A white background with black text

Description automatically generated

Department of Computer Science

Institute of Business Administration, Karachi

Lab #8: Moore Sequence Detector & Temperature

Sensor

Digital Logic Design

# Objective:

The objectives of this experiment are:

1. Use Logisim-evolution to model and simulate the Moore sequence detector.
2. Designing Temperature Sensor as a Moore Machine on Logisim evolution.

# Introduction:

*Logisim-Evolution* is a Java application, so a Java runtime environment will need to be installed before using the application. Many students who are taking a digital logic class already have a Java runtime on their computer and can skip this step, but those who do not will need to install the Java runtime. That process is not covered in this man- ual but information about installing the Java runtime environment is available at <http://www.oracle.com/technetwork/java/javase/downloads/index.html>. It can be confusing to know which version of Java to download but students working on the labs in this manual only need the runtime, called *JRE* on the website. Students who are also in programming classes will likely already have the runtime as part of the Java Developer’s Kit (JDK). It can be tricky testing the Java installation since the Chrome, Firefox, and Edge browsers will not run Java apps, but students can open a command prompt and enter java -version to see what version of Java their computers are running, if any.

*Logisim-Evolution (*[logisim-evolution download | SourceForge.net](https://sourceforge.net/projects/logisimevolution/)) is available as a free download. Visit the website and about halfway down the page find a section named “Running Logisim-evolution.” Click the “here” link at the end of the first sentence in that section.

Since the *Logisim-Evolution* file is a Java application, it does not need to be installed like most software. To start *Logisim-Evolution,* double- click the *Logisim-Evolution* shortcut. That will start Java and then run the *Logisim-Evolution* application. Also, *Logisim-Evolution* will not need to be uninstalled when it is no longer needed since it is not actually installed, the *Logisim-Evolution* file can simply be deleted.

* + 1. *Beginner’s Tutorial*

*Logisim-Evolution* comes with a beginner’s tutorial available in Help

-> Tutorial. That tutorial only takes a few minutes and introduces.

students to the major components of the application. Students should complete that tutorial before starting this lab.

* + 1. *Logisim-evolution Workspace*

Start *Logisim-Evolution* by double-clicking its icon. The initial *Logisim- Evolution* window will be similar to Figure [1.1](#_bookmark18).

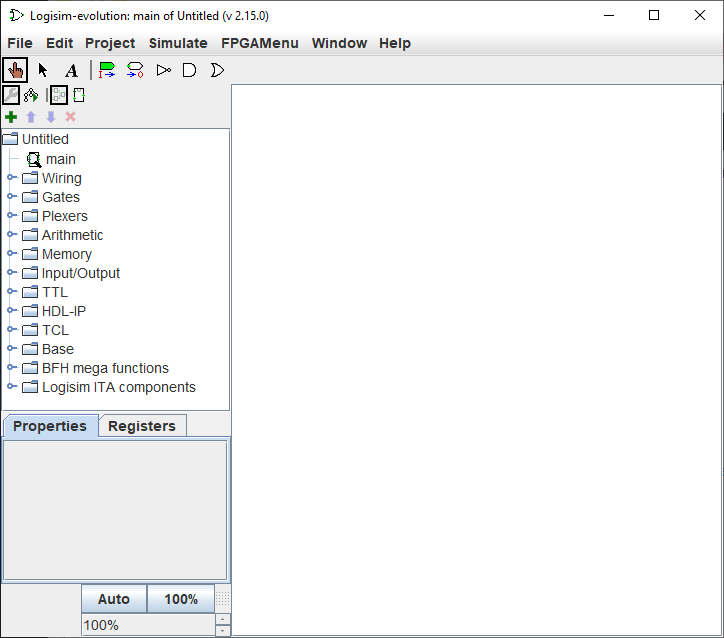


Figure 1.1: Logisim-evolution Initial Screen

The *Logisim-Evolution* space is divided into several areas. Along the top is a text menu that includes the types of selections found in most programs. For example, the “File” menu includes items like “Save” and “Exit.” The “Edit” menu includes an “Undo” option that is useful. In later labs, the various options under “Project” and “Simulate” will be described and used. Items in the “FPGAMenu” are beyond the scope of this class and will not be used. Of particular importance at this point is “Library Reference” in the “Help” menu. It contains information about every logical device available in *Logisim-Evolution* and is very useful while using those components in new circuits.

