COMSM1302
Overview of Computer Architecture

Lecture 9

ModuleSim

In the previous lecture

1. Computer systems layers.

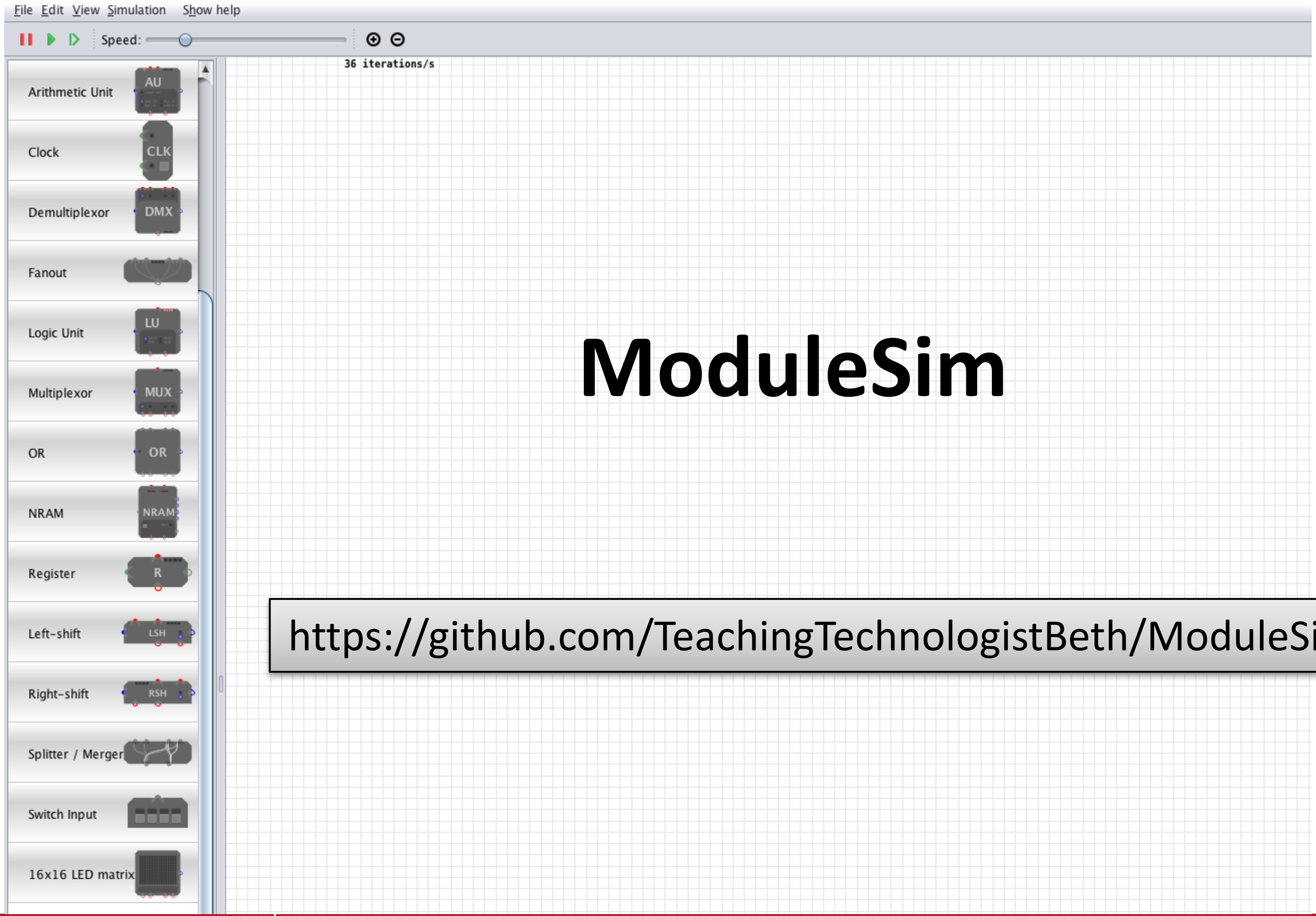
2. 4-bit CPU from 4-bit counter.

3. ModuleSim simulation software

In this lecture

1. Modulesim
    1. Bus in Modulesim
    2. Different components in ModuleSim
    3. Solve some design problems
2. At the end of this lecture:
    1. To use ModuleSim to implement and test your designs.

Simulation

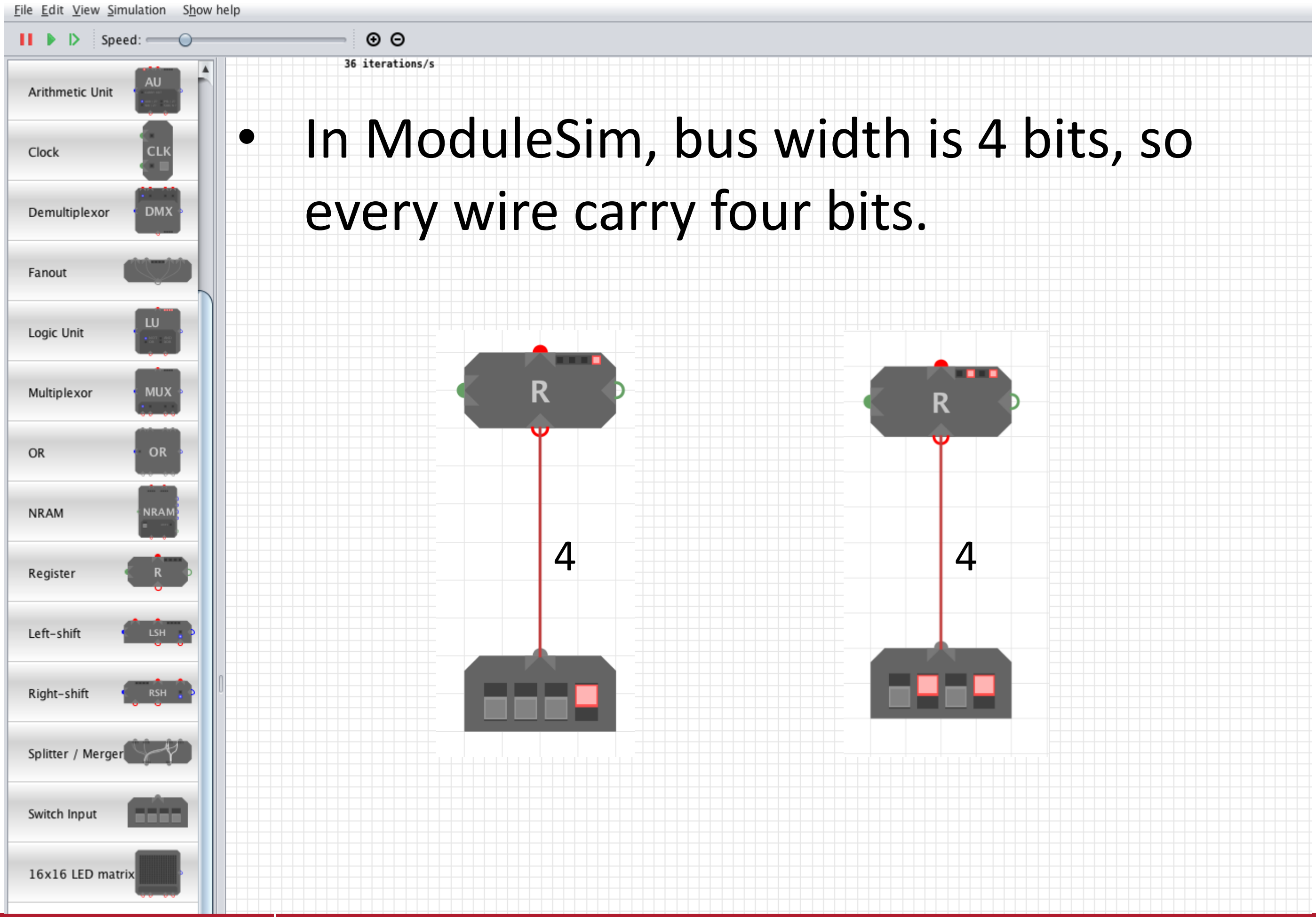


ModuleSim

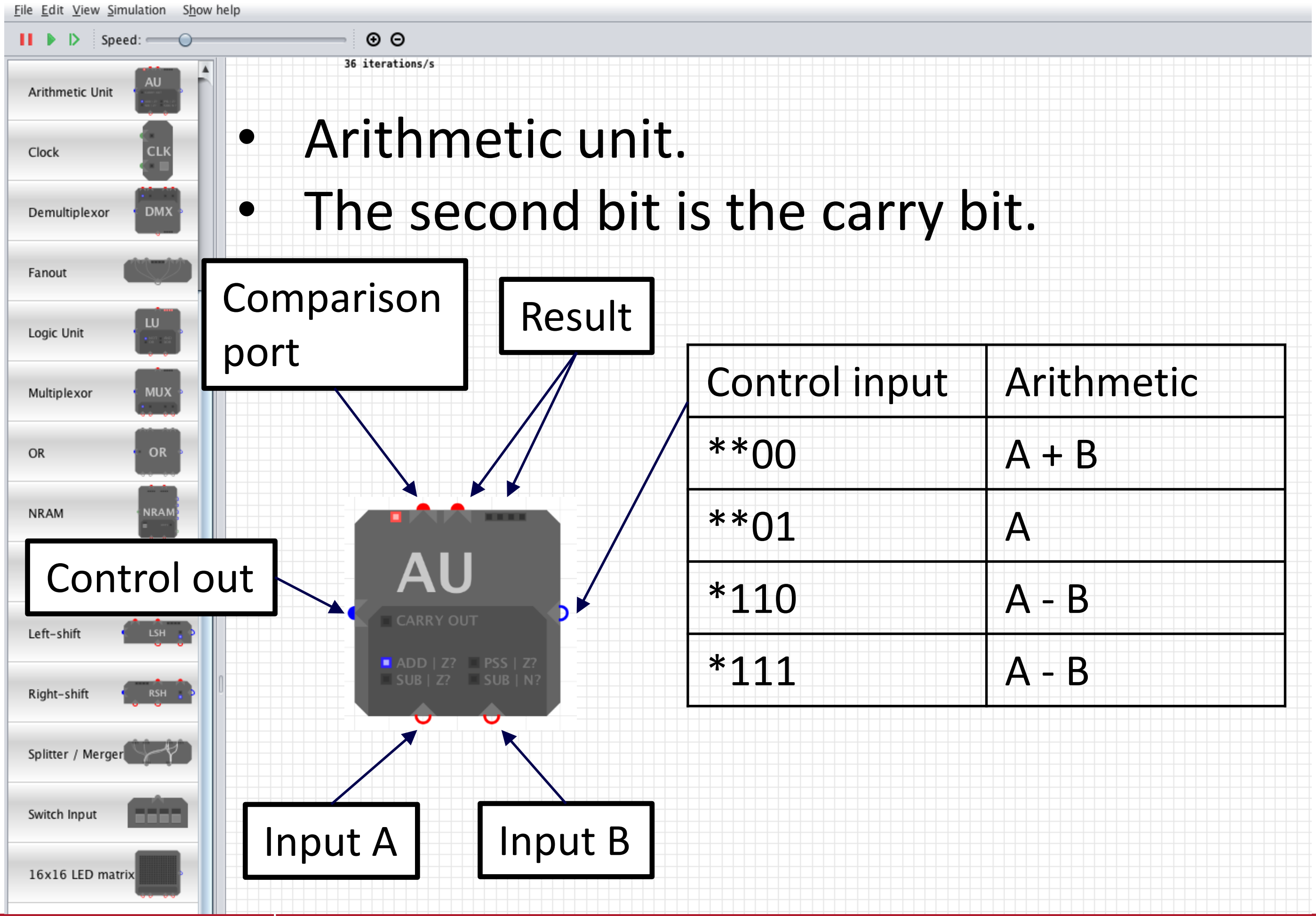
https://github.com/TeachingTechnologistBeth/ModuleSim/releases

Bus

- In ModuleSim, bus width is 4 bits, so every wire carry four bits.



AU



- Arithmetic unit.
- The second bit is the carry bit.

