MobiLogic Simulator

Team Members: Raymond Chan (7541), Hunter Kennedy (0099), Andy Li (1380), Yifan Liu (7182), Brandon Lau (5096), Ashwin Muralidharan (9217)

1. Introduction

1.1 Problem Statement

Students first entering a Computer Architecture class will learn about how to build complex circuits using software such as Logisim. However, such software solutions are often complex and have a steep learning curve. Therefore, removing most of the heavyweight features in those circuit building software and only including the essentials as a precursor to complex circuit building software is beneficial for the students.

The result would be a mobile application that can run/test simple circuits and provide feedback to the user for them to have a better comprehension of logic circuits, gates, etc.

1.2 Background

Logic simulators are widely used for the verification of integrated circuits. Users could use simulators to test or find problems in their designed circuits. Logic gates are also one of the most important parts of computer architecture. Operations like and, or, not and etc. provided in an arithmetic logic unit (ALU) can also be simulated by logic gates or the combination of gates. In addition, logic gates are the building blocks of digit systems.

1.3 Customer

The customers that this MobiLogic application is for are students/other individuals who are first entering a Computer Architecture class or want to play around and run simple logic circuits.

1.4 User Story Mapping

1. **Users can select a type of gate, move this gate onto a virtual circuit board, and connect it with other components in this circuit, or delete existing components.**

User can select a type of gate	Place selected gates onto board	Connect/disconnect gates via/from their inputs and outputs	Can choose to move gates around	Able to delete gates

List of buttons for the gates	Track mouse click as location to place gate	Track mouse click selected input and output pairs	Track mouse drag	When mouse click on gate, gate disappear
Visualize which button is being pressed	Create object when button is pressed	Visualize wire connection as straight line between the selected endpoints	Move gate image	Free existing connections
Create images for each gate		The selected endpoints should be highlighted?	Move wires with the gate	Visualize the deletion

2. **Users can save and load circuits, as well as use a saved circuit as a subcircuit for the current circuit.**

User can save circuits	User can load circuits	User can use a saved circuit as a sub-circuit
Design a circuit storage class	Design a circuit storage class	Design a circuit class with the same interface as a gate. This will enable circuits to be used as components.
Implement Codename One's external saving with this storage class	Load from Codename One's External Saving	Testing
Testing	Testing	

3. **As users edit the circuit, they can see the critical path of the current circuit, its propagation delay, and maximum possible operating frequency.**

User can view propagation delay for circuit	User can view critical path of circuit	User can view maximum possible operating frequency
Design traversal algorithm for circuit representation	Allow for visualization of critical path (recoloring of wires, etc.)	Set up area for visualizing/displaying this information and work out

		formula for finding this frequency
Use traversal algorithm to add propagation delays for each component in each path and give the highest of these paths	Implement algorithm for creating a linked list/tree/(whatever we use to represent circuit) that references all components on the critical path	Pull data from critical path propagation delay to compute operating frequency
Implement (recursive?) algorithm for including the propagation delay of subcircuits	Link visualization with algorithm	Bug fixing/testing

4. **Users can run the circuit simulation to check how changing inputs affect outputs.**

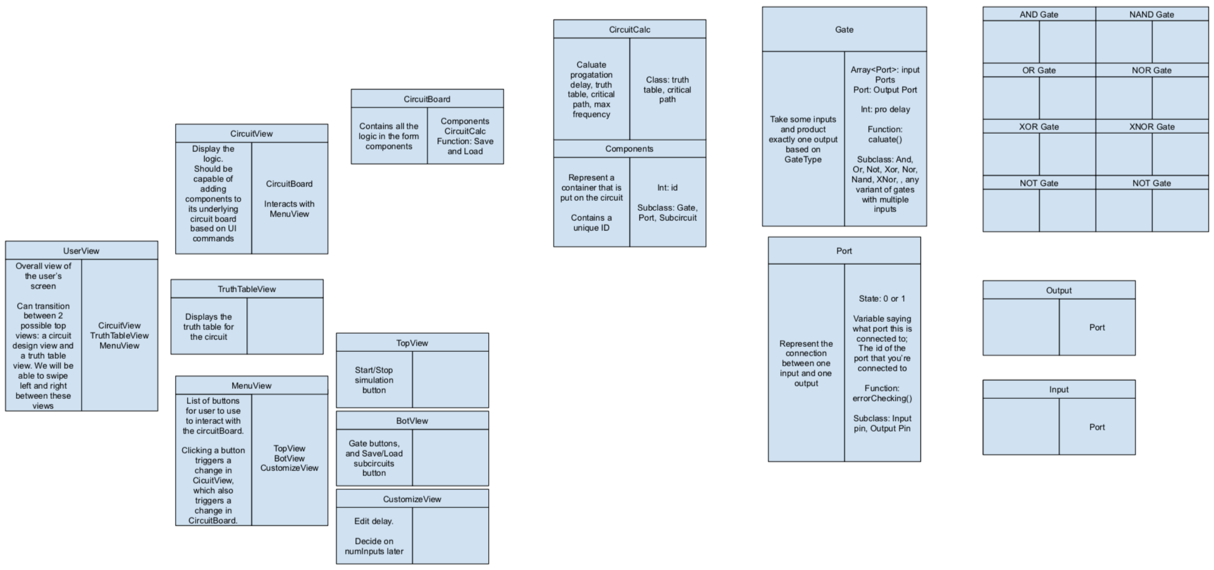
Users can run the simulation	Users can change the inputs on the circuit that affect the output
User can press a button for the simulation (just UI, no behavior yet)	User can toggle the inputs, 0 or 1, on the circuit for a single gate (no behavior yet, only toggling)
Implement running the circuit diagram on the given input, which produces some result	User can toggle the inputs on a single gate to achieve an output
Implement the on/off of LEDs while the circuit is running	User can toggle inputs on multiple gates (AND, OR, NOT, etc.) to achieve the respective behavior as the output
	User can connect subcircuits as inputs to a gate, and achieve an output (which then may be another input to another gate)

5. **Users can view the logic state (i.e. truth table) for the circuit diagram.**

User can view a truth table for the circuit
A button should be assigned to show the truth table of the current circuit
I/O should be saved to generate a truth table
Truth Table should be generated by iterating through all possible input combinations and calculating possible outputs

1.5 CRC Cards

(Will be uploaded as a separate PDF document in the submission because the text is too small here)



UserView
Overall view of the user's screen

Can transition between 2 possible top views: a circuit design view and a truth table view. We will be able to swipe left and right between these views
CircuitView TruthTableView MenuView
CircuitView
Display the logic. Should be capable of adding components to its underlying circuit board based on UI commands
CircuitBoard

Interacts with MenuView
CircuitBoard
Contains all the logic in the form components
Components CircuitCalc Function: Save and Load
TruthTableView
Displays the truth table for the circuit
MenuView
List of buttons for user to use to interact with the circuitBoard.

Clicking a button triggers a change in CicuitView, which also triggers a change in CircuitBoard.
TopView BotView CustomizeView
TopView
Start/Stop simulation button
BotView
Gate buttons, and Save/Load subcircuits button
CustomizeView
Edit delay.

