University of New South Wales

School of Electrical Engineering and Telecommunications

INDIVIDUAL Assessment Task: ______________________________

Course Code		Course Name	
Week/Session/Year		Lecturer	

Student Number	
Family Name	
Given Names	

Mark/Grade given (For official use only)

Marker's Comments:

Since this work counts toward your formal assessment for this course, please write your name and student number where indicated above, and sign the declaration below. Attach this cover sheet to the front of your submission, so that your name and student number can be seen without any cover needing to be opened.

For further information, please see http://www.lc.unsw.edu.au/plagiarism/index.html

---

I declare that this assessment item is my own work, except where acknowledged, and has not been submitted for academic credit elsewhere, and acknowledge that the assessor of this item may, for the purpose of assessing this item:

- ***Reproduce this assessment item and provide a copy to another member of the University ; and/or,***
- ***Communicate a copy of this assessment item to a plagiarism checking service (which may then retain a copy of the assessment item on its database for the purpose of future plagiarism checking).***

I certify that I have read and understood the University Rules in respect of Student Academic Misconduct.

Signature of student ____________________ Date: _______________

August 2005

Zachary Hamid
z5059915

ELEC2141 – Assignment 2

Assumptions

For the design that was implemented, it was assumed that after pressing the start button, the start button would not be pressed a second time until one of two things has occurred:

- The coins have been refunded
- The newspaper has been released

Until one of these has happened, the user will only deposit coins into the system. If the user does press the start button, the state will simply loop to itself, effectively meaning nothing has happened in the overall system.
It is also assumed that the REL and REF signals are handled externally to release the newspaper and refund the inserted coins, respectively.

State Diagram

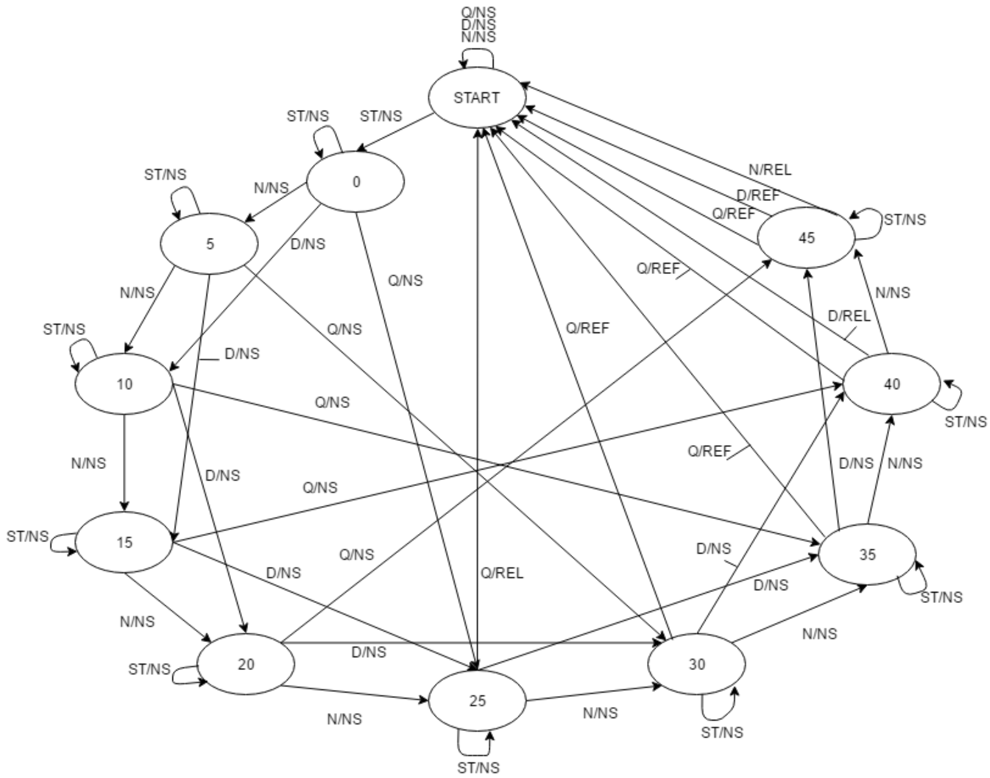


Fig 1. State diagram of the design

The states in the above diagram represent the following:

State	Meaning
START	START state, only progresses to next state when START button is pressed
0	0c total deposited
5	5c total deposited
10	10c total deposited
15	15c total deposited
20	20c total deposited
25	25c total deposited
30	30c total deposited
35	35c total deposited
40	40c total deposited
45	45c total deposited

