EE M16 and CS M51A Winter 2019 Section 2

Logic Design of Digital Systems

Dr. Yutao He

Verilog Lab #3 - Design of Sequential Systems

Due: March 10, 2019

Team ID: _____P26_____

(1) Name: _____Huynh_____Derrick_____
Last First

Student ID: _____705101279_____

Signature: _____Derrick Huynh_____

(2) Name: _____Xia_____Lucas_____
Last First

Student ID: _____005099109_____

Signature: _____Lucas Xia_____

Date: _____03/10/19_____

Result	
Correctness	
Creativity	
Report	
Total Score	

Verilog Lab #3 Project

(1) *Abstract*

Implement a vending machine that takes in nickels and dimes and deposits gum and nickels with a finite state machine that uses JK flip flops. The FSM has 4 states, which represents empty, 5c, 10c, and 15c. The transition of states depends on the input of x1 and x0, the encoding for the coin values.

Inputs		**Outputs**	
Variables	**Values**	**Variables**	**Values**
Reset	{True (T),False (F)}	Release Gum (RG)	{T,F}
Coin	{Empty (E), Nickel (N), Dime (D) }	Return Nickel (RN)	{T,F}

Table 1: The Inputs and Outputs of the iKon Controller

(2) *The Functions of the Circuit*

$J_2 = x_0 r(s_0 + x_1)$

$K_2 = (x_0 + r)(s_0' + x_1 + r)$

$J_1 = s_1' x_0 r$

$K_1 = (x_0 + r)(s_1 + x_1 + r)$

$z_1 = s_1 x_0 r'(s_0' + x_1)$

$z_0 = s_1 s_0' x_1 r$

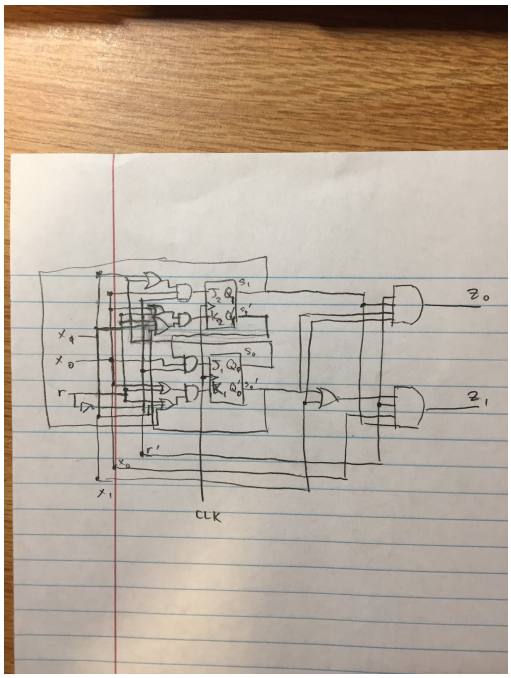


Figure 1: Schematic of the FSM

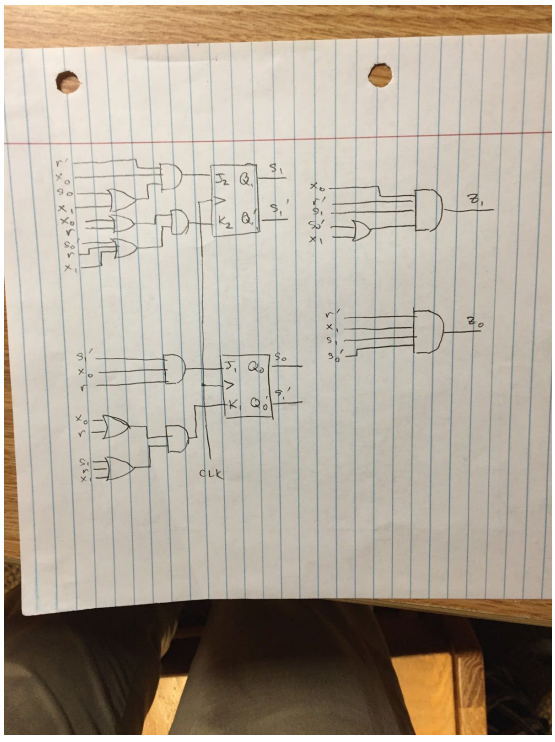


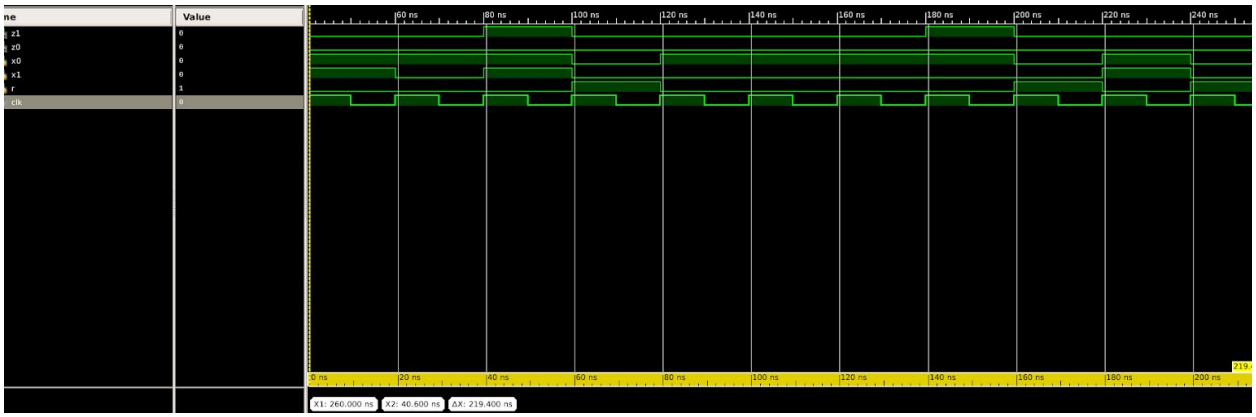
Figure 2: A Simplified Schematic of the FSM

The appendix contains the handwritten notes that provide work in determining the switching expressions (Figures 3, 4, 5, 6, 7).