Decide on numInputs later
CircuitCalc
Caluate progatation delay, truth table, critical path, max frequency
Class: truth table, critical path
Components
Represent a container that is put on the circuit

Contains a unique ID
Int: id

Subclass: Gate, Port, Subcircuit
Gate
Take some inputs and product exactly one output based on GateType
Array<Port>: input Ports
Port: Output Port

Int: pro delay

Function: caluate()

Subclass: And, Or, Not, Xor, Nor, Nand, XNor, , any variant of gates with multiple inputs
Port
Represent the connection between one input and one output
State: 0 or 1

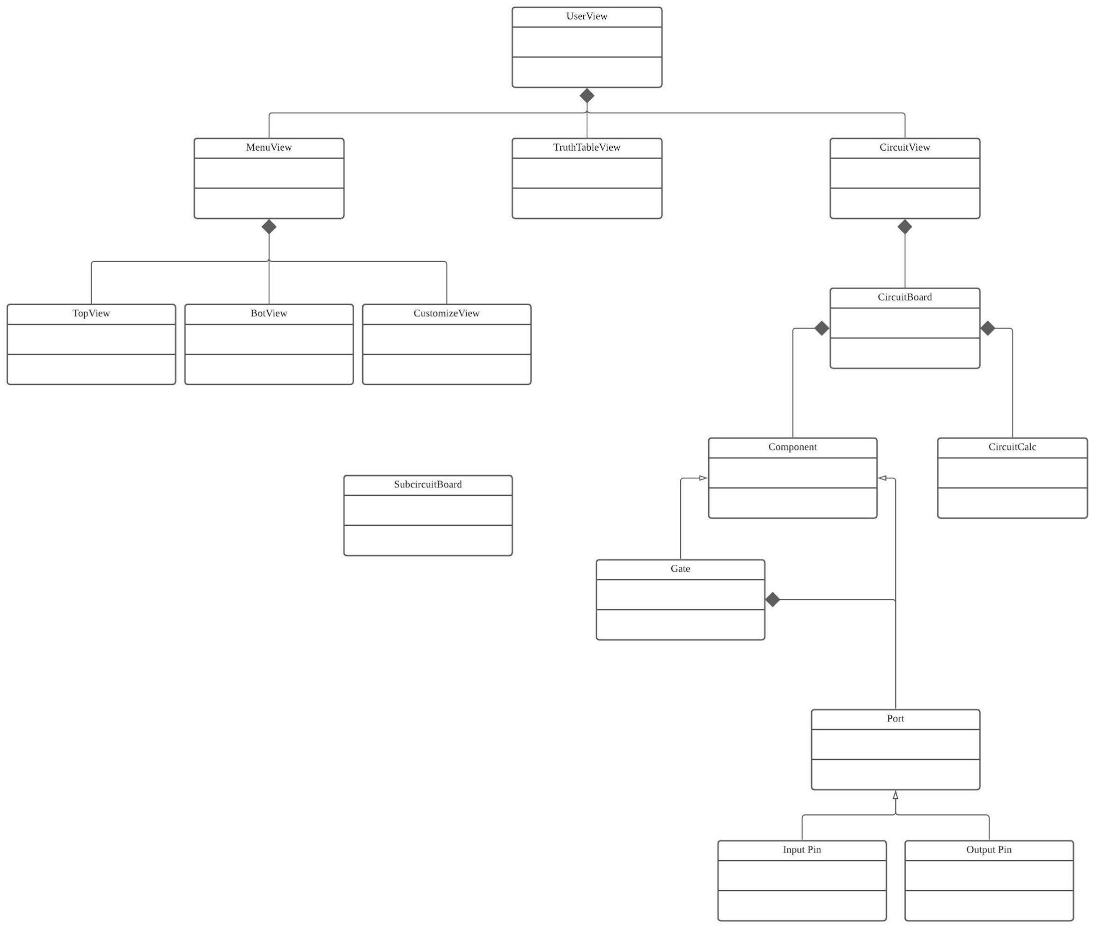
Variable saying what port this is connected to; The id of the port that you're connected to

Function: errorChecking()

Subclass: Input pin, Output Pin
AND Gate
NAND Gate
OR Gate
NOR Gate
XOR Gate
XNOR Gate
NOT Gate
NOT Gate
Output
Port
Input
Port

1.6 Class Diagram Template

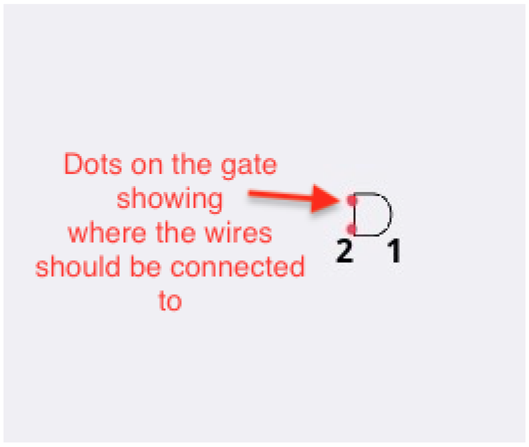
(Will be uploaded as a separate PDF document in the submission because the text is too small here)



1.7 Design Decisions

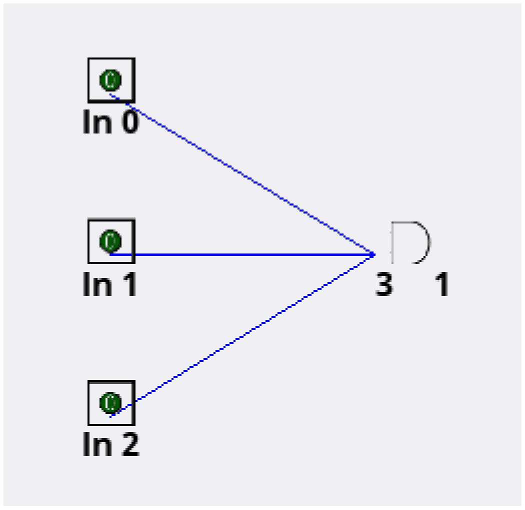
Input Pin on Gates (UI)

When designing the mobile logic simulator, we went with the approach of not explicitly showing the input pins on the gates for the UI, unlike in other logic circuit building software like Logisim, which is shown in the screenshot below.


Dots on the gate showing where the wires should be connected to
2
1

We did not include these dots for the input pins for the user to connect to because theoretically, these dots are purely for the frontend. The backend does not need to know the order or positioning of the input pins for common logic gates (not for subcircuits though).

Because this is a mobile application, we had to keep things simple and not overly complicate the UI. We decided to not include the UI input pins but chose to implement it like the following.



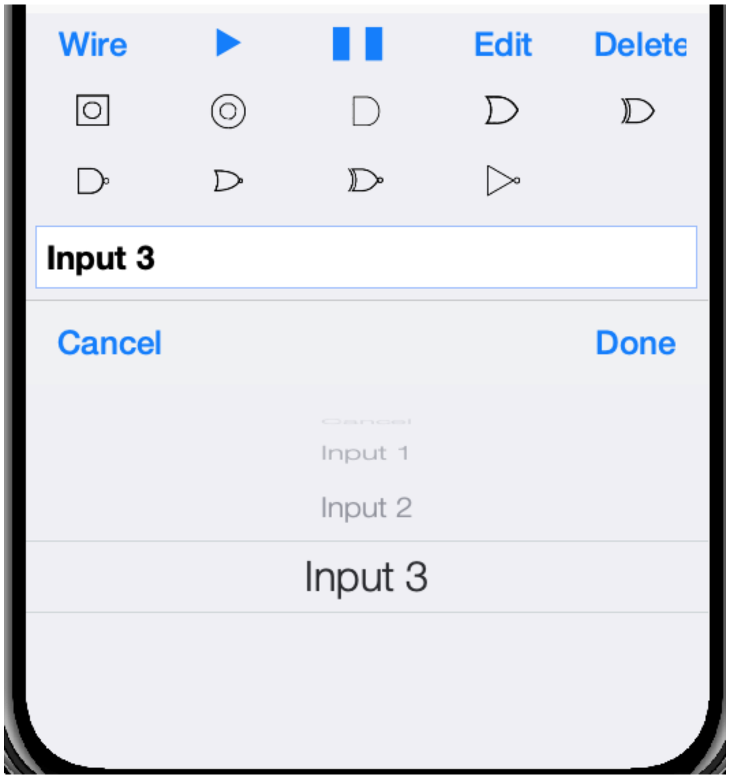
The number 3 on the bottom left of the AND gate is the number of inputs that are attached to it. This approach for the input pins is much better because

1. User can attach any amount of inputs to the gate (note that common logic gates except NOT gate, it needs 2 or more inputs to operate on)
2. We do not need to keep updating the gate UI every time a new input pin is added, which could take up a lot of room, and also the gate UI will need to scale in size. This is not good on a mobile device where the room is already limited by the small screen
3. Functionally, it works the same way as when the input pin UI was displayed on the gates since the order does not matter

Subcircuit Input Pin Ordering

Unlike for common logic gates, subcircuit input pin ordering matters. However, with the approach we have implemented as described above where the ordering of input pins for common logic gates does not matter, we had to allow a way to specify the ordering for subcircuits.

