PRACTICAL – XII

APPLICATIONS USING LOGISM

WHAT IS LOGISM SIMULATOR?

Logisim is an educational tool for designing and simulating digital logic circuits. With its simple toolbar interface and simulation of circuits as you build them, it is simple enough to facilitate learning the most basic concepts related to logic circuits. With the capacity to build larger circuits from smaller subcircuits, and to draw bundles of wires with a single mouse drag, Logisim can be used (and is used) to design and simulate entire CPUs for educational purposes.

Logisim is used by students at colleges and universities around the world in many types of classes, ranging from a brief unit on logic in general-education computer science surveys, to computer organization courses, to full-semester courses on computer architecture.

FEATURES OF LOGISM SIMULATOR

- The drawing interface is based on an intuitive toolbar. Color-coded wires aid in simulating and debugging a circuit.
- The wiring tool draws horizontal and vertical wires, automatically connecting to components and to other wires. It's very easy to draw circuits!
- Completed circuits can be saved into a file, exported to a GIF file, or printed on a printer.
- Circuit layouts can be used as "subcircuits" of other circuits, allowing for hierarchical circuit design.
- Included circuit components include inputs and outputs, gates, multiplexers, arithmetic circuits, flip-flops, and RAM memory.
- The included "combinational analysis" module allows for conversion between circuits, truth tables, and Boolean expressions.

COMPONENTS OF LOGISM SIMULATOR

1.TTY:
This component implements a very simple dumb terminal. It receives a sequence of ASCII codes and displays each printable character. When the current row becomes full, the cursor moves to the following line, possibly scrolling all current rows up if the cursor was already in the bottom row. The only supported control sequences are: backspace (ASCII 8), which deletes the last character in the final row, unless the final row is already empty; newline (ASCII 10), which moves the cursor to the beginning of the following line,

scrolling if necessary; and form-feed (ASCII 12, typed as control-L), which clears the screen.

2. Counter:
This register holds a single value, whose value is emitted on the output Q. Each time the clock input (diagrammed with a triangle on the component's south edge) triggers according to its Trigger attribute, the value in the register may update based on the two inputs on the component's west edge: The upper input is called load and the lower is called count, and they are interpreted as follows.

3. ROM:
The ROM component stores up to 16,777,216 values (specified in the Address Bit Width attribute), each of which can include up to to 32 bits (specified in the Data Bit Width attribute). A circuit can access the current values in ROM, but it cannot change them. The user can modify individual values interactively via the Poke Tool, or the user can modify the entire contents via the Menu Tool.

Unlike the RAM component, the ROM component's current contents are stored as an attribute of the component. Thus, if a circuit containing a ROM component is used twice, then both ROM components will hold the same values. Also because of this behavior, the current ROM contents are stored in files created by Logisim.

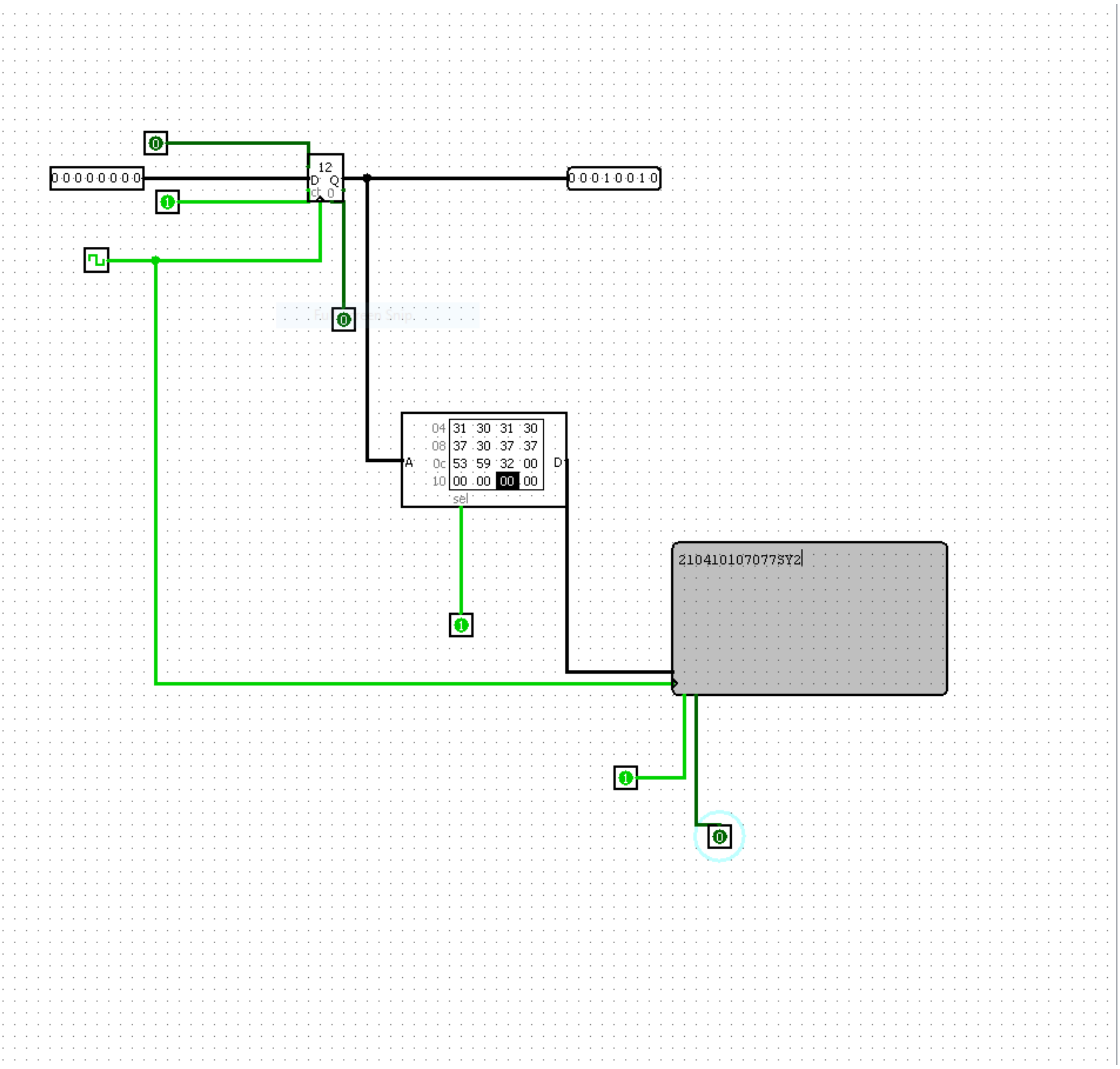
Current values are displayed in the component. Addresses displayed are listed in gray to the left of the display area. Inside, each value is listed using hexadecimal. The value at the currently selected address will be displayed in inverse text (white on black).

4. CLK:

Simulates the clock.

WORKING

We have used ROM element which stores the enrolment number and class of students which in separated by a dash (-) and terminated by a '\n'. The ROM element has Address bit width of 5 bits and Data bit width of 7 bits.
For selecting Address we have used an counter which counter from 0 to 12 (hex). The count is incremented on each consecutive clock pulse and the next ASCII character is printed on screen via a TTY display.


0
00000000
12
D Q
ct 0
1
0
00010010
04 31 30 31 30
08 37 30 37 37
0c 53 59 32 00
10 00 00 00 00
sel
A
D
1
210410107077SY2
1
0