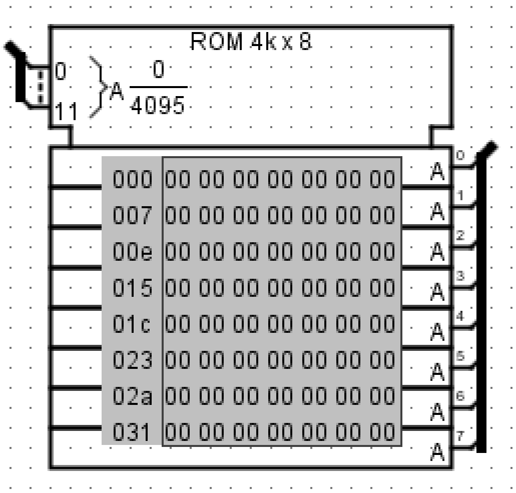
11 ROM

11.1 PURPOSE

This lab introduces students to Read Only Memory (ROM) and builds a fun application: The *Magic 8-Ball*. This was a toy that was developed in the 1950s and was popular throughout the 1960s. It was a small plastic sphere with the markings of an 8-ball. If the user "asked it a question" and then turned the toy upside down the answer would magically appear in a small window on the bottom of the ball.

11.2 PROCEDURE

Start a new *Logisim-Evolution* project and create a subcircuit named `Magic_8_Ball`. Open that circuit and place a ROM (*Memory* library) device near the center of the drawing canvas. Set the ROM properties for an *Address Bit Width* of 12 and a *Data Bit Width* of 8[1].



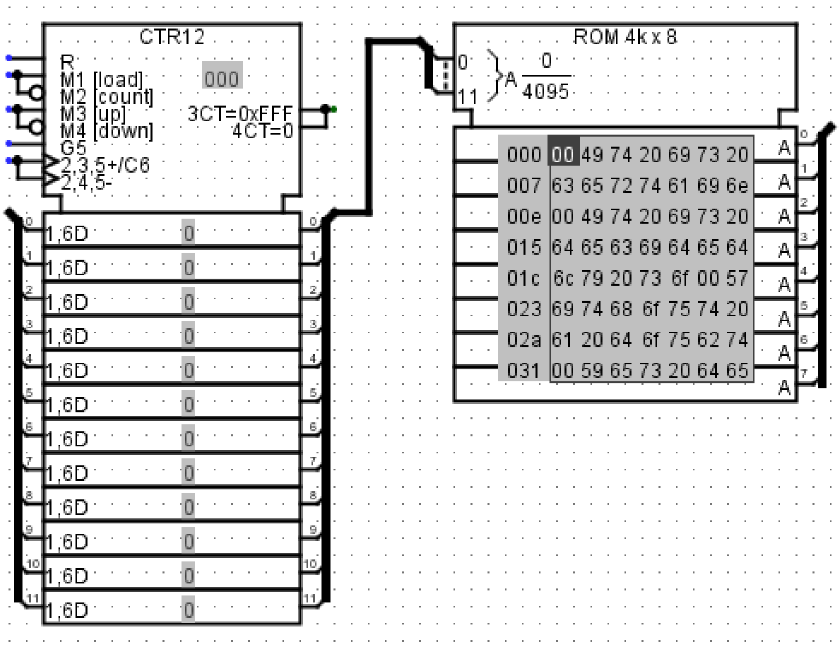
```
000 00 00 00 00 00 00 00
007 00 00 00 00 00 00 00
00e 00 00 00 00 00 00 00
015 00 00 00 00 00 00 00
01c 00 00 00 00 00 00 00
023 00 00 00 00 00 00 00
02a 00 00 00 00 00 00 00
031 00 00 00 00 00 00 00
```

Figure 11.1: Placing ROM

A ROM stores data that is accessed by setting an address on the inputs at the top left of the device and then reading the contents of that address on the 8-bit bus on the right side of the device. By attaching a counter to the ROM address port several consecutive addresses can be "stepped through" to output a message. Attach a Counter (*Mem-*