Comparison port

Result

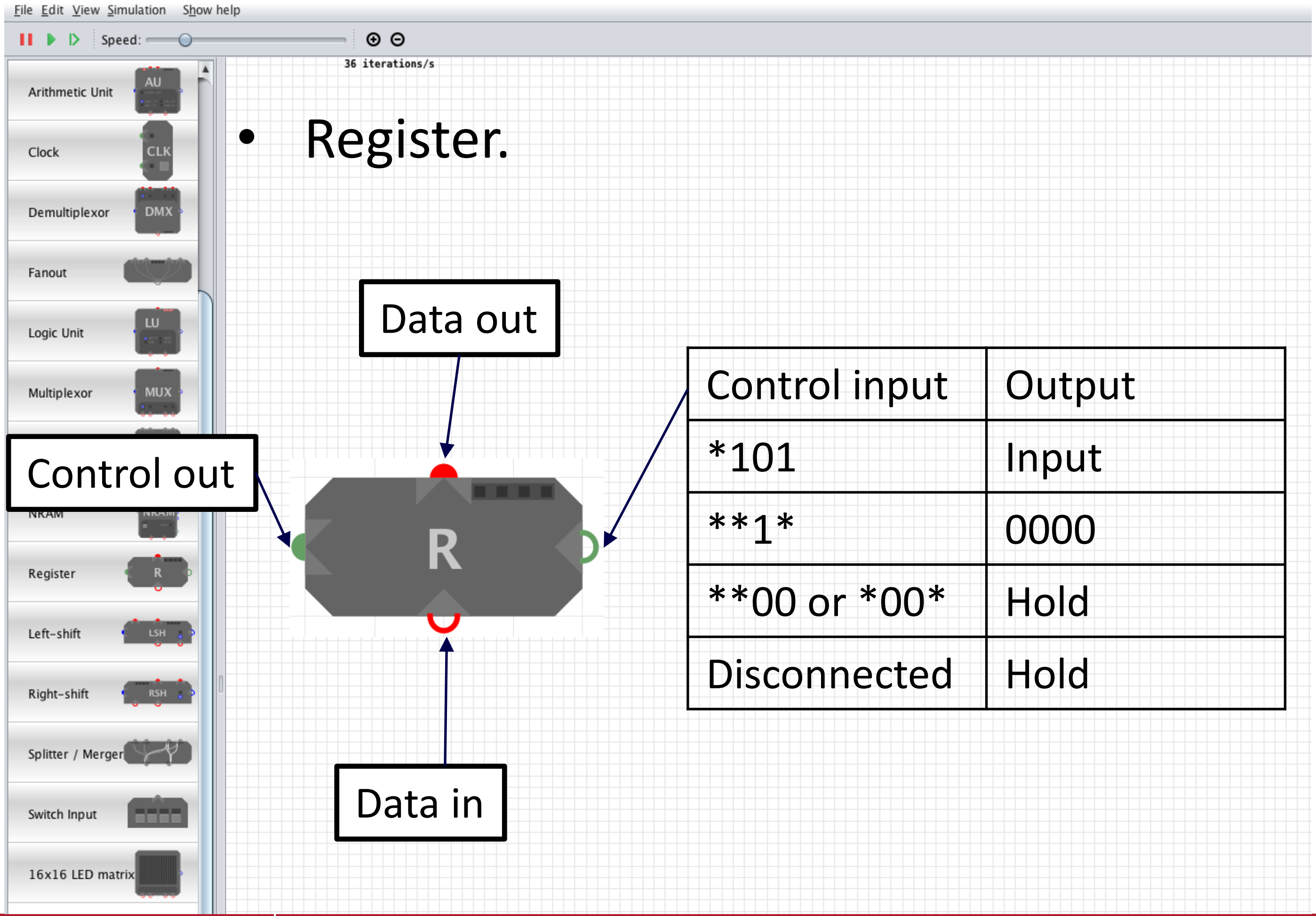
Control out

Input A

Input B

Control input	Arithmetic
**00	A + B
**01	A
*110	A - B
*111	A - B

Register



- Register.

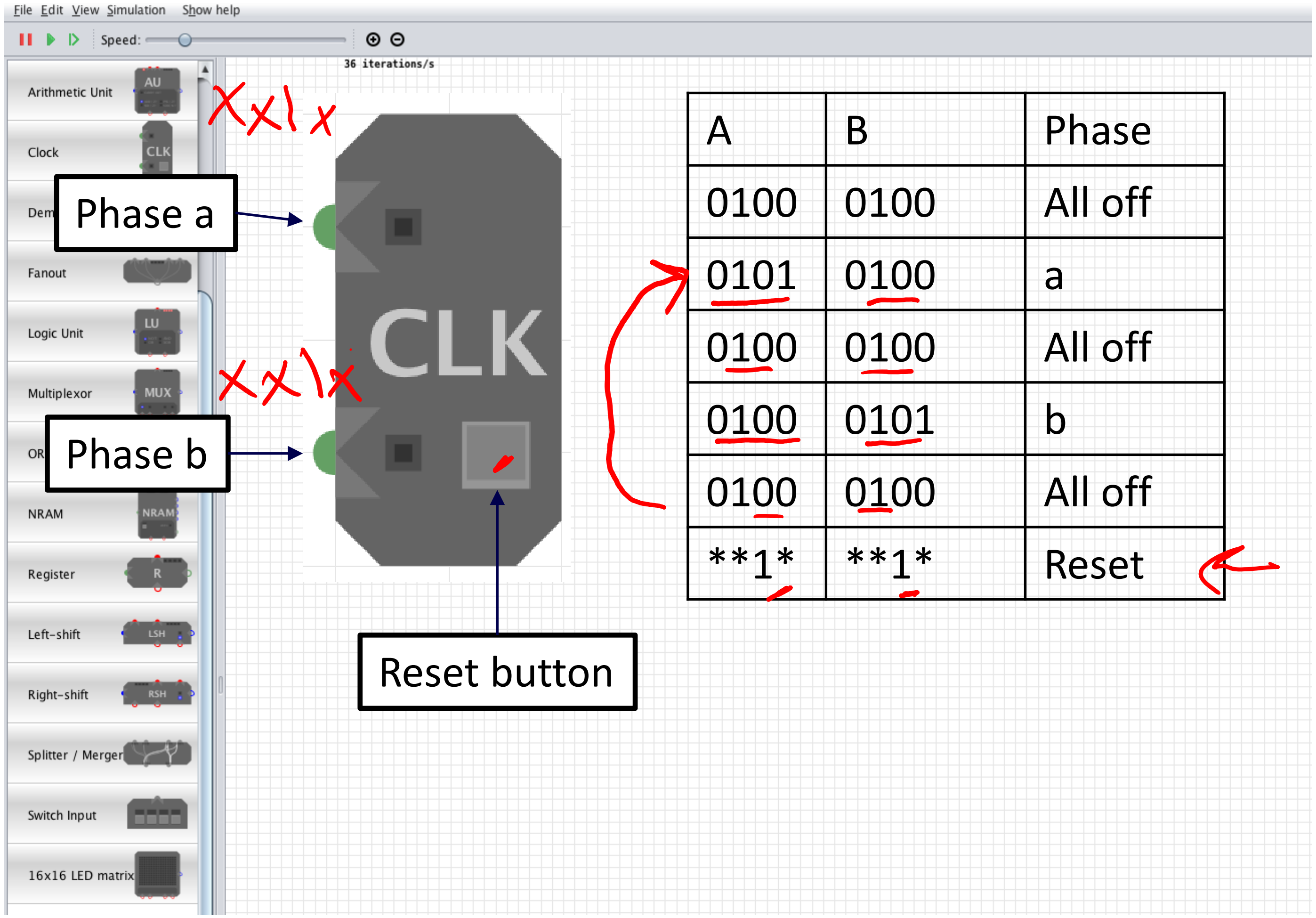
Data out

Control out

Data in

Control input	Output
*101	Input
**1*	0000
**00 or *00*	Hold
Disconnected	Hold

Clock



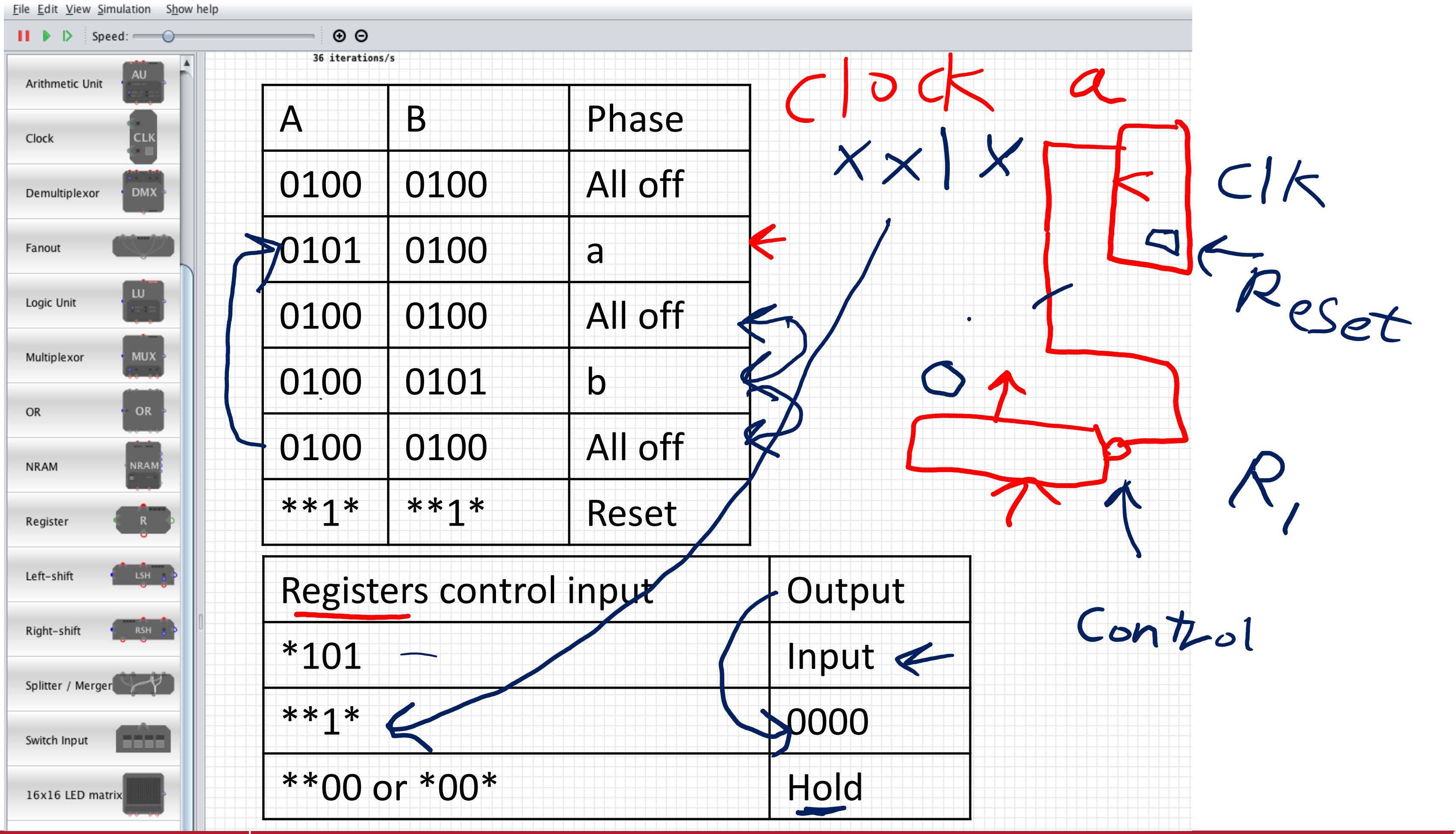
Phase a

Phase b

Reset button

A	B	Phase
0100	0100	All off
0101	0100	a
0100	0100	All off
0100	0101	b
0100	0100	All off
**1*	**1*	Reset

