

BIT CHIP:

```
CHIP Bit {
    IN in, load;
    OUT out;

    PARTS:
        // Use the Muxer to repeat values when load is ON and use DFF to relay
previous
        Mux(a=loop, b=in, sel=load, out=s);
        DFF(in=s, out=out, out=loop);
}
```

BIT TEST:

The screenshot shows the Hardware Simulator (2.5) window. The title bar indicates the file path: C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\03\a\Bit.hdl.

File View Run Help

Simulation Controls: Animate (Program flow), Format (Decimal), View (Script). Speed controls: Slow, Fast.

Chip Name: Bit (Clocked) **Time:** 107

Input pins		Output pins	
Name	Value	Name	Value
in	1	out	0
load	0		

HDL		Internal pins	
Name	Value	Name	Value
loop	0	s	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/03/a/Bit.hdl

/**
 * 1-bit register:
 * If load[t] == 1 then out[t+1] = in[t]
 * else out does nothing
 */

CHIP Bit {
    IN in, load;
    OUT out;
```

Script Output:

```
output;
tock,
output;

set in 1,
set load 0,
tick,
output;

tock,
output;

set in 1,
set load 0,
tick,
output;

tock,
output;

set in 1,
set load 0,
tick,
output;

tock,
output;

set in 1,
set load 0,
tick,
output;

tock,
output;
```

End of script - Comparison ended successfully

REGISTER CHIP:

```
CHIP Register {
    IN in[16], load;
    OUT out[16];

    PARTS:
        // 16 bit storer
        Bit(in=in[0], load=load, out=out[0]);
        Bit(in=in[1], load=load, out=out[1]);
        Bit(in=in[2], load=load, out=out[2]);
        Bit(in=in[3], load=load, out=out[3]);
        Bit(in=in[4], load=load, out=out[4]);
        Bit(in=in[5], load=load, out=out[5]);
        Bit(in=in[6], load=load, out=out[6]);
        Bit(in=in[7], load=load, out=out[7]);
        Bit(in=in[8], load=load, out=out[8]);
        Bit(in=in[9], load=load, out=out[9]);
        Bit(in=in[10], load=load, out=out[10]);
        Bit(in=in[11], load=load, out=out[11]);
        Bit(in=in[12], load=load, out=out[12]);
        Bit(in=in[13], load=load, out=out[13]);
        Bit(in=in[14], load=load, out=out[14]);
        Bit(in=in[15], load=load, out=out[15]);
}
```

REGISTER TEST:


```

Mux8Way16(a=t1, b=t2, c=t3, d=t4, e=t5, f=t6, g=t7, h=t8, sel=address,
out=out);
}

```

RAM8 TEST:

Hardware Simulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\03\RAM8.t

File View Run Help

The screenshot shows the Hardware Simulator (2.5) interface. At the top, there's a toolbar with icons for running, stepping through code, and other simulation controls. Below the toolbar, the 'Chip Name' is set to 'RAM8 (Clocked)' and the 'Time' is 46. The main window is divided into several sections: 'Input pins', 'Output pins', 'HDL', 'Internal pins', and a script editor on the right.

Input pins		Output pins	
Name	Value	Name	Value
in[16]	21845	out[16]	21845
load	0		
address[3]	7		

Internal pins	
Name	Value
s1	0
s2	0
s3	0
s4	0
s5	0
s6	0
s7	0
s8	0
t1[16]	21845
t2[16]	21845
t3[16]	21845
t4[16]	21845
t5[16]	21845

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/03/ram8/ram8.t

/**
 * Memory of 8 registers, each 16 bits wide.
 * stored at the memory locations 0..7.
 * the in value is loaded into t1.
 * (the loaded value will be emitted to out)
 */

CHIP RAM8 {
    IN in[16], load, address[3];
    OUT out[16];

    PARTS:
        // Putting 8 8-bit RAMs together
        DMux8Way(in=load, sel=address[0..2], a=s1, b=s2, c=s3, d=s4, e=s5,
f=s6, g=s7, h=s8);
        RAM8(in=in, load=s1, address=address[3..5], out=t1);
        RAM8(in=in, load=s2, address=address[3..5], out=t2);
        RAM8(in=in, load=s3, address=address[3..5], out=t3);
        RAM8(in=in, load=s4, address=address[3..5], out=t4);
        RAM8(in=in, load=s5, address=address[3..5], out=t5);
        RAM8(in=in, load=s6, address=address[3..5], out=t6);
        RAM8(in=in, load=s7, address=address[3..5], out=t7);
        RAM8(in=in, load=s8, address=address[3..5], out=t8);
        Mux8Way16(a=t1, b=t2, c=t3, d=t4, e=t5, f=t6, g=t7, h=t8, sel=address,
out=out);
}

