EEE102-02 Lab 7 Report:
Finite State Machine

Kaan Ermertcan - 22202823

27.11.2023

Purpose:

In this lab, we will design our own finite state machine and implement it on a breadboard using logic gates and flip-flop ICs.

Design Specifications:

My design represents a vending machine that sells souvenirs at 2€ each. The machine only accepts 2€, 1€ and 50 cent coins as payment. My design has a two-bit input (“00” -> No coin is inserted, “01”-> 50 cent is inserted, “10” -> 1€ is inserted, “11” -> 2€ is inserted), and a one-bit output (“0”-> Product is not sold, “1”-> Product is sold). The vending machine keeps track of the balance. If total money put into the machine is greater than or equal to 2€, output is set as “1” for one clock cycle and if it exceeds 2€, the change is given back (Balance resets to 0). If enough coins are not inserted consecutively (one in each clock cycle), the inserted coins are given back (Balance resets to 0).

Methodology:

First, I decided that my design can be achieved with a 4-state mealy machine. I drew the state diagram of my FSM as in the figure below.

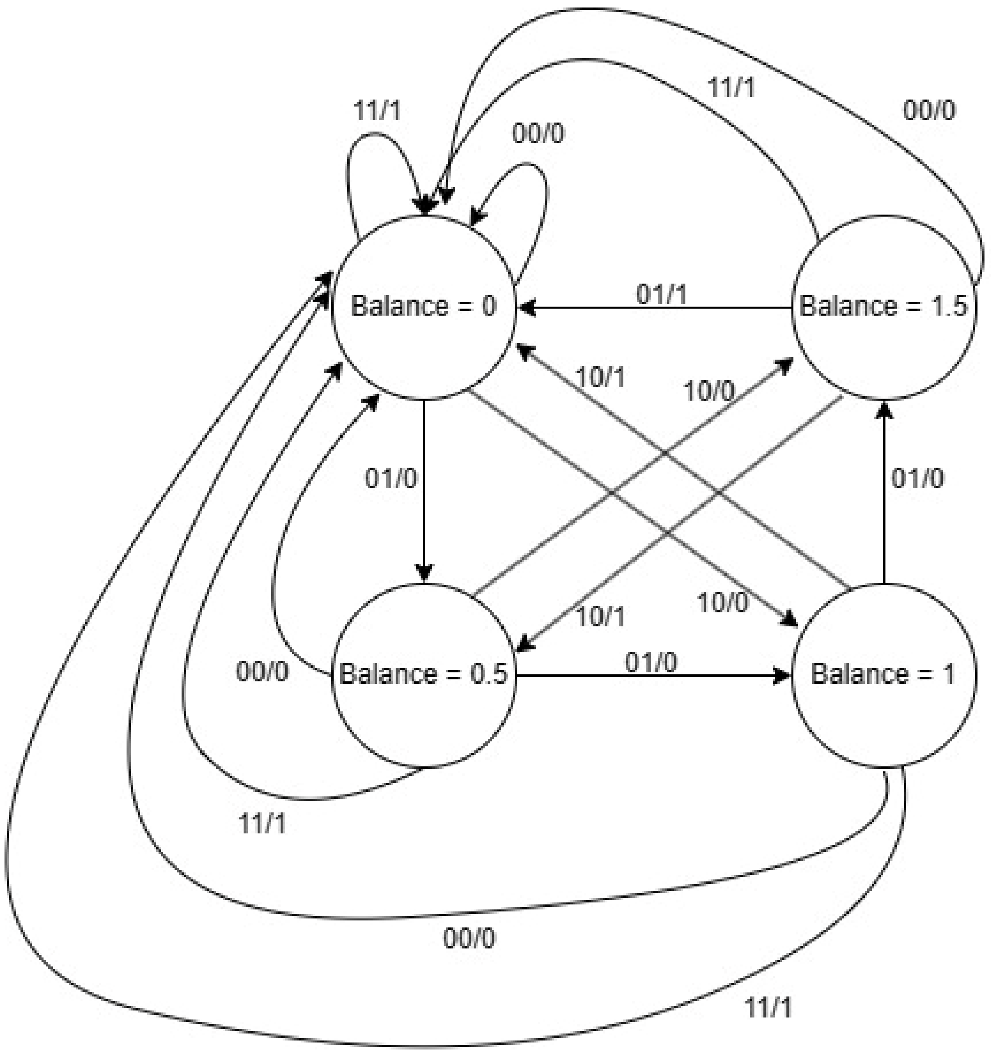

Figure 1.1: State diagram.

Then, I created the state table of my FSM from the state diagram.