[1] The provided starter circuit already contains the `Magic_8_Ball` subcircuit along with two devices needed in the early part of the build.

ory library) with 12 Data Bits to the address port of the ROM, as in Figure 11.2.



```
000 00 49 74 20 69 73 20
007 63 65 72 74 61 69 6e
00e 00 49 74 20 69 73 20
015 64 65 63 69 64 65 64
01c 6c 79 20 73 6f 00 57
023 69 74 68 6f 75 74 20
02a 61 20 64 6f 75 62 74
031 00 59 65 73 20 64 65
```

Figure 11.2: ROM With Counter

According to Wikipedia[2], the Magic 8-Ball featured 20 sayings:

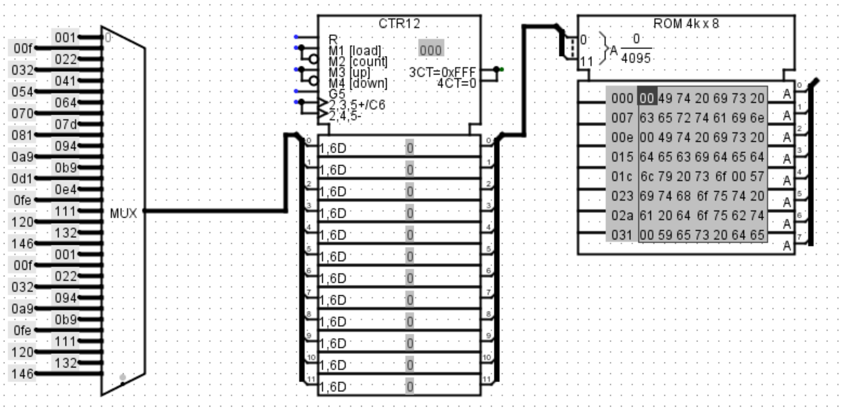
```
001 It is certain
00f It is decidedly so
022 Without a doubt
032 Yes definitely
041 You may rely on it
054 As I see it yes
064 Most likely
070 Outlook good
07d Yes
081 Signs point to yes
094 Reply hazy try again
0a9 Ask again later
0b8 Better not tell you now
0d1 Cannot predict now
0e4 Concentrate and ask again
0fe Do not count on it
111 My reply is no
120 My sources say no
132 Outlook not so good
146 Very doubtful
```

[2] https://en.wikipedia.org/wiki/Magic_8-Ball

The *Magic 8-Ball* simulator built in this lab uses those same 20 saying. In the above chart, each saying is numbered and the start point in ROM (using hexadecimal notation) for each saying is also noted. Thus, saying one starts on ROM byte 001, saying two starts on ROM byte 00f, saying three starts on ROM byte 022, and so forth.

The content of the ROM device must be loaded before it can be used and that content is provided in `Lab09_ROM.txt` accompanying this lab. To load the ROM device, click it one time and then click the "(click to edit)" link in its properties panel. In the ROM editor window that pops up, click the "open" button and navigate to the ROM memory file. Click "close window" to load the ROM device and make it ready for service[3].

The start point for each saying, as indicated on the above table, is stored in a Constant (*Wiring* library) then a Mux (*Plexers* library) with five select bits is used to transmit a message start location to the counter so it can be read from the ROM device. Figure 11.3 illustrates the circuit at this point[4].



```
000 00 49 74 20 69 73 20
007 63 65 72 74 61 69 6e
00e 00 49 74 20 69 73 20
015 64 65 63 69 64 65 64
01c 6c 79 20 73 6f 00 57
023 69 74 68 6f 75 74 20
02a 61 20 64 6f 75 62 74
031 00 59 65 73 20 64 65
```

Figure 11.3: ROM Filter Mux

A five-bit Random Generator (*Memory* library) is used to select a random message. Figure 11.4 illustrates the placement of the random generator.