(3) *The Verilog Code*

It is attached.

(4) *The Simulation Result*



The timing diagram above shows the outputs z1 and z0 when different inputs of x1x0r are input into the system. The first line is z1 and the second line represents z0. z1 and z0 follow the encodings for our output as described. z1 is meant to represent RG or release gum. z0 is RN, or return nickel. The third line is x0, and the fourth line is x1. This is meant to show the different coin inputs, as following the encoding, with 00 being nothing, 01 being a nickel, and 11 being a dime. The fifth line is r, with represents the reset variable. In our implementation, we decided to include reset as a variable and not include it as an asynchronous reset in our verilog file. The last line represents the clock signal. In the timing diagram above, the inputs DND (Dime, Nickel, Dime) are input, which results in z1 outputting 11 as the controller should release a gum and return a nickel. Then the controller is reset, then NNNN (4 Nickels) is input, and z1 again outputs 1. The controller is reset and a dime and reset is input, resulting in nothing.

(5) *The Design Review*

The work we did for designing the circuit implementation was a strong reinforcement of what we had learned in class and problems involving flip-flops done on the homework. The most difficult part of the handwritten work was the K-maps. We opted for K-maps instead of the Quine-McCluskey method because Quine-McCluskey requires a lot of comparisons and work. The K-maps needed to be 3-dimensional but it was a far easier method, despite our inexperience with 5 or more input K-maps.

