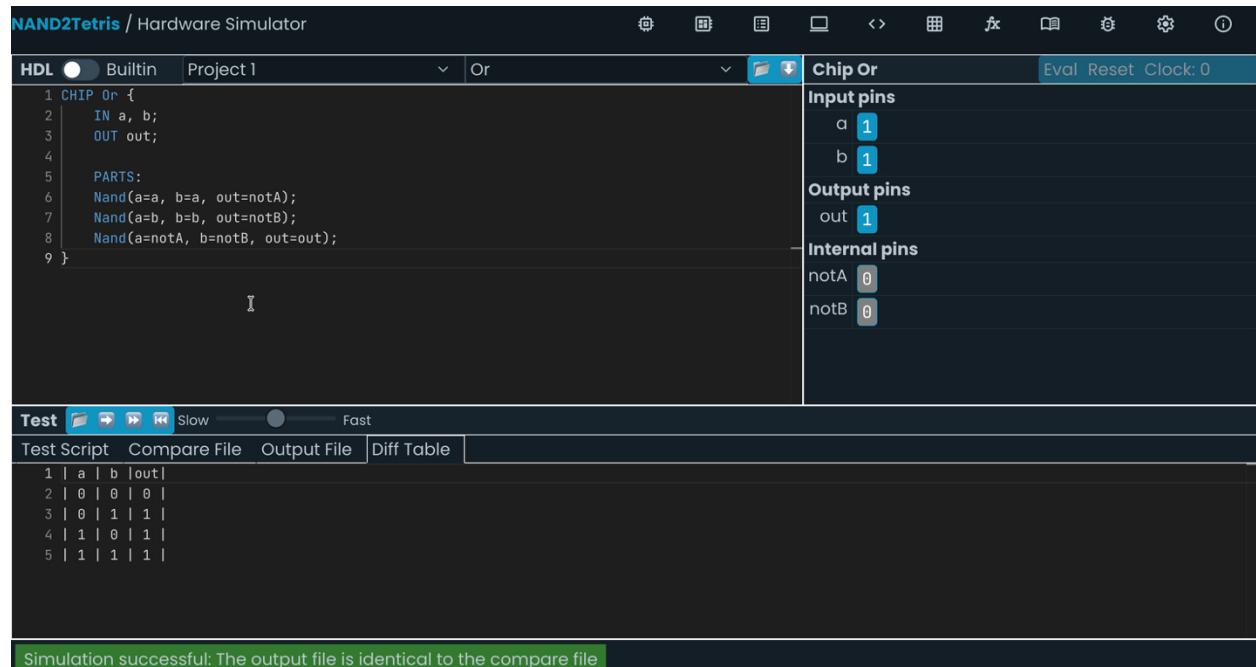
```

CHIP Or {
    IN a, b;
    OUT out;

    PARTS:
        Nand(a=a, b=a, out=notA);
        Nand(a=b, b=b, out=notB);
        Nand(a=notA, b=notB, out=out);
}

