

# 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

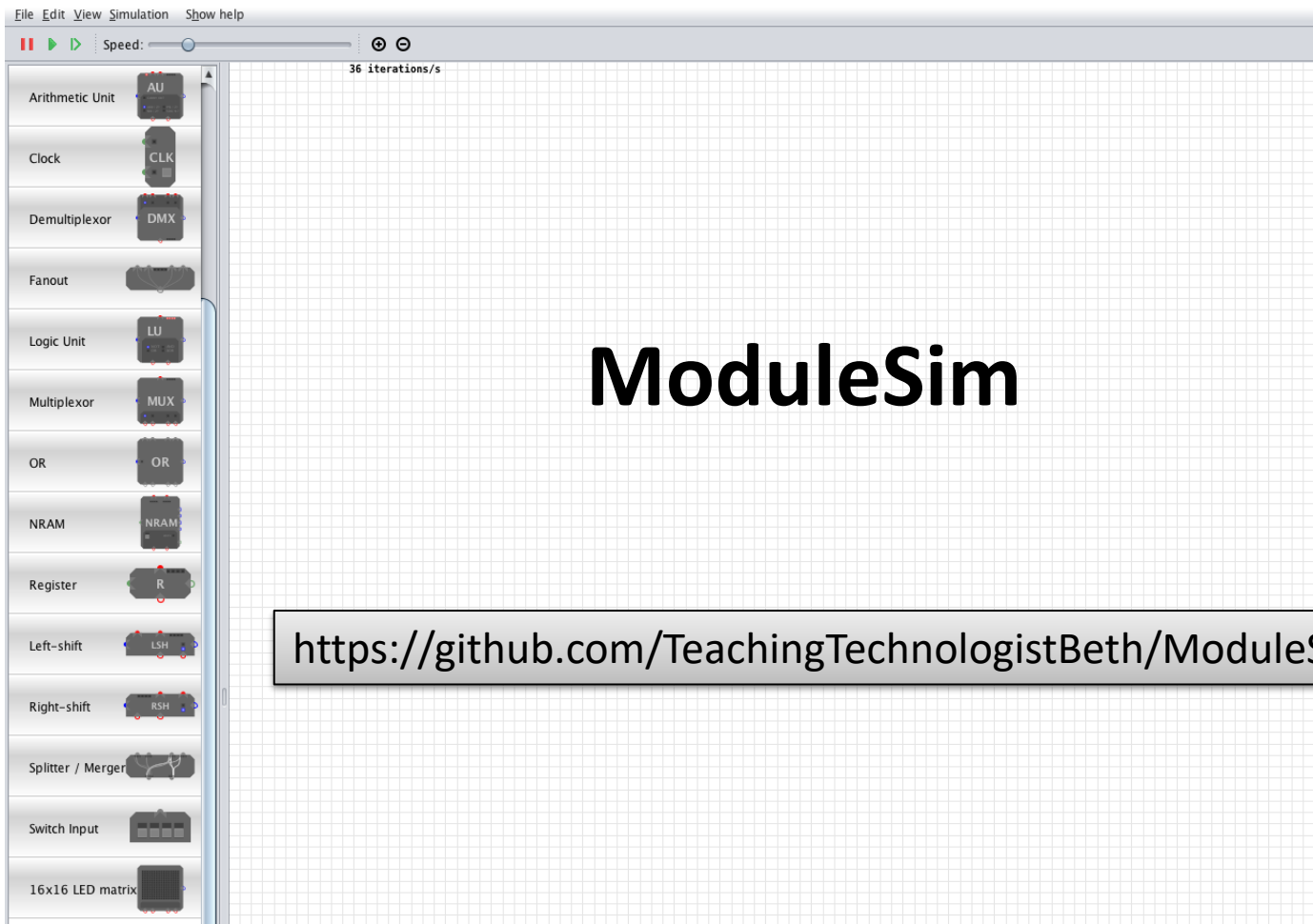
## 1. Modulesim

1. Bus in Modulesim
2. Different components in ModuleSim
3. Solve some design problems

## 2. At the end of this lecture:

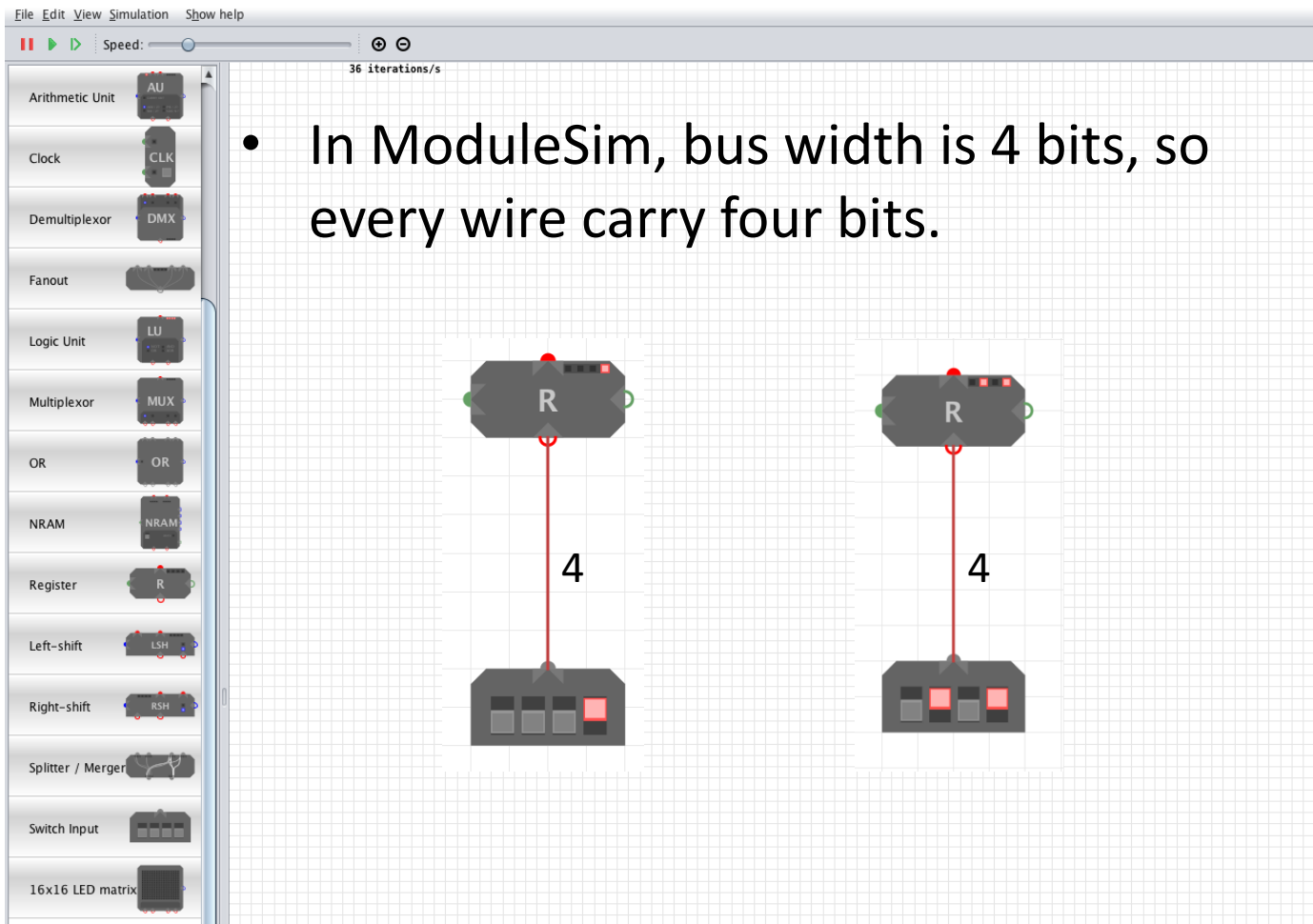
1. To use ModuleSim to implement and test your designs.