```

Script Editor:

```

set load 1,
set address 7,
set in %B0101010101010101,
tick,
output,
tock,
output;

set load 0,
set address 0,
tick,
output;
tock,
output;
set address 1,
eval,
output;
set address 2,
eval,
output;
set address 3,
eval,
output;
set address 4,
eval,
output;
set address 5,
eval,
output;
set address 6,
eval,
output;
set address 7,
eval,
output;

```

End of script - Comparison ended successfully

RAM64 CHIP:

```

CHIP RAM64 {
    IN in[16], load, address[6];
    OUT out[16];

    PARTS:
        // Putting 8 8-bit RAMs together
        DMux8Way(in=load, sel=address[0..2], a=s1, b=s2, c=s3, d=s4, e=s5,
f=s6, g=s7, h=s8);
        RAM8(in=in, load=s1, address=address[3..5], out=t1);
        RAM8(in=in, load=s2, address=address[3..5], out=t2);
        RAM8(in=in, load=s3, address=address[3..5], out=t3);
        RAM8(in=in, load=s4, address=address[3..5], out=t4);
        RAM8(in=in, load=s5, address=address[3..5], out=t5);
        RAM8(in=in, load=s6, address=address[3..5], out=t6);
        RAM8(in=in, load=s7, address=address[3..5], out=t7);
        RAM8(in=in, load=s8, address=address[3..5], out=t8);
        Mux8Way16(a=t1, b=t2, c=t3, d=t4, e=t5, f=t6, g=t7, h=t8, sel=address,
out=out);
}

```

```

RAM8(in=in, load=s4, address=address[3..5], out=t4);
RAM8(in=in, load=s5, address=address[3..5], out=t5);
RAM8(in=in, load=s6, address=address[3..5], out=t6);
RAM8(in=in, load=s7, address=address[3..5], out=t7);
RAM8(in=in, load=s8, address=address[3..5], out=t8);
Mux8Way16(a=t1, b=t2, c=t3, d=t4, e=t5, f=t6, g=t7, h=t8,
sel=address[0..2], out=out);
}

```

RAM64 TEST:

Hardware Simulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\03\A\RAM64.

File View Run Help

Animate: Program flow
 Format: Decimal
 View: Script

Chip Name: RAM64 (Clocked)
 Time: 81

Input pins		Output pins	
Name	Value	Name	Value
in[16]	21845	out[16]	21845
load	0		
address[6]	61		

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/03/a/RAM64.nasm

/**
 * Memory of 64 registers, each
 * stored at the memory location
 * the in value is loaded into t
 * (the loaded value will be emulated)
 */
CHIP RAM64 {
    IN in[16], load, address[6];

```

Internal pins

Name	Value
s1	0
s2	0
s3	0
s4	0
s5	0
s6	0
s7	0
s8	0
t1[16]	0
t2[16]	0
t3[16]	0
t4[16]	0
t5[16]	0

```

set load 1,
set address $B111101,
set in $B0101010101010101,
tick,
output,
tock,
output;

set load 0,
set address $B000101,
tick,
output;
set address $B001101,
eval,
output;
set address $B010101,
eval,
output;
set address $B011101,
eval,
output;
set address $B100101,
eval,
output;
set address $B101101,
eval,
output;
set address $B110101,
eval,
output;
set address $B111101,
eval,
output;

```

End of script - Comparison ended successfully

RAM512 CHIP:

```

CHIP RAM512 {
    IN in[16], load, address[9];
    OUT out[16];