Zachary Hamid
z5059915

The table below describes the meaning of the input and outputs:

Input	Meaning	Output	Meaning
ST	START button pressed	NS	No signal
N	Nickel inserted	REF	Refund
D	Dime inserted	REL	Release
Q	Quarter inserted	-----------	----------

Deriving a state table from Fig. 1 yields:

Table 1: State table of the design

	Next State				Output			
State	In = ST	In = N	In = D	In = Q	In = ST	In = N	In = D	In = Q
START	0	START	START	START	NS	NS	NS	NS
0	0	5	10	25	NS	NS	NS	NS
5	5	10	15	30	NS	NS	NS	NS
10	10	15	20	35	NS	NS	NS	NS
15	15	20	25	40	NS	NS	NS	NS
20	20	25	30	45	NS	NS	NS	NS
25	25	30	35	START	NS	NS	NS	REL
30	30	35	40	START	NS	NS	NS	REF
35	35	40	45	START	NS	NS	NS	REF
40	40	45	START	START	NS	NS	REL	REF
45	45	START	START	START	NS	REL	REF	REF

State minimization

In order to attempt to minimize the number of states, an implication table must be generated from the state table above. The first and second pass of the implication table generation is shown below:

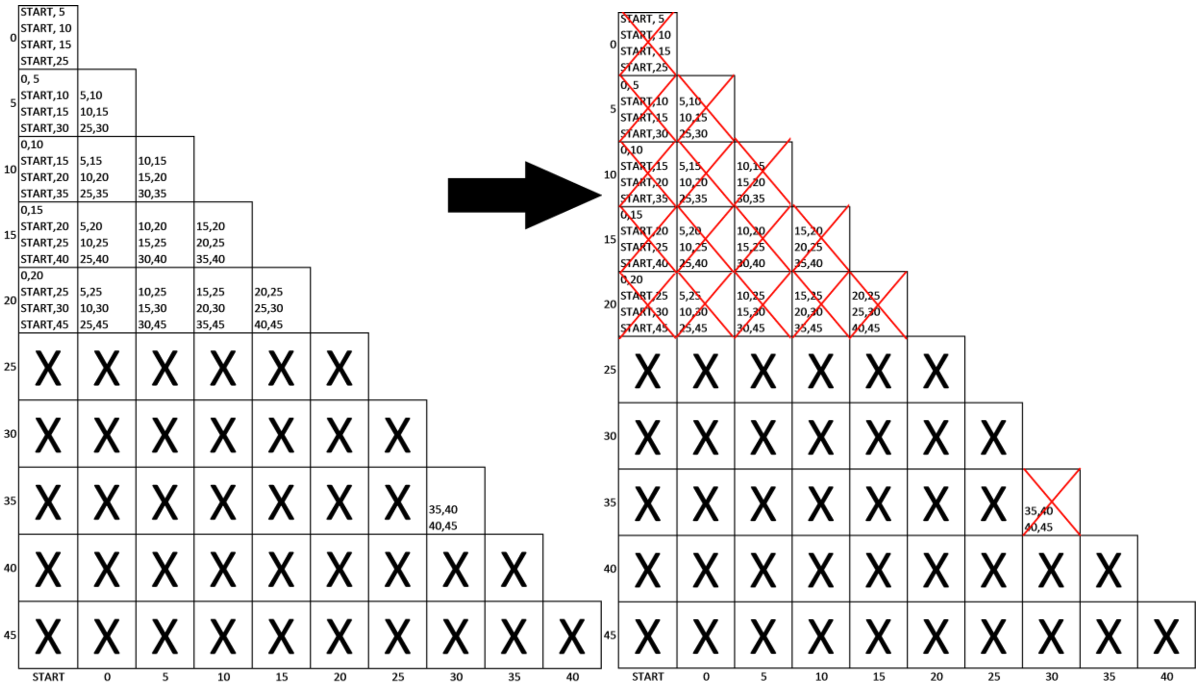


Fig. 2 – Implication table from State table (Table 1)

Zachary Hamid
z5059915

As can be seen from the above implication table, no state minimization can occur as each state is unique.