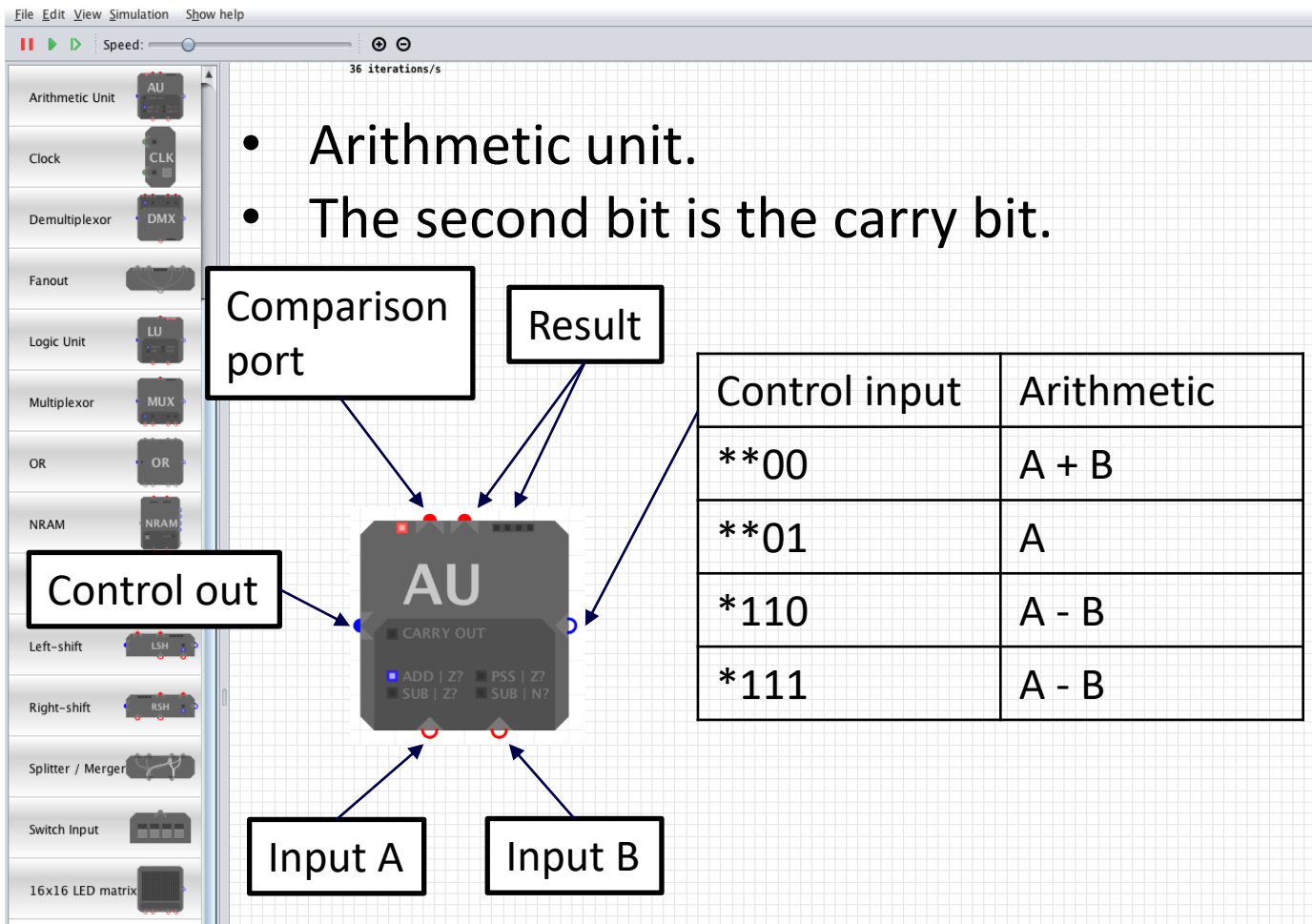
# Simulation



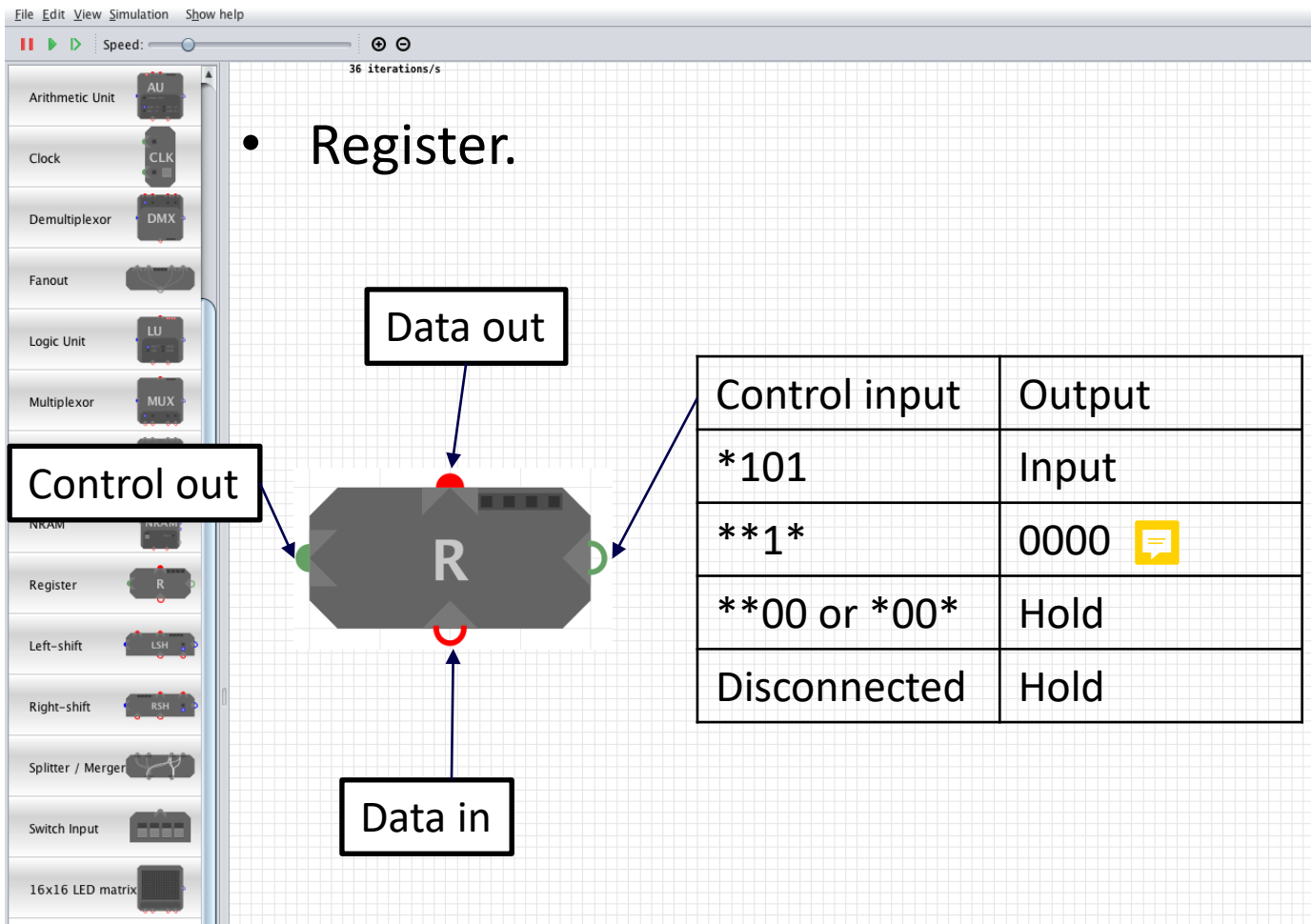
<https://github.com/TeachingTechnologistBeth/ModuleSim/releases>

# Bus





# Register



# Clock



File Edit View Simulation Show help

Speed:

36 iterations/s

Arithmetic Unit AU

Clock CLK

Demultiplexer DEM

Fanout FANOUT

Logic Unit LU

Multiplexer MUX

OR OR

NRAM NRAM

Register R

Left-shift LSH

Right-shift RSH

Splitter / Merger SPLITTER


Switch Input SWITCH

16x16 LED matrix LED

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



File Edit View Simulation Show help

Speed:

36 iterations/s

Arithmetic Unit

Clock

Demultiplexor

Fanout

Logic Unit

Multiplexor

OR

NRAM

Register

Left-shift

Right-shift

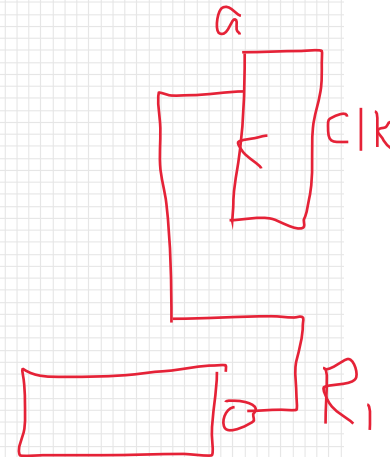
Splitter / Merger

Switch Input

16x16 LED matrix

| 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 – use 2-1

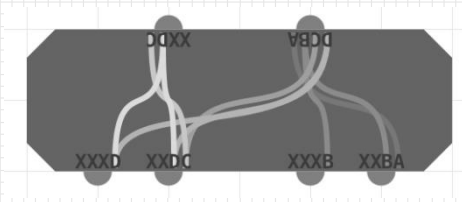


File Edit View Simulation Show help

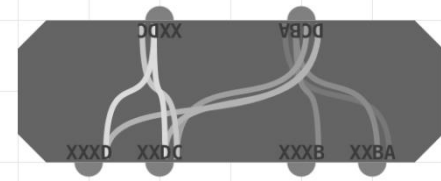
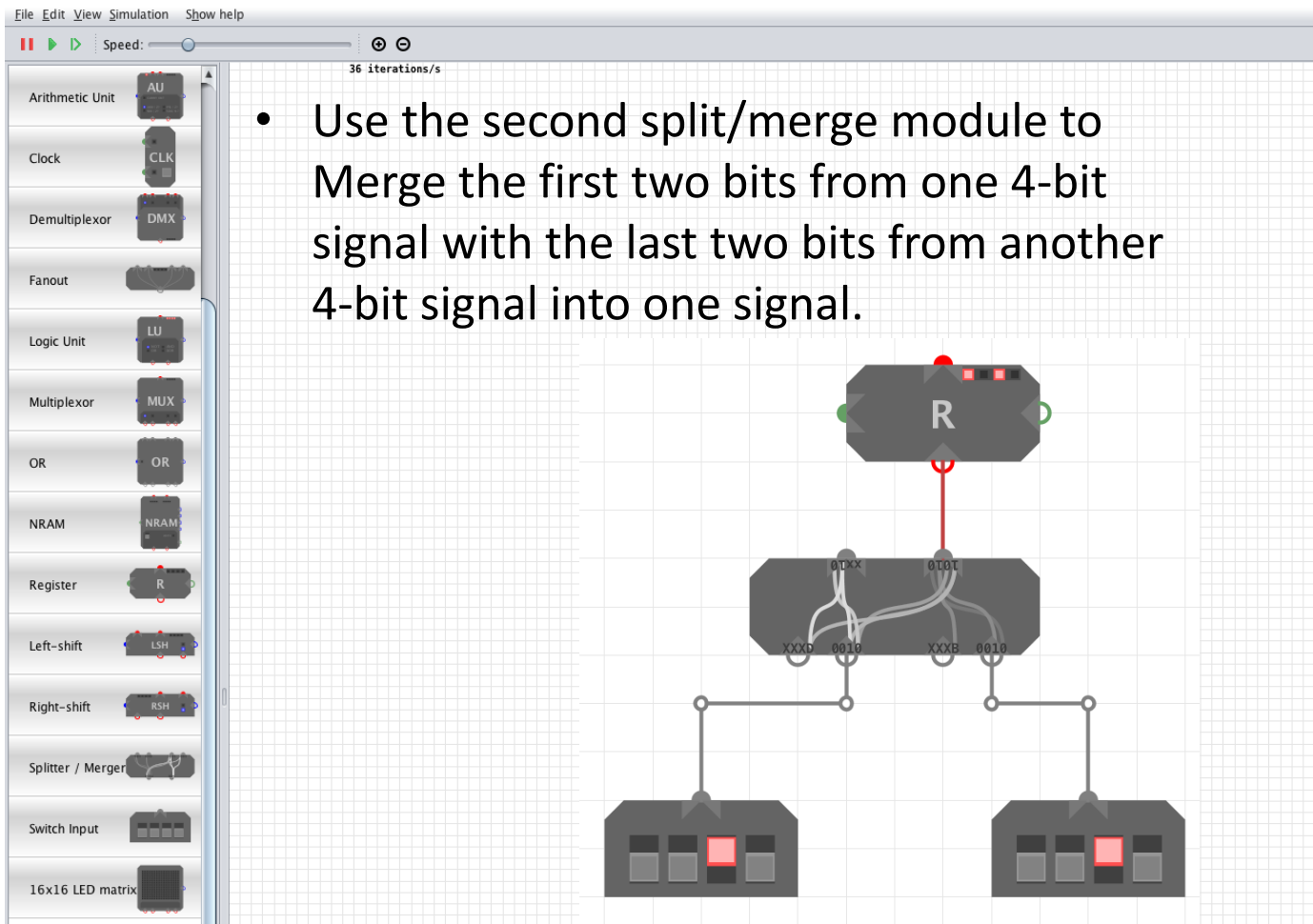
Speed: 36 iterations/s

Arithmetic Unit (AU)  
Clock (CLK)  
Demultiplexor (DMX)  
Fanout  
Logic Unit (LU)  
Multiplexor (MUX)  
OR  
NRAM  
Register (R)  
Left-shift (LSH)  
Right-shift (RSH)  
Splitter / Merger  
Switch Input  
16x16 LED matrix

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



# Split/Merge – use 1-1

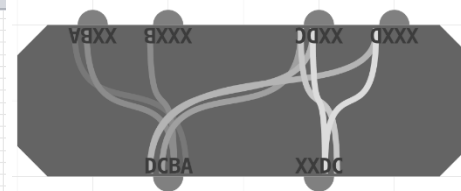


File Edit View Simulation Show help

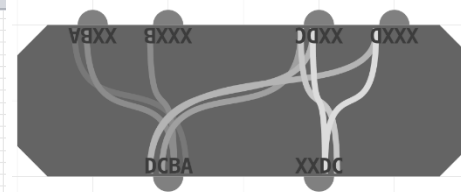
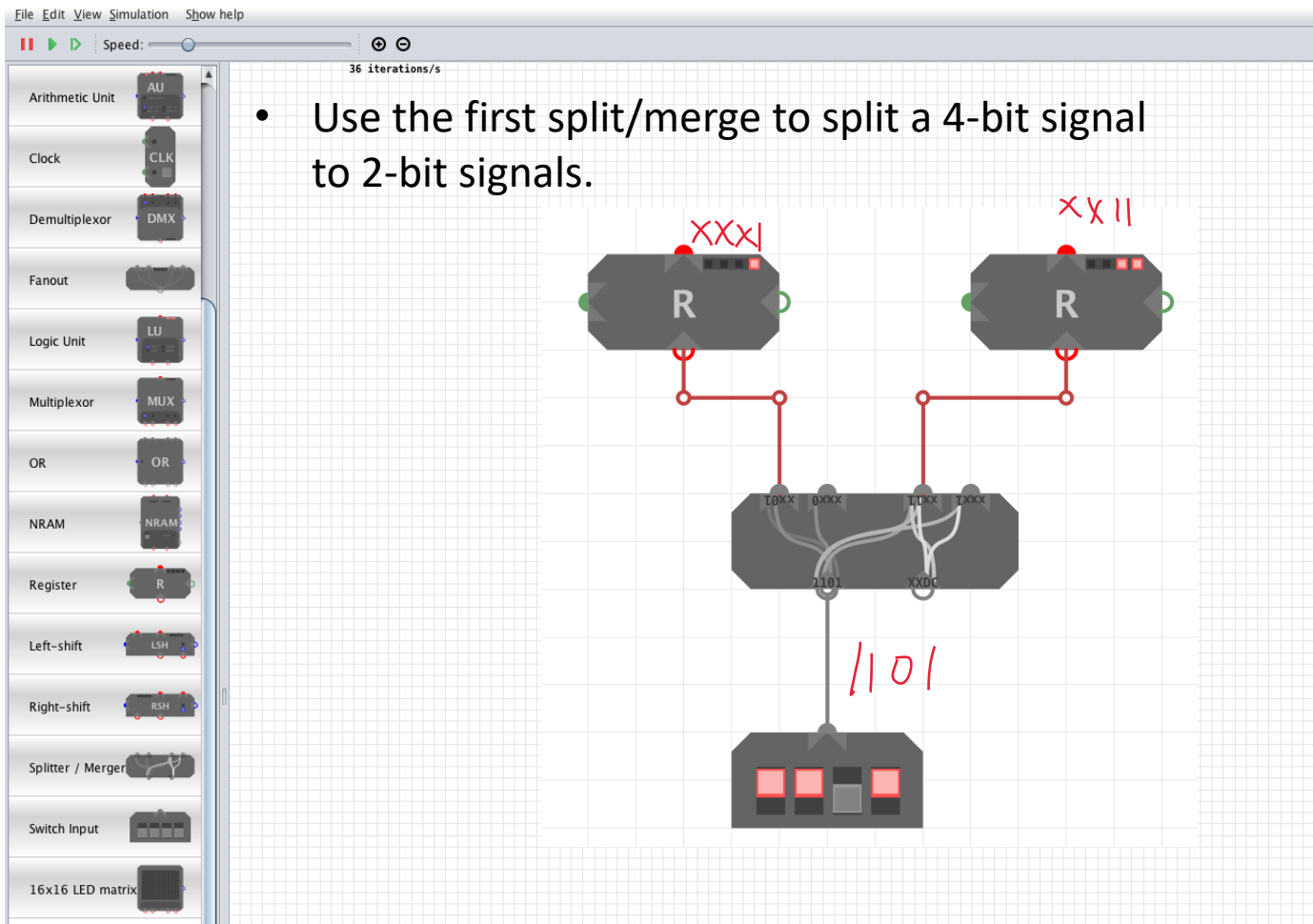
Speed: 36 iterations/s