```

Result



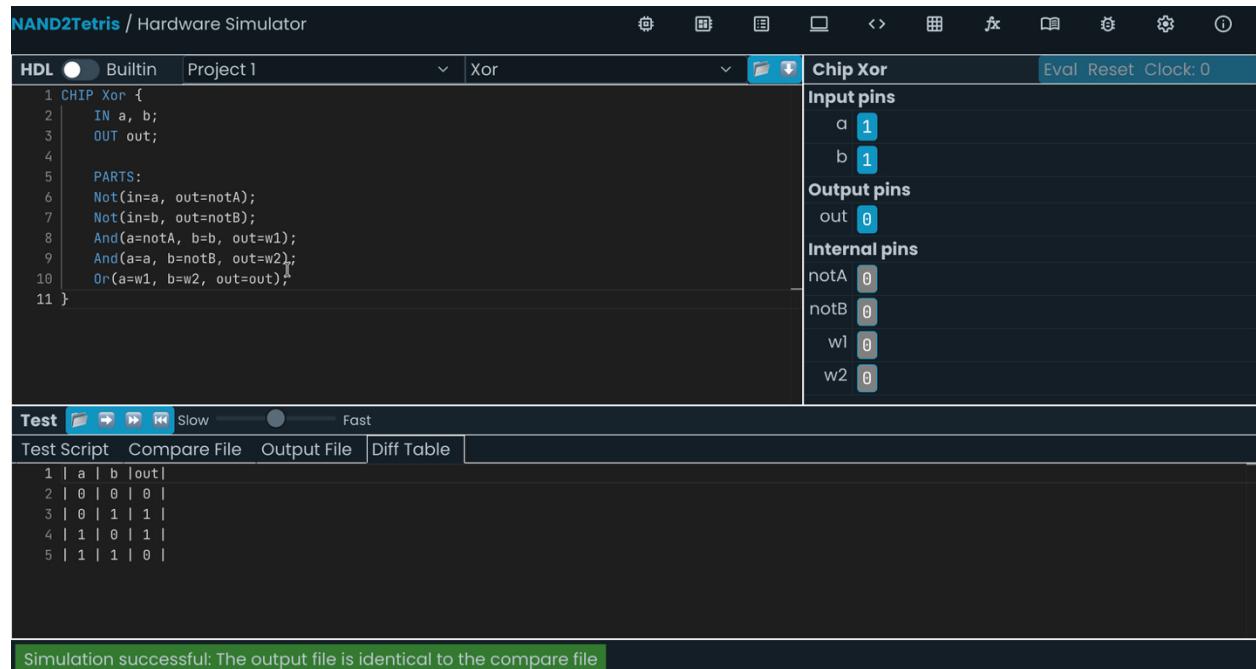
XOR Gate

Code

```
CHIP Xor {
    IN a, b;
    OUT out;

    PARTS:
        Not(in=a, out=notA);
        Not(in=b, out=notB);
        And(a=notA, b=b, out=w1);
        And(a=a, b=notB, out=w2);
        Or(a=w1, b=w2, out=out);
}
```

Result



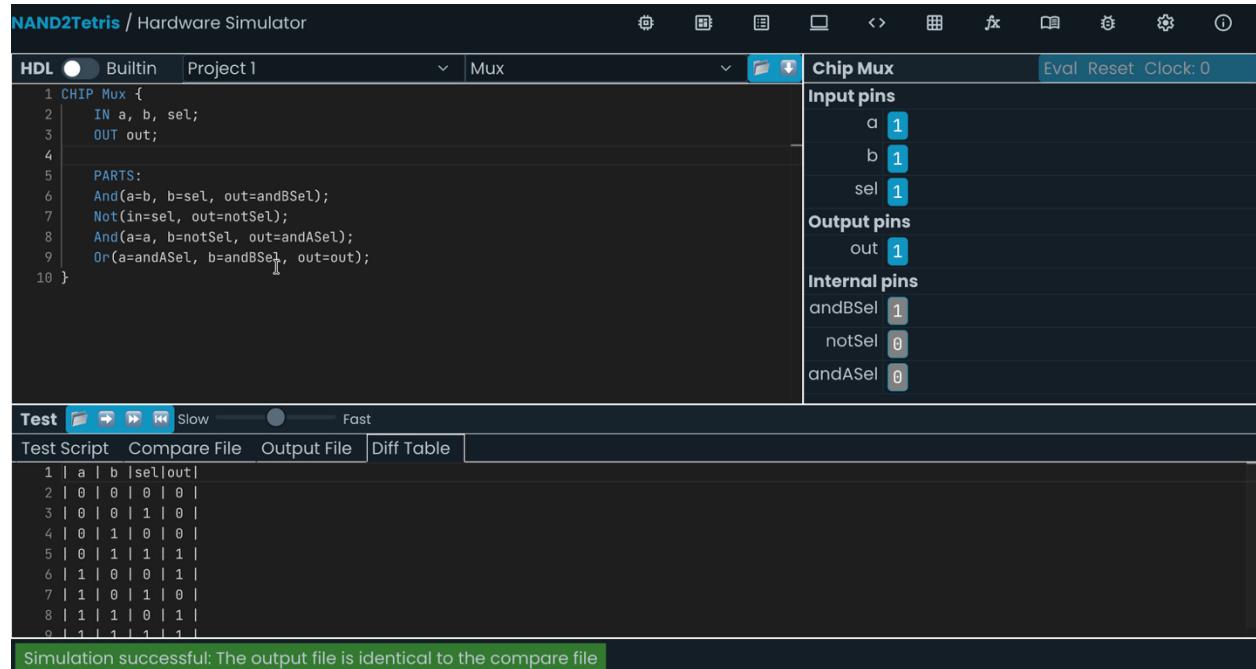
MUX Gate

Code

```
CHIP Mux {
    IN a, b, sel;
    OUT out;

    PARTS:
    And(a=b, b=sel, out=andBSel);
    Not(in=sel, out=notSel);
    And(a=a, b=notSel, out=andASel);
    Or(a=andASel, b=andBSel, out=out);
}
```