Implementation

The inputs and outputs to the FSM and the states can be assigned as follows (using gray code for the states and regular binary code for the inputs/outputs):

State	Assignment
START	0000
0	0001
5	0011
10	0010
15	0110
20	0111
25	0101
30	0100
35	1100
40	1101
45	1111

Input	Assignment	Output	Assignment
ST	00	NS	00
N	01	REF	01
D	10	REL	10
Q	11	------	----------

Redrawing the state table with the new assignments gives the following:

Table 1: State table with state assignments

	Next State				Output			
State	In = 00	In = 01	In = 10	In = 11	In = 00	In = 01	In = 10	In = 11
0000	0001	0000	0000	0000	00	00	00	00
0001	0001	0011	0010	0101	00	00	00	00
0011	0011	0010	0110	0100	00	00	00	00
0010	0010	0110	0111	1100	00	00	00	00
0110	0110	0111	0101	1101	00	00	00	00
0111	0111	0101	0100	1111	00	00	00	00
0101	0101	0100	1100	0000	00	00	00	10
0100	0100	1100	1101	0000	00	00	00	01
1100	1100	1101	1111	0000	00	00	00	01
1101	1101	1111	0000	0000	00	00	10	01
1111	1111	0000	0000	0000	00	10	01	01

Using the excitation tables for a D flip-flop, T flip-flop and JK flip-flop, the following tables were generated:

	D Flip-Flop				T Flip-Flop			
State	In = 00	In = 01	In = 10	In = 11	In = 00	In = 01	In = 10	In = 11
0000	0001	0000	0000	0000	0001	0000	0000	0000
0001	0001	0011	0010	0101	0000	0010	0011	0100
0011	0011	0010	0110	0100	0000	0001	0101	0111
0010	0010	0110	0111	1100	0000	0100	0101	1110
0110	0110	0111	0101	1101	0000	0001	0011	1011
0111	0111	0101	0100	1111	0000	0010	0011	1000
0101	0101	0100	1100	0000	0000	0001	1001	0101
0100	0100	1100	1101	0000	0000	1000	1001	0100
1100	1100	1101	1111	0000	0000	0001	0011	1100
1101	1101	1111	0000	0000	0000	0010	1101	1101
1111	1111	0000	0000	0000	0000	1111	1111	1111

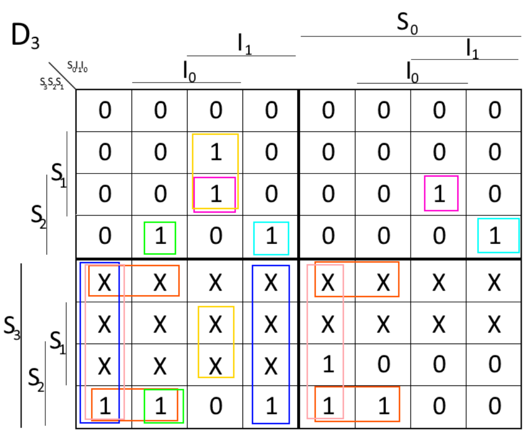
Zachary Hamid
z5059915

	JK Flip-Flop							
	In = 00		In = 01		In = 10		In = 11	
State	J	K	J	K	J	K	J	K
0000	0001	XXXX	0000	XXXX	0000	XXXX	0000	XXXX
0001	000X	XXX0	001X	XXX0	001X	XXX1	010X	XXX0
0011	00XX	XX00	00XX	XX01	01XX	XX01	01XX	XX11
0010	00X0	XX0X	01X0	XX0X	01X1	XX0X	11X0	XX1X
0110	0XX0	X00X	0XX1	X00X	0XX1	X01X	1XX1	X01X
0111	0XXX	X000	0XXX	X010	0XXX	X011	1XXX	X000
0101	0X0X	X0X0	0X0X	X0X1	1X0X	X0X1	0X0X	X1X1
0100	0X00	X0XX	1X00	X0XX	1X01	X0XX	0X00	X1XX
1100	XX00	00XX	XX01	00XX	XX11	00XX	XX00	11XX
1101	XX0X	00X0	XX1X	00X0	XX0X	11X1	XX0X	11X1
1111	XXXX	0000	XXXX	1111	XXXX	1111	XXXX	1111

The flip-flop input excitation equations were found using 6 variable K-maps assuming the state bits are $S_3S_2S_1S_0$, and input bits are I_1I_0, a K-map was made for each flip-flop for each flip-flop bit ($D_3D_2D_1D_0$ for D flip-flop, $T_3T_2T_1T_0$ for T flip-flop $J_3J_2J_1J_0$ and $K_3K_2K_1K_0$ for JK flip-flop). So in total, four 6-variable K-maps were required for the D and T flip-flops and eight 6-variable K-maps for the JK flip-flop (4 for J and 4 for K). The output equations had to be found as well in order to draw logic diagrams for each implementation, the output bits were denoted as F_1F_0.