Under the menu bar is the Toolbar, which is a row of eight buttons that are the most commonly used tools in *Logisim-Evolution* :

* + - * **Pointing Finger**: Used to “poke” and change input values while the simulator is running.
* **Arrow**: Used to select components or wires in order to modify, move, or delete them.
* **A**: Activates the Text tool so text information can be added to the circuit.
* **Green Input Port**: Creates an input port for a circuit.
* **White Output Port**: Creates an output port for a circuit.
* **NOT Gate**: Creates a NOT gate.
* **AND Gate**: Creates an AND gate.
* **OR Gate**: Creates an OR gate.

The Explorer Pane is on the left side of the workspace and contains a folder list. The folders contain “libraries” of components organized in a logical manner. For example, the “Gates” folder contains vari- ous gates (AND, OR, XOR, etc.) that can be used in a circuit. The four icons across the top of the Explorer Pane are used for advanced operations and will be covered as they are needed.

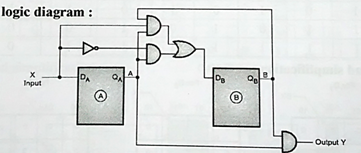
The Properties panel on the lower left side of the screen is where the properties for any selected component can be read and set. For ex- ample, the number of inputs for an AND gate can be set to a specific number.

The drawing canvas is the largest part of the screen. It is where circuits are constructed and simulated.

**Moore Sequence Detector:**

A Moore Sequence Detector is a digital circuit that detects a specific sequence of input signals and generates an output accordingly.

Following is the circuit diagram for implementation of 101 sequence detector.



**Method:**

* + - 1. Set Simulate -> Tick Frequency to 4 Hertz. This will simulate clock that ticks once per second.
      2. Click Simulate -> Chronogram to set up the *chronogram*.
      3. Click *Start Chronogram* and the screen illustrated in Figure 1.2 pops up.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated Figure 1.2: Chronogram Starting

* + - 1. The *chronogram* has five buttons that control the simulator.

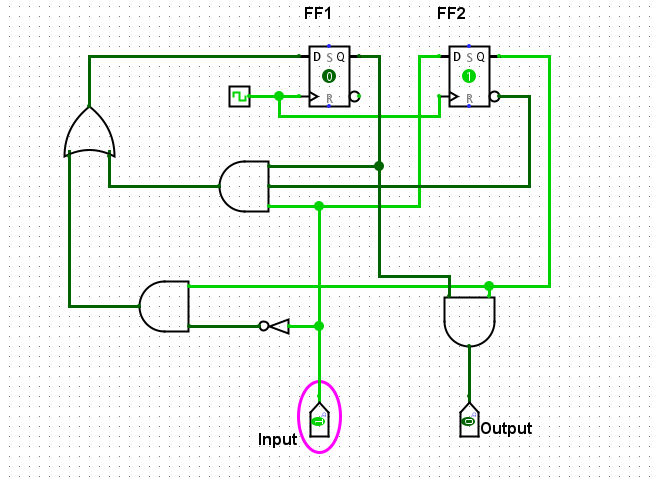
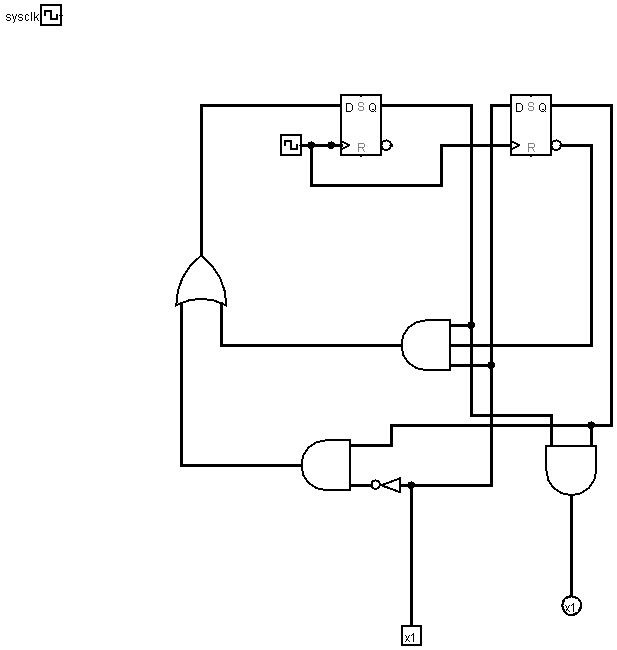