Result

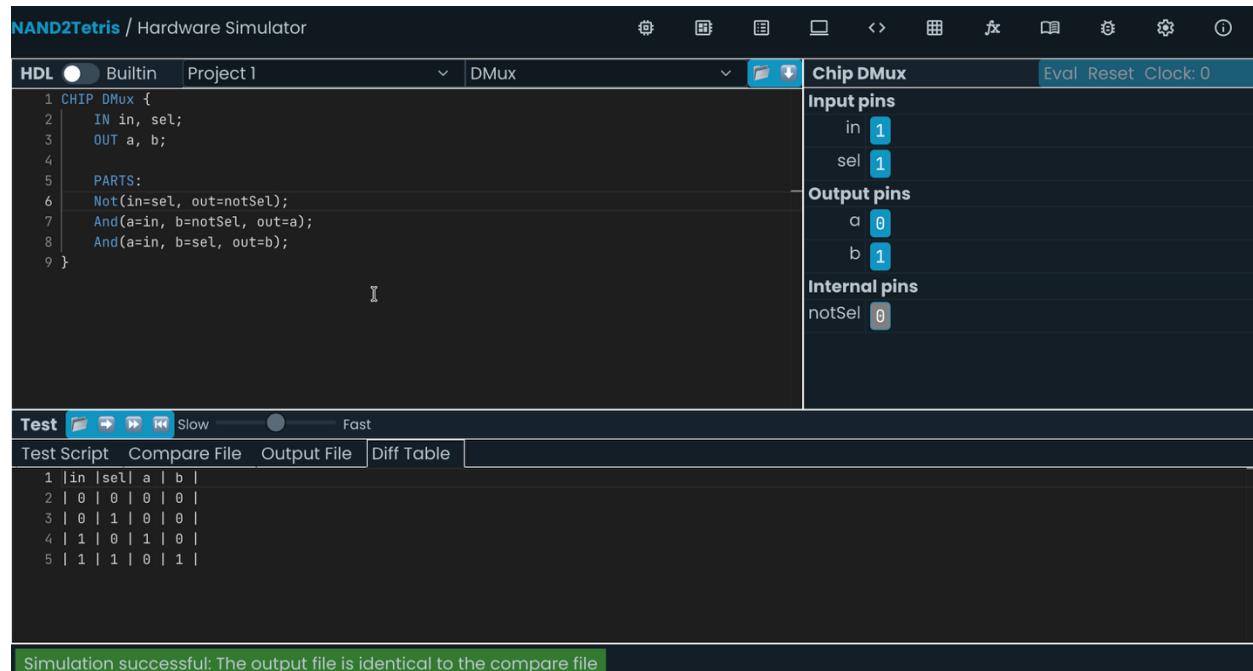


DMUX Gate

Code

```
CHIP DMux {  
IN in, sel;  
OUT a, b;  
  
PARTS:  
Not(in=sel, out=notSel);  
And(a=in, b=notSel, out=a);  
And(a=in, b=sel, out=b);  
}
```

Result



NOT16 Gate

Code

```
CHIP Not16 {
    IN in[16];
    OUT out[16];

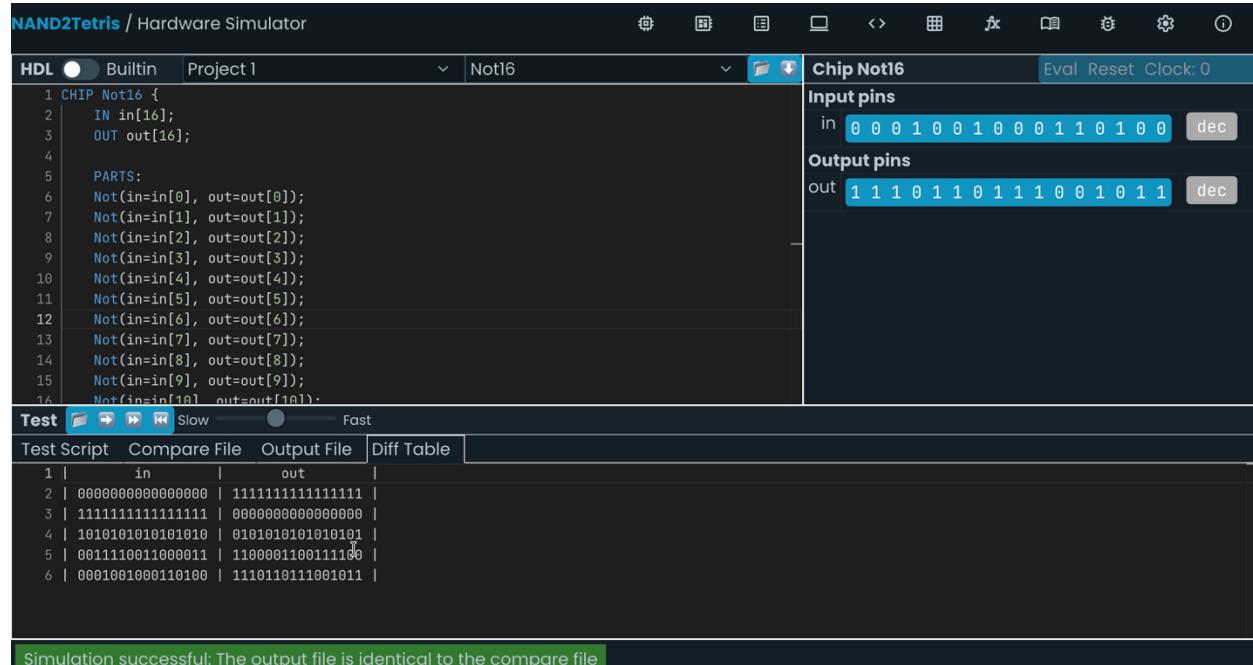
    PARTS:
        Not(in=in[0], out=out[0]);
        Not(in=in[1], out=out[1]);
        Not(in=in[2], out=out[2]);
        Not(in=in[3], out=out[3]);
        Not(in=in[4], out=out[4]);
        Not(in=in[5], out=out[5]);
        Not(in=in[6], out=out[6]);
        Not(in=in[7], out=out[7]);
        Not(in=in[8], out=out[8]);
        Not(in=in[9], out=out[9]);
        Not(in=in[10], out=out[10]);
        Not(in=in[11], out=out[11]);
```

```

    Not(in=in[12], out=out[12]);
    Not(in=in[13], out=out[13]);
    Not(in=in[14], out=out[14]);
    Not(in=in[15], out=out[15]);
}