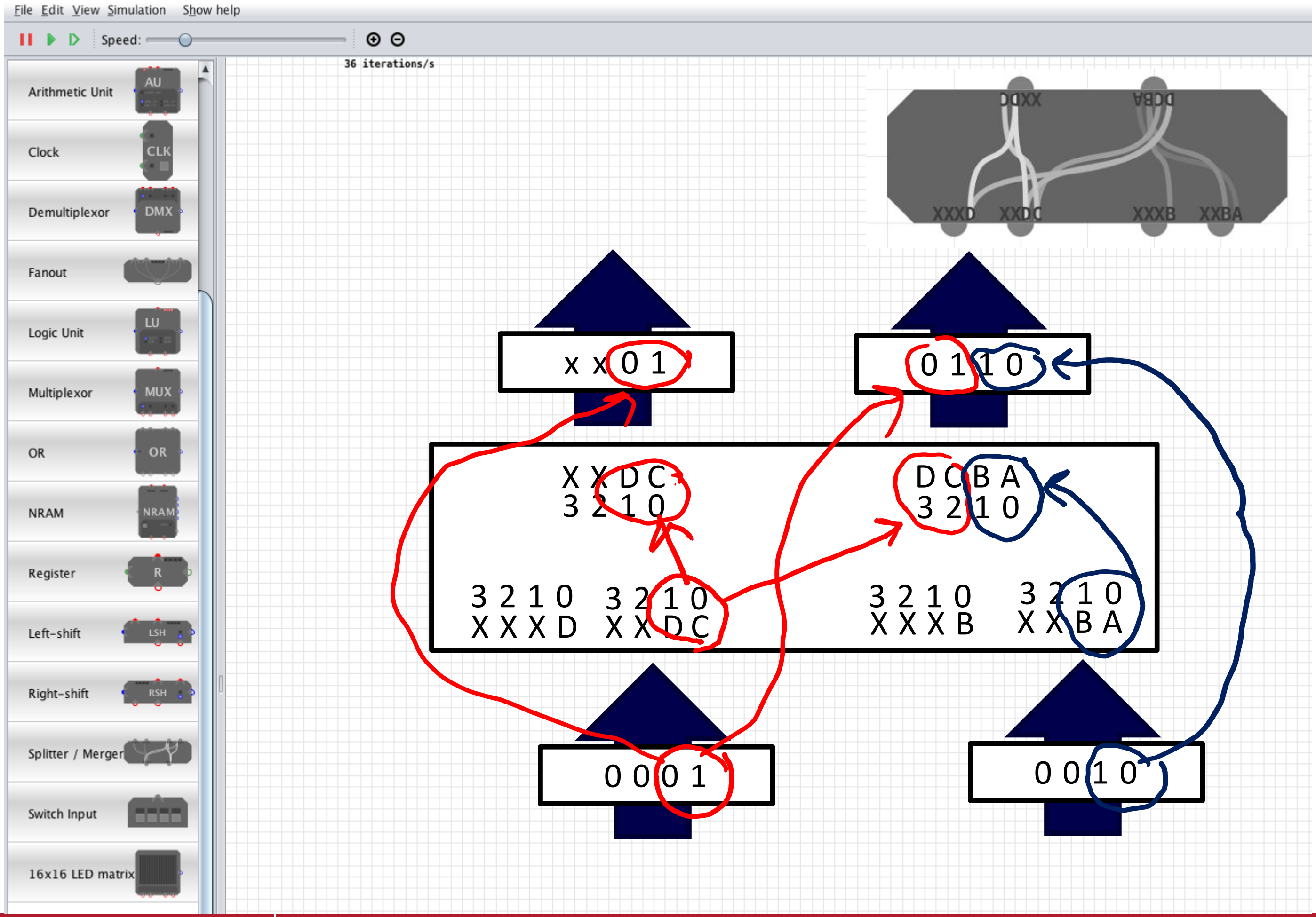
Clock use



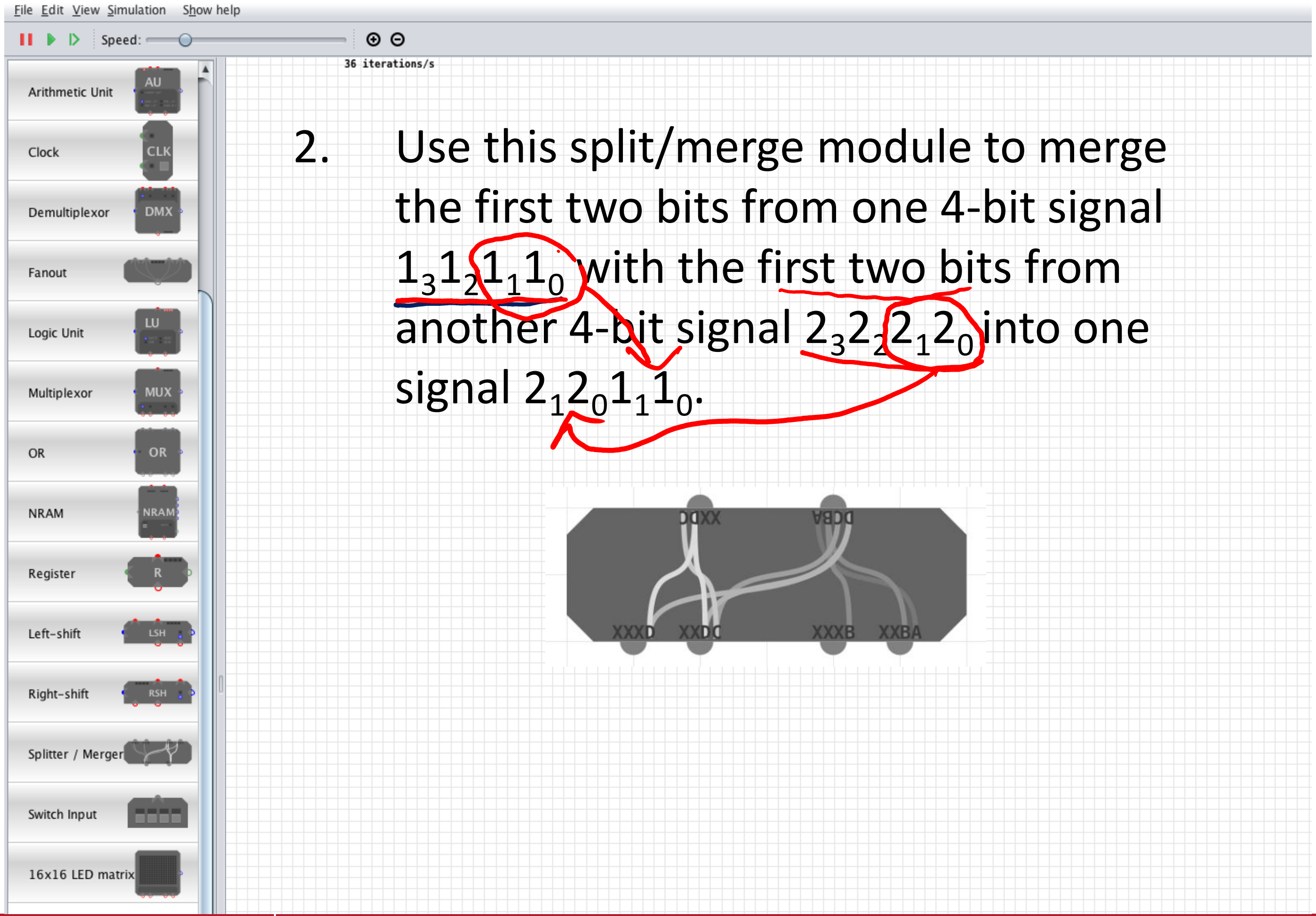
A	B	Phase
0100	0100	All off
0101	0100	a
0100	0100	All off
0100	0101	b
0100	0100	All off
**1*	**1*	Reset

Registers control input	Output
*101	Input
**1*	0000
**00 or *00*	Hold

Split/Merge

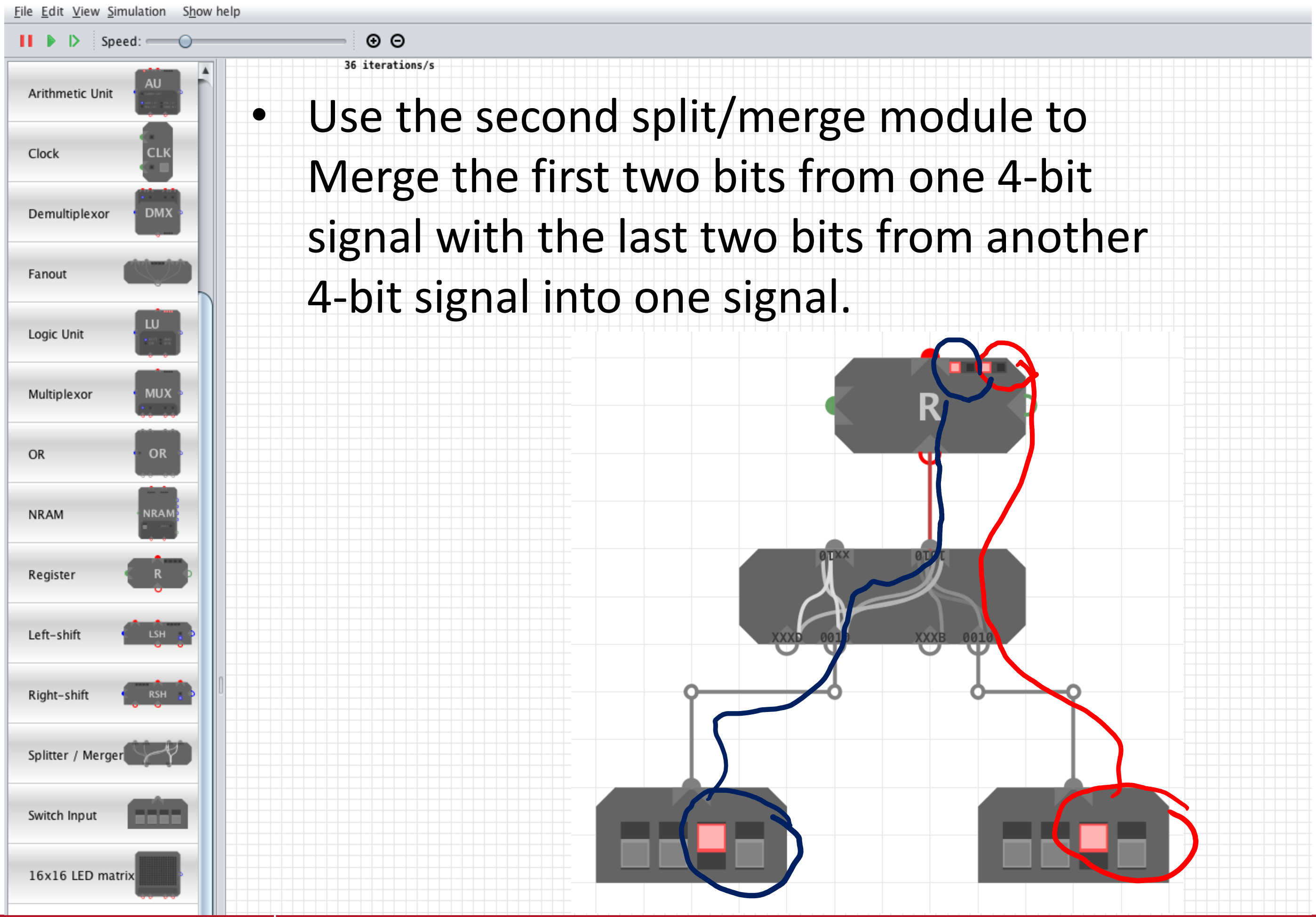

File Edit View Simulation Show help
Speed:
36 iterations/s
Arithmetic Unit
AU
Clock
CLK
Demultiplexor
DMX
Fanout
Logic Unit
LU
Multiplexor
MUX
OR
OR
NRAM
NRAM
Register
R
Left-shift
LSH
Right-shift
RSH
Splitter / Merger
Switch Input
16x16 LED matrix
DDXX
XXXD
XXDC
XXXB
XXBA
x x 0 1
0 1 1 0
X X D C
3 2 1 0
D C B A
3 2 1 0
3 2 1 0
X X X D
3 2 1 0
X X D C
3 2 1 0
X X X B
3 2 1 0
X X B A
0 0 0 1
0 0 1 0

Split/Merge – use 2-1

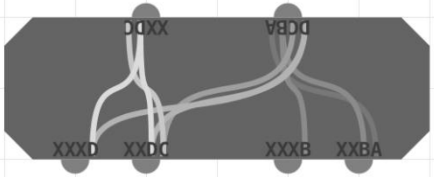


2. Use this split/merge module to merge the first two bits from one 4-bit signal $1_3 1_2 1_1 1_0$ with the first two bits from another 4-bit signal $2_3 2_2 2_1 2_0$ into one signal $2_1 2_0 1_1 1_0$.

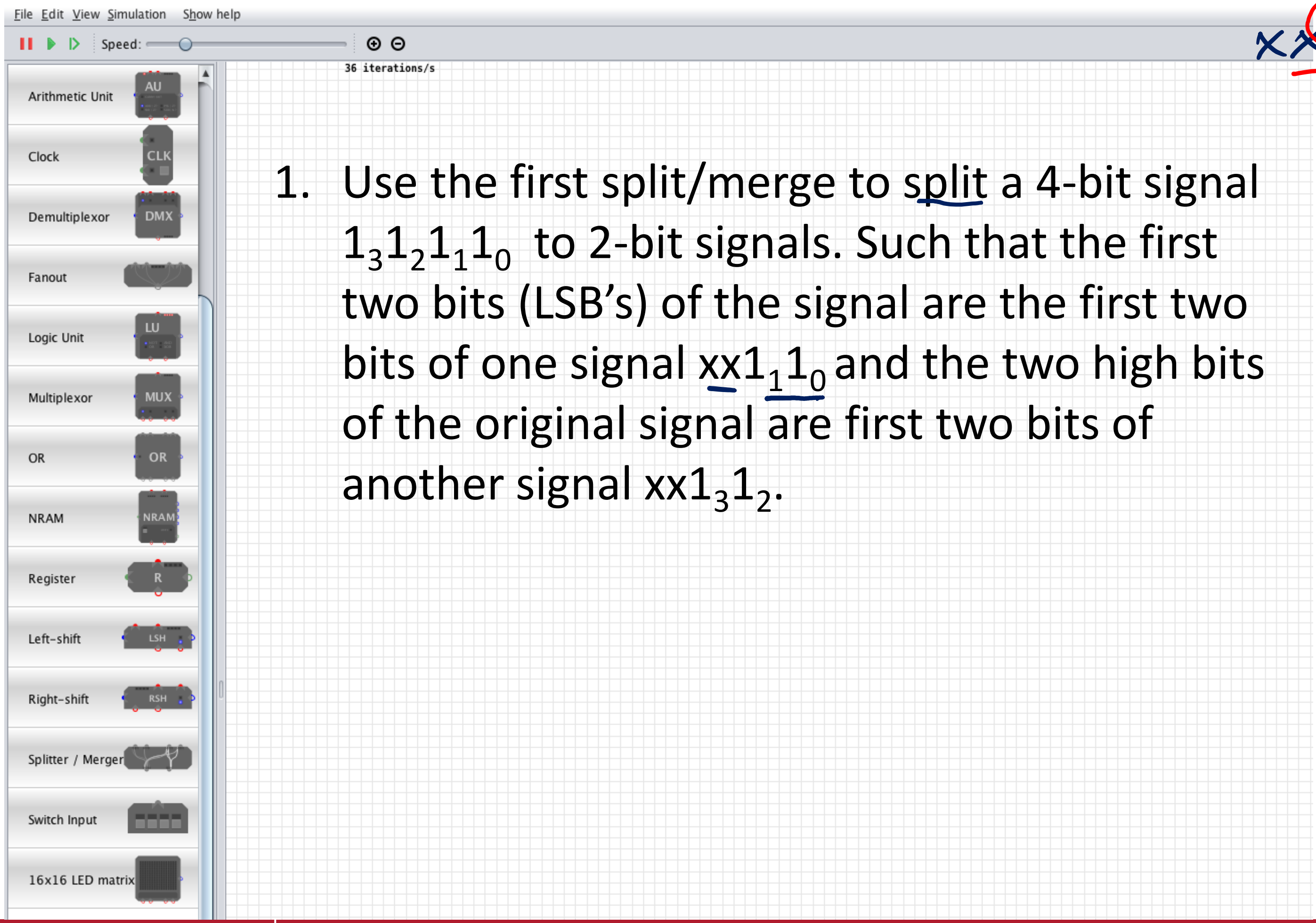
Split/Merge – use 2-2



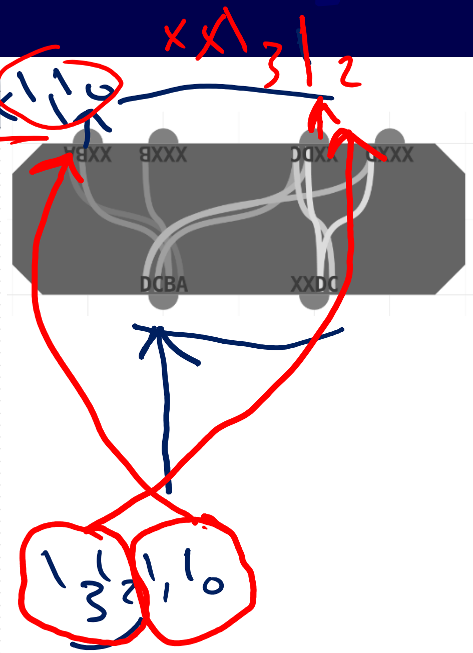
- Use the second split/merge module to Merge the first two bits from one 4-bit signal with the last two bits from another 4-bit signal into one signal.