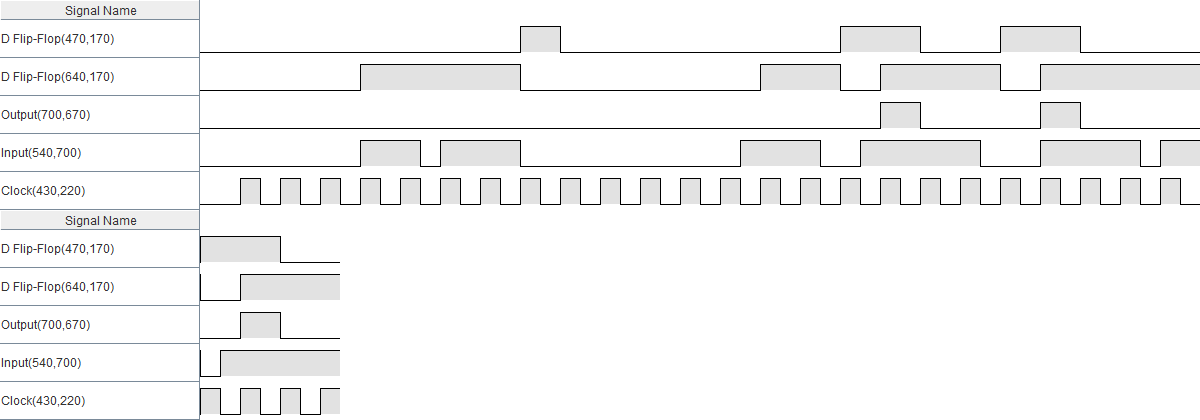
Figure 1.13: Chronogram Controls

* Button One: Start/Stop the simulation.
* Button Two: Simulate one step.
* Button Three: Start/Stop *sysclk*. This will “turn on” the chronogram and begin creating a timing diagram.
* Button Four: Step one *sysclk* tick. This will tick the *sysclk* one time. Since this lab set up the *sysclk* for four ticks per second this button would need to be clicked four times to extend the timing diagram one second.
* Button Five: Step one *clk* tick. This extends the timing di- agram by one complete clock tick, or one second in this circuit.
  + - 1. Click button three to start the *chronogram* and watch the timing diagram unfold. After a few seconds click that button a second time to stop the *chronogram*.
      2. The following can be done once the timing diagram is complete.
* Click on the timing diagram to set the cursor (indicated by a red line). Once the cursor is set the values for each signal at the cursor’s location are printed next to the signal’s label on the left edge of the timing diagram.
* Hover the mouse over the timing diagram and roll the mouse wheel to zoom the timing diagram appearance.
  + Click “Export” to save the timing diagram signal levels in a text file. That file can later be loaded to reevaluate the timing diagram.
  + Click “Export as image” to save the timing diagram as a PNG file.

# Tasks for the Lab

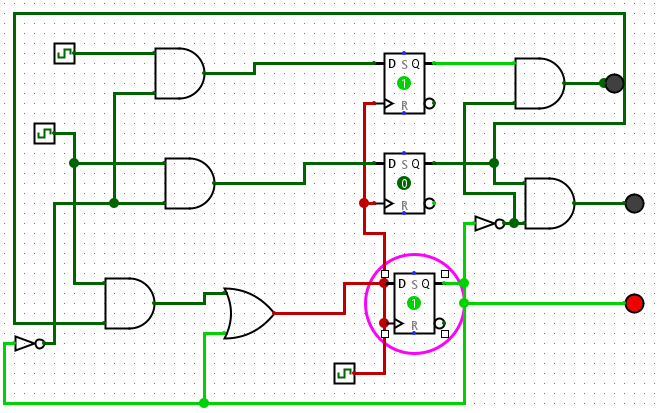
1. Design and simulate the Moore sequence detector for 101 sequences as shown below in the Logisim-evolution and attach the screenshots of their timing diagram as well.





1. Design a temperature sensor as Moore machine as shown below in the Logisim evolution along with their timing diagrams in such a way that for

* 000 (all LEDs should be OFF)
* 100 (First LED ON)
* 010 (Second LED ON)
* 11O (First & Second LED ON)
* 101 (Third LED ON)
* 001 (Third LED ON)
* 011 (Third LED ON)
* 111 (Third LED ON)



Attach screenshots of the timing diagram for all the given combinations just like the given timing diagram for 101 (Third LED ON).