```

Result



AND16 Gate

Code

```

CHIP And16 {
    IN a[16], b[16];
    OUT out[16];

    PARTS:
    And(a=a[0], b=b[0], out=out[0]);
    And(a=a[1], b=b[1], out=out[1]);
    And(a=a[2], b=b[2], out=out[2]);
    And(a=a[3], b=b[3], out=out[3]);
    And(a=a[4], b=b[4], out=out[4]);
    And(a=a[5], b=b[5], out=out[5]);
}

```

```

    And(a=a[6], b=b[6], out=out[6]);
    And(a=a[7], b=b[7], out=out[7]);
    And(a=a[8], b=b[8], out=out[8]);
    And(a=a[9], b=b[9], out=out[9]);
    And(a=a[10], b=b[10], out=out[10]);
    And(a=a[11], b=b[11], out=out[11]);
    And(a=a[12], b=b[12], out=out[12]);
    And(a=a[13], b=b[13], out=out[13]);
    And(a=a[14], b=b[14], out=out[14]);
    And(a=a[15], b=b[15], out=out[15]);
}

```

Result

NAND2Tetris / Hardware Simulator

CHIP And16 {
 IN a[16], b[16];
 OUT out[16];
}

PARTS:
 And(a=a[0], b=b[0], out=out[0]);
 And(a=a[1], b=b[1], out=out[1]);
 And(a=a[2], b=b[2], out=out[2]);
 And(a=a[3], b=b[3], out=out[3]);
 And(a=a[4], b=b[4], out=out[4]);
 And(a=a[5], b=b[5], out=out[5]);
 And(a=a[6], b=b[6], out=out[6]);
 And(a=a[7], b=b[7], out=out[7]);
 And(a=a[8], b=b[8], out=out[8]);
 And(a=a[9], b=b[9], out=out[9]);
 And(a=a[10], b=b[10], out=out[10]);
}

Chip And16

Input pins

a	0 0 0 1 0 0 1 0 0 0 1 1 0 1 0 0	dec
b	1 0 0 1 1 0 0 0 0 1 1 1 0 1 1 0	dec

Output pins

out	0 0 0 1 0 0 0 0 0 0 1 1 0 1 0 0	dec
-----	---------------------------------	-----

Test Script Compare File Output File Diff Table

	a	b	out
1	0000000000000000	0000000000000000	0000000000000000
2	0000000000000000	1111111111111111	0000000000000000
3	1111111111111111	1111111111111111	1111111111111111
4	1010101010101010	0101010101010101	0000000000000000
5	0011110011000011	0000111111110000	0000110011000000
6	0001001000110100	1001100001110110	000100000110100
7			

