# 12 VENDING MACHINE

## 12.1 PURPOSE

One of the important benefits of working with *Logisim-Evolution* is being able to simulate real-world circuits before they are physically built. This lab simulates a vending machine that meets these requirements:

1. The customer can input the following coins: 5-cent, 10-cent, 25-cent.
2. When 75 cents is input, the machine will activate the dispenser and permit the customer to select a product.
3. When at least 75 cents is input no more coins will be accepted.
4. Change will be returned to the customer if more than 75 cents is deposited.
5. A reset button will return the customer's money.
6. When a product is dispensed, 75 cents will be added to the machine's "Total Money Collected" register.
7. No product is dispensed if less than 75 cents is deposited.
8. The current number of items available for each product is stored in a counter.
9. When a service technician restocks the machine the item count for each product is set to 15, which is the maximum number of items that can be stocked.
10. If the number of products available is zero for any one product the machine will light a "sold out" light and no action will be taken if that product is selected.

This circuit uses only combinational logic and is an example of a reasonably complex system.

## 12.2 PROCEDURE

The starter circuit for this lab is almost complete, but three of the requirements have not been met.

- Requirement three is that the coin input will stop once 75 cents is reached but this is not working so customers can continue depositing coins into the machine.
- When a product is dispensed, the coins deposited and change returned is not reset back to zero. This means that a customer could deposit 75 cents and then keep selecting products until the machine is empty.
- Requirement six is that the machine totals all of the money collected but that is not functional.

### 12.2.1 *Testing the Circuit*

To test the circuit:

1. Ensure simulation is enabled at SIMULATE -> SIMULATION ENABLED.
2. Poke the *Ena* input pin to enable the vending machine simulator.
3. Notice that the *SoldOut1*, *SoldOut2*, and *SoldOut3* LEDs are lit, indicating that those products are sold out.
4. Restock products by poking the *Restock1* and *Restock2* buttons. For this test, do not poke *Restock3* to keep that product empty. As a product is restocked the "SoldOut" LED for that product goes out and the *Prod01* and *Prod02* counts change to 15.
5. Poke the *In5*, *In10*, and *In25* buttons to deposit coins. The total deposited is displayed and any amount over 75 cents is shown as change. Notice that the deposit circuit is not disabled after 75 cents is reached so customers can continue depositing coins.
6. Once at least 75 cents is deposited, poke *Vend1* to vend that product. When the button is poked the *Dispense1* LED momentarily lights to indicate that a product was sold. The number of items available for that product decreases. Notice that once a product is dispensed the amount of money deposited is not reset and the machine can dispense additional products without additional money being deposited.
7. Poke *Vend3* and notice that nothing happens since that product is sold out.
8. Poke *Reset* to reset the amount of money deposited.

### 12.2.2 *Subcircuit Descriptions*

This simulator contains five subcircuits in addition to the `main` circuit and this section describes all of those components.

#### 12.2.2.1 *main*

The `main` circuit is the interface between a human customer and the simulator, as shown in Figure 12.1.

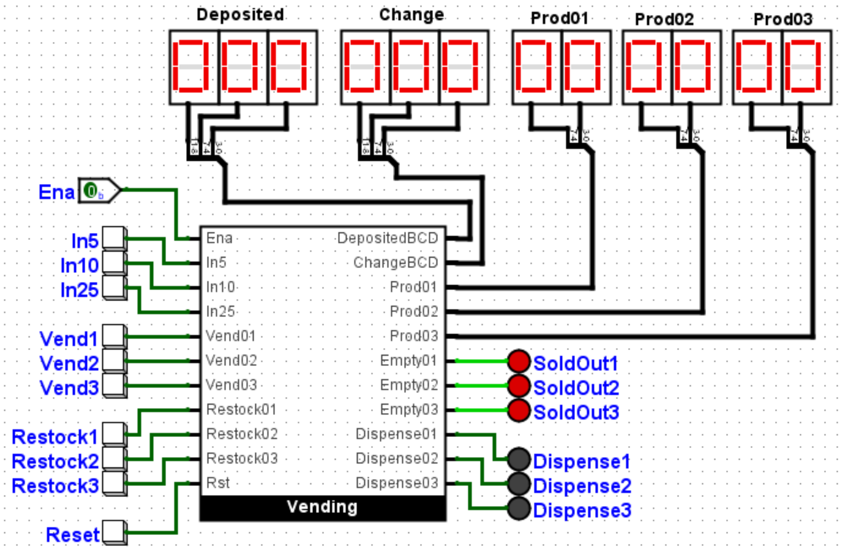

Figure 12.1: Vending Machine Main Circuit

The `main` circuit includes the following components.