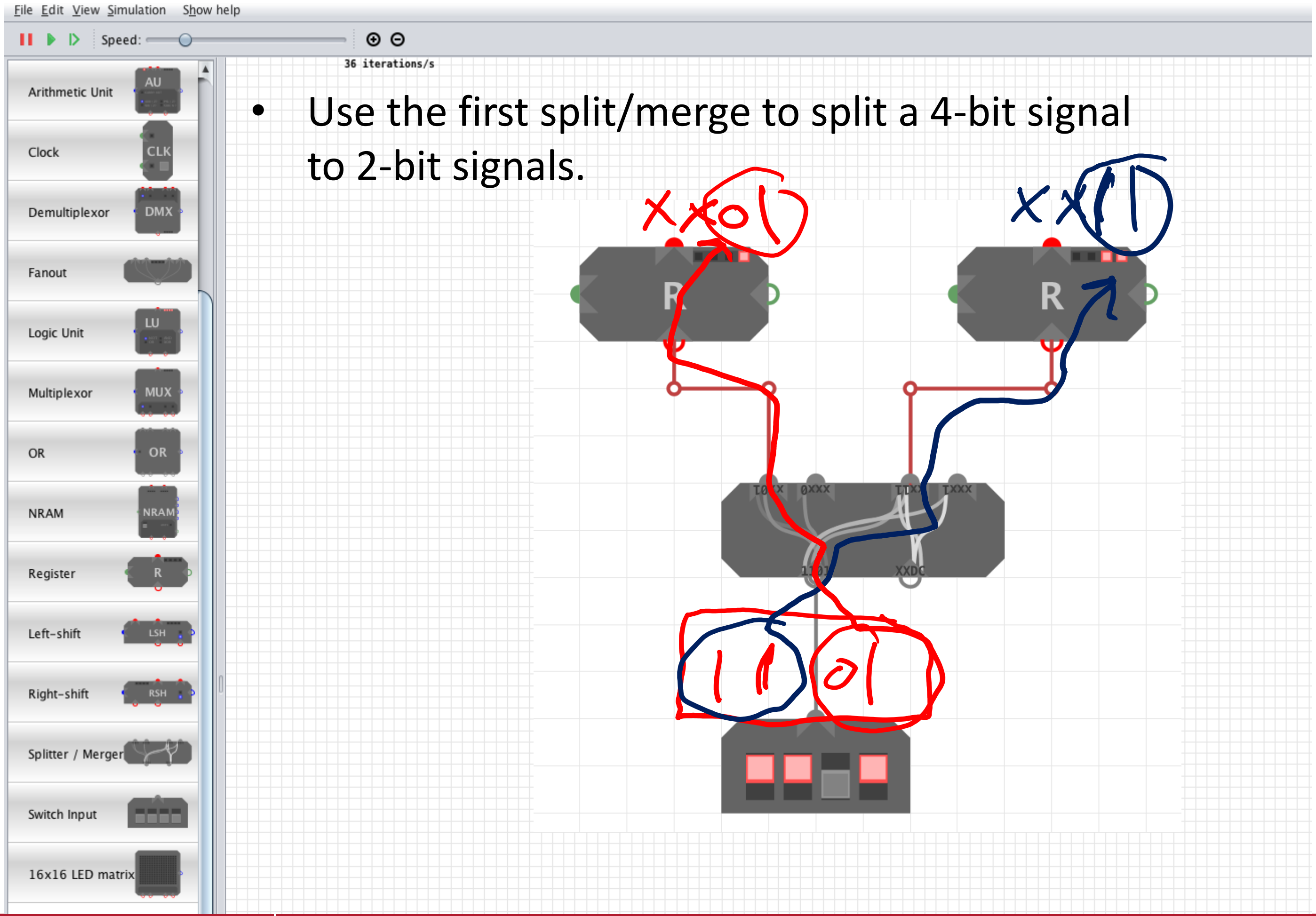
Split/Merge – use 1-1



1. Use the first split/merge to split a 4-bit signal $1_3 1_2 1_1 1_0$ to 2-bit signals. Such that the first two bits (LSB's) of the signal are the first two bits of one signal $xx1_1 1_0$ and the two high bits of the original signal are first two bits of another signal $xx1_3 1_2$.

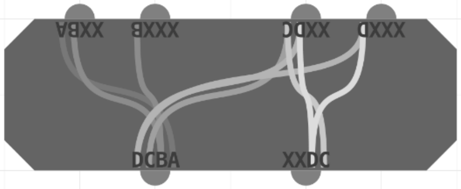


Split/Merge – use 1-2

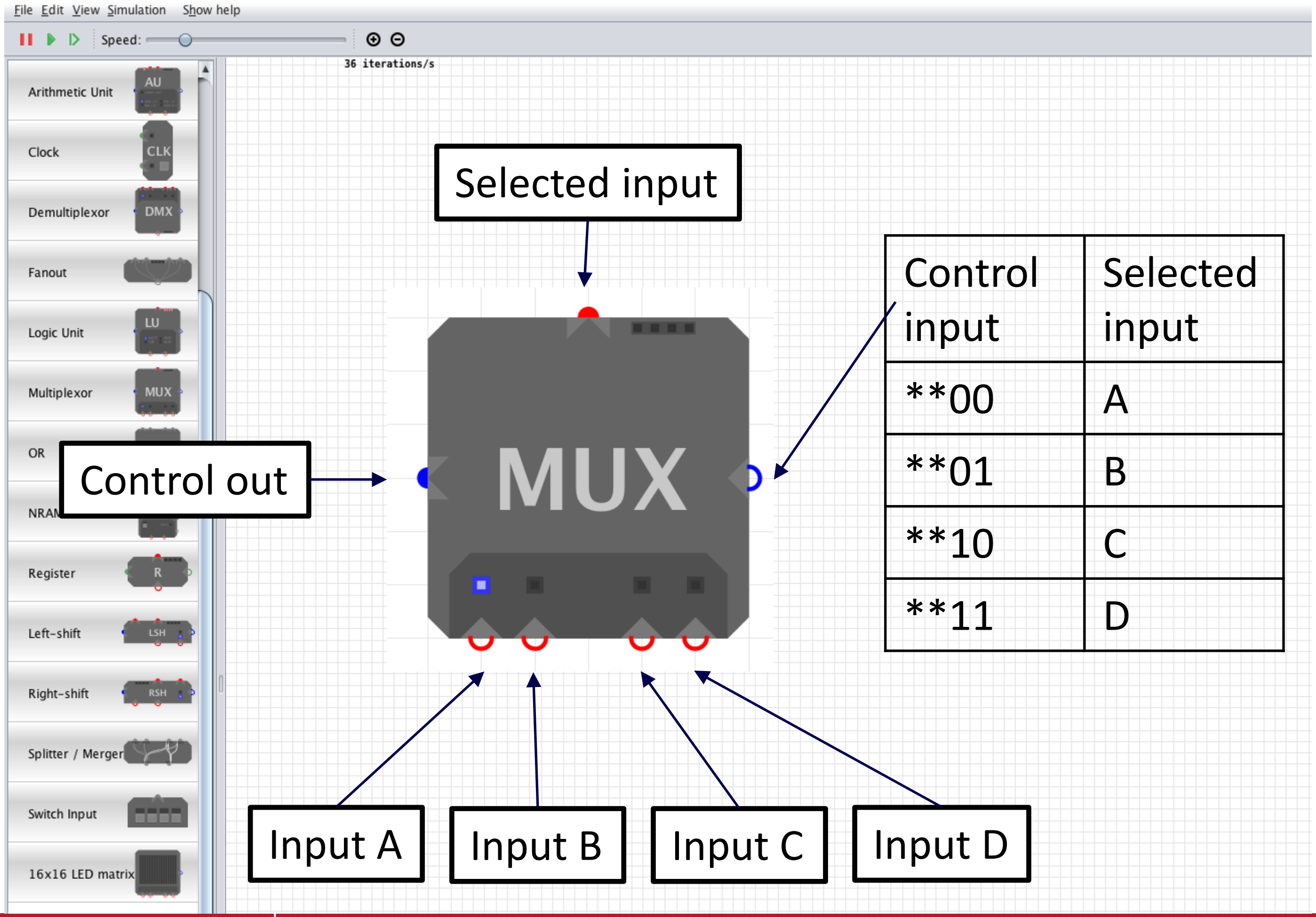


36 iterations/s

- Use the first split/merge to split a 4-bit signal to 2-bit signals.