Present State (S)		**Next State (S(t+1))**				**Output (F)**			
		I = 00	**I = 01**	**I = 10**	**I = 11**	**I = 00**	**I = 01**	**I = 10**	**I = 11**
Balance = 0	**00**	00	01	10	00	0	0	0	1
Balance = 0.5	**01**	00	10	11	00	0	0	0	1
Balance = 1	**10**	00	11	00	00	0	0	1	1
Balance = 1.5	**11**	00	00	01	00	0	1	1	1

Table 1.1: State table.

After that, we can determine the next state logic and output logic by deriving SOP equations using 3 K-maps. The SOP equations are below:

$$S(t+1)_1 = S_1'I_1I_0' + S_1'S_0I_1'I_0 + S_1S_0'I_1'I_0$$

$$S(t+1)_0 = S_0'I_1'I_0 + S_0I_1I_0'$$

$$\mathrm{F} = I_1I_0 + S_1I_1 + S_1S_0I_0$$

I decided to use NAND gates instead of AND and OR gates in the above equations to reduce the number of needed ICs (The output does not change if all gates are replaced with NAND, derived from bubble to bubble logic). The circuit diagram can be seen in the figure below.

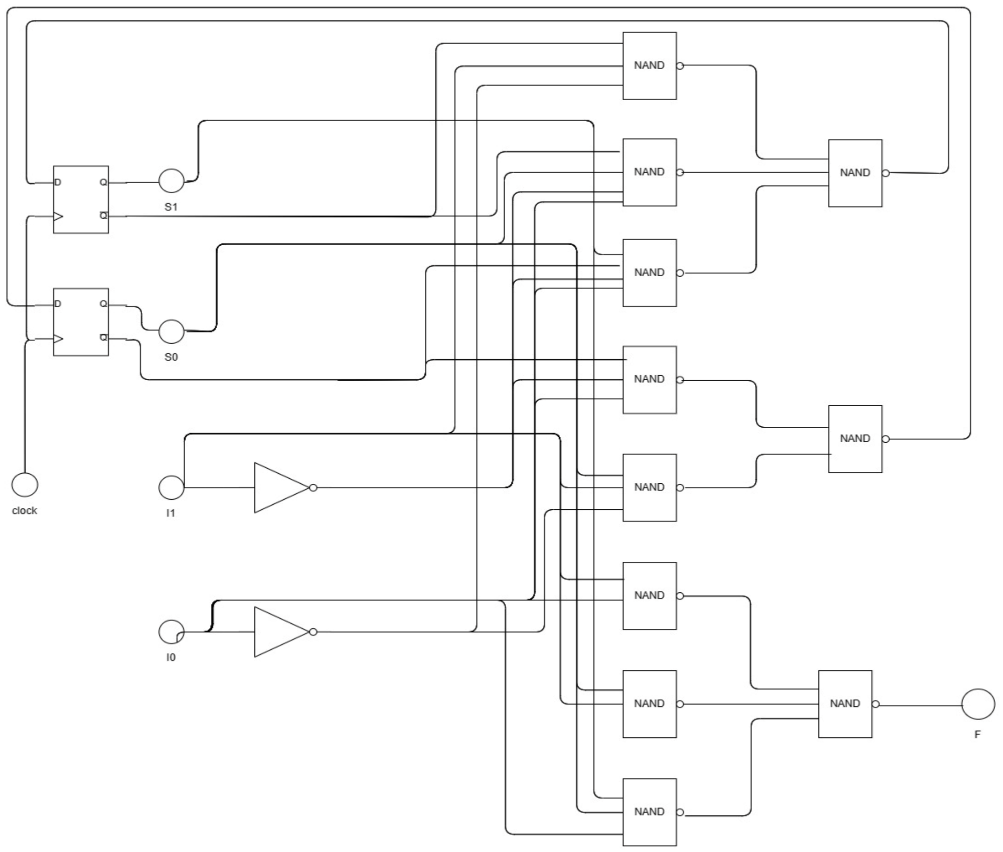

Figure 1.2: Circuit diagram.

The list of required logic ICs is below.

Nb	Description	Code
1	Hex Inverter	74HC04
1	Dual D-Flip flop	74HC74
1	Dual 4-input NAND	74HC20
1	Quad 2-input NAND	74HC00
2	Triple 3-input NAND	74HC10

Table 1.2: List of required ICs.

Two pushbuttons are connected to the inputs. Two red LEDs are connected to the flip flops' outputs to visualize the current state of the FSM, one red LED is connected to the clock, one green LED is connected to the output.