We went with the approach of including a popup from the bottom of the screen where the user can specify exactly where to connect the external input pin to at the input pin located inside the subcircuit when the user creates the wire connection.



Popup where the user specifies which input pin located inside the subcircuit that the external input pin connects to

We went with the approach of the popup to incorporate some easy-to-use app-centric designs into the mobile application. Additionally, the user had to explicitly state which internal input pin to connect to inside the subcircuit because this is how it is done in other circuit logic software like Logisim. The difference between ours and Logisim is that Logisim uses physical input pins on the gate where the user specifically drags a wire to connect to at a certain location on the subcircuit, but we cannot do that because we do not have the physical input pins on the gates/subcircuits for reasons explained in the above section.

Fixing the Circuit Board (Non-scrollable)

We decided to make the circuit board fixed (non-scrollable) on this mobile application because the use case for this application are students/individuals who want to test out simple circuits (See Background and Customer sections above for more details). If the circuit board was scrollable, then the user will be able to have a larger canvas to build circuits. However, because the user will be creating simple circuits, scrolling for the circuit board would not be necessary and we believe that the current fixed circuit board is a good amount of room already. In addition, the user can save and load subcircuits if they want to create slightly complex circuits while trying to preserve space.

Adding Wires between Gates/Input/Output Pins

In Logisim, to connect a wire between two gates/input/output pins, the user would simply drag a wire from the first object and connect it to the second object. However, on mobile, because of the small screen size, dragging a finger across the screen to connect two objects would not be the best approach since the gates/input/output pins are small, and it would be hard to connect or drag the wire to a specific location.

Therefore, we went with the approach of including a physical [Wire] button. After the user presses this button, they can then proceed with clicking on the first gate and then click on the second gate. A wire will automatically be connected between the two gates. This approach is more mobile friendly and does not require the user to drag a finger to connect the wire, which will be hard on a mobile device with a small screen.

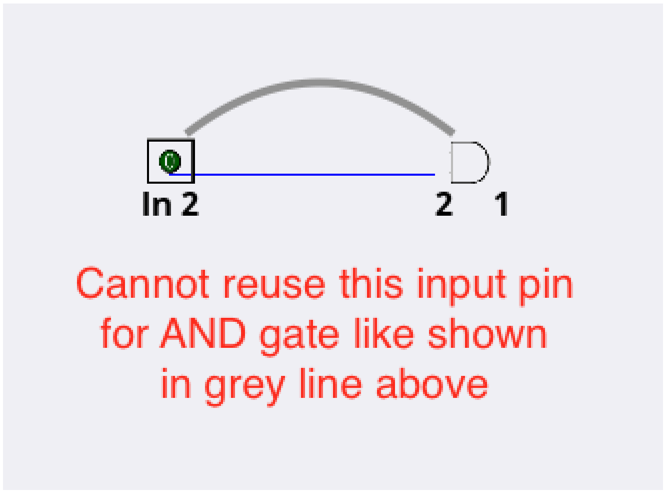
Deleting a Wire

Building off the previous section on how to add a wire between gates/input/output pins, to delete a wire, the user would first click on [Wire], then click on the first object (gates/input/output pins) that the wire is connected to and following click on the second object that the other end of the wire is attached to. This will delete the wire. We chose to do it this way because it was simply the inverse of adding a wire. When there is no wire connection between two objects, a wire will be created. If there is a wire connection between the two objects, then the wire will be created.

So, the [Wire] button technically has two functionalities— creating the wire or deleting it if it already exists. This approach is both intuitive, and also on a mobile device where room is limited, we save on valuable space because we do not need an extra button.

Reusing Same Input Pin on Gates

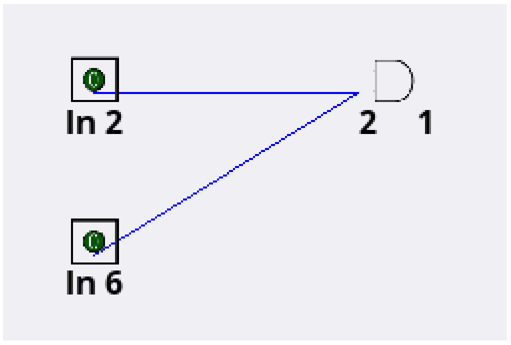
Let’s say we have an AND gate, and we would like to connect two input pins with value 0 into the AND gate to produce an output. The user first creates an input pin with the value 0 and connects it to the AND gate. For the second value 0 input to the AND gate, the user would not be able to reuse the first input pin (although it has the same value), and the user would need to create another, separate input pin to connect to the AND gate.



We decided to go with this approach of not being able to reuse input pins on gates due to our approach of how we implemented the wires (see above sections for more details). The [Wire] button has the two functionalities of either creating a wire between two objects or deleting the wire if it exists. So, the user would not be able to simply reuse an input pin twice because the second attempt to connect a wire from the input pin to a gate would simply remove it.

In addition, we did not support reusing the same input pins on gates because if we did, then the wire drawn (using the algorithm we currently use), would simply overlay on the existing wire. To the user, this just looks like one wire, and it would be hard to distinguish. We could, theoretically, do some wire algorithm that makes the wire connect non-overlapping like shown in the grey line in the above screenshot, but this was a low priority feature given the limited time we have on this project.

Users will need to do the following instead if they want to connect the same value inputs to a gate.