MUX



Selected input

Control out

Input A

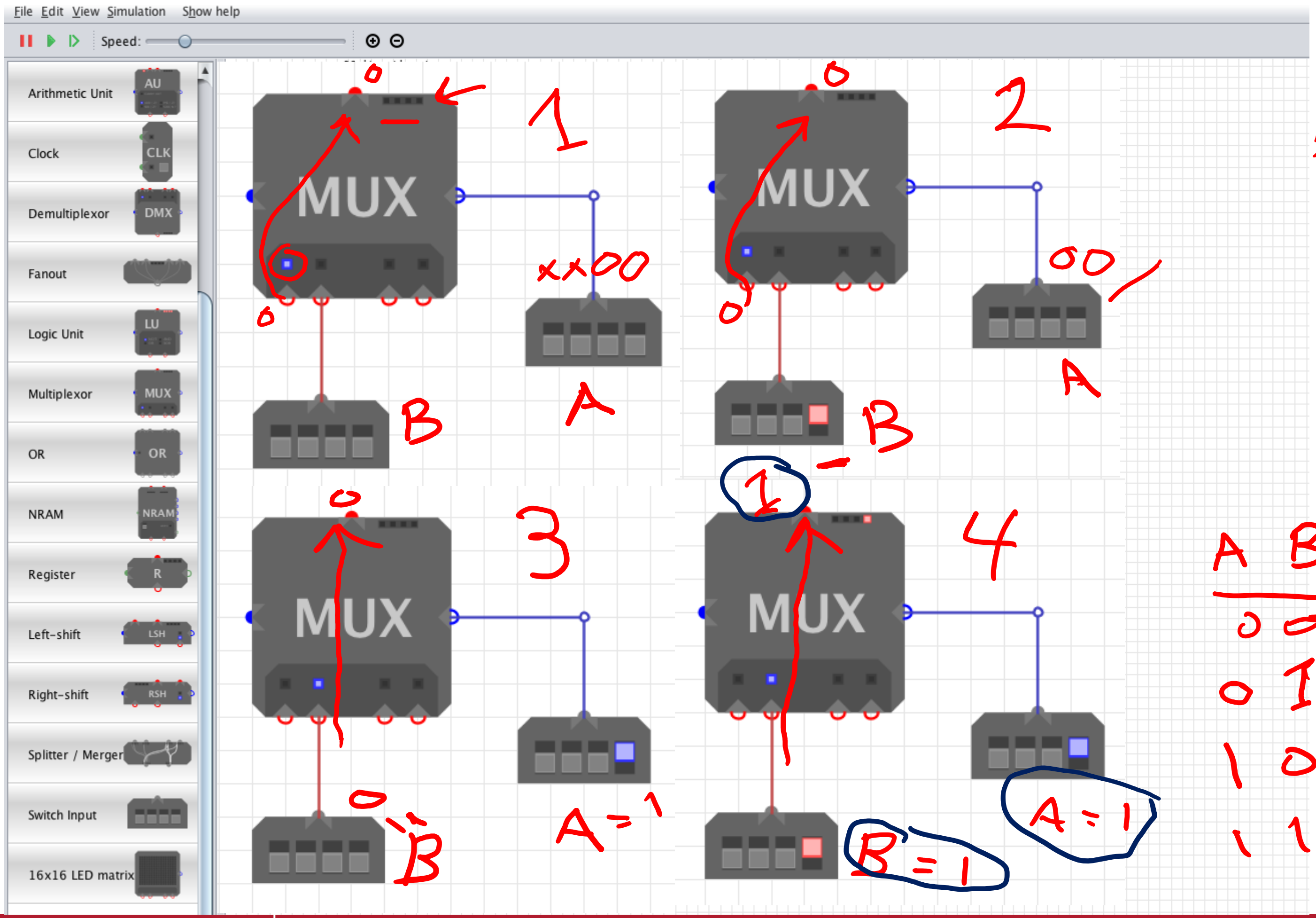
Input B

Input C

Input D

Control input	Selected input
**00	A
**01	B
**10	C
**11	D

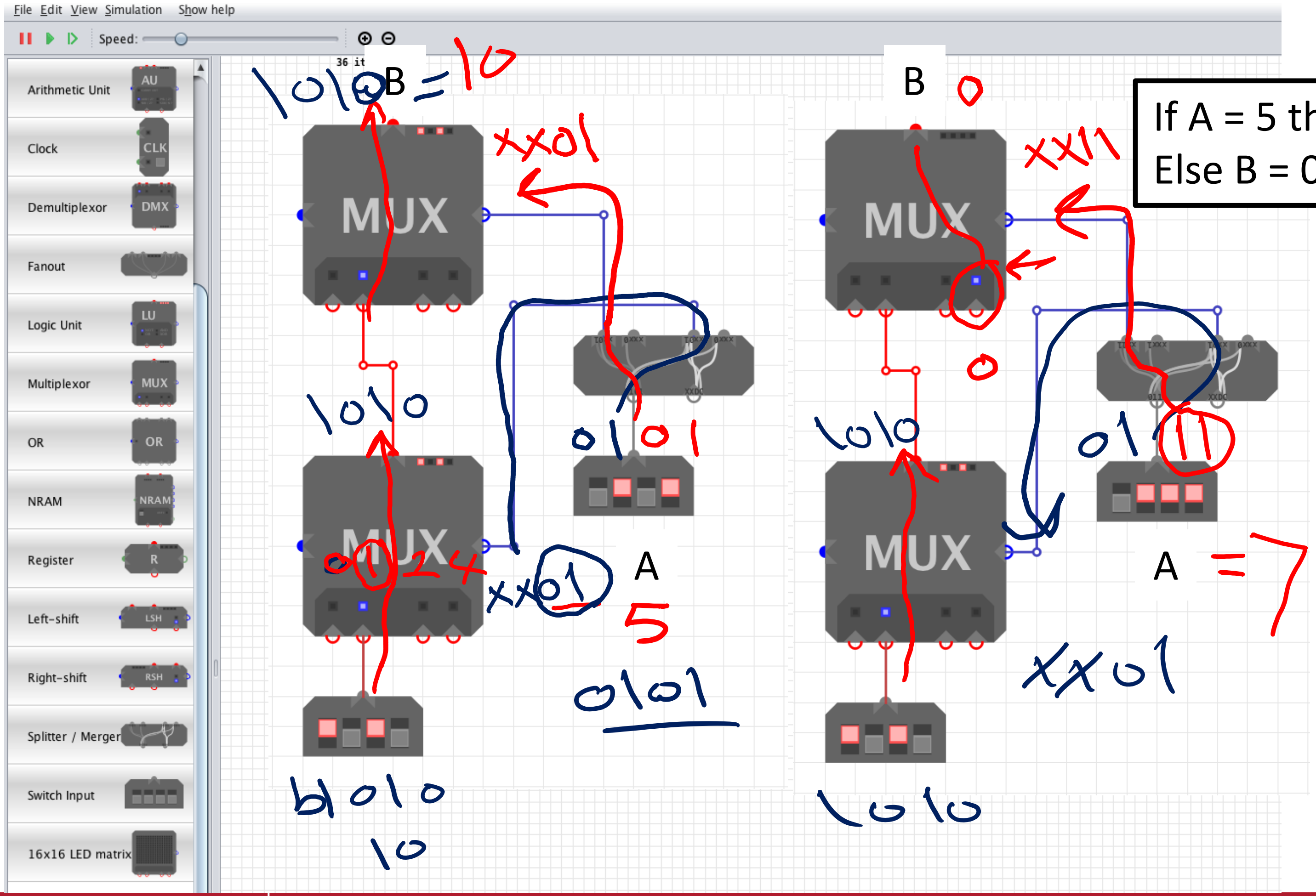
AND from MUX



xx00	A
xx01	B
xx10	C
xx11	D

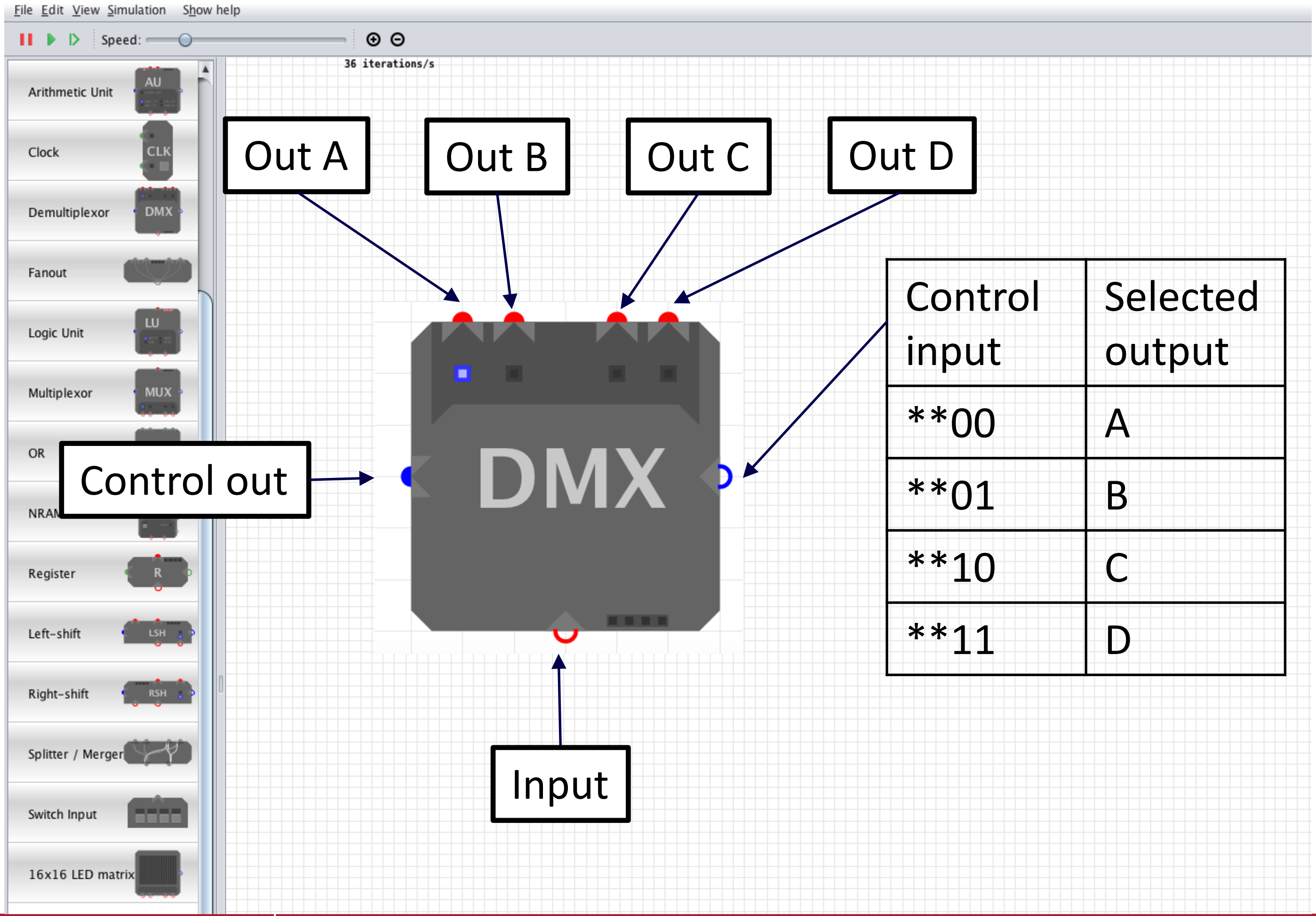
A	B	A∧B
0	0	0
0	1	0
1	0	0
1	1	1

If – Else with MUX



If A = 5 then B = 10
Else B = 0

DMX



Out A

Out B

Out C

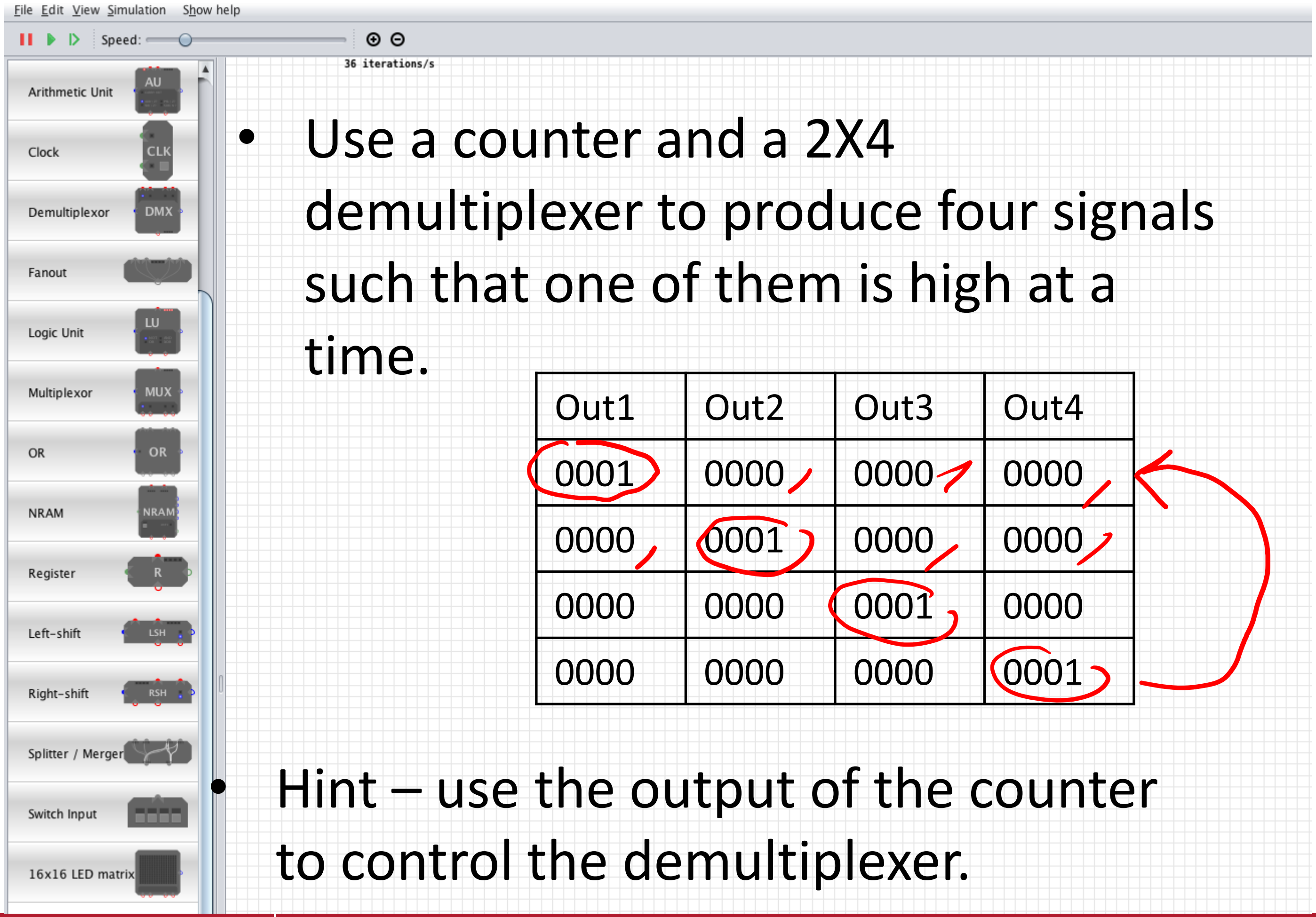
Out D

Control out

Input

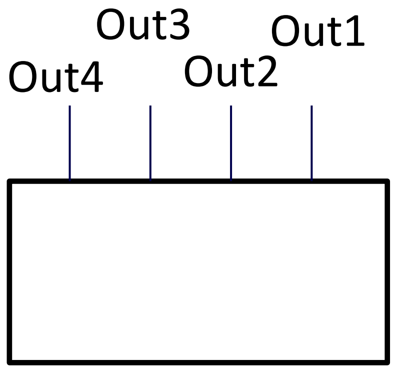
Control input	Selected output
**00	A
**01	B
**10	C
**11	D