The Boolean expressions derived from the k-maps were simplified slightly by Boolean factorization. The k-maps are as follows (note that all implicants on the k-maps are essential prime implicants):

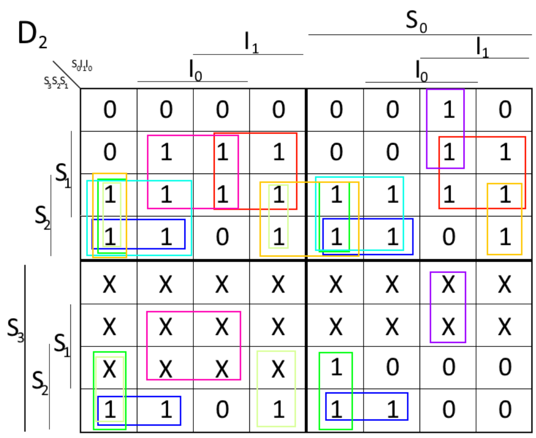
D Flip-flop Inputs



Essential prime implicants: $S_3\overline{S_1}\overline{I_1}$, $S_3\overline{S_0}\overline{I_0}$, $S_3\overline{I_1}\overline{I_0}$, $S_1\overline{S_0}I_1I_0$, $S_2\overline{S_1}\overline{S_0}\overline{I_1}I_0$, $\overline{S_3}S_2\overline{S_1}I_1\overline{I_0}$, $\overline{S_3}S_2S_1I_1I_0$

$$\therefore D_3 = S_3\overline{S_1}\overline{I_1} + S_3\overline{S_0}\overline{I_0} + S_3\overline{I_1}\overline{I_0} + S_1\overline{S_0}I_1I_0 + S_2\overline{S_1}\,\overline{S_0}\overline{I_1}I_0 + \overline{S_3}S_2\overline{S_1}I_1\overline{I_0} + \overline{S_3}S_2S_1I_1I_0$$

$$D_3 = S_3(\overline{S_1}\overline{I_1} + \overline{I_0}(\overline{S_0} + \overline{I_1})) + I_0(\overline{S_0}(S_1I_1 + S_2\overline{S_1}\,\overline{I_1})) + \overline{S_3}S_2I_1(S_1 \oplus I_0)$$



Essential prime implicants: $\overline{S_3}S_1I_1$, $\overline{S_3}S_2\overline{I_1}$, $S_2\overline{S_1}\overline{I_1}$, $\overline{S_3}S_2\overline{I_0}$, $S_1\overline{S_0}I_0$, $S_2\overline{S_0}\overline{I_0}$, $S_2\overline{I_1}\overline{I_0}$, $\overline{S_2}S_0I_1I_0$

$$\therefore D_2 = \overline{S_3}S_1I_1 + \overline{S_3}S_2\overline{I_1} + S_2\overline{S_1}\overline{I_1} + \overline{S_3}S_2\overline{I_0} + S_1\overline{S_0}I_0 + S_2\overline{S_0}\overline{I_0} + S_2\overline{I_1}\overline{I_0} + \overline{S_2}S_0I_1I_0$$

$$D_2 = \overline{S_3}(S_1I_1 + S_2(\overline{I_1} + \overline{I_0})) + S_2(\overline{S_1}\overline{I_1} + \overline{I_0}(\overline{S_0} + \overline{I_1})) + I_0(\overline{S_2}S_0I_1 + S_1\overline{S_0})$$

Zachary Hamid
z5059915

D_1

0	0	0	0	0	1	0	1
1	1	0	1	1	1	0	1
1	1	0	0	1	0	1	0
0	0	0	0	0	0	0	0
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	1	0	0	0
0	0	0	1	0	1	0	0