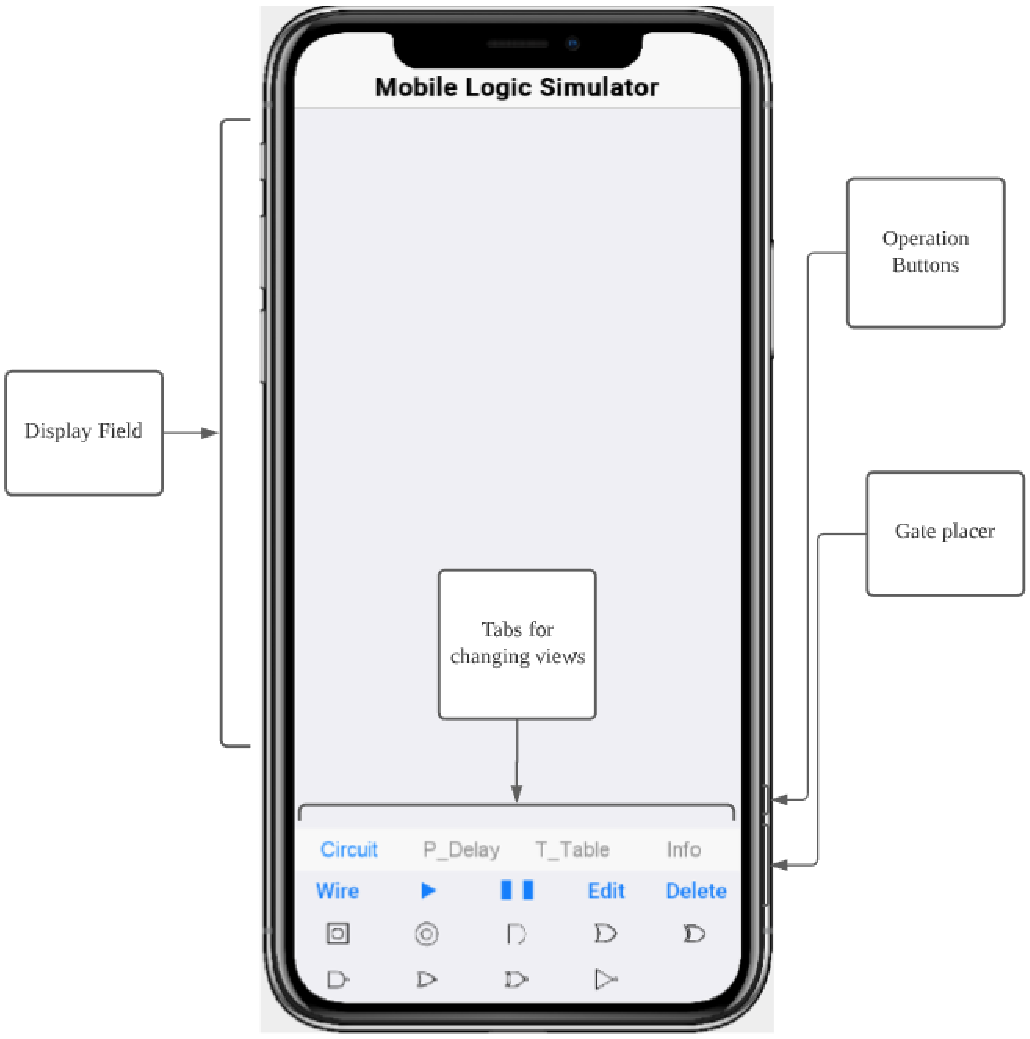


(Both input pins have value 0)

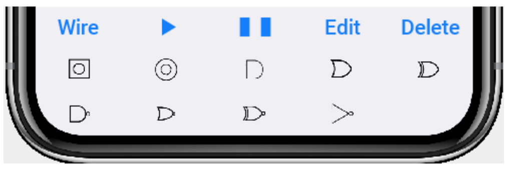
2. Description

2.1 Interface Construction

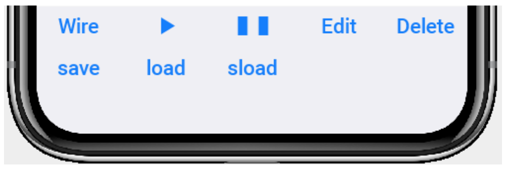
Our simulator's user interface has been divided into two major sections, the display field at the top and the operation field at the bottom. The information displayed in the top display field could be switched by clicking the tabs below. "Circuit" will show the current circuit layout, "P_delay" will show the propagation delay of the current circuit, and "T_Table" will show the truth table of the current circuit. The other field is the operation field, users can do the simulation, edit the component on the circuit board, and create new gates on the circuit board.



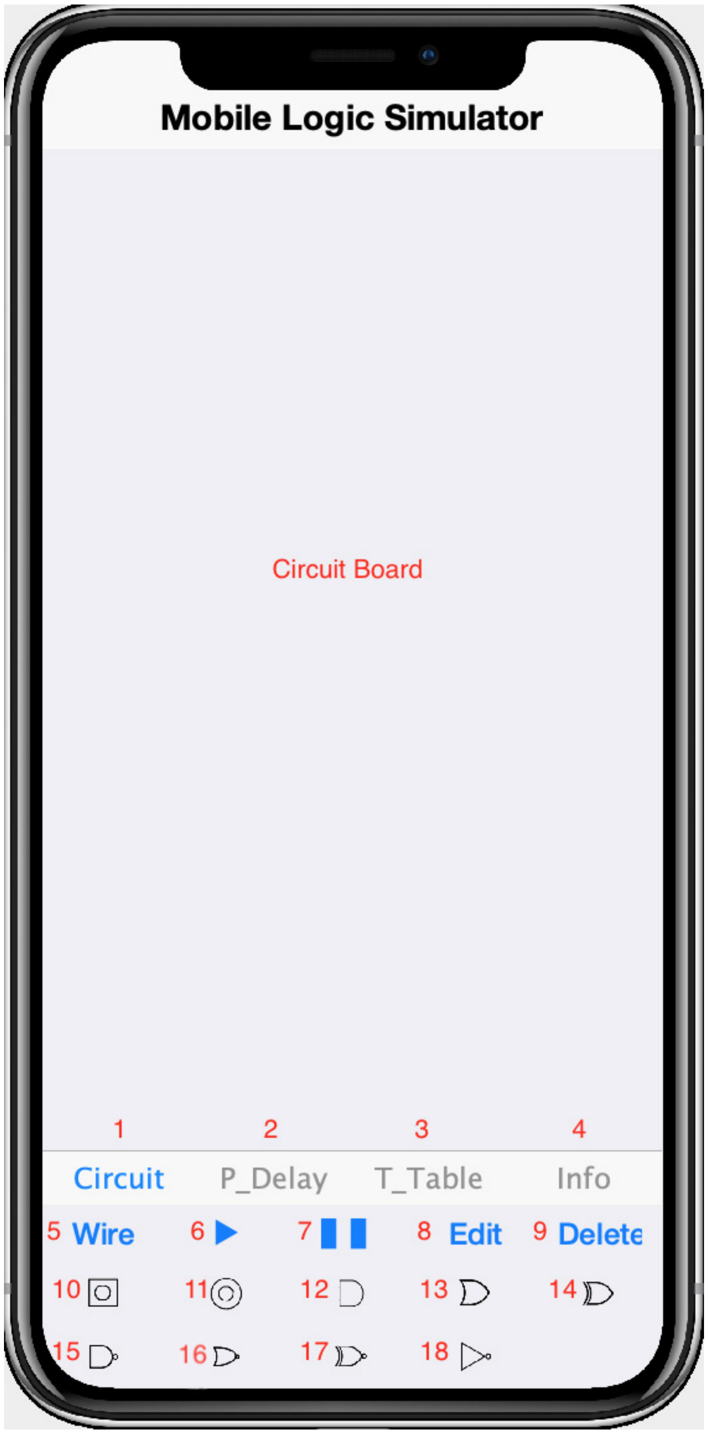
- Reasoning: Mobile, user will just be creating simple circuits, not complex ones



Swipe to view Save/Load

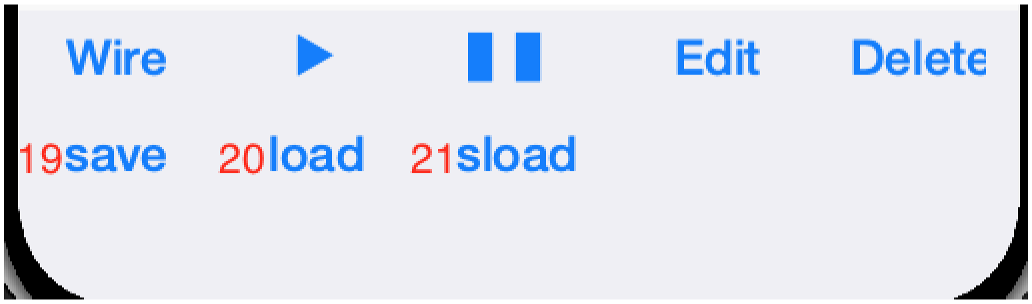


2.2 Feature Description



1. **[Circuit]** - Circuit Board view, which is what is currently displayed in image above. This is where the logic gates and visualization will be.
2. **[P_Delay]** - Propagation Delay view to edit the propagation delay and view critical path
3. **[T_Table]** - Truth table view, which displays the truth table for the currently circuit on the Circuit Board view