Results:

I built the circuit on a breadboard and connected the power supply to test it. When testing, I used set and reset connections of the flip-flops to set them to a particular state and checked if Q(t+1) and F is correct. If not, I tracked the voltages using a multimeter until I find the wrong connection. After correcting some wire connections, the testing is successful. I connected the set and reset connections to high and connected the clock pin to a signal generator set to 1 Hz. My circuit is completed.

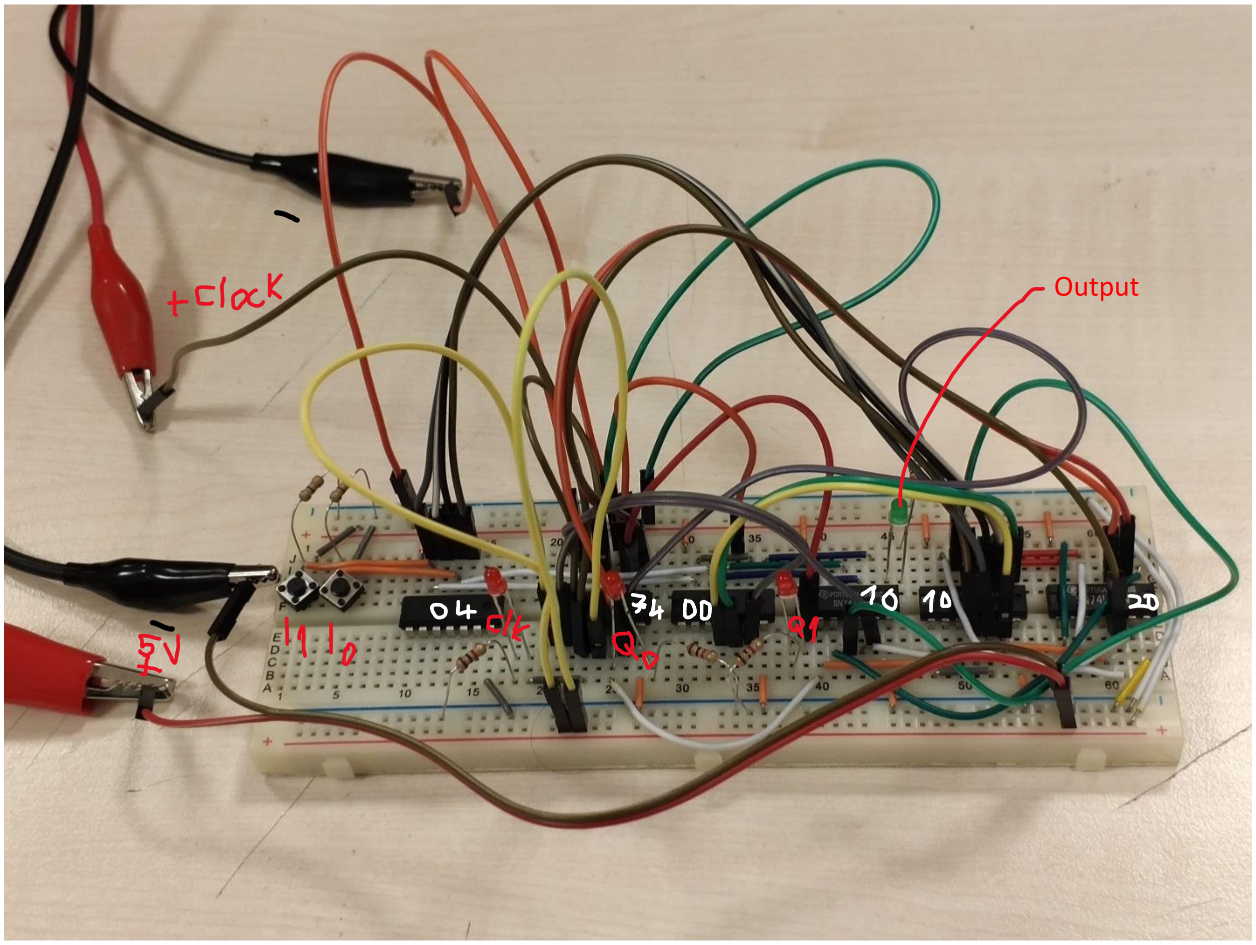

Figure 2.1: The completed circuit.

Now, some example photos of the working circuit can be found below.