Arithmetic Unit (AU), Clock (CLK), Demultiplexor (DMX), Fanout, Logic Unit (LU), Multiplexor (MUX), OR, NRAM, Register (R), Left-shift (LSH), Right-shift (RSH), Splitter / Merger, Switch Input, 16x16 LED matrix

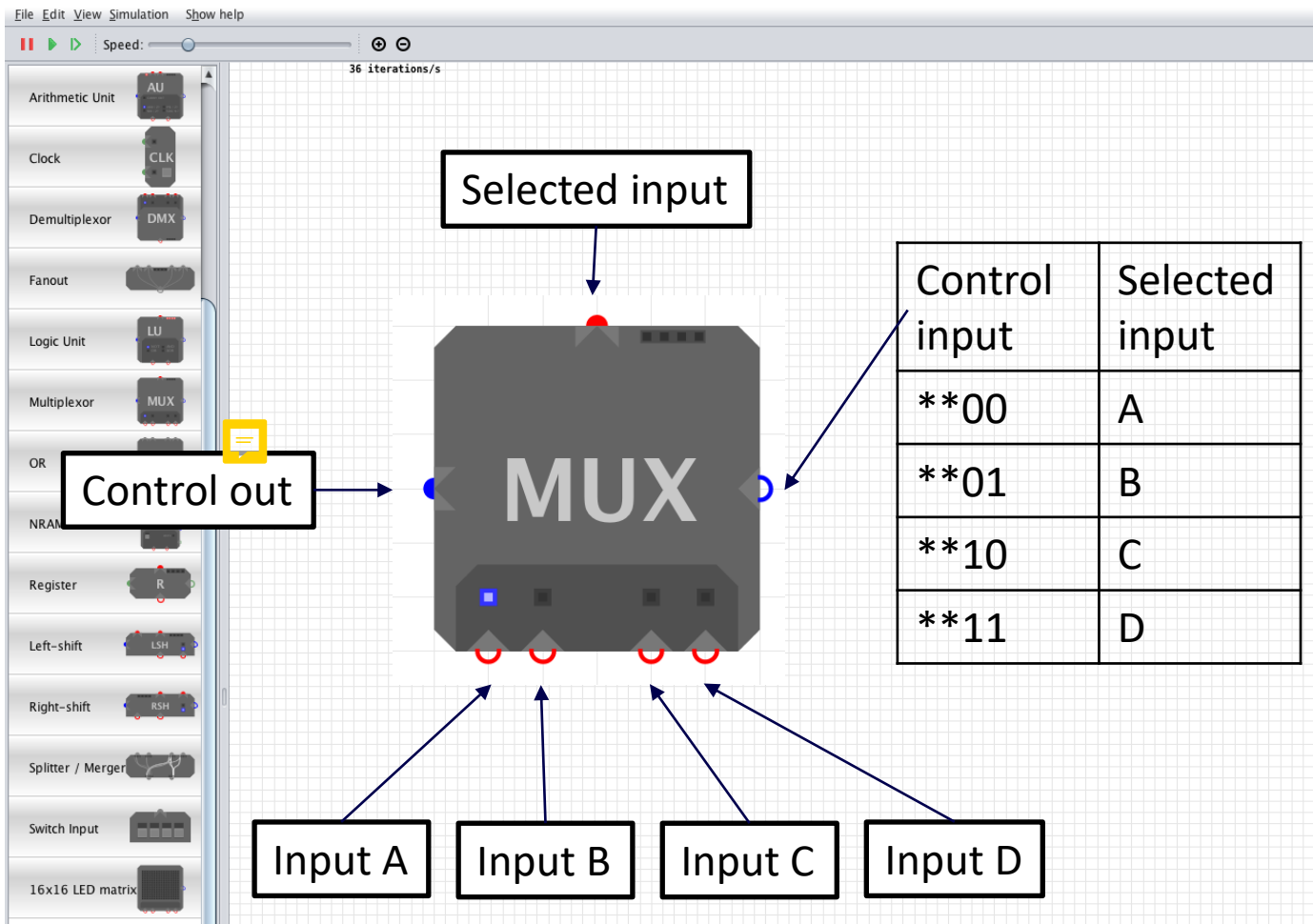
1. Use the first split/merge to split a 4-bit signal  $1_3 1_2 1_1 1_0$  to 2-bit signals. Such that the first two bits (LSB's) of the signal are the first two bits of one signal  $xx 1_1 1_0$  and the two high bits of the original signal are first two bits of another signal  $xx 1_3 1_2$ .