4. **[Info] -** To view the overall propagation delay and maximum operating frequency for current circuit
5. **[Wire]** - Adds wire between gates and input/output pins (which are buttons 9 - 17)
6. **[▶]** - Play button to simulate what's currently on the circuit board
7. **[▌▌]** - Pause button to pause the running simulation
8. **[Edit]** - To enter edit mode and modify the circuit
9. **[Delete]** - To delete a gate or input/output pin on the circuit board
10. **[Input Pin]**
11. **[Output Pin]**
12. **[And Gate]**
13. **[Or Gate]**
14. **[Xor Gate]**
15. **[Nand Gate]**
16. **[Nor Gate]**
17. **[Xnor Gate]**
18. **[Not Gate]**



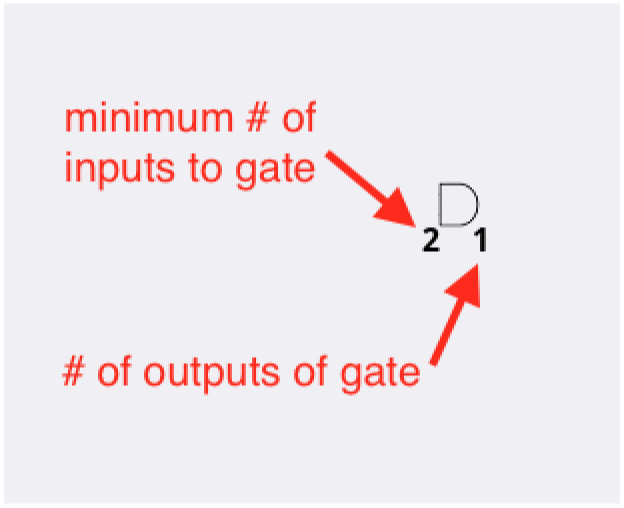
19. **[save]** - saving current circuit into a subcircuit
20. **[load]** - load a saved circuit (all the saved components and wires) to the current circuit board
21. **[sload]** - save circuit on the circuit board as a subcircuit

2.3 Numbers on Gates/Subcircuits

Gates or subcircuits on the circuit board display two numbers on the bottom of it.

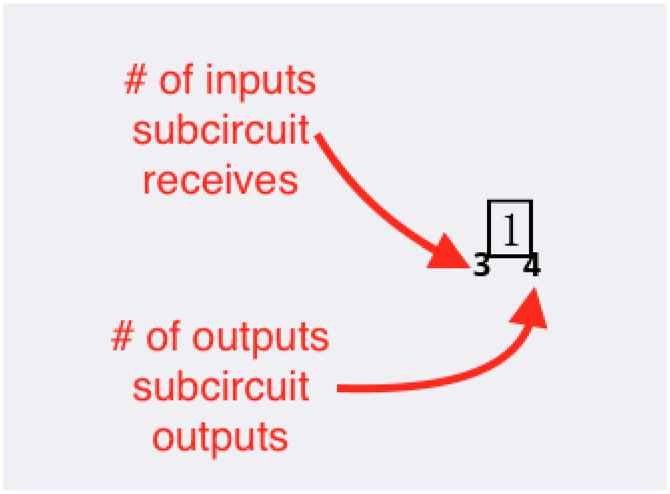
<u>Gates</u>
For gates, the bottom left number represents the minimum number of inputs the gate needs to work. In all common logic gates (AND, OR, XOR, etc.) except for NOT gate, they have the same minimum number of inputs to the gate and number of outputs. When greater than 2 inputs are attached to the common logic gates (except for the NOT gate), the bottom left number increases to the respective count of the inputs to that gate.

For the NOT gate, the bottom left number represents both the minimum and maximum number of inputs the NOT gate receives. In order words, it can only take in one input, no more no less, for it to function properly, and it outputs one output.



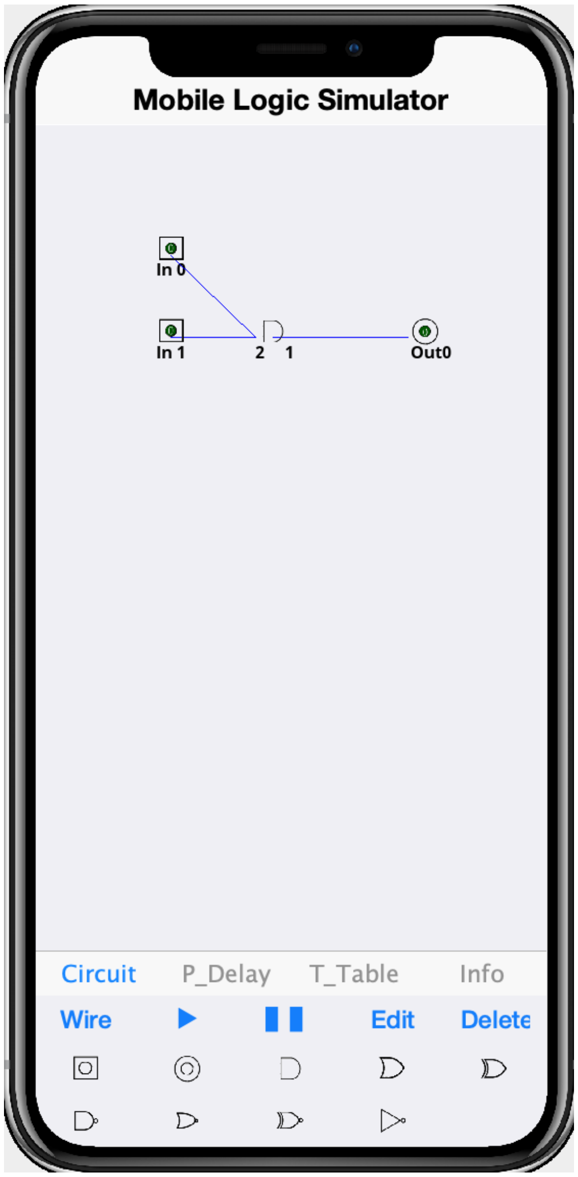
Subcircuits

For subcircuits, the bottom left number represents the exact number of inputs the subcircuit receives, and the bottom right number represents the number of outputs the subcircuit outputs.


# of inputs subcircuit receives
1
3
4
# of outputs subcircuit outputs

The bottom left number is the number of inputs that the gate/subcircuit receives. The bottom right number is the number of outputs that the gate/subcircuit outputs.

2.4 Building a Circuit



Placing Gates onto the Circuit Board

1. Go into the circuit board view by pressing [Circuit]
2. Press on the desired gates or input/output pins (buttons 9 - 17) and it will be automatically added onto the circuit board
3. Click and drag the gates or input/output pins in the circuit board to move them around and reposition them

Adding Wires Between the Gates or Input/Output Pins

1. Press [Wire]
2. Click on the first gate/input/output pin that you would like to have a wire drawn **FROM**
3. Click on the second gate/input/output pin that you would like to have the wire connect **TO**

   a. A wire will now be automatically made from the first gate/input/output pin to the second gate/input/output pin.
   b. If you would like to reposition the wires, press [Edit] to go into edit mode then drag the gate/input/output pins around, and the wires will automatically shrink/grow/position itself.