Scenario 1: 4x50 cents is inserted:

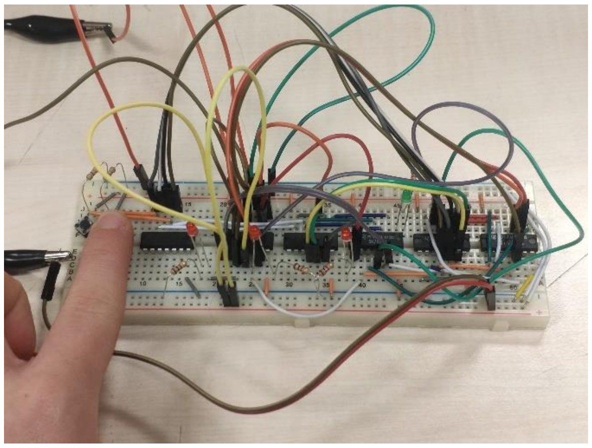
Figure 3.1: User inserts first 50 cent.

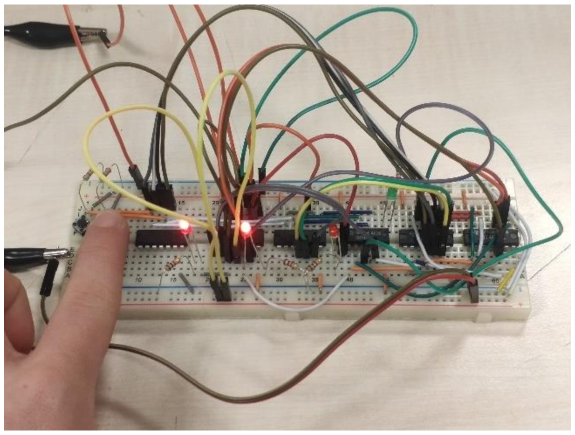
Figure 3.2: Balance = 0.5, User inserts second 50 cent.

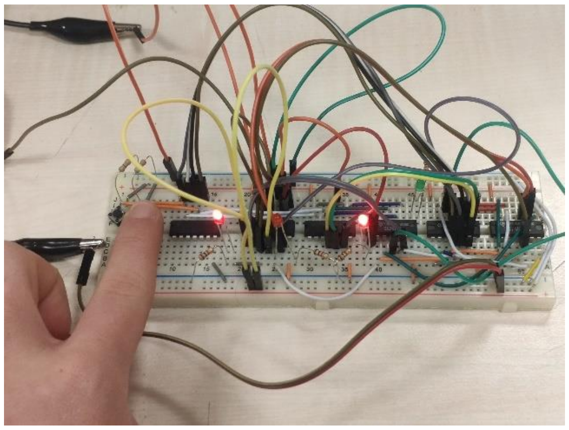
Figure 3.3: Balance = 1, User inserts third 50 cent.

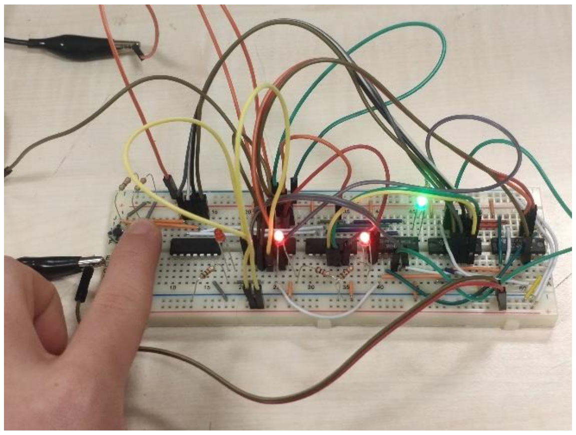
Figure 3.4: Balance = 1.5, User inserts fourth 50 cent.

Scenario 2: 2x1€ is inserted:

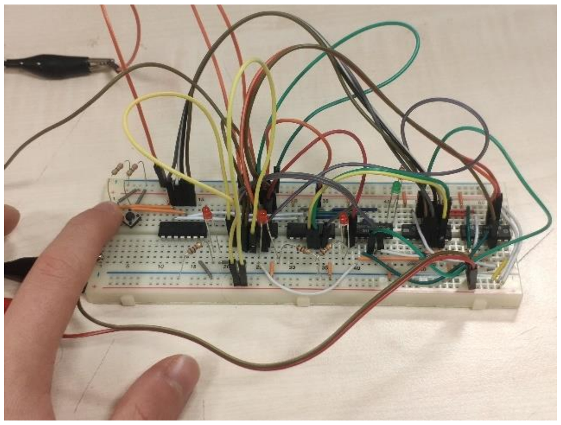
Figure 3.5: User inserts first 1€.