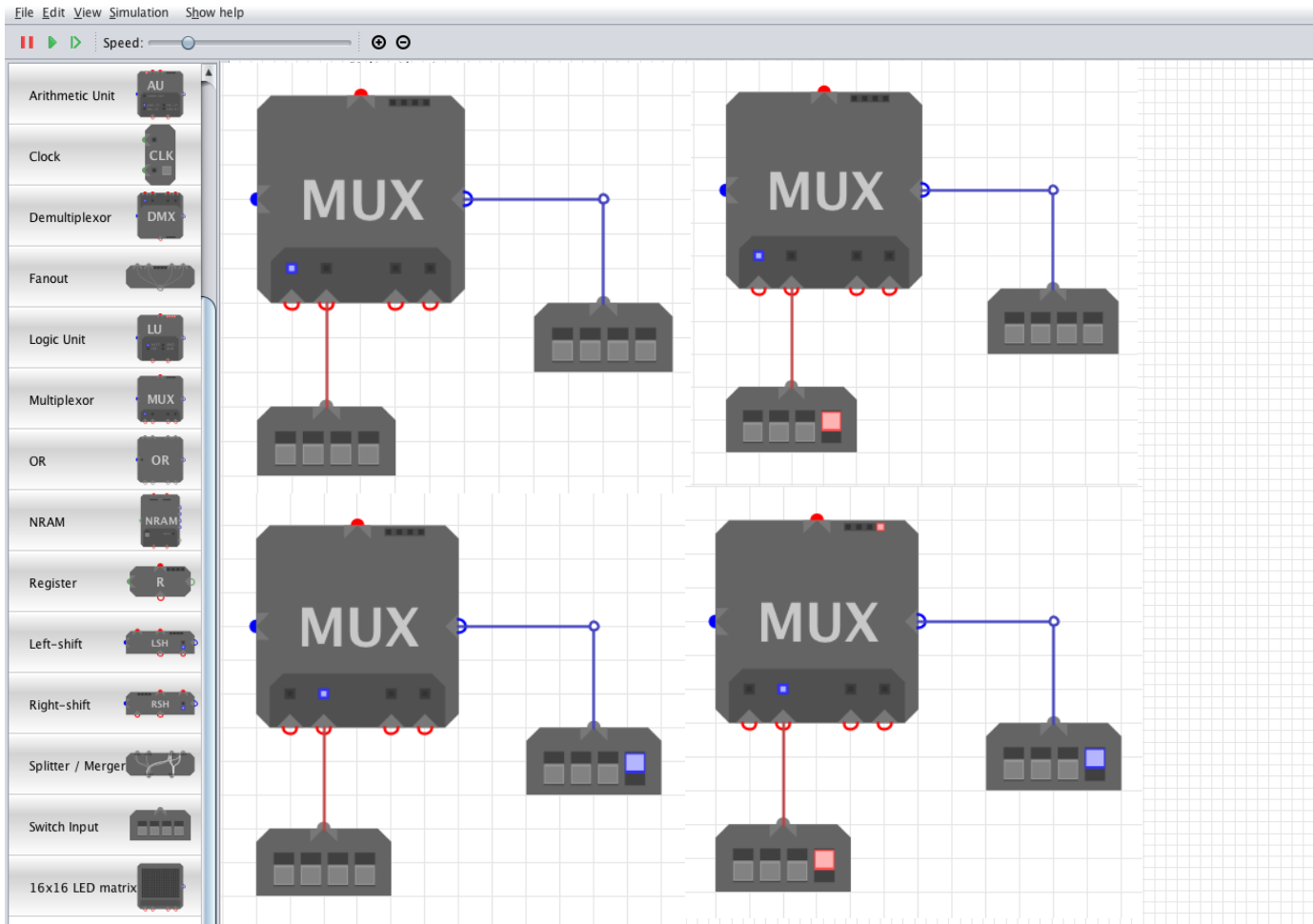
# Split/Merge – use 1-2



# MUX

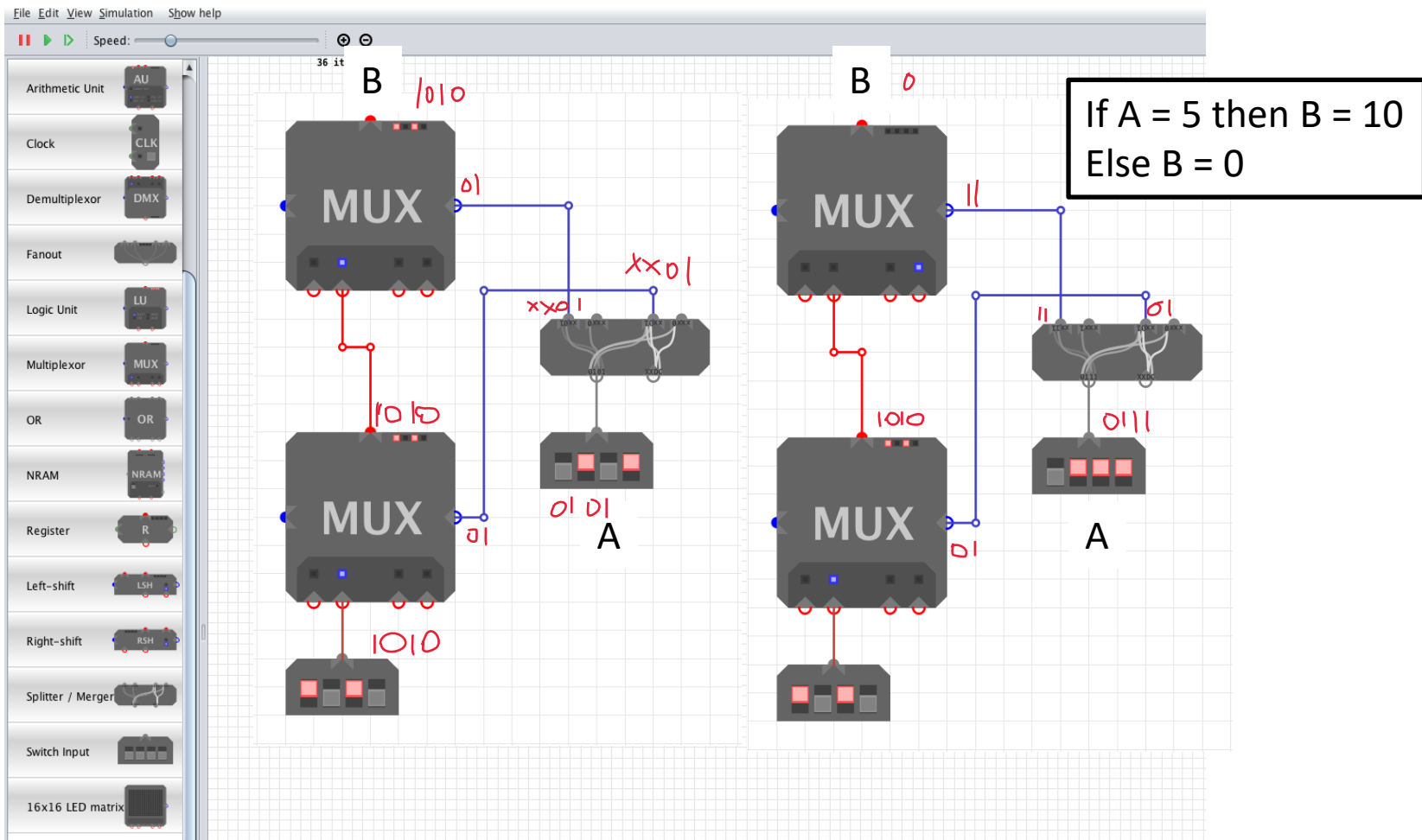


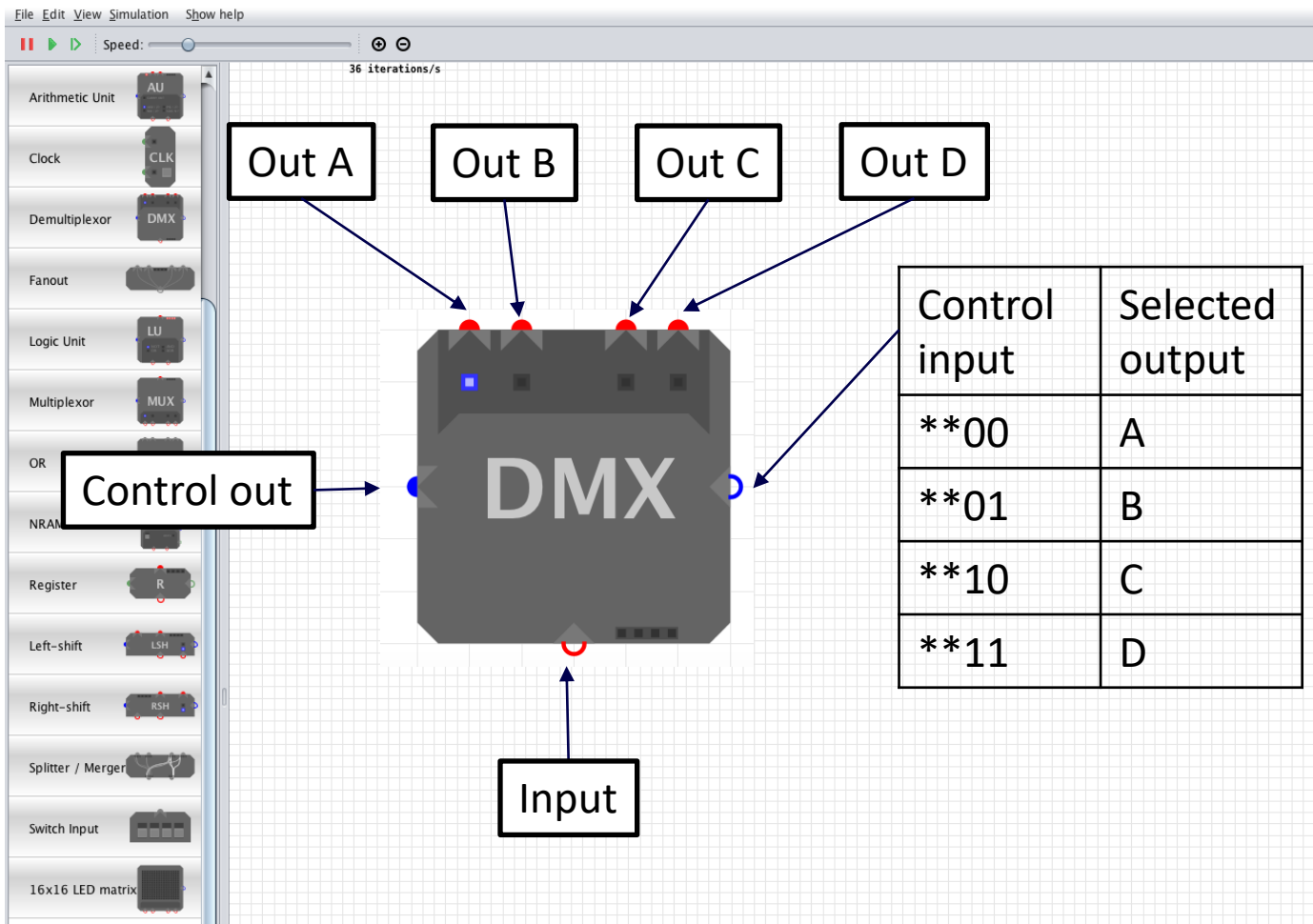
# AND from MUX





# 🔥 If – Else with MUX





# DMX with counter

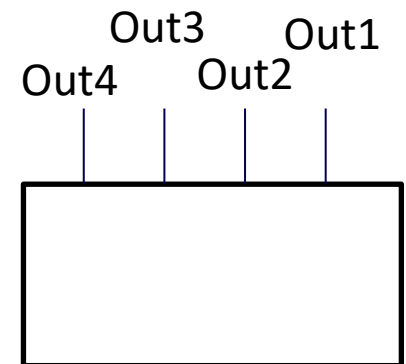


File Edit View Simulation Show help  
Speed: 36 iterations/s