Essential prime implicants: $S_1\bar{S}_0\bar{I}_1$, $\bar{S}_2S_1\bar{I}_0$, $S_1I_1\bar{I}_0$, $\bar{S}_2S_0\bar{I}_1I_0$, $\bar{S}_2S_0I_1\bar{I}_0$, $S_3\bar{S}_0I_1\bar{I}_0$, $S_3\bar{S}_1S_0\bar{I}_1I_0$, $\bar{S}_3S_2S_1S_0I_1I_0$

$$\therefore D_1 = S_1\bar{S}_0\bar{I}_1 + \bar{S}_2S_1\bar{I}_0 + S_1I_1\bar{I}_0 + \bar{S}_2S_0\bar{I}_1I_0 + \bar{S}_2S_0I_1\bar{I}_0 + S_3\bar{S}_0I_1\bar{I}_0 + S_3\bar{S}_1S_0\bar{I}_1I_0 + \bar{S}_3S_2S_1S_0I_1I_0$$

$$D_1 = S_1(\bar{S}_0\bar{I}_1 + \bar{I}_0(\bar{S}_2 + \bar{I}_1)) + I_1\bar{I}_0(\bar{S}_2S_0 + S_3\bar{S}_0) + S_0I_0(\bar{I}_1(\bar{S}_2 + S_3\bar{S}_1) + \bar{S}_3S_2S_1I_1)$$

D_0

1	0	0	0	1	1	1	0
0	0	0	1	1	0	0	0
0	1	1	1	1	1	1	0
0	0	0	1	1	0	0	0
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	1	0	0	0
0	1	0	1	1	1	0	0

Essential prime implicants: $S_0\bar{I}_1\bar{I}_0$, $\bar{S}_3S_2S_1I_0$, $\bar{S}_2\bar{S}_1\bar{I}_1\bar{I}_0$, $\bar{S}_2\bar{S}_1S_0I_0$, $S_1\bar{S}_0I_1\bar{I}_0$, $S_2\bar{S}_0\bar{I}_1\bar{I}_0$, $S_3\bar{S}_1\bar{I}_1I_0$

$$\therefore D_0 = S_0\bar{I}_1\bar{I}_0 + \bar{S}_3S_2S_1I_0 + \bar{S}_2\bar{S}_1\bar{I}_1\bar{I}_0 + \bar{S}_2\bar{S}_1S_0I_0 + S_1\bar{S}_0I_1\bar{I}_0 + S_2\bar{S}_0\bar{I}_1\bar{I}_0 + S_3\bar{S}_1\bar{I}_1I_0$$

$$D_0 = \bar{I}_0(S_0\bar{I}_1 + \bar{S}_0I_1(S_1 + S_2)) + \bar{S}_2\bar{S}_1(\bar{I}_1\bar{I}_0 + S_0I_0) + I_0(S_3\bar{S}_1\bar{I}_1 + \bar{S}_3S_2S_1)$$

T Flip-flop Inputs

T_3

0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	1	0	0	0	1
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	0	1	1	1
0	0	1	0	0	0	1	1

Essential prime implicants: $S_3I_1I_0$, $S_3S_0I_1$, $S_3S_1I_0$, $\bar{S}_0I_1I_0$, $S_2S_1I_1I_0$, $\bar{S}_3S_2\bar{S}_1I_1\bar{I}_0$, $\bar{S}_3S_2\bar{S}_1\bar{S}_0\bar{I}_1I_0$

$$\therefore T_3 = S_3I_1I_0 + S_3S_0I_1 + S_3S_1I_0 + \bar{S}_0I_1I_0 + S_2S_1I_1I_0 + \bar{S}_3S_2\bar{S}_1I_1\bar{I}_0 + \bar{S}_3S_2\bar{S}_1\bar{S}_0\bar{I}_1I_0$$

$$T_3 = S_3(S_0I_1 + S_1I_0) + I_1I_0(S_3 + \bar{S}_0 + S_2S_1) + \bar{S}_3S_2\bar{S}_1(I_1\bar{I}_0 + \bar{S}_0\bar{I}_1I_0)$$

T_2

0	0	0	0	0	0	1	0
0	1	1	1	0	0	1	1
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	0	1	1	1
0	0	1	0	0	0	1	1

Essential prime implicants: $\bar{S}_2S_1I_1$, $S_3S_0I_1$, $S_3S_1I_0$, $S_2\bar{S}_1I_1I_0$, $\bar{S}_2S_0I_1I_0$, $\bar{S}_2S_1\bar{S}_0I_0$

$$\therefore T_2 = \bar{S}_2S_1I_1 + S_3S_0I_1 + S_3S_1I_0 + S_2\bar{S}_1I_1I_0 + \bar{S}_2S_0I_1I_0 + \bar{S}_2S_1\bar{S}_0I_0$$

$$T_2 = I_1((\bar{S}_2S_1 + S_3S_0) + I_0(S_2\bar{S}_1 + \bar{S}_2S_0)) + S_1I_0(S_3 + \bar{S}_2\bar{S}_0)$$