DMX with counter



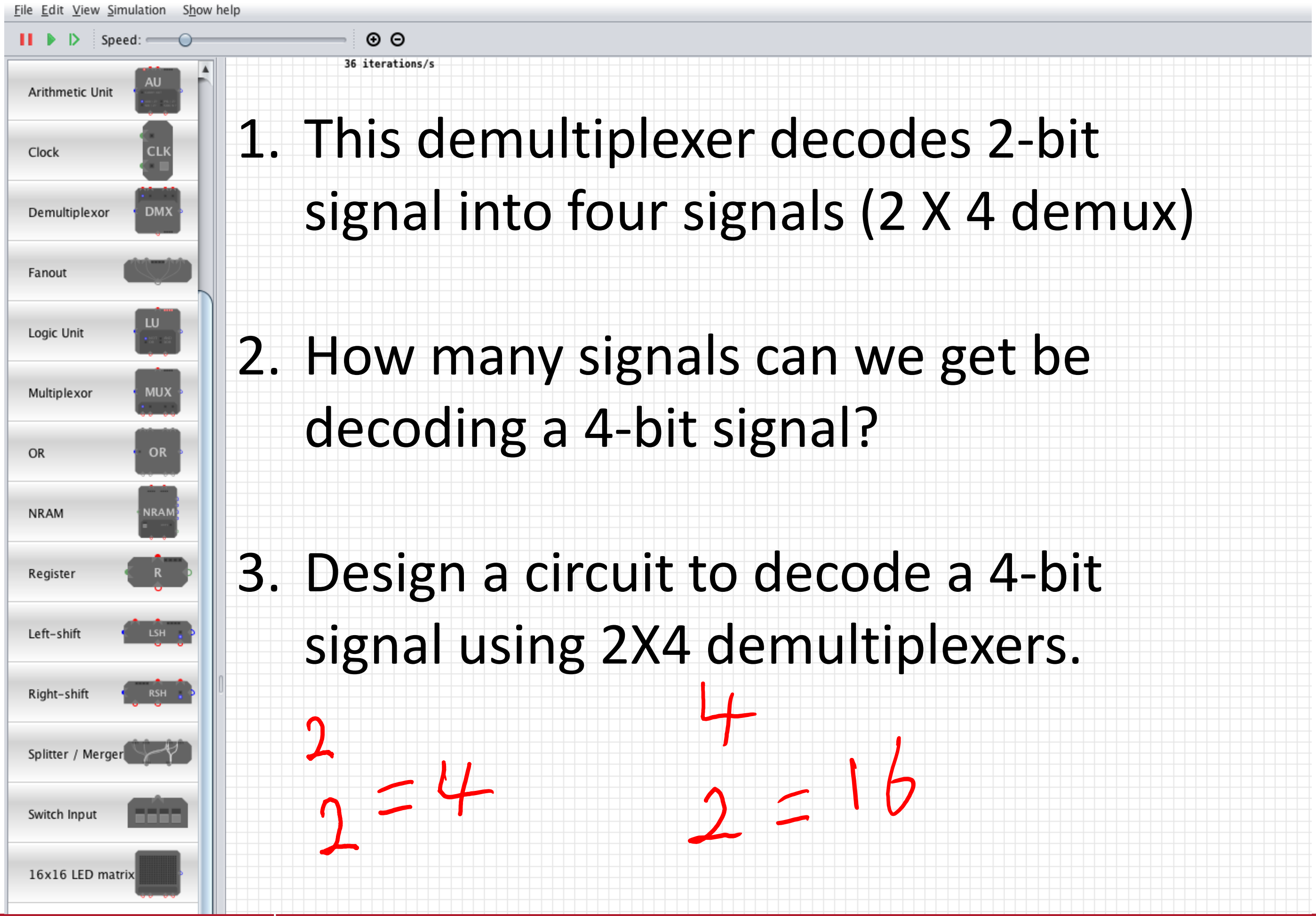
- Use a counter and a 2X4 demultiplexer to produce four signals such that one of them is high at a time.

Out1	Out2	Out3	Out4
0001	0000	0000	0000
0000	0001	0000	0000
0000	0000	0001	0000
0000	0000	0000	0001



- Hint – use the output of the counter to control the demultiplexer.

4 X 16 DMX



1. This demultiplexer decodes 2-bit signal into four signals (2 X 4 demux)

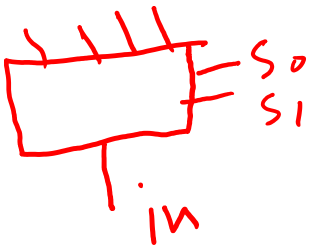
2. How many signals can we get be decoding a 4-bit signal?

3. Design a circuit to decode a 4-bit signal using 2X4 demultiplexers.

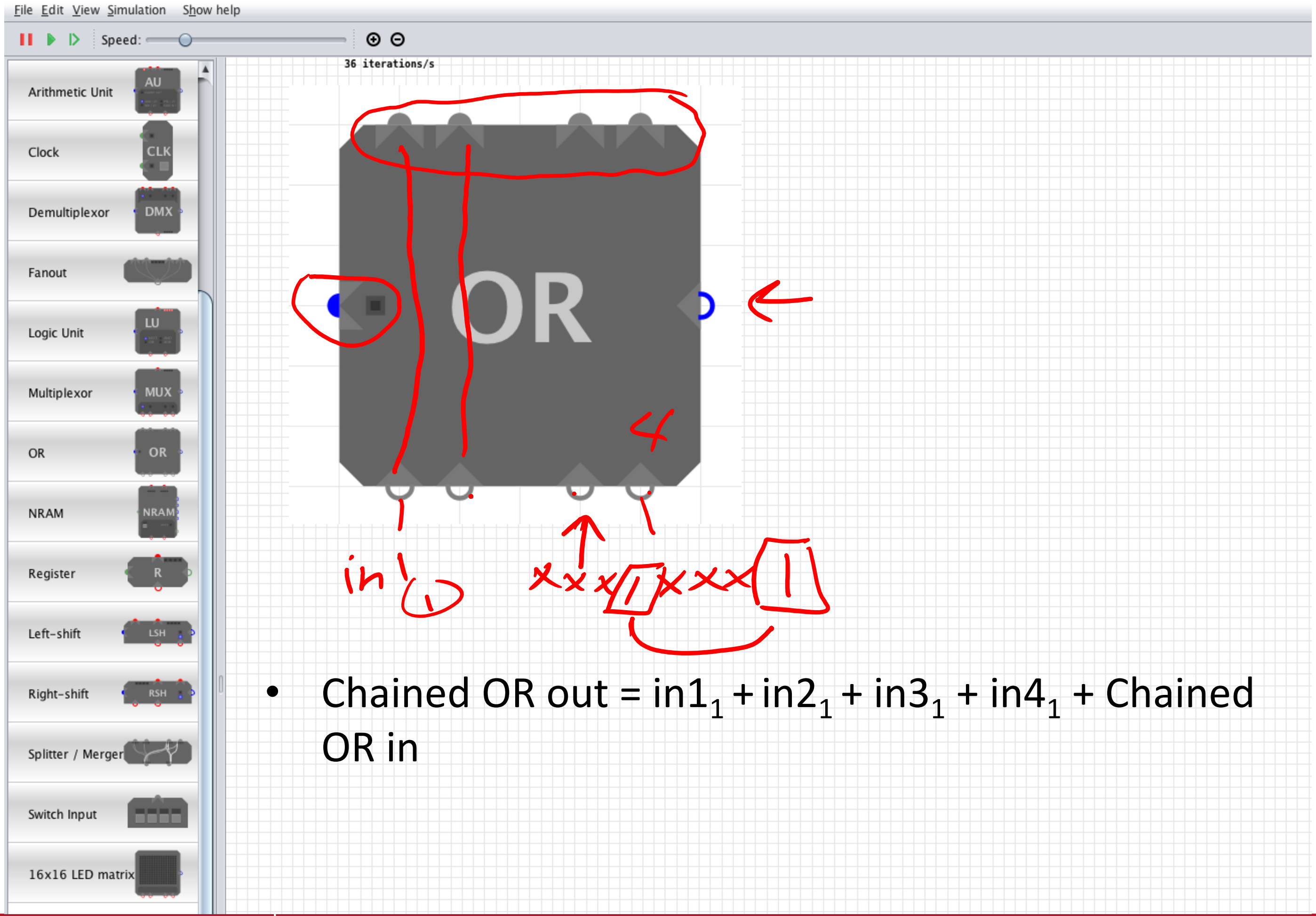
$2^2 = 4$

$2^4 = 16$

Control input	Selected output
**00	A
**01	B
**10	C
**11	D

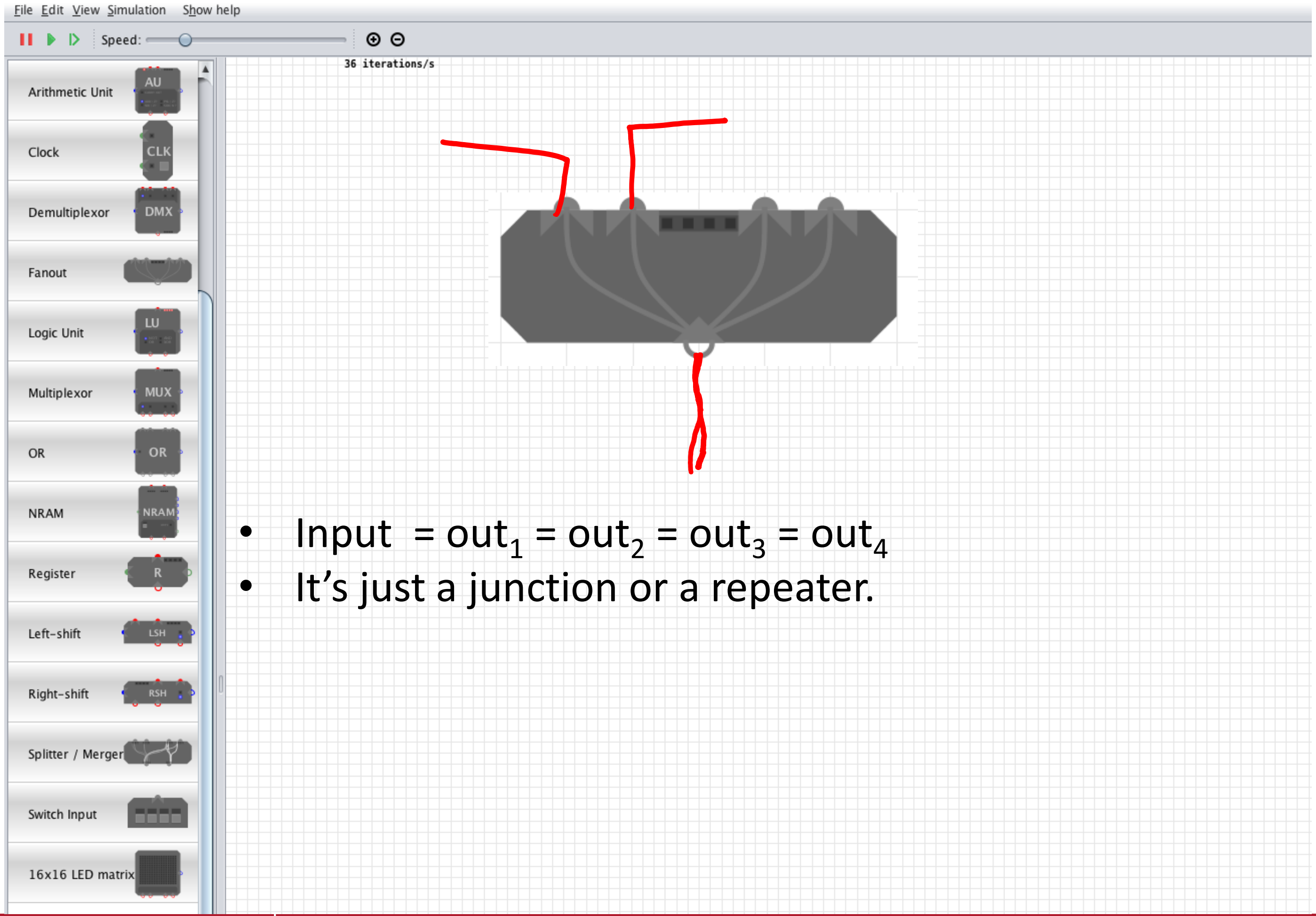


OR



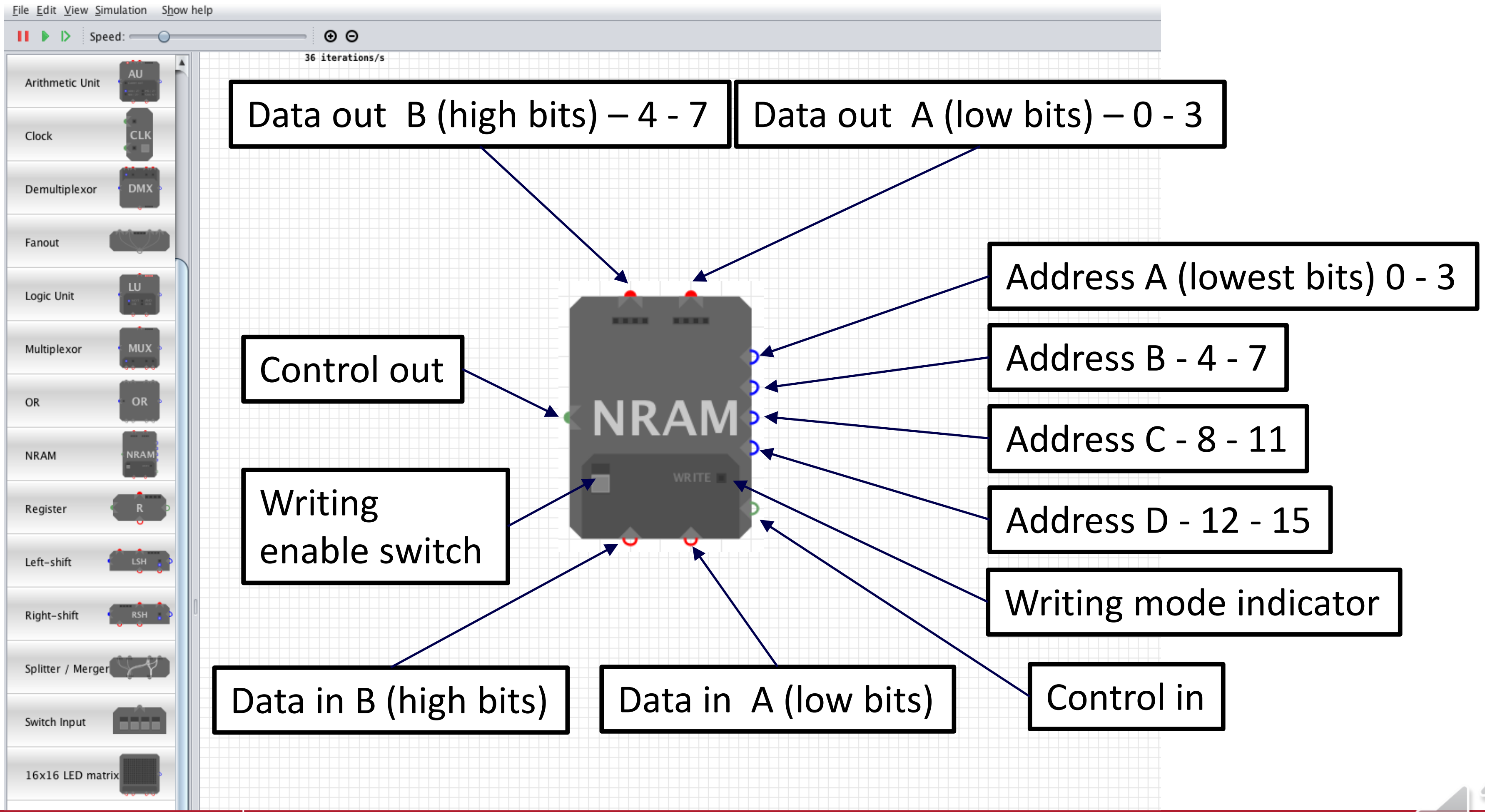
- Chained OR out = $in1_1 + in2_1 + in3_1 + in4_1$ + Chained OR in

Fanout



- Input = $out_1 = out_2 = out_3 = out_4$
- It's just a junction or a repeater.