Deleting Wires Between the Gates or Input/Output Pins

1. Press [Wire]
2. Click on the first gate/input/output pin that has an *existing wire* attached to it that you wish to delete
3. Click on the second gate/input/output pin that the other end of the wire the attached to
   a. The wire will now be deleted

Note: The steps for adding/deleting a wire are the same. The difference is that if a wire exists between the two gate/input/output pins, it will be deleted. If no wire exists, then a wire will be connected between the two..

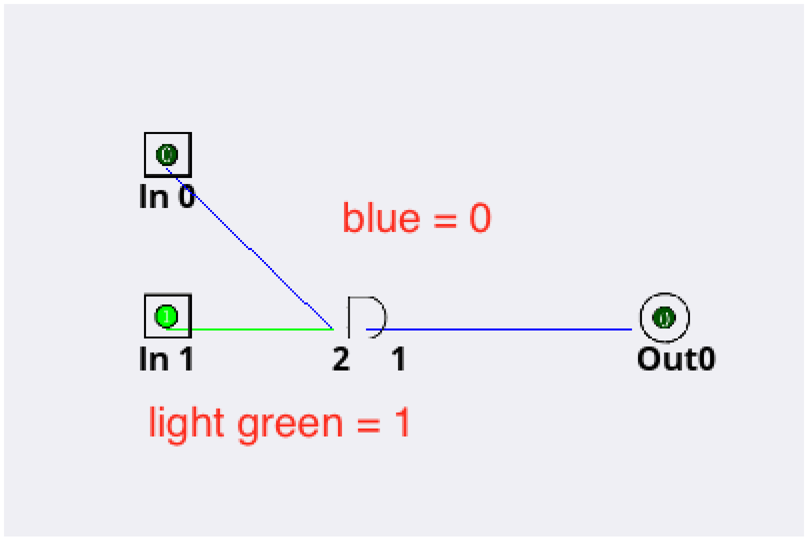
Simulating the Circuit

1. Press [▶] to simulate/run the circuit
   a. Based on the input(s) to the circuit, the output will be displayed in the output pin(s)

Toggling the Inputs

By default, the input pins will have value 0. To toggle it to between 0 or 1, do the following:

1. Make sure we are running the circuit by first pressing [▶]
2. Click on the input pins and it will toggle them between 0 and 1
   a. Note: Blue color is 0, Light green color is 1
   b. After toggling the inputs, the output(s) will automatically update accordingly

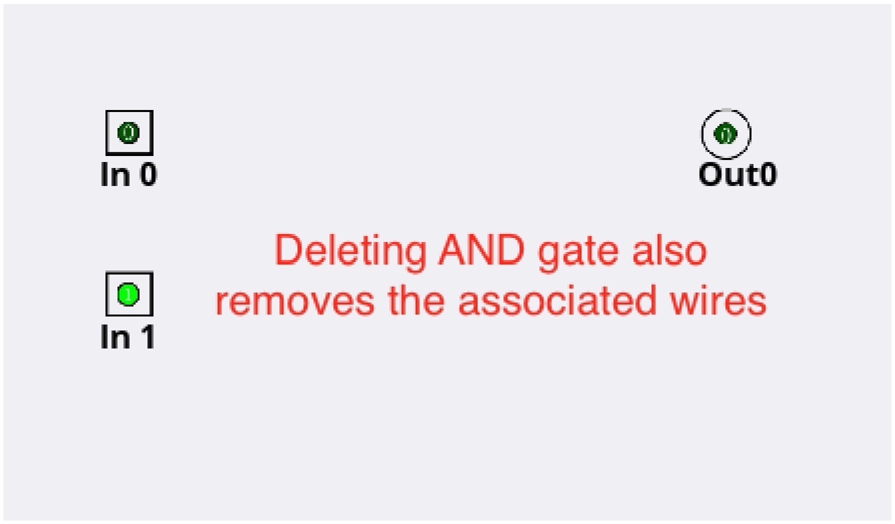


Pausing the Simulation

1. Press [▌▌]

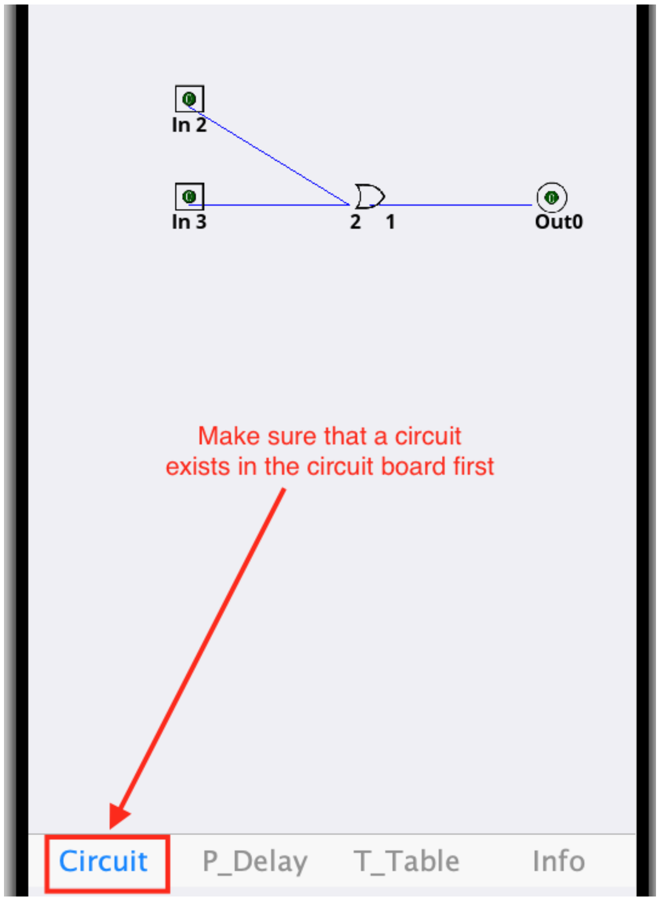
Deleting Gates/Input/Output Pins

1. Press [Delete]
2. Click on the gate/input/output pin that you want to delete
    a. The associated wires connected to that gate/input/output pin, if any, will automatically be removed