    PARTS:
        // Putting 8 64-bit RAMs together

```

```

    DMux8Way(in=load, sel=address[0..2], a=s1, b=s2, c=s3, d=s4, e=s5,
f=s6, g=s7, h=s8);

    RAM64(in=in, load=s1, address=address[3..8], out=t1);
    RAM64(in=in, load=s2, address=address[3..8], out=t2);
    RAM64(in=in, load=s3, address=address[3..8], out=t3);
    RAM64(in=in, load=s4, address=address[3..8], out=t4);
    RAM64(in=in, load=s5, address=address[3..8], out=t5);
    RAM64(in=in, load=s6, address=address[3..8], out=t6);
    RAM64(in=in, load=s7, address=address[3..8], out=t7);
    RAM64(in=in, load=s8, address=address[3..8], out=t8);
    Mux8Way16(a=t1, b=t2, c=t3, d=t4, e=t5, f=t6, g=t7, h=t8,
sel=address[0..2], out=out);
}

```

RAM512 TEST:

Hardware Simulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\03\b\RAM512

File View Run Help

The screenshot shows the Hardware Simulator (2.5) interface. At the top, there are icons for running the simulation (play, step, pause, reset) and a speed control slider (Slow to Fast). Below these are tabs for 'Animate' (Program flow), 'Format' (Decimal), and 'View' (Script). The main window is divided into several sections:

- Chip Name:** RAM512 (Clocked) | **Time:** 81
- Input pins:** A table with columns 'Name' and 'Value'. It shows 'in[16]' with value 21845, 'load' with value 0, and 'address[9]' with value 490.
- Output pins:** A table with columns 'Name' and 'Value'. It shows 'output;' with value 0.
- HDL:** A text area containing Verilog code for the RAM512 chip. It includes comments about the memory size (512 registers) and the test procedure.
- Internal pins:** A table with columns 'Name' and 'Value'. It lists internal signals s1 through s8 (all 0), t1[16] through t5[16] (all 0), and t3[16] with value 21845.
- Script:** A text area on the right showing the sequence of operations performed during the test, including setting addresses, loading data, and evaluating outputs.

End of script - Comparison ended successfully

RAM4K CHIP:

```
CHIP RAM4K {
```

```

    IN in[16], load, address[12];
    OUT out[16];

    PARTS:
    // Putting 8 512-bit RAM together
    DMux8Way(in=load, sel=address[0..2], a=s1, b=s2, c=s3, d=s4, e=s5,
f=s6, g=s7, h=s8);
    RAM512(in=in, load=s1, address=address[3..11], out=t1);
    RAM512(in=in, load=s2, address=address[3..11], out=t2);
    RAM512(in=in, load=s3, address=address[3..11], out=t3);
    RAM512(in=in, load=s4, address=address[3..11], out=t4);
    RAM512(in=in, load=s5, address=address[3..11], out=t5);
    RAM512(in=in, load=s6, address=address[3..11], out=t6);
    RAM512(in=in, load=s7, address=address[3..11], out=t7);
    RAM512(in=in, load=s8, address=address[3..11], out=t8);
    Mux8Way16(a=t1, b=t2, c=t3, d=t4, e=t5, f=t6, g=t7, h=t8,
sel=address[0..2], out=out);
}

```

RAM4K TEST:

File View Run Help

Chip Name: **RAM4K (Clocked)** Time: **81**

Input pins		Output pins	
Name	Value	Name	Value
in[16]	21845	out[16]	21845
load	0		
address[12]	3925		

Internal pins	
Name	Value
s1	0
s2	0
s3	0
s4	0
s5	0
s6	0
s7	0
s8	0
t1[16]	0
t2[16]	0
t3[16]	0
t4[16]	0
t5[16]	0

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/03/b/RAM4K.nand2tetris.hdl

/**
 * Memory of 4K registers, each
 * stored at the memory location
 * the in value is loaded into t
 * (the loaded value will be emulated)
 */

CHIP RAM4K {
    IN in[16], load, address[12];
    OUT out[16];

    // ... (HDL code for RAM4K) ...
}
```