Memory



Data out B (high bits) – 4 - 7

Data out A (low bits) – 0 - 3

Address A (lowest bits) 0 - 3

Address B - 4 - 7

Address C - 8 - 11

Address D - 12 - 15

Control out

Writing enable switch

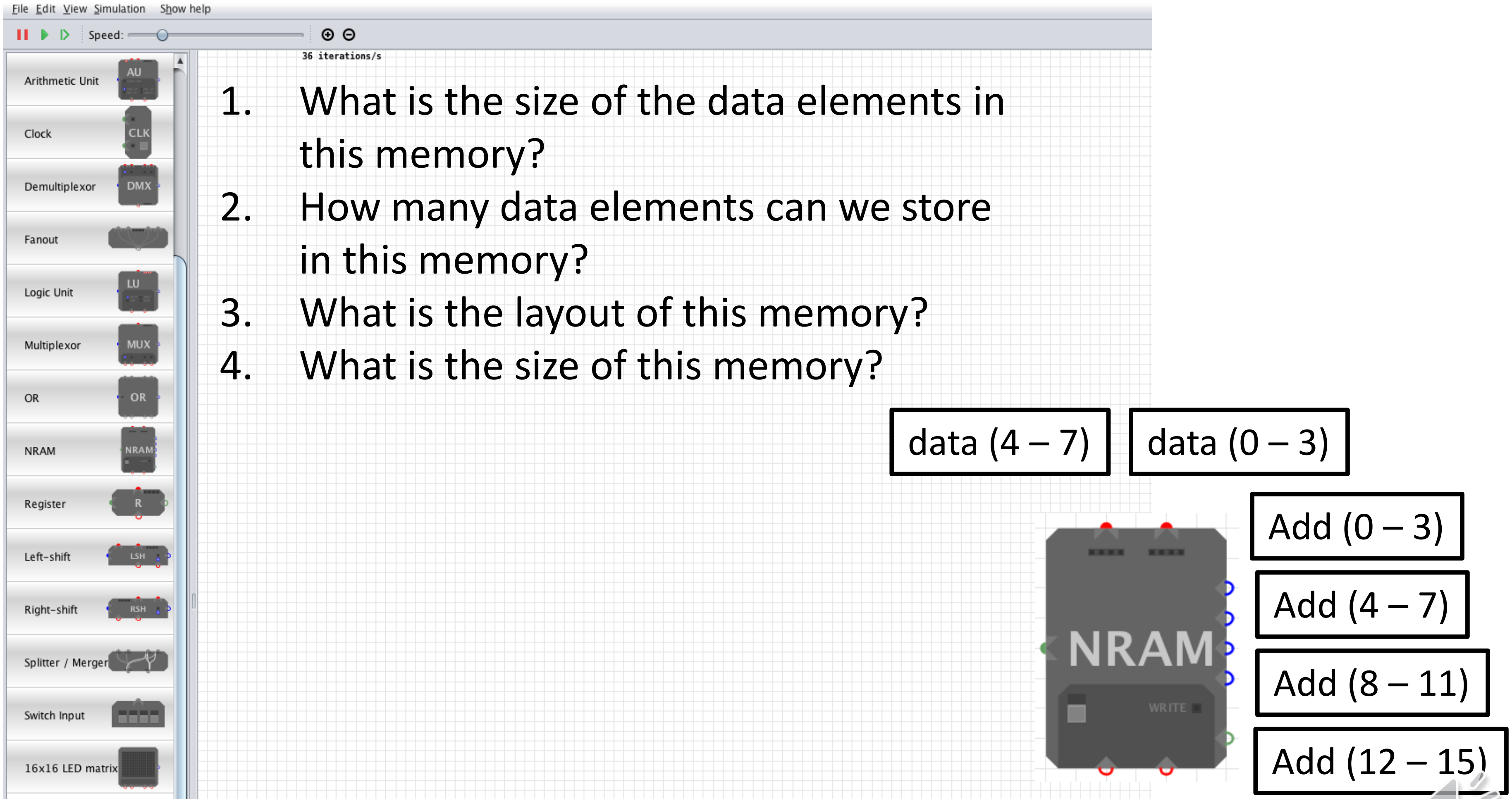
Writing mode indicator

Data in B (high bits)

Data in A (low bits)

Control in

Memory - size



1. What is the size of the data elements in this memory?
2. How many data elements can we store in this memory?
3. What is the layout of this memory?
4. What is the size of this memory?

data (4 – 7)

data (0 – 3)

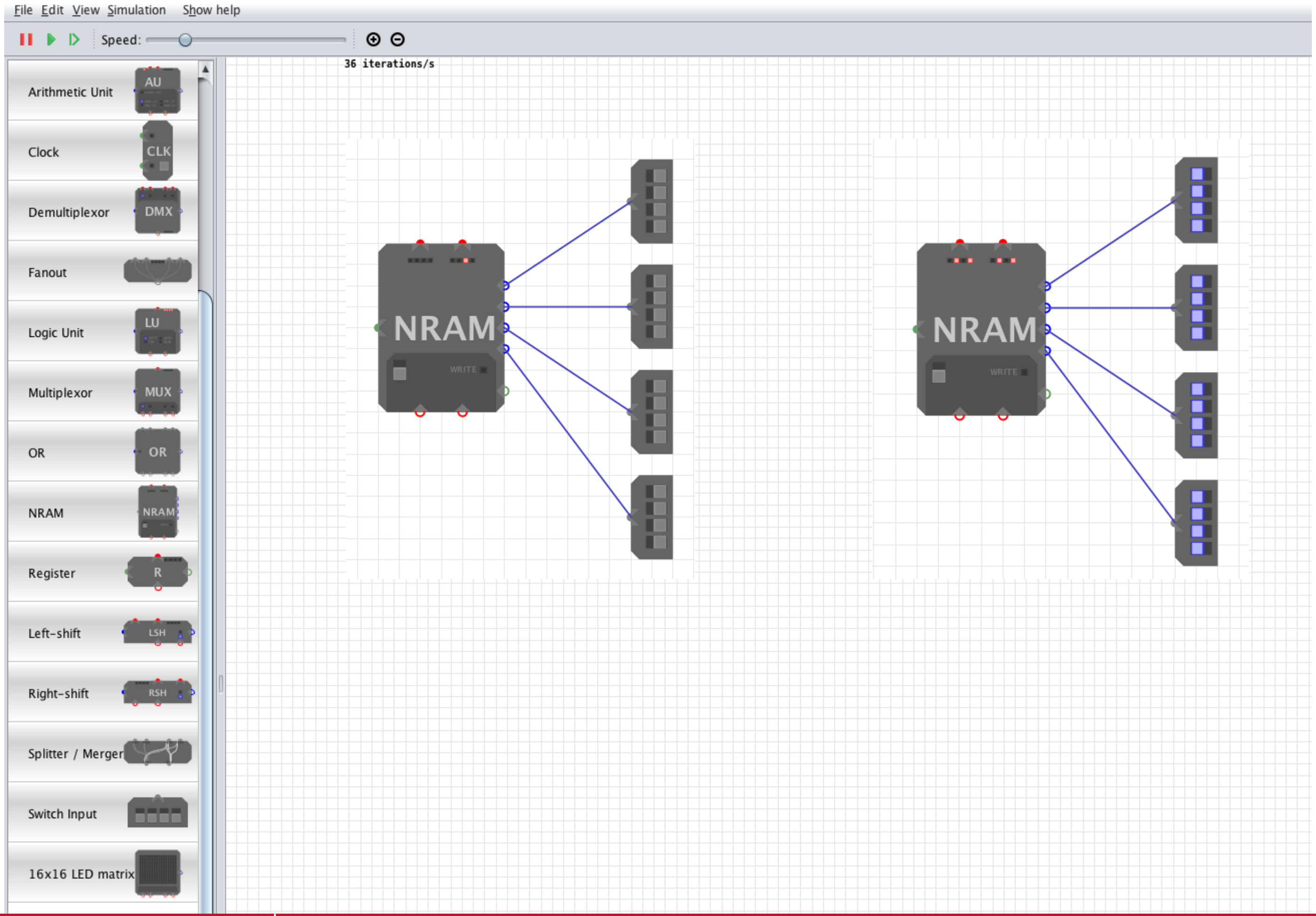
Add (0 – 3)

Add (4 – 7)

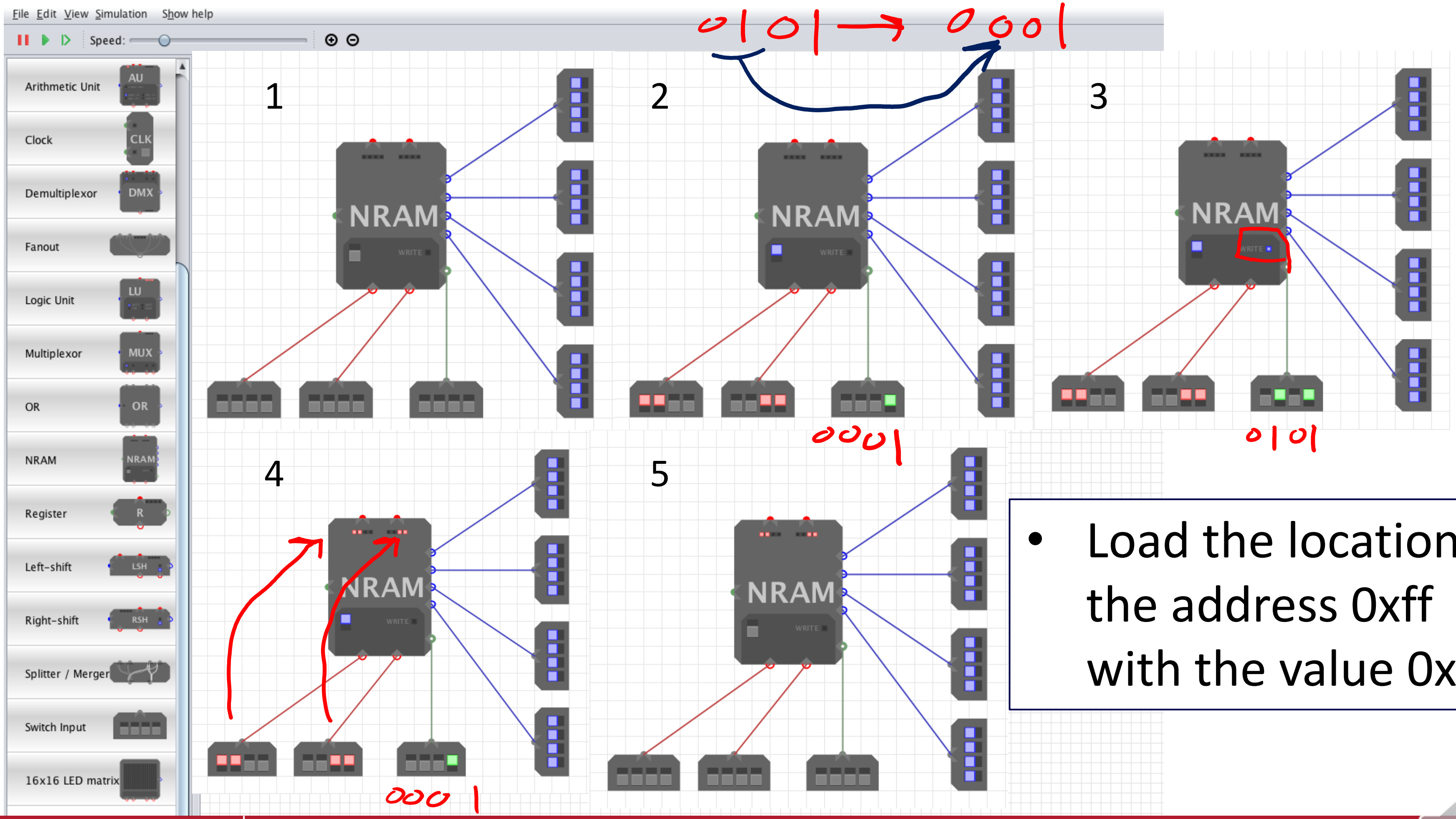
Add (8 – 11)

Add (12 – 15)