[3] The ROM device provided with the starter circuit is pre-loaded so it will not be necessary to load it again. However, this information is left here for students who may want to load the ROM for practice.

[4] The multiplexer provided with the starter circuit already has the various constants attached. Students who wish to do so can create their own multiplexer by using the start addresses in the "Sayings" listing above.

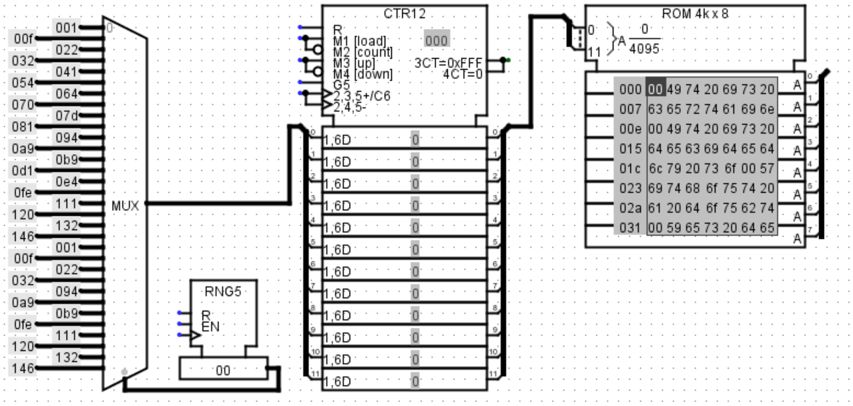


Figure 11.4: Random Generator Added

To complete the circuit, a few odds-and-ends were added. Figure 11.5 shows the completed circuit, but details from that figure are used below to describe how to complete the circuit.

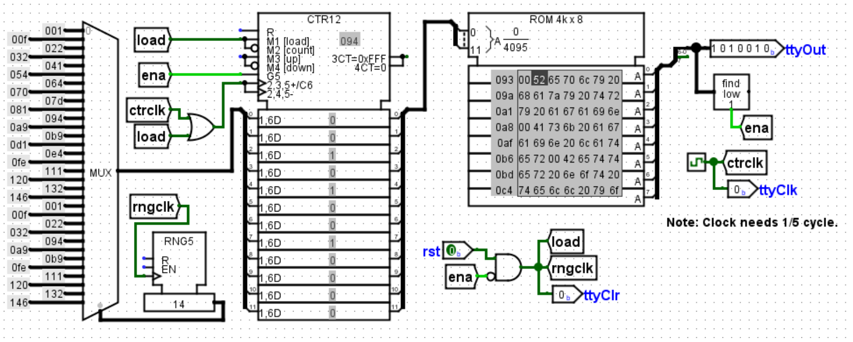


Figure 11.5: Completed Magic 8-Ball Circuit

To set up the counter, four signals are needed. These are all from tunnels (*Wiring* library) connected to other spots on the circuit. (See Figure 11.6.)

- **load**. The load signal goes high when the counter should be loaded with a new number from the multiplexer. The number loaded is the location in ROM for the start of a message. Notice that the load signal is used on two pins. The top pin places the counter in load mode while the bottom pin uses the load signal as a clock pulse.
- **ena**. The enable signal turns the counter on/off. When enable is high then the counter functions normally and when it is low the counter is disabled.
- **ctrclk**.The counter clock provides the clock signal for the counter.

Connect the random number generator as follows. (See Figure 11.6.)

- The clock input pin is connected to a “rngclk” tunnel.

- The generator output is wired to the select port of the multiplexer.