- Numeric displays for the amount deposited, the change returned, and the number of items available for each of three products.
- An *Ena* (*Enable*) input so a technician can disable the machine for servicing.
- Buttons to simulate depositing coins, vending products, and restocking the machine.
- LEDs to indicate when products are sold out and dispensed.

#### 12.2.2.2 *Activator*

The `Activator` subcircuit receives a signal from the `Bank` subcircuit that indicates how much money has been collected. The `Activator` returns the BCD Total and Change values and sets a signal to activate the `Dispenser` subcircuit once 75 cents has been deposited. Figure 12.2 illustrates the `Activator` subcircuit.

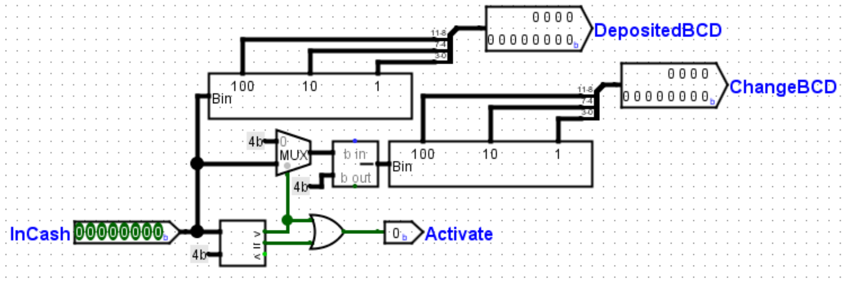

Figure 12.2: Activator Subcircuit

The `Activator` subcircuit has only one input, *InCash*. That input is connected to the `Bank` subcircuit output and contains the total amount of cash deposited. That input is connected to a Bin2BCD (*BFH mega functions* library) device and is then output as a BCD number on the *DepositedBCD* output pin.

The *InCash* input is also sent to a comparator where the amount is compared to 75. If the amount in the bank is equal to or greater than 75 then the `Activate` output goes high.

Finally, the *InCash* input is sent to a mux that outputs 75 until the comparator indicates that more than 75 is in the bank, then the mux passes the *InCash* amount to a subtractor where 75 is subtracted from it and the result sent to the *ChangeBCD* output.

### 12.2.2.3 *Bank*

The `Bank` subcircuit keeps a running total of the amount deposited and sends that total to the `Activator` subcircuit. Figure 12.3 illustrates the `Bank` subcircuit.

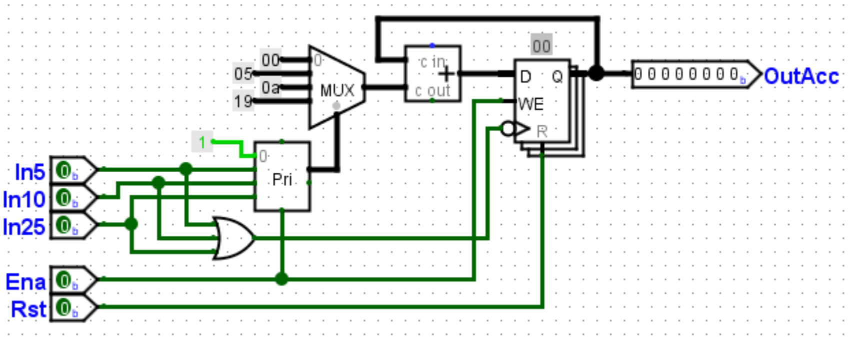

Figure 12.3: Bank Subcircuit

The `Bank` subcircuit has five inputs. *In5*, *In10*, and *In25* indicate the value of the coin dropped into the machine. When high, the *Ena* input enables the `Bank`. When high, the *Rst* input resets the total to zero.