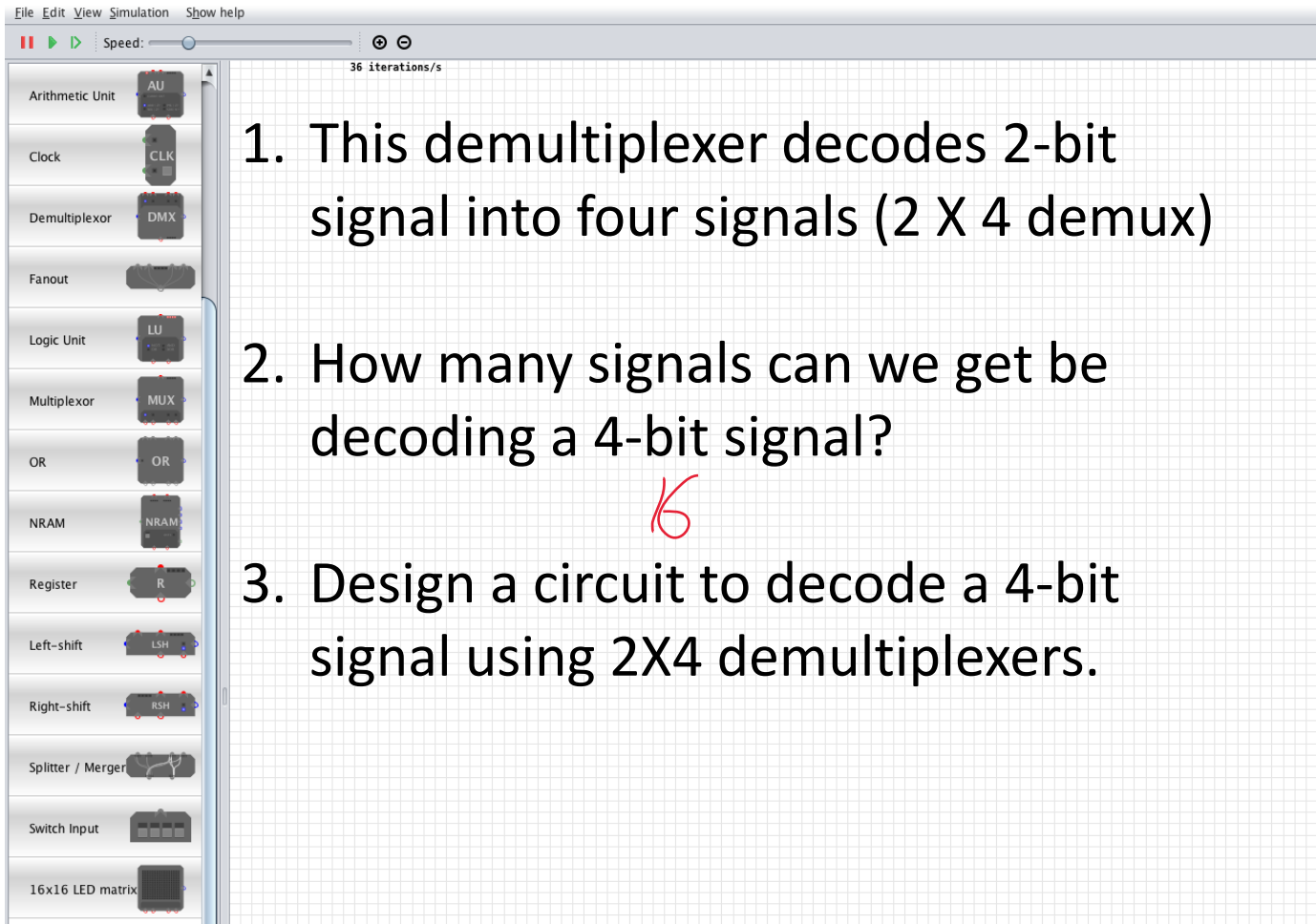
- Use a counter and a 2X4 demultiplexer to produce four signals such that one of them is high at a time.

| 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



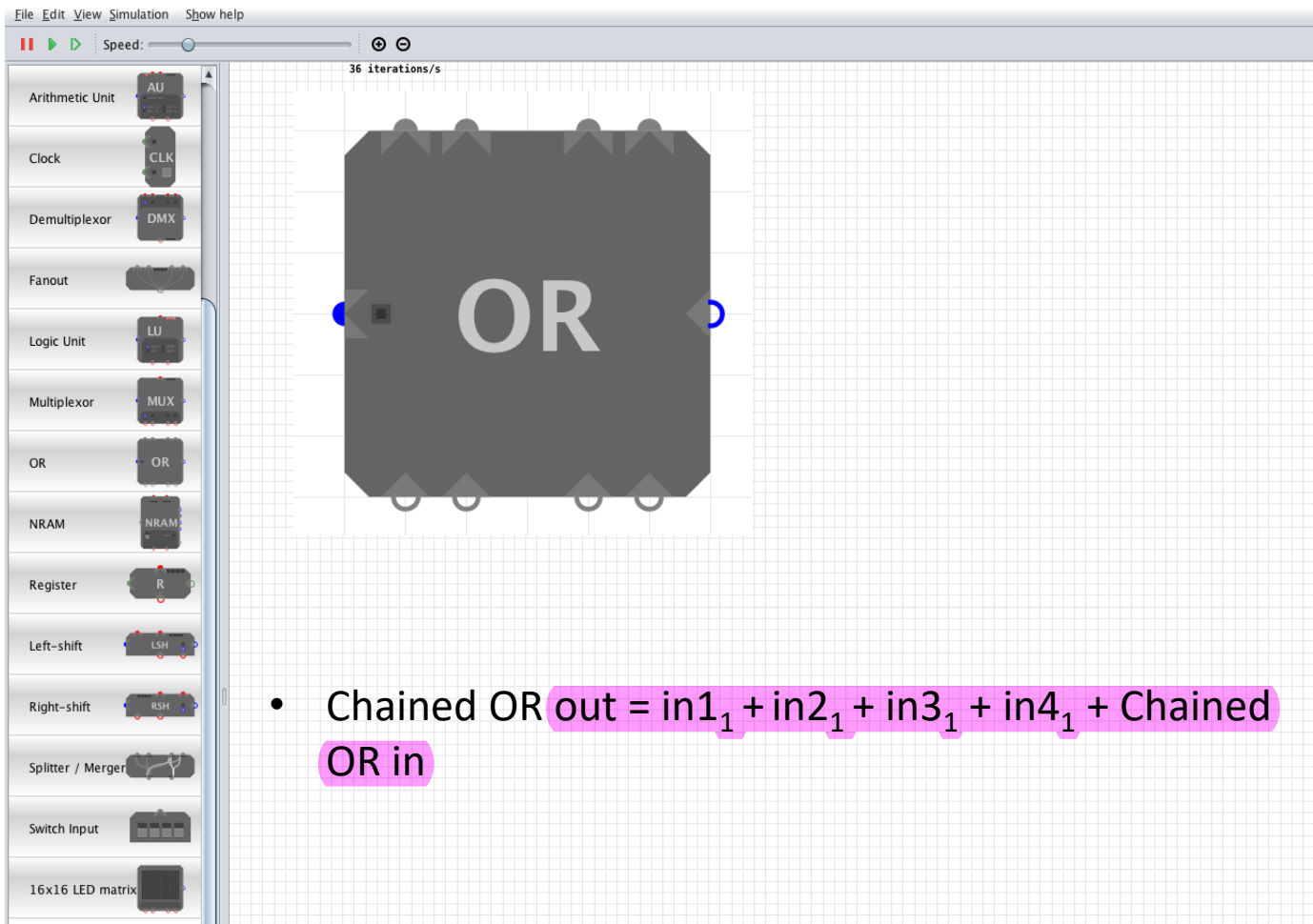
1. This demultiplexer decodes 2-bit signal into four signals (2 X 4 demux)