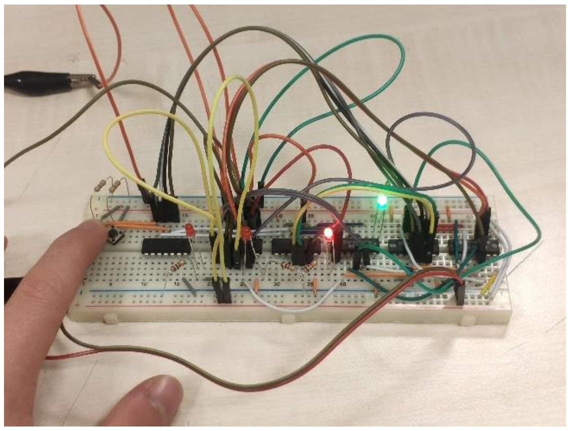
Figure 3.6: Balance = 1, User inserts second 1€.

Scenario 3: 1€ + 2x50 cents is inserted:

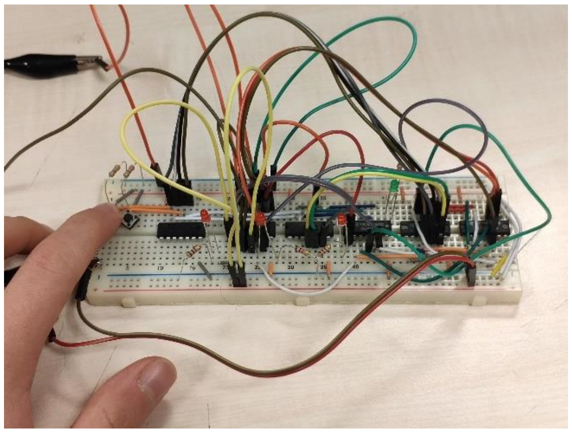
Figure 3.7: User inserts 1€.

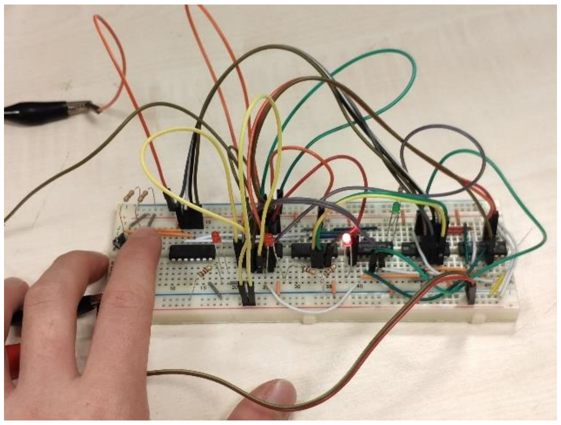
Figure 3.8: Balance = 1, User inserts first 50 cent.

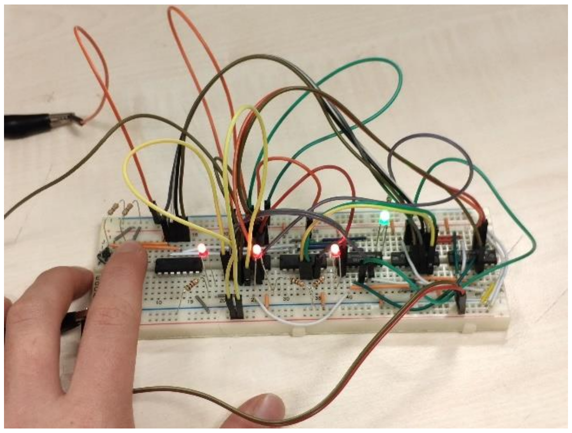
Figure 3.9: User inserts second 50 cent.

Scenario 4: 2€ is inserted:

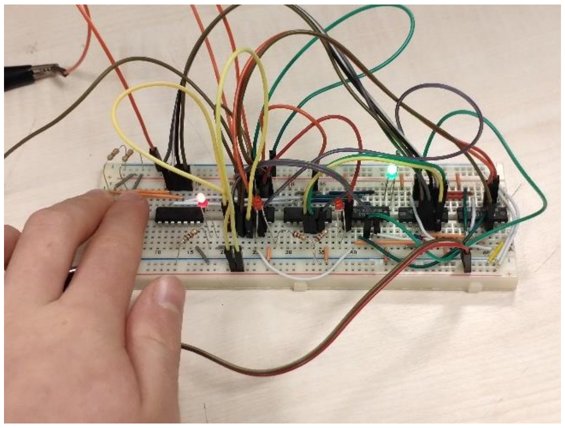
Figure 3.10: User inserts 2€.

Conclusion:

In this lab, I successfully designed and implemented a finite state machine on a breadboard. We learned how to analyze and design FSMs using state diagrams and state tables in the lectures which I used in implementing this lab’s design.