The `Bank` subcircuit has only one output, *OutAcc*, that makes the total cash accumulated available to the `Activator` subcircuit.

For this description, imagine that a 5-cent coin is deposited. *In5* goes high which changes the output of the priority encoder from zero to one. That output is sent to a mux control where the number five, on mux input one, is passed to an adder. The output of the adder is sent to a register where it is remembered. The output of the register is sent to the *OutAcc* pin but is also looped back to the adder so each new coin is added to the previous total. Thus, the register keeps a running total of the money deposited.

The final logic function in this subcircuit is a three-input OR gate where each of the coin input pins are sent to the clock input of the register. As coins are dropped into the machine the register is clocked in order to capture each new deposit. It is important to note that *the register is set to activate on a falling edge* in order to give the input signal enough time to propagate through the priority encoder, mux, and adder.

#### 12.2.2.4 *Dispenser*

The Dispenser subcircuit dispenses the three products available in the machine. Figure 12.4 illustrates the Dispenser subcircuit.

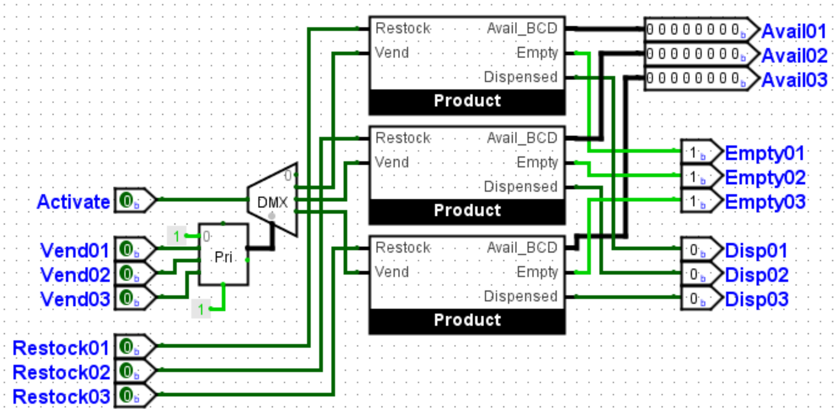

Figure 12.4: Dispenser Subcircuit

The Dispenser subcircuit has seven inputs and nine outputs.

Inputs:

- **Activate**. A high input on this pin permits a product to be dispensed. This signal is generated in the Activator subcircuit.
- **Vend**. These inputs cause one of three products to be dispensed.
- **Restock**. This resets the product count to 15, simulating a service technician restocking the machine.

Outputs:

- **Avail**. This is an 8-bit number (not BCD) that shows how many items each of the products have available for sale.
- **Empty**. This pin goes high when any product is sold out.

- **Disp**. This pin goes high when an item is dispensed.

Overall, this is a rather simple subcircuit. When one of the *Vend* inputs goes high the priority encoder sends the number for that input to the demux control port. Thus, if a customer selects product one then the priority encoder transmits a one to the demux.

The demux will transmit the value present on the *Activate* input to one of three `Product` subcircuits. When *Activate* is low then a zero is transmitted to the `Product` subcircuit which effectively disables the dispenser function. However, if *Activate* is high then a one is transmitted to one of the `Product` subcircuits and that will cause a product to be dispensed.

### 12.2.2.5 *Product*

The `Product` subcircuit keeps count of the number of items available for a product. There are two inputs and three outputs.

Inputs:

- **Restock**. This resets the count of the item to 15. It is designed to simulate a service technician restocking the machine.
- **Vend**. When this goes high a single item is dispensed.

Outputs:

- **AvailBCD**. This is a count, in BCD, of the number of items available for sale.
- **Empty**. This goes high when there are no items available for sale.
- **Dispensed**. This goes high when an item is dispensed. It represents an item physically dropping out of the machine for the customer to retrieve.