Simulation successful: The output file is identical to the compare file

OR16 Gate

Code

```

CHIP Or16 {
    IN a[16], b[16];
    OUT out[16];
}

```

PARTS:

```

Or(a=a[0], b=b[0], out=out[0]);
Or(a=a[1], b=b[1], out=out[1]);
Or(a=a[2], b=b[2], out=out[2]);
Or(a=a[3], b=b[3], out=out[3]);
Or(a=a[4], b=b[4], out=out[4]);
Or(a=a[5], b=b[5], out=out[5]);
Or(a=a[6], b=b[6], out=out[6]);
Or(a=a[7], b=b[7], out=out[7]);
Or(a=a[8], b=b[8], out=out[8]);
Or(a=a[9], b=b[9], out=out[9]);
Or(a=a[10], b=b[10], out=out[10]);
Or(a=a[11], b=b[11], out=out[11]);
Or(a=a[12], b=b[12], out=out[12]);
Or(a=a[13], b=b[13], out=out[13]);
Or(a=a[14], b=b[14], out=out[14]);
Or(a=a[15], b=b[15], out=out[15]);
}

```

Result

The screenshot shows the NAND2Tetris Hardware Simulator interface. The top bar displays 'NAND2Tetris / Hardware Simulator'. The main window is divided into several sections:

- Code Editor:** Shows the Verilog code for the `Or16` chip, which includes a `CHIP` block defining inputs `a` and `b`, output `out`, and 16 `Or` gates for each bit position.
- Chip Configuration:** A table for the `Or16` chip with columns for **Input pins** and **Output pins**. The **Input pins** table shows two 16-bit binary values: `a` (0001001000100010) and `b` (1001100000000000). The **Output pins** table shows the resulting 16-bit output `out` (1001110000000000).
- Test Section:** Includes a **Test** button, a speed slider from **Slow** to **Fast**, and a table for running a test script. The table has columns for **Test Script**, **Compare File**, **Output File**, and **Diff Table**. The **Diff Table** section shows a 7x4 grid of binary values, with the first row being 1 | a | b | out |.
- Status Bar:** At the bottom, a green bar indicates "Simulation successful: The output file is identical to the compare file".