One main issue that we encountered while working on this project was the implementation of our system in Verilog code. There were few tutorials available online on how to create a sequential system using JK flip flops. To overcome this issue, we had to utilize multiple tutorials online to understand the logic behind the verilog code. We combined tutorials for implementing a JK flip flop by itself, creating a Finite State Machine (FSM), and combining multiple Verilog Modules together. After writing our main .v files for both the JK Flip Flop and the overall controller, we were tasked with creating a test bench file to test our system. This proved incredibly difficult as even fewer tutorials were available. In addition, multiple bugs occured with certain implementations that were hard to understand. The most time was spent trying to debug our test bench file. One such issue with the test bench file that we encountered was the outputs not outputting a logical value. In our simulation, z1 and z0 outputted x, which in verilog means an illogical statement.

(6) *Team Member Contributions*

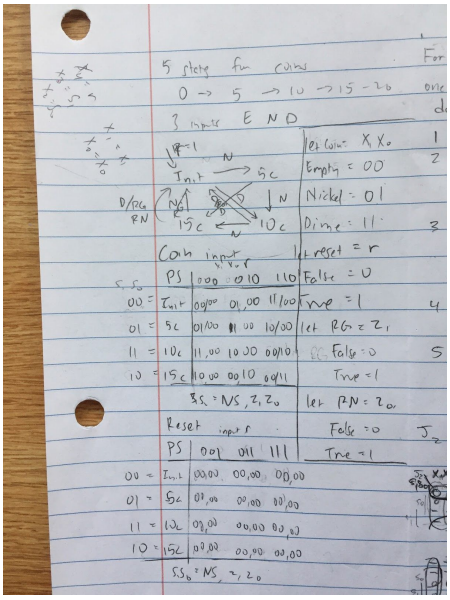
Lucas- 46%

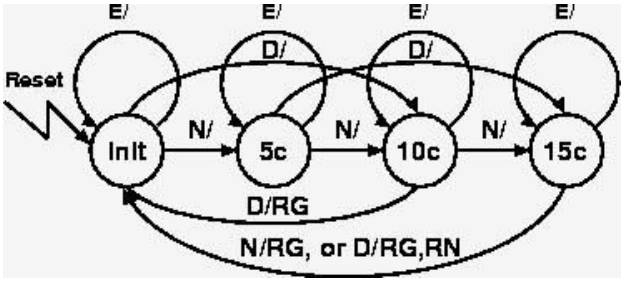
Derrick- 54%

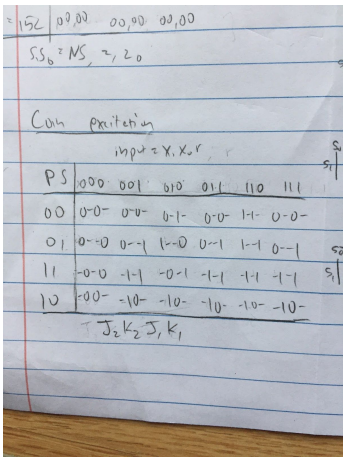
Lucas- helped write out state tables and excitation table, did minimization of the expressions with 5-input K-maps, wrote large parts of the report

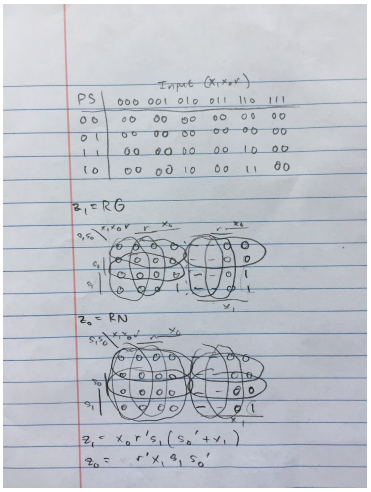
Derrick- helped write the encodings and tables, coded the verilog file and the test bench file, wrote large parts of the report

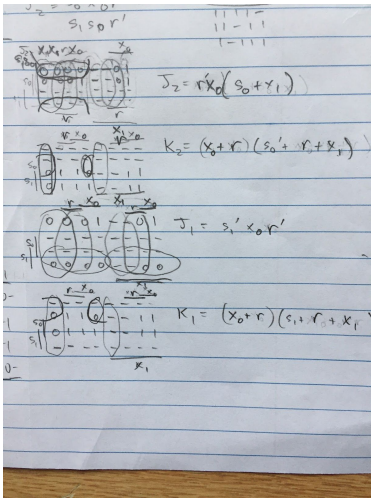
(7) *Appendix - The detailed design worksheet*