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.

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

# OR



# Fanout

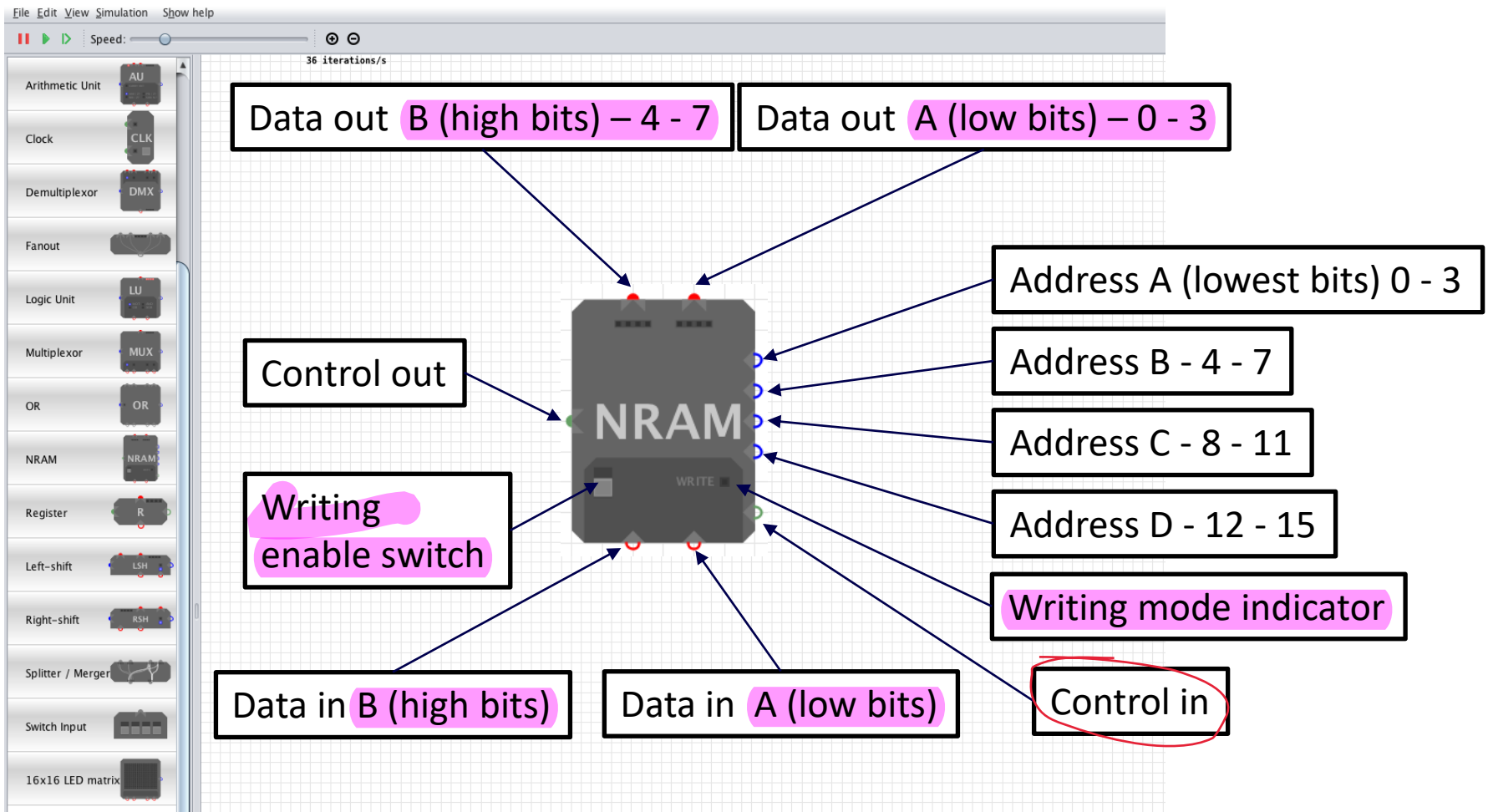


File Edit View Simulation Show help

Speed: 36 iterations/s

- Input = out<sub>1</sub> = out<sub>2</sub> = out<sub>3</sub> = out<sub>4</sub>
- It's just a junction or a repeater.

# Memory



# Memory - size



File Edit View Simulation Show help

Speed: 36 iterations/s

Arithmetic Unit AU

Clock CLK

Demultiplexor DMX

Fanout

Logic Unit LU

Multiplexor MUX

OR

NRAM

Register R

Left-shift LSH

Right-shift RSH

Splitter / Merger

Switch Input

16x16 LED matrix

1. What is the size of the data elements in this memory? 8
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)

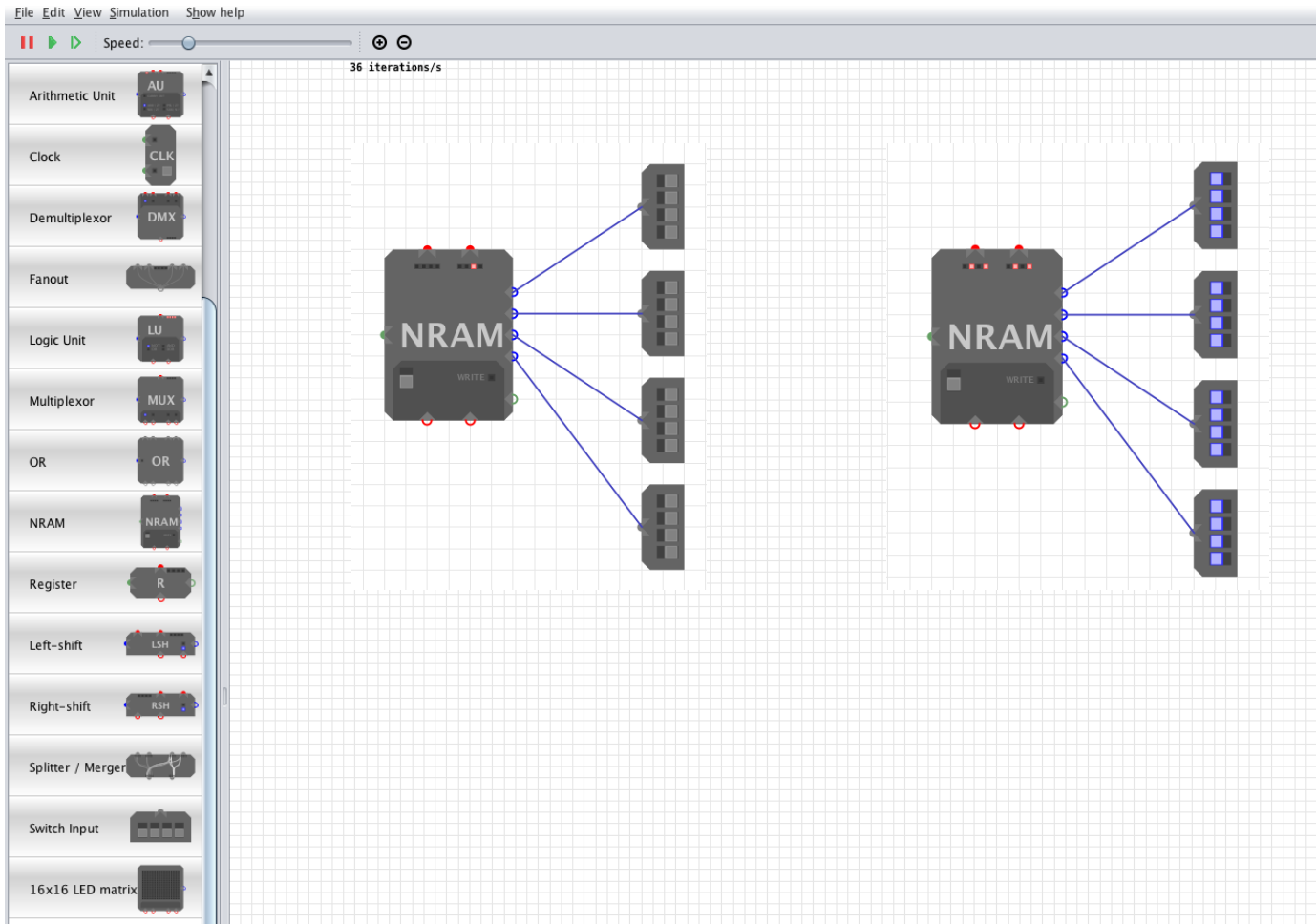
Add (4 – 7)