Script

```
set load 1,
set address %B111101010101,
set in %B0101010101010101,
tick,
output,
tock,
output;

set load 0,
set address %B000101010101,
tick,
output;
tock,
output;
set address %B001101010101,
eval,
output;
set address %B010101010101,
eval,
output;
set address %B011101010101,
eval,
output;
set address %B100101010101,
eval,
output;
set address %B101101010101,
eval,
output;
set address %B110101010101,
eval,
output;
set address %B111101010101,
eval,
output;
```

End of script - Comparison ended successfully

RAM16K CHIP:

```
CHIP RAM16K {
    IN in[16], load, address[14];
    OUT out[16];

    PARTS:
        // Putting 4 4K-RAM together
        DMux4Way(in=load, sel=address[0..1], a=s1, b=s2, c=s3, d=s4);
        RAM4K(in=in, load=s1, address=address[2..13], out=t1);
        RAM4K(in=in, load=s2, address=address[2..13], out=t2);
        RAM4K(in=in, load=s3, address=address[2..13], out=t3);
        RAM4K(in=in, load=s4, address=address[2..13], out=t4);
        Mux4Way16(a=t1, b=t2, c=t3, d=t4, sel=address[0..1], out=out);
}
```


File View Run Help

Chip Name: **RAM16K (Clocked)** Time: **81**

Input pins		Output pins	
Name	Value	Name	Value
in[16]	21845	out[16]	21845
load	0		
address[14]	15701		

Internal pins	
Name	Value
s1	0
s2	0
s3	0
s4	0
t1[16]	0
t2[16]	21845
t3[16]	0
t4[16]	0

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/03/b/RAM16K.nand2tetris.hdl

/**
 * Memory of 16K registers, each
 * stored at the memory location
 * the in value is loaded into t
 * (the loaded value will be emulated)
 */

CHIP RAM16K {
    IN in[16], load, address[14];

```

Script

```
set load 1,
set address $B11110101010101,
set in $B0101010101010101,
tick,
output,
tock,
output;

set load 0,
set address $B00010101010101,
tick,
output;
tock,
output;
set address $B00110101010101,
eval,
output;
set address $B01010101010101,
eval,
output;
set address $B01110101010101,
eval,
output;
set address $B10010101010101,
eval,
output;
set address $B10110101010101,
eval,
output;
set address $B11010101010101,
eval,
output;
set address $B11110101010101,
eval,
output;
```

End of script - Comparison ended successfully

PC CHIP:

```
CHIP PC {
    IN in[16], load, inc, reset;
    OUT out[16];

    PARTS:
        // I had to look up assistance for this chip
        // computes increment of input
        Inc16(in=loop, out=incre);
        // gate for t input with increment
        Mux16(a=loop, b=incre, sel=inc, out=s1);
        // gate for recent decision and t-1 input
        Mux16(a=s1, b=in, sel=load, out=s2);
        //gate for resetting the clock or not
        Mux16(a=s2, b[0..15]=false, sel=reset, out=s3);
        // storing result no matter what, along with assigning a variable for
        the loop and output

```

```
Register(in=s3, load=true, out=out, out=loop);
}
```

PC TEST:

Hardware Simulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\03\a\PC.hdl

File View Run Help

Chip Name: PC (Clocked)Time: 15

Input pins

Name	Value
in[16]	22222
load	0
inc	0
reset	1

Output pins

Name	Value
out[16]	0

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/03/a/PC.hdl

/**
 * A 16-bit counter with load and increment.
 * if (reset == 1) out = 0
 * else if (load == 1) out = in
 * else if (inc == 1) out = out + 1
 * else out = out
 */
CHIP PC {
```

Internal pins

Name	Value
loop[16]	0
incr[16]	1
s1[16]	0
s2[16]	0
s3[16]	0

Animate: Program flowFormat: DecimalView: Script

tock, output;

set reset 1, tick, output;

tock, output;

set in 0, set reset 0, set load 1, tick, output;

tock, output;

set load 0, set inc 1, tick, output;

tock, output;

set in 22222, set reset 1, set inc 0, tick, output;

tock, output;

output;

End of script - Comparison ended successfully