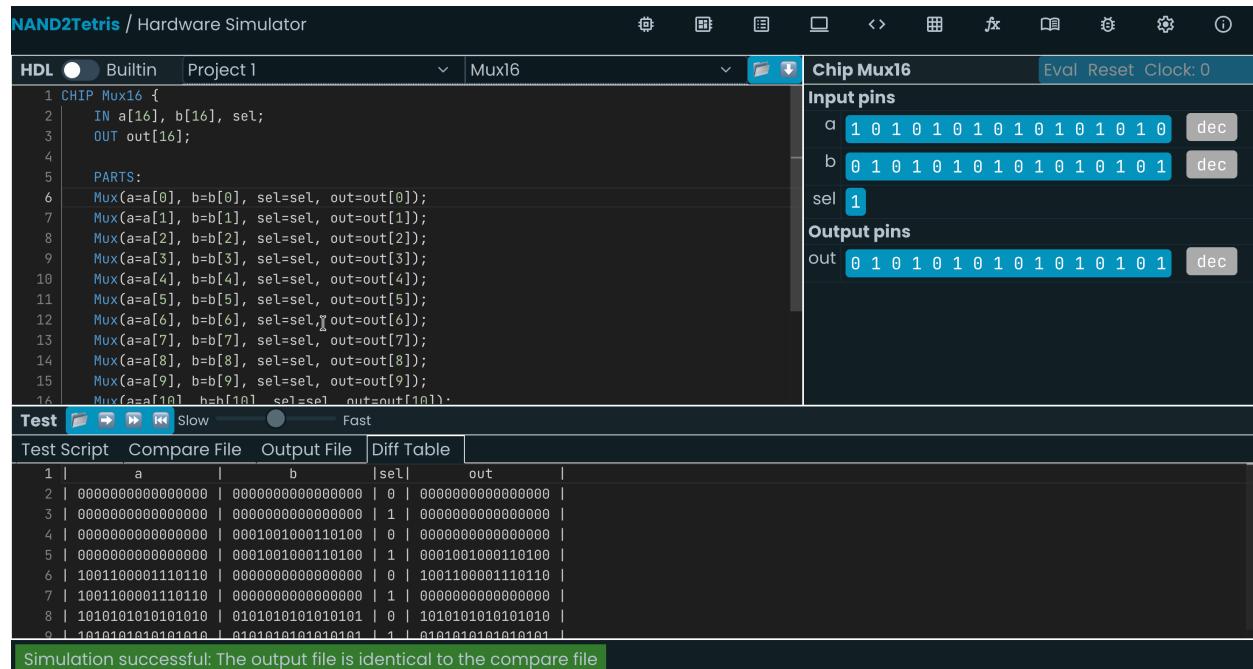
MUX16 Gate

Code

```
CHIP Mux16 {
    IN a[16], b[16], sel;
    OUT out[16];

    PARTS:
    Mux(a=a[0], b=b[0], sel=sel, out=out[0]);
    Mux(a=a[1], b=b[1], sel=sel, out=out[1]);
    Mux(a=a[2], b=b[2], sel=sel, out=out[2]);
    Mux(a=a[3], b=b[3], sel=sel, out=out[3]);
    Mux(a=a[4], b=b[4], sel=sel, out=out[4]);
    Mux(a=a[5], b=b[5], sel=sel, out=out[5]);
    Mux(a=a[6], b=b[6], sel=sel, out=out[6]);
    Mux(a=a[7], b=b[7], sel=sel, out=out[7]);
    Mux(a=a[8], b=b[8], sel=sel, out=out[8]);
    Mux(a=a[9], b=b[9], sel=sel, out=out[9]);
    Mux(a=a[10], b=b[10], sel=sel, out=out[10]);
    Mux(a=a[11], b=b[11], sel=sel, out=out[11]);
    Mux(a=a[12], b=b[12], sel=sel, out=out[12]);
    Mux(a=a[13], b=b[13], sel=sel, out=out[13]);
    Mux(a=a[14], b=b[14], sel=sel, out=out[14]);
    Mux(a=a[15], b=b[15], sel=sel, out=out[15]);
}
```

Result



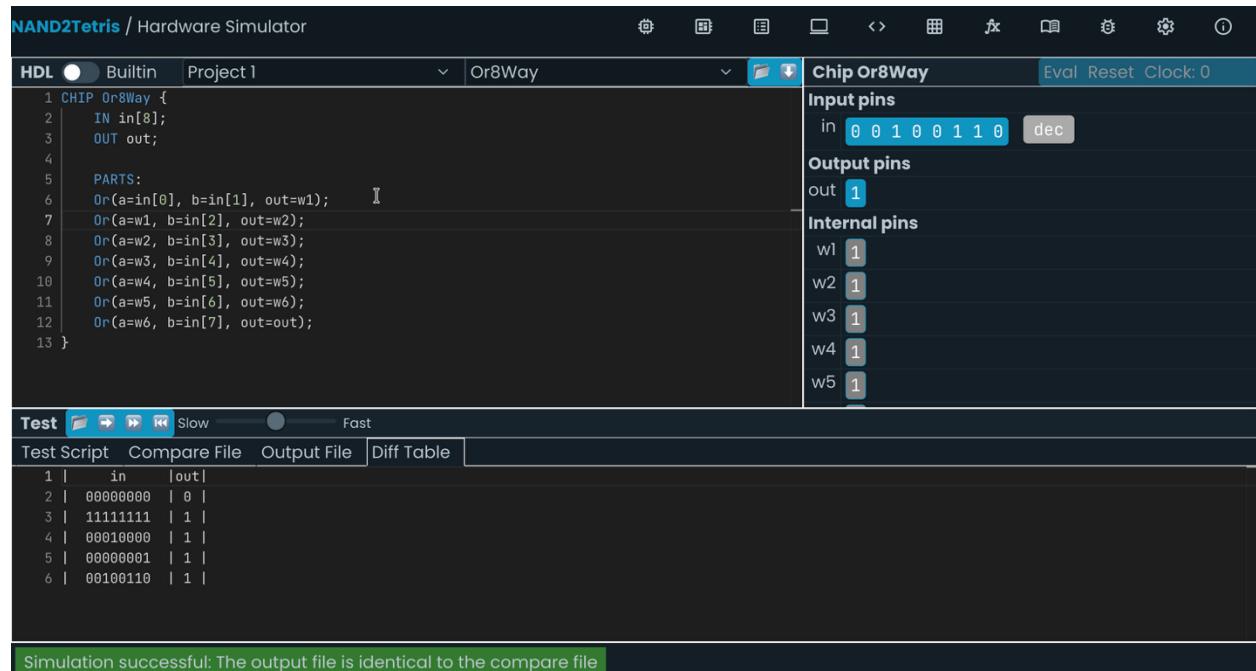
OR8WAY Gate

Code

```
CHIP Or8Way {
    IN in[8];
    OUT out;

    PARTS:
    Or(a=in[0], b=in[1], out=w1);
    Or(a=w1, b=in[2], out=w2);
    Or(a=w2, b=in[3], out=w3);
    Or(a=w3, b=in[4], out=w4);
    Or(a=w4, b=in[5], out=w5);
    Or(a=w5, b=in[6], out=w6);
    Or(a=w6, b=in[7], out=out);
}
```