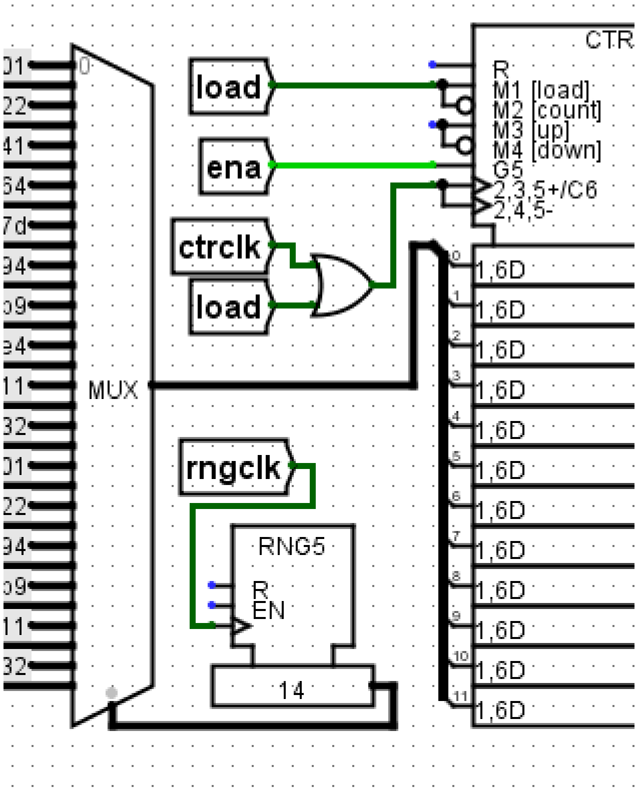


Figure 11.6: Counter Inputs

The counter control signals are generated and distributed from a small group located under the ROM device. The purpose of this tiny group is to transmit a high signal through the AND gate when the reset pin goes high while enable is low. This generates the signals needed to select a new random message and put the starting address of that message in the counter. (See Figure 11.7.)

- **rst**. The reset pin is an external signal that originates from the `main` circuit.

- **ena′**. Enable Not originates from the ROM output group.

- **load**. This signal is used to load a message starting address into the counter. When it goes high it activates the “load” function and also becomes a single clock pulse for the counter.

- **rngclk**. The random number generator clock signal activates that device so it generates a random number. That number is then used to select a single line from the multiplexer so a message starting address can be loaded.

- **ttyClr**. This sends a high signal to the TTY "clear" pin on the main circuit. That signal is used to clear the TTY device.

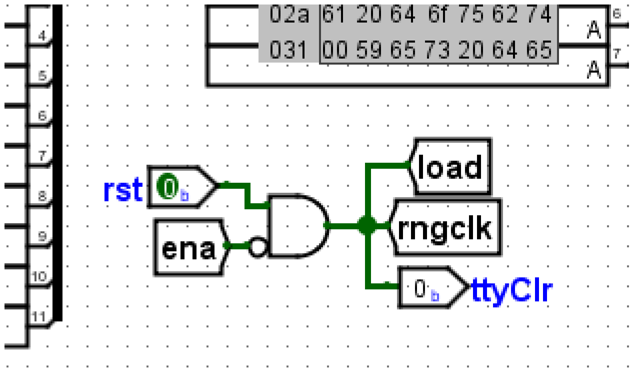


Figure 11.7: Counter Control Generation and Distribution

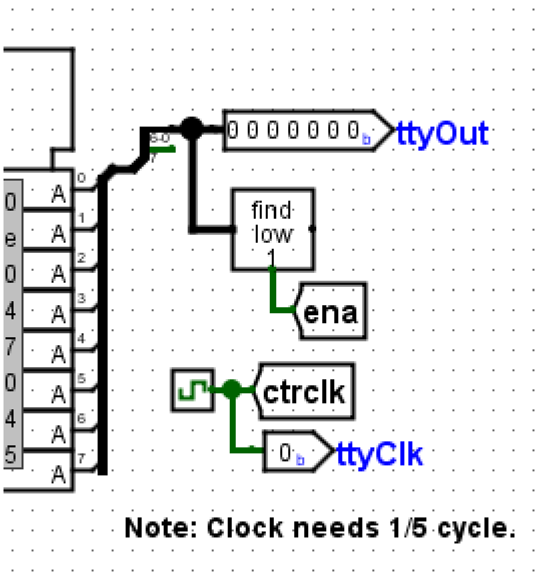
There are two functions found at the ROM device output. (See Figure 11.8.)