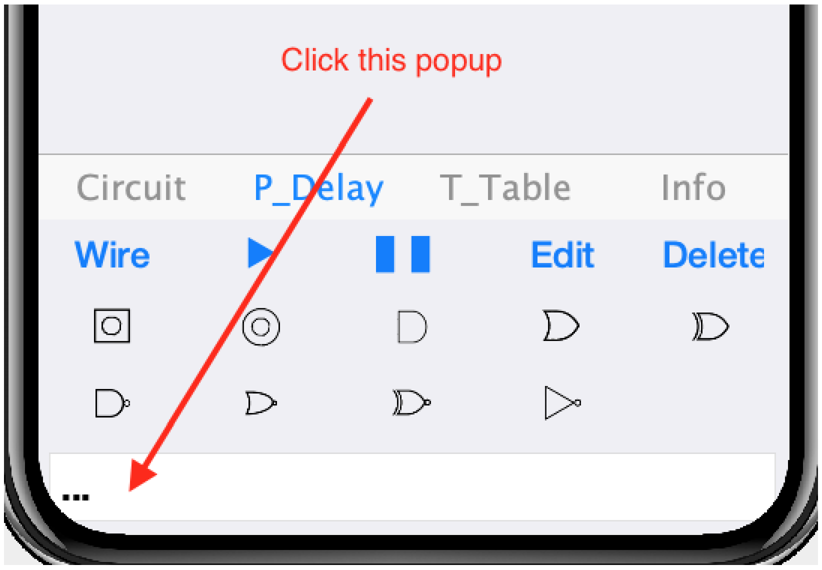


2.5 Modifying Propagation Delay

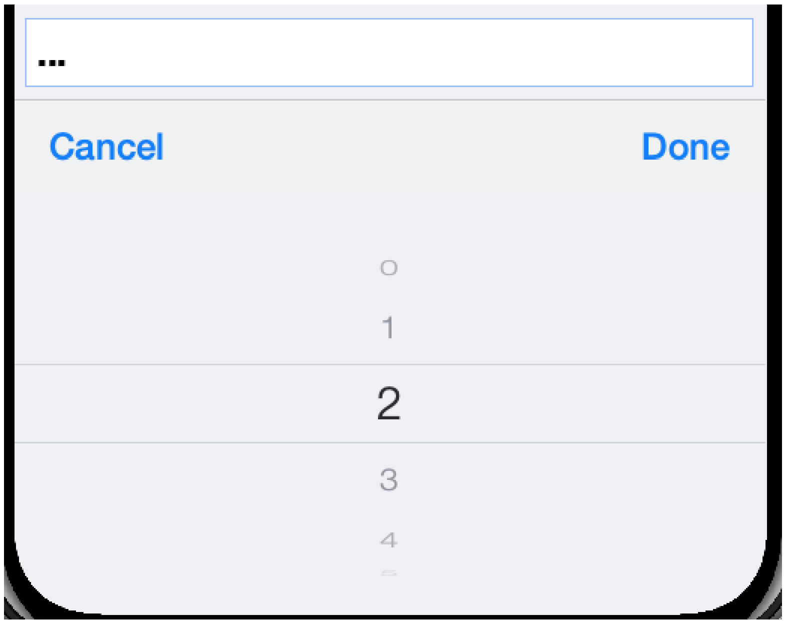
To modify the propagation delay on a circuit, first have a circuit in the circuit board, like shown below.



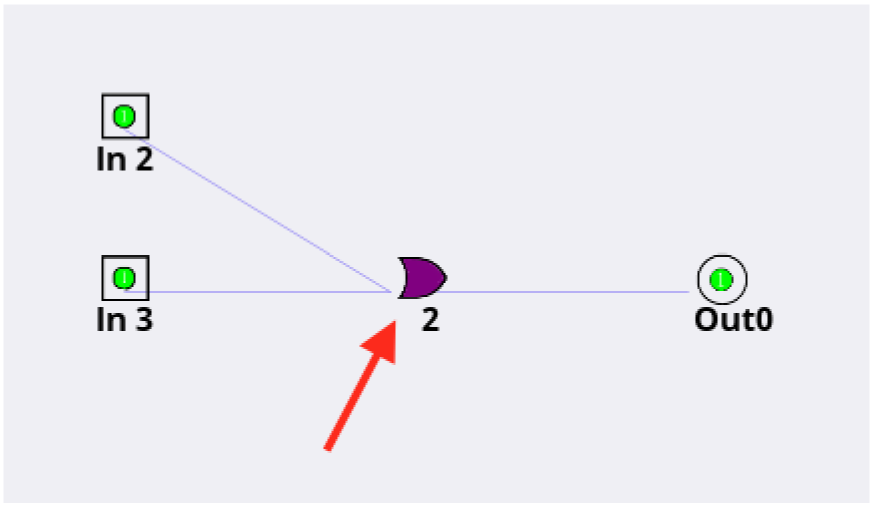
1. Now, navigate to the P_Delay tab. An exact copy of the circuit from the “Circuit” view should be displayed
2. Click on the gate that you wish to change the propagation delay for (units are in nanoseconds and default value for all gates is 1 nanosecond)
3. A popup is displayed at the bottom of the screen. Click it

a. 

4. Now, using the selector, you may change the propagation delay of the gate you have chosen. When you are done, press [Done] on the top right of the selector popup.

   a. 

Propagation for the gate is now updated to the amount of nanoseconds you have selected.

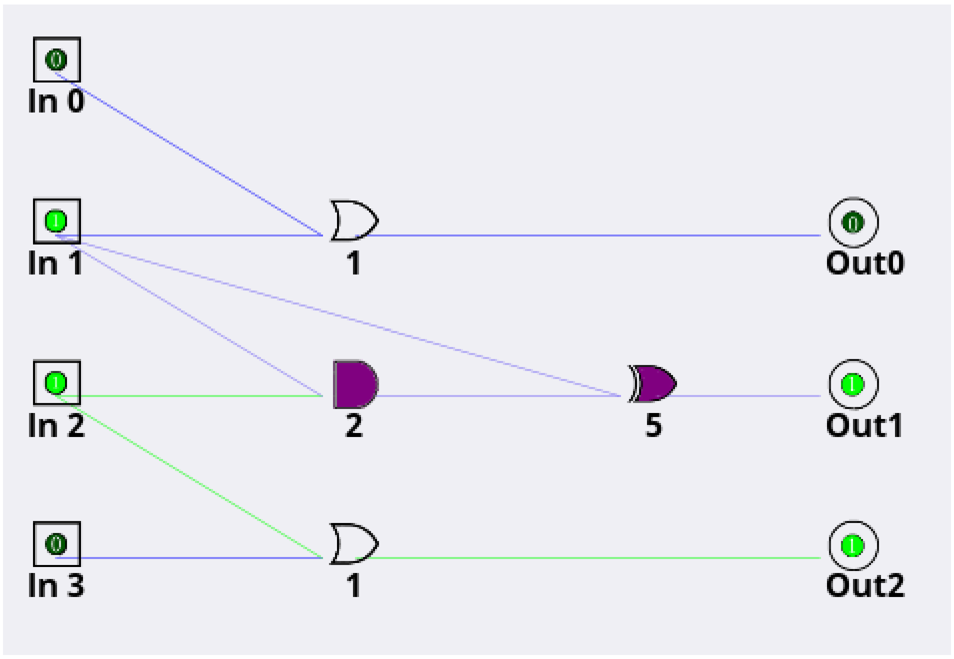


If you wish to modify the propagation delay for multiple kinds of the same gate, you need to do so individually for each gate. This is because sometimes, in certain cases, a 2-input gate might have a different propagation delay than a 5-input gate for example, and this approach allows for that.

2.6 Viewing Critical Path

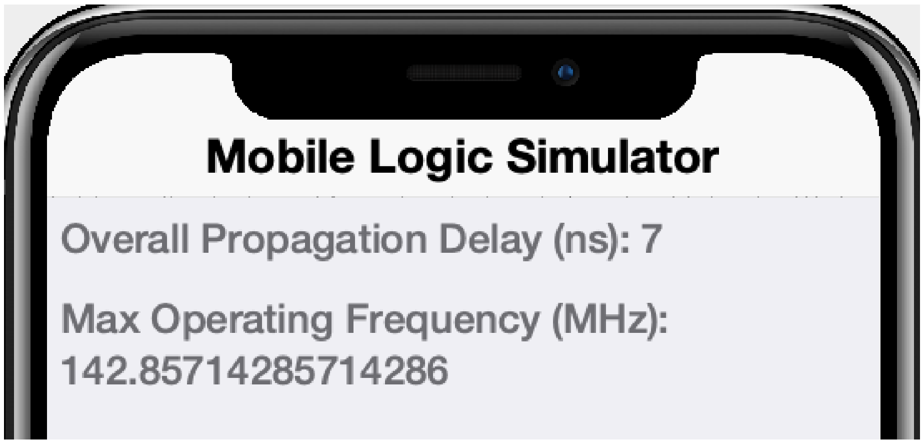
1. Make sure a circuit exists in the circuit board, [Circuit] tab
2. Navigate to the [P_Delay] tab