Result



MUX4WAY16 Gate

Code

```
CHIP Mux4Way16 {
    IN a[16], b[16], c[16], d[16], sel[2];
    OUT out[16];

    PARTS:
    Mux16(a=a, b=b, sel=sel[0], out=muxAB);
    Mux16(a=c, b=d, sel=sel[0], out=muxCD);
    Mux16(a=muxAB, b=muxCD, sel=sel[1], out=out);
}
```

Result

NAND2Tetris / Hardware Simulator

Chip Mux4Way16 Eval Reset Clock: 0

Input pins

a	0 0 0 1 0 0 1 0 0 0 1 1 0 1 0 0	dec
b	1 0 0 1 1 0 0 0 0 1 1 1 0 1 1 0	dec
c	1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0	dec
d	0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1	dec
sel	1 1	dec

Output pins

out	0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1	dec
-----	---------------------------------	-----

Internal pins

muxAB	1 0 0 1 1 0 0 0 0 1 1 0 1 0 1 0	dec
-------	---------------------------------	-----

Test Script Slow Fast

Diff Table

	a	b	c	d	sel	out
1	0000000000000000	0000000000000000	0000000000000000	0000000000000000	00	0000000000000000
2	0000000000000000	0000000000000000	0000000000000000	0000000000000000	01	0000000000000000
3	0000000000000000	0000000000000000	0000000000000000	0000000000000000	10	0000000000000000
4	0000000000000000	0000000000000000	0000000000000000	0000000000000000	11	0000000000000000
5	0001001000110100	100110001110110	1010101010101010	0101010101010101	00	0001001000110100
6	0001001000110100	100110001110110	1010101010101010	0101010101010101	01	100110001110110
7	0001001000110100	100110001110110	1010101010101010	0101010101010101	10	1010101010101010
8	0001001000110100	100110001110110	1010101010101010	0101010101010101	11	0101010101010101

Simulation successful: The output file is identical to the compare file

MUX8WAY16 Gate

Code

```
CHIP Mux8Way16 {
    IN a[16], b[16], c[16], d[16],
        e[16], f[16], g[16], h[16],
        sel[3];
    OUT out[16];

    PARTS:
        Mux4Way16(a=a, b=b, c=c, d=d, sel=sel[0..1], out=w1);
        Mux4Way16(a=e, b=f, c=g, d=h, sel=sel[0..1], out=w2);
        Mux16(a=w1, b=w2, sel=sel[2], out=out);
}
```

Result

NAND2Tetris / Hardware Simulator

HDL **Builtin** **Project1** **Mux8Way16** **Chip Mux8Way16** **Eval** **Reset** **Clock: 0**

```

1 CHIP Mux8Way16 {
2     IN a[16], b[16], c[16], d[16],
3     | e[16], f[16], g[16], h[16],
4     | sel[3];
5     OUT out[16];
6
7     PARTS:
8     Mux4Way16(a=a, b=b, c=c, d=d, sel=sel[0..1], out=w1);
9     Mux4Way16(a=e, b=f, c=g, d=h, sel=sel[0..1], out=w2);
10    Mux16(a=w1, b=w2, sel=sel[2], out=out);
11 }

```

Input pins

d	0 0 0 1 0 0 1 0 0 0 1 1 0 1 0 0	dec
b	0 0 1 0 0 0 1 1 0 1 0 0 0 1 0 1	dec
c	0 0 1 1 0 1 0 0 0 1 0 1 0 1 1 0	dec
d	0 1 0 0 0 1 0 1 0 1 0 0 1 1 1 1	dec
e	0 1 0 1 0 1 1 0 0 1 1 1 1 0 0 0	dec
f	0 1 1 0 0 1 1 1 1 0 0 0 1 0 0 1	dec
g	0 1 1 1 1 0 0 0 1 0 0 1 1 0 1 0	dec
h	1 0 0 0 1 0 0 1 1 0 1 0 1 0 1 1	dec

Test **Slow** **Fast**

Test Script **Compare File** **Output File** **Diff Table**

	a	b	c	d	e	f	g	h
2	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000
3	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000
4	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000
5	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000
6	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000
7	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000
8	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000
9	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000