Memory - read

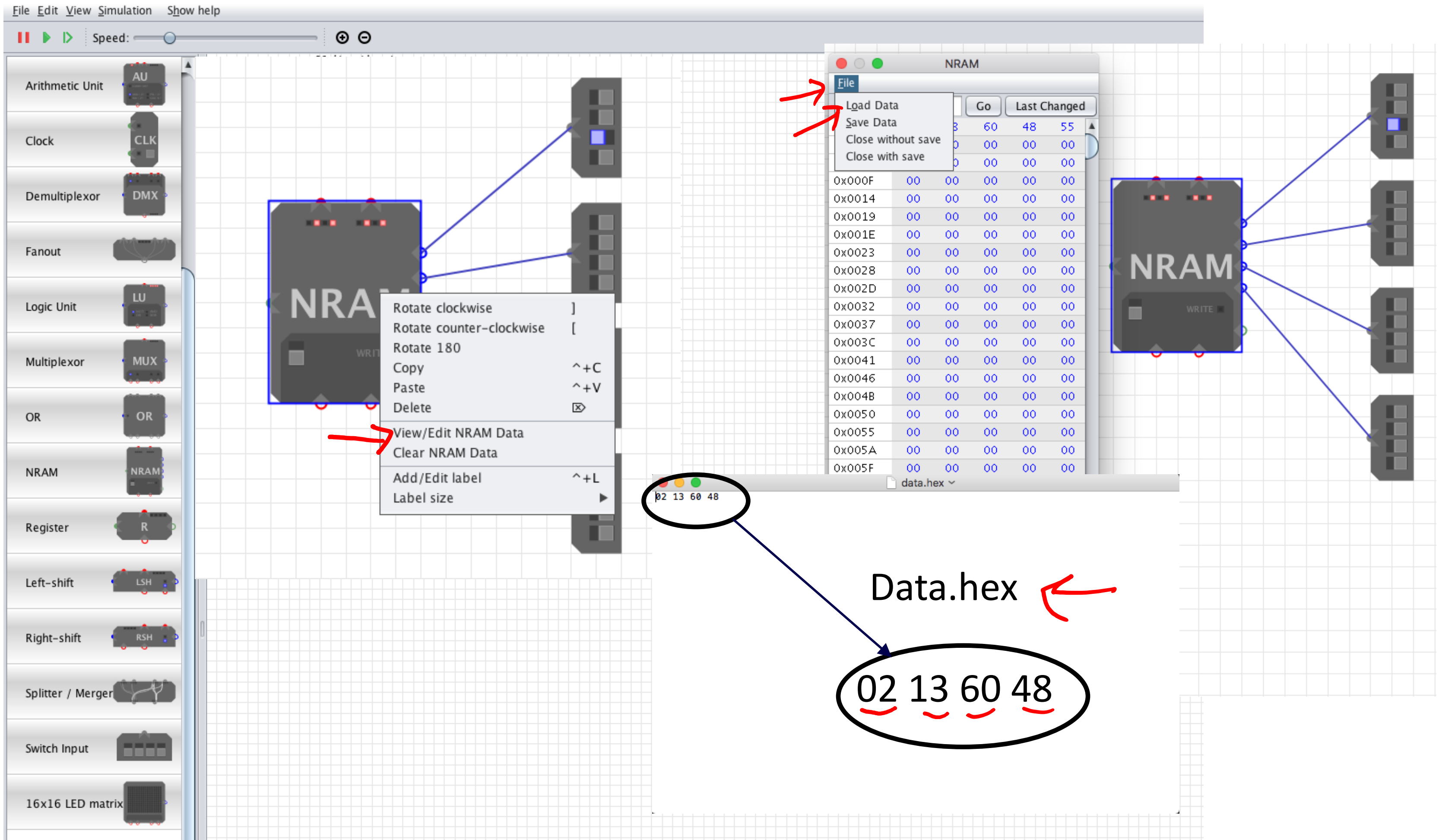

File Edit View Simulation Show help
Speed:
36 iterations/s
Arithmetic Unit
AU
Clock
CLK
Demultiplexor
DMX
Fanout
Logic Unit
LU
Multiplexor
MUX
OR
OR
NRAM
NRAM
Register
R
Left-shift
LSH
Right-shift
RSH
Splitter / Merger
Switch Input
16x16 LED matrix
NRAM
WRITE
NRAM
WRITE

Memory – write



- Load the location at the address 0xff with the value 0xc3.

Memory – load data



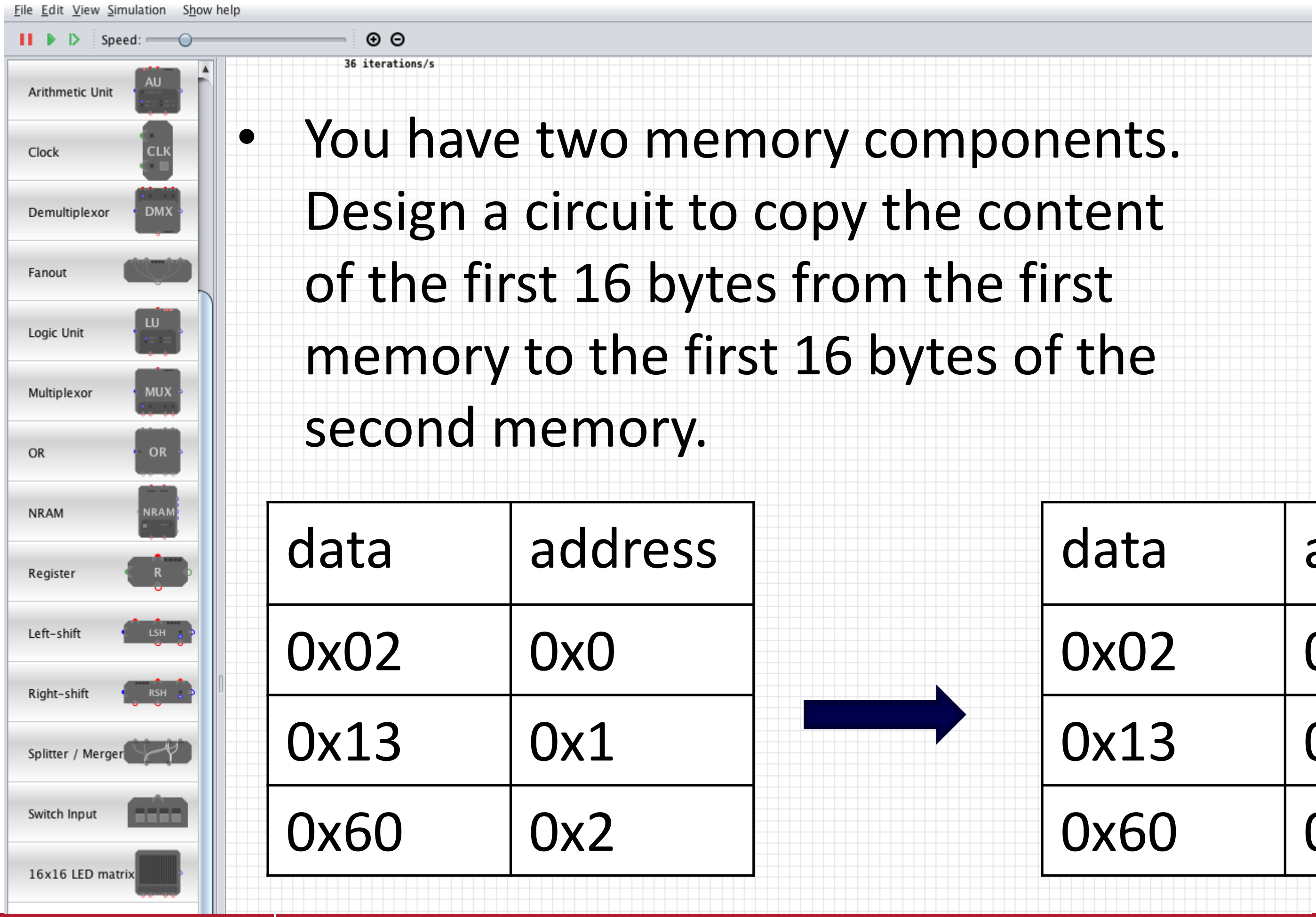
		3	60	48	55
		0	00	00	00
		0	00	00	00
0x000F	00	00	00	00	00
0x0014	00	00	00	00	00
0x0019	00	00	00	00	00
0x001E	00	00	00	00	00
0x0023	00	00	00	00	00
0x0028	00	00	00	00	00
0x002D	00	00	00	00	00
0x0032	00	00	00	00	00
0x0037	00	00	00	00	00
0x003C	00	00	00	00	00
0x0041	00	00	00	00	00
0x0046	00	00	00	00	00
0x004B	00	00	00	00	00
0x0050	00	00	00	00	00
0x0055	00	00	00	00	00
0x005A	00	00	00	00	00
0x005F	00	00	00	00	00

```
02 13 60 48
```

Data.hex

02 13 60 48

Memory - design problem -1



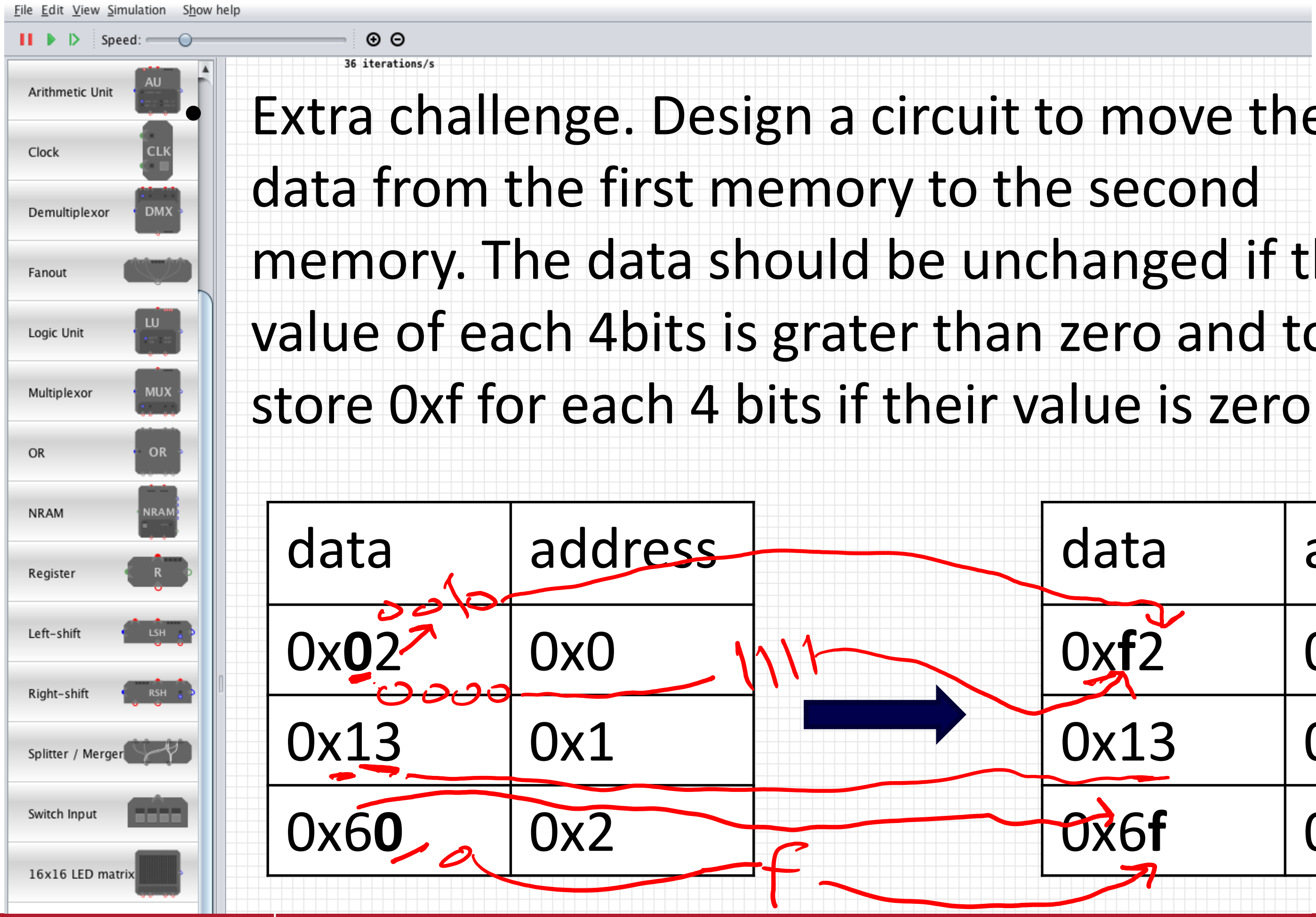
- You have two memory components. Design a circuit to copy the content of the first 16 bytes from the first memory to the first 16 bytes of the second memory.

data	address
0x02	0x0
0x13	0x1
0x60	0x2

→

data	address
0x02	0x0
0x13	0x1
0x60	0x2

Memory - design problem -2



- Extra challenge. Design a circuit to move the data from the first memory to the second memory. The data should be unchanged if the value of each 4bits is grater than zero and to store 0xf for each 4 bits if their value is zero.

data	address
0x**0**2	0x0
0x13	0x1
0x6**0**	0x2

→

data	address
0x**f**2	0x0
0x13	0x1
0x6**f**	0x2

0010

0000

1111

0

f

Summary

1. ModuleSim and some of its components.
   1. How clock control registers
   2. How to use Split/Merge
   3. Some applications of multiplexers and demultiplexers.
2. Memory component and two design problem.