- The output of the ROM device is connected to the *ttyOut* pin in order to drive the teletype device on the main circuit.[5]

- The Bit Finder (*Arithmetic* library) attached to the output of the ROM device is used to find the lowest-order *one* in the ROM byte output. If the ROM byte includes at least one *one* then the south port of the finder is high. If the ROM output is all zeros then the Finder output it goes low and that is used as the *ena* signal for the counter and random number generator. When the enable signal is low it also permits a *rst* signal (generated on the main circuit when the user "asks another question") to create a new answer.

- Near the output of the ROM device a clock signal is split to two outputs. One is the *ctrclk* tunnel that is used by the counter and the other is the *ttyClk* pin, which is used on the main circuit to clock the teletype device. **It is important to note that the clock properties are set for a 1 tick high duration and 5 ticks low duration (a 1/5 clock).**

[5] Note that at the output of the ROM device is a splitter. ASCII letters are only seven bits wide so this splitter passes bits 0-6 to the *ttyOut* port but bit 7 (the most significant bit) is simply discarded. The provided starter circuit includes the splitter.



Note: Clock needs 1/5 cycle.

Figure 11.8: ROM Output

The only remaining step is to create the `main` circuit. As in all labs in this manual, the `main` circuit does nothing more than provide a user interface for the *Magic 8-Ball* Circuit. Figure 11.9 illustrates the `main` circuit.

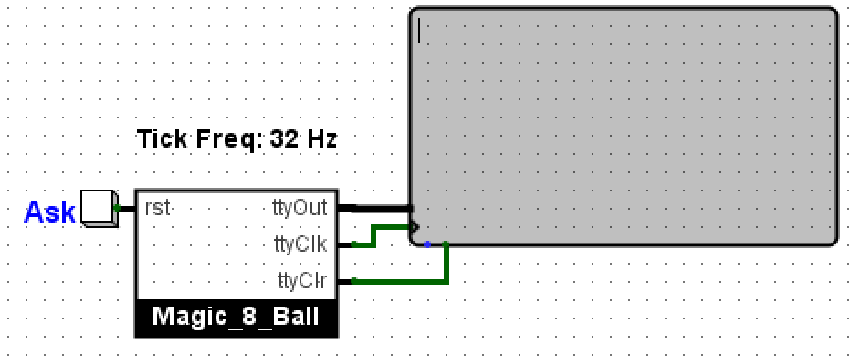


Figure 11.9: Magic 8-Ball Main Circuit

11.2.1 *Testing the Circuit*

Before the circuit can be tested the ROM device must be loaded. The ROM was loaded earlier in the lab but in case it does not have any content (it is filled with zeros), then load it with *`Lab09_ROM.txt`*, which was provided with the lab. To load the ROM device, click it one time and then click the "(click to edit)" link in its properties panel. In the ROM editor window that pops up, click the "open" button and find the ROM memory file. Click "close window" to load the ROM device and make it ready for service.

The circuit should be tested by enabling the simulator clock at a frequency of 32 Hertz. Every time the *Ask* button is pressed a new random message will be displayed on the teletype screen[6].

11.3 DELIVERABLE

To receive a grade for this lab, build this circuit. Be sure the standard identifying information is at the top left of the `main` circuit, similar to:

```
George Self
Lab 11: ROM
September 13, 2019
```

Save the file with this name: *Lab11_ROM* and submit that file for grading.

[6] Due to the way this circuit is constructed one out of six button presses will fail and no message will be displayed. The failures are random events so the circuit may fail several times in a row but then not fail for the next 20 or more presses. Students may want to investigate this bug but that is not required.