Simulation successful: The output file is identical to the compare file

DMUX4WAY Gate

Code

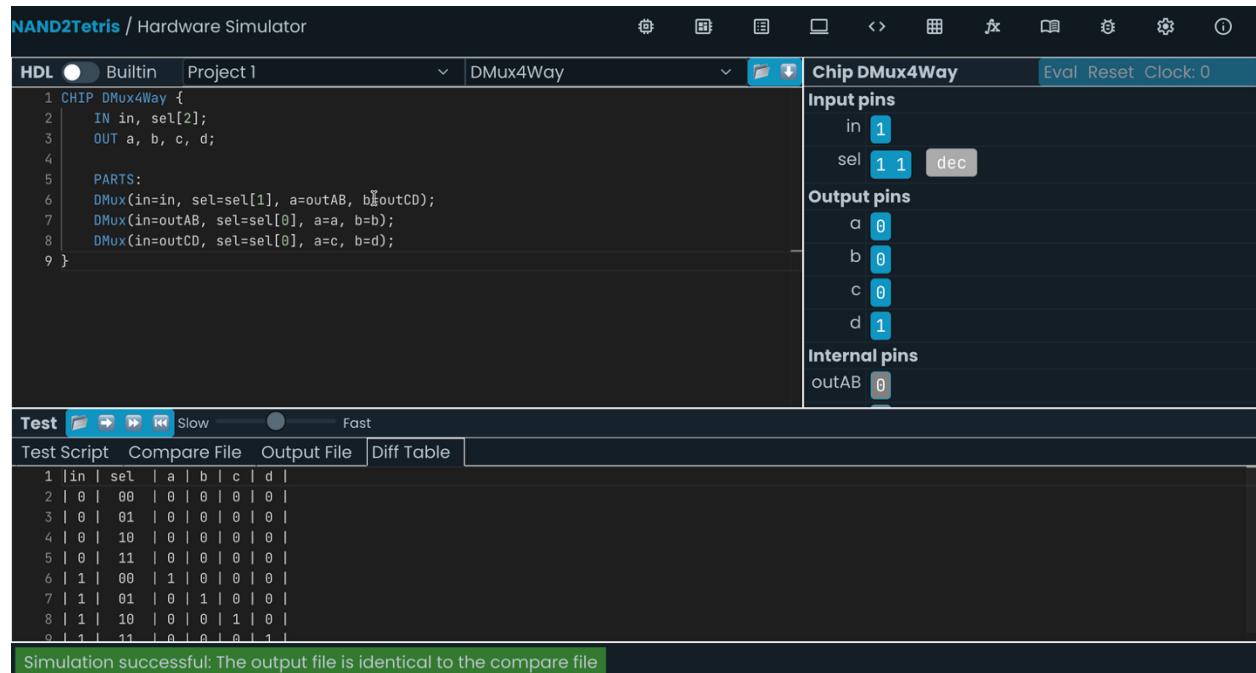
```

CHIP DMux4Way {
    IN in, sel[2];
    OUT a, b, c, d;

    PARTS:
    DMux(in=in, sel=sel[1], a=outAB, b=outCD);
    DMux(in=outAB, sel=sel[0], a=a, b=b);
    DMux(in=outCD, sel=sel[0], a=c, b=d);
}

```

Result



DMUX8WAY Gate

Code

```
CHIP DMux8Way {
    IN in, sel[3];
    OUT a, b, c, d, e, f, g, h;

    PARTS:
    DMux(in=in, sel=sel[2], a=outABCD, b=outEFGH);
    DMux4Way(in=outABCD, sel=sel[0..1], a=a, b=b, c=c, d=d);
    DMux4Way(in=outEFGH, sel=sel[0..1], a=e, b=f, c=g, d=h);
}
```

Result

NAND2Tetris / Hardware Simulator

HDL Builtin Project 1 ▾ DMux8Way ▾ Chip DMux8Way Eval Reset Clock: 0

```
1 CHIP DMux8Way {
2     IN in, sel[3];
3     OUT a, b, c, d, e, f, g, h;
4
5     PARTS:
6     DMux(in=in, sel=sel[2], a=outABCD, b=outEFGH);
7     DMux4Way(in=outABCD, sel=sel[0..1], a=a, b=b, c=c, d=d);
8     DMux4Way(in=outEFGH, sel=sel[0..1], a=e, b=f, c=g, d=h);
9 }
```

Input pins

in	1
sel	1 1 1

Output pins

a	0
b	0
c	0
d	0
e	0
f	0

Test Slow Fast

Test Script Compare File Output File Diff Table

	in	sel	a	b	c	d	e	f	g	h
1	0	000	0	0	0	0	0	0	0	0
2	0	001	0	0	0	0	0	0	0	0
3	0	010	0	0	0	0	0	0	0	0
4	0	011	0	0	0	0	0	0	0	0
5	0	100	0	0	0	0	0	0	0	0
6	0	101	0	0	0	0	0	0	0	0
7	0	110	0	0	0	0	0	0	0	0
8	0	111	0	0	0	0	0	0	0	0

Simulation successful: The output file is identical to the compare file