Add (8 – 11)

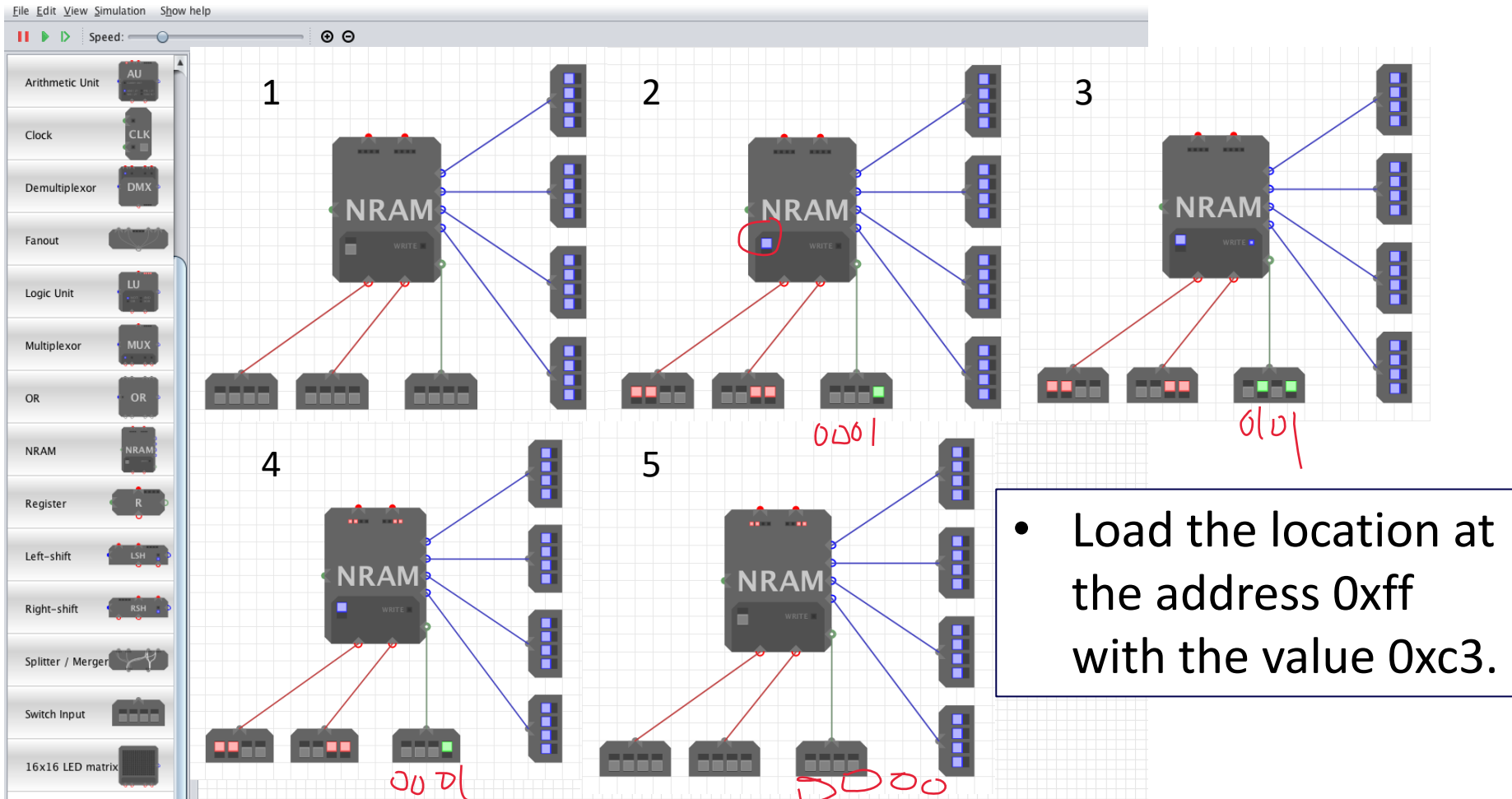
Add (12 – 15)



# Memory - read



# Memory – write



# Memory – load data

The screenshot displays a digital logic simulator interface. On the left is a component palette with various logic blocks. The main workspace shows a circuit with several NRAM components connected to a bus. A context menu is open over one NRAM component, listing actions like 'Rotate', 'Copy', 'Paste', and 'View/Edit NRAM Data'. The 'View/Edit NRAM Data' option is selected, opening a window titled 'NRAM'. This window contains a table of memory addresses and their corresponding data values. A callout points to a specific memory location, showing the hex value '02 13 60 48' being loaded into it.

| Address | 00 | 01 | 02 | 03 | 04 | 05 |
|---------|----|----|----|----|----|----|
| 0x000F  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x0014  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x0019  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x001E  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x0023  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x0028  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x002D  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x0032  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x0037  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x003C  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x0041  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x0046  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x004B  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x0050  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x0055  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x005A  | 00 | 00 | 00 | 00 | 00 | 00 |
| 0x005F  | 00 | 00 | 00 | 00 | 00 | 00 |

File Edit View Simulation Show help  
Speed: [Slider]  
[Buttons: Stop, Play, Pause]

Arithmetic Unit (AU), Clock (CLK), Demultiplexor (DMX), Fanout, Logic Unit (LU), Multiplexor (MUX), OR, NRAM, Register (R), Left-shift (LSH), Right-shift (RSH), Splitter / Merger, Switch Input, 16x16 LED matrix

NRAM Data Window:

| File               | Go | Last Changed |    |
|--------------------|----|--------------|----|
| Load Data          | 60 | 48           | 55 |
| Save Data          | 00 | 00           | 00 |
| Close without save | 00 | 00           | 00 |
| Close with save    | 00 | 00           | 00 |

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

data.hex

02 13 60 48

Data.hex

02 13 60 48


# Memory - design problem -1



File Edit View Simulation Show help  
Speed: 36 iterations/s

- 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

File Edit View Simulation Show help  
Speed: 36 iterations/s

Arithmetic Unit (AU)  
Clock (CLK)  
Demultiplexor (DMX)  
Fanout  
Logic Unit (LU)  
Multiplexor (MUX)  
OR  
NRAM  
Register (R)  
Left-shift (LSH)  
Right-shift (RSH)  
Splitter / Merger  
Switch Input  
16x16 LED matrix

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 greater than zero and to store 0xf for each 4 bits if their value is zero.

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

→

| data | address |
|------|---------|
| 0xf2 | 0x0     |
| 0x13 | 0x1     |
| 0x6f | 0x2     |

# 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.