3. The critical path is the path highlighted in purple



2.7 Viewing Overall Propagation Delay and Maximum Operating Frequency

1. Make sure a circuit exists in the circuit board, [Circuit] tab
2. Navigate to the [Info] tab

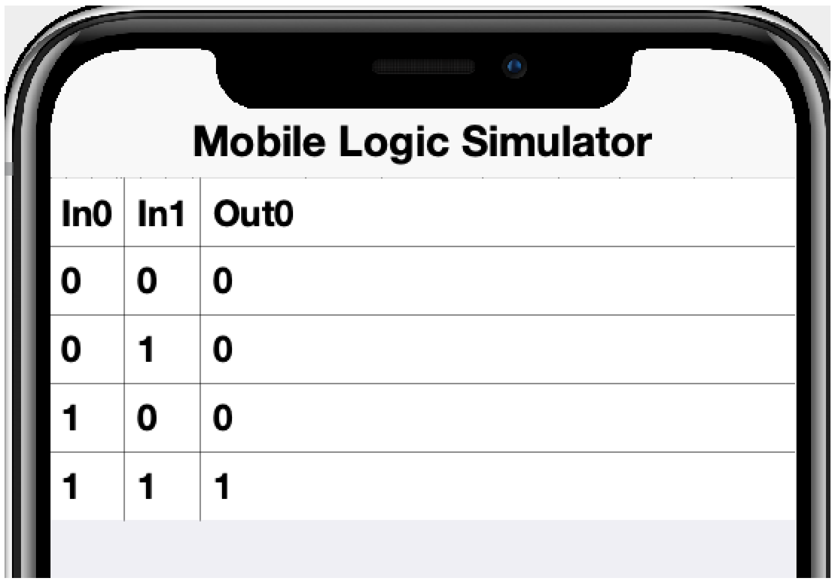

Mobile Logic Simulator

Overall Propagation Delay (ns): 7

Max Operating Frequency (MHz): 142.85714285714286

2.8 Truth Table

To view the truth table of a circuit, first make sure that there is a circuit on the circuit board view, and then press [T_Table] to view the truth table for that circuit board. The truth table is updated automatically every time the user presses [▸].

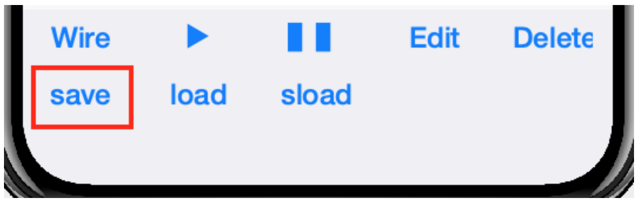


Mobile Logic Simulator

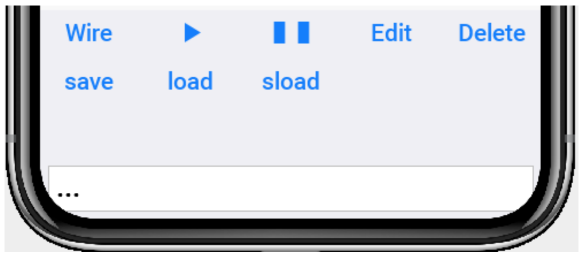
In0	In1	Out0
0	0	0
0	1	0
1	0	0
1	1	1

2.9 Subcircuit Saving and Loading

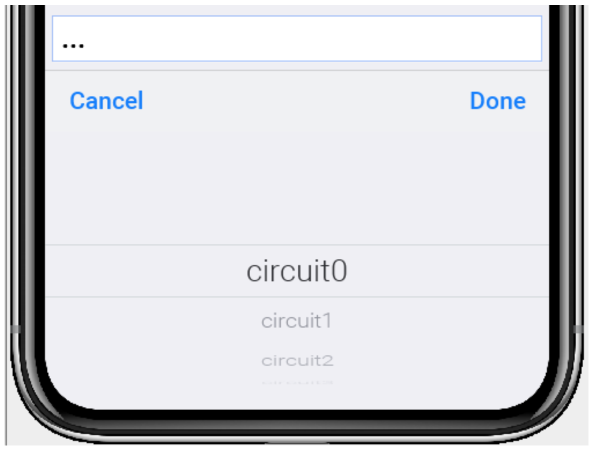
2.9.1 Saving



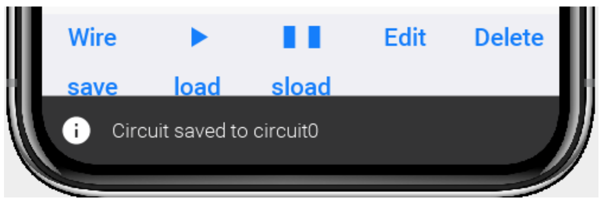
The save button is located in the second tab of the operation field.



A picker will pop up when the user clicks the save button.



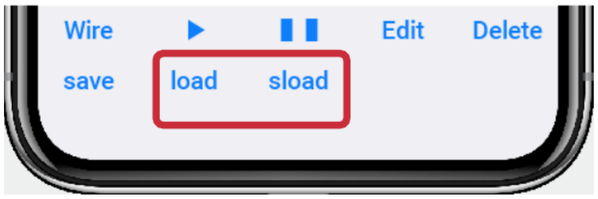
Users can then choose a circuit register to save the current circuit on the circuit board.



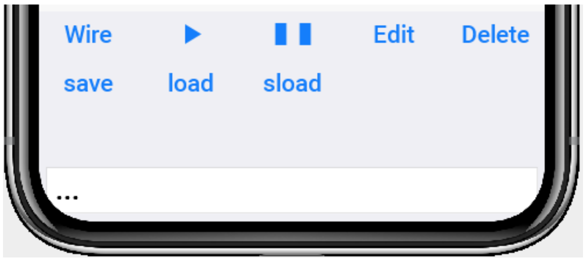
Circuit saved to circuit0

After choosing the circuit register to save and clicking done, a notification bar will also pop up to tell the user which register the current circuit is saved.

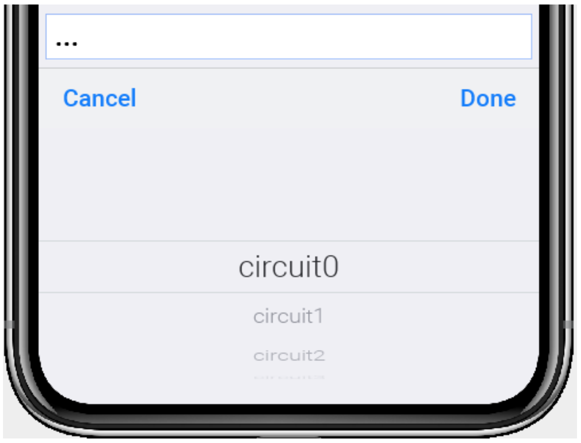
2.9.2 Loading



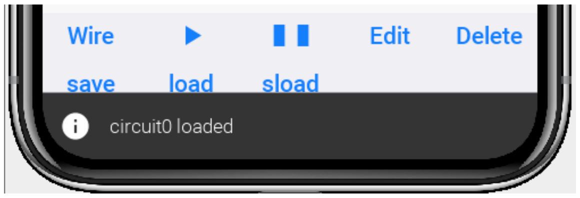
The load button is located in the second tab of the operation field, there are two choices for loading a circuit— [load] will load all the saved components and wires to the current circuit board, and [sload] will load the saved circuit as a subcircuit.



A picker will pop up when the user clicks the load button.



Users can then choose a circuit register to load circuit from to the circuit board.



circuit0 loaded

After choosing the circuit register to load from and clicking done, a notification bar will also pop up to tell the user which register the current circuit is loaded from.