Figures 3, 4, 5, 6, 7: Handwritten/Given Work of the Inputs, Outputs, Encodings, State Tables, State Diagram, Minimization Process With K-maps

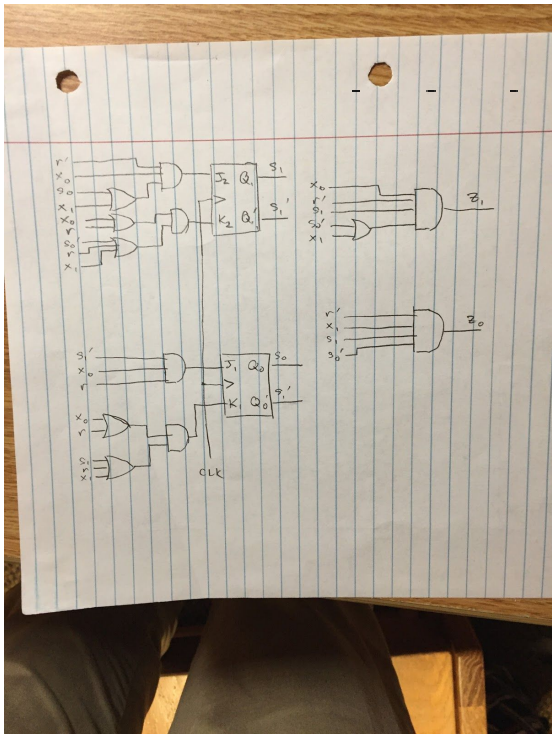


Figure 2: A Simplified Schematic of the FSM

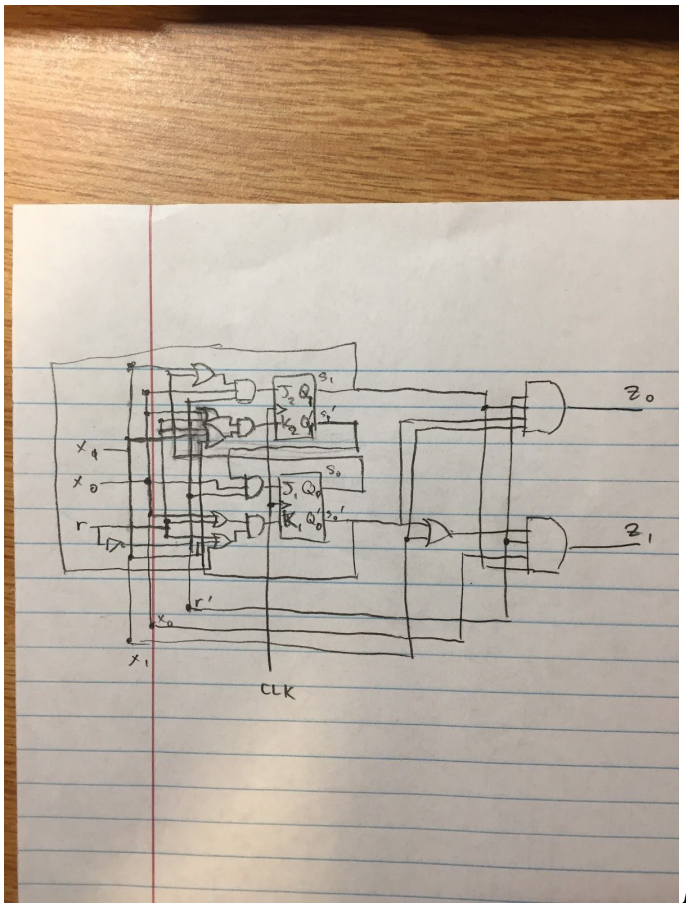


Figure 1: Schematic of the FSM