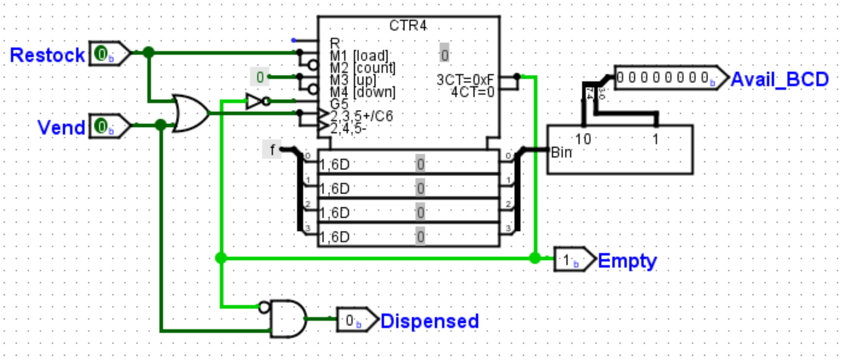

Figure 12.5: Product Subcircuit

This subcircuit is nothing more than a counter with a few controlling signals. The counter has a constant zero input on the *M3* port. That sets the counter to decrement the count on each clock pulse.

The *Restock* input is wired to the counter's reset port and a high input will reset the counter to 15. Note, the counter's properties are pre-set for a maximum count of 15.

The *Vend* input is wired to the counter's clock port so when an item is sold the count will decrease. This input is also wired to the *Dispensed* output to indicate that an item was sold.

The counter has two outputs. The *3CT=0xF* output goes high when the count reaches zero (the item is sold out). That signal is used to disable the counter so no further sales are made. The second counter output is the count it contains and that is wired to a Bin2BCD (*BFH mega functions* library) device. The output of that device is sent to the *AvailBCD* port for other subcircuits to use.

### 12.2.2.6 *Vending*

The `Vending` subcircuit consolidates the other subcircuits into an IC that is used in the `main` circuit. Figure 12.6 illustrates the `Vending` subcircuit.

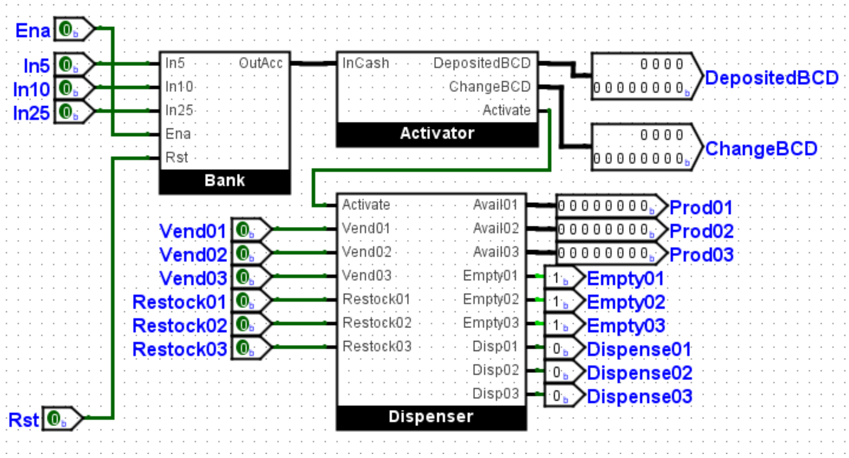

Figure 12.6: Vending Subcircuit

No further explanation is given for this subcircuit since it only wires the other subcircuits together and introduces no new logic.

## 12.3 challenge

The Vending Machine simulator has three vital flaws that must be corrected.

- Requirement three is that the coin input will stop once 75 cents is reached but this is not working so customers can continue depositing coins into the machine.

- When a product is dispensed, the coins deposited and change returned is not reset back to zero. This means that a customer could deposit 75 cents and then keep selecting products until the machine is empty.
- Requirement six is that the machine totals all of the money collected but that is not functional.

## 12.4 DELIVERABLE

To receive a grade for this lab, correct all three flaws identified in the Challenge. Be sure the standard identifying information is at the top left of the *main* circuit, similar to:

```
George Self
Lab 12: Vending Machine
February 16, 2018
```

Save the file with this name: *Lab12_Vend* and submit that file for grading.