T_1

0	0	0	0	0	1	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	1	0	1
0	0	0	0	0	0	0	0
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	0	1	1	1
0	0	0	1	0	1	0	0

Essential prime implicants: $S_3S_1I_0$, $\bar{S}_2S_1I_1I_0$, $S_1\bar{S}_0I_1I_0$, $S_2S_1I_1\bar{I}_0$, $S_3\bar{S}_0I_1\bar{I}_0$, $S_3S_0\bar{I}_1I_0$, $S_2S_1S_0\bar{I}_1I_0$, $\bar{S}_2\bar{S}_1S_0\bar{I}_1I_1$, $\bar{S}_2\bar{S}_1S_0I_1\bar{I}_0$

$$\therefore T_1 = S_3S_1I_0 + \bar{S}_2S_1I_1I_0 + S_1\bar{S}_0I_1I_0 + S_2S_1I_1\bar{I}_0 + S_3\bar{S}_0I_1\bar{I}_0 + S_3S_0\bar{I}_1I_0 + S_2S_1S_0\bar{I}_1I_0 + \bar{S}_2\bar{S}_1S_0\bar{I}_1I_0 + \bar{S}_2\bar{S}_1S_0I_1\bar{I}_0$$

$$T_1 = I_0(S_3S_1 + S_0\bar{I}_1(S_3 + \overline{\bar{S}_1 \oplus S_2})) + I_1(I_0(S_1(\bar{S}_2 + \bar{S}_0)) + \bar{I}_0(S_2S_1 + \bar{S}_2\bar{S}_1S_0 + S_3\bar{S}_0))$$

T_0

1	0	0	0	0	0	0	1
0	0	0	1	0	1	1	1
0	1	1	1	0	0	0	1
0	0	0	1	0	1	1	1
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	0	1	1	1
0	1	0	1	0	0	1	1

Essential prime implicants: $S_0I_1\bar{I}_0$, $S_1I_1\bar{I}_0$, $S_2I_1\bar{I}_0$, $S_3S_0I_1$, $S_3S_1I_0$, $\bar{S}_2S_1S_0I_0$, $S_2\bar{S}_1S_0I_0$, $S_3\bar{S}_0\bar{I}_1I_0$, $\bar{S}_3S_2\bar{S}_1S_0I_0$, $\bar{S}_2\bar{S}_1\bar{S}_0\bar{I}_1\bar{I}_0$

$$\therefore T_0 = S_0I_1\bar{I}_0 + S_1I_1\bar{I}_0 + S_2I_1\bar{I}_0 + S_3S_0I_1 + S_3S_1I_0 + \bar{S}_2S_1S_0I_0 + S_2\bar{S}_1S_0I_0 + S_3\bar{S}_0\bar{I}_1I_0 + \bar{S}_3S_2\bar{S}_1S_0I_0 + \bar{S}_2\bar{S}_1\bar{S}_0\bar{I}_1\bar{I}_0$$

$$T_0 = \bar{I}_0(I_1(S_0 + S_1 + S_2) + \bar{S}_0\bar{I}_1(S_3 + \bar{S}_2\bar{S}_1)) + S_1I_0(S_3 + S_0 \oplus S_2) + S_0(\bar{S}_3S_2\bar{S}_1I_0 + S_3I_1)$$

Zachary Hamid
z5059915

JK Flip-flop Inputs

J_3

0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	1	0	0	0	1
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X

Essential prime implicants: $S_1\bar{S}_0 I_1 I_0,\ S_2\bar{S}_1 I_1\bar{I}_0,\ S_2 S_1 I_1 I_0,\ S_2\bar{S}_1\bar{S}_0\bar{I}_1 I_0$

$$\therefore J_3 = S_1\bar{S}_0 I_1 I_0 + S_2\bar{S}_1 I_1\bar{I}_0 + S_2 S_1 I_1 I_0 + S_2\bar{S}_1\bar{S}_0\bar{I}_1 I_0$$

$$J_3 = I_1 I_0(S_1(\bar{S}_0 + S_2)) + S_2\bar{S}_1(I_1\bar{I}_0 + \bar{S}_0\bar{I}_1 I_0)$$

J_2

0	0	0	0	0	0	1	0
0	1	1	1	0	0	1	1
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X

Essential prime implicants: $S_1 I_1,\ S_0 I_1 I_0,\ S_1\bar{S}_0 I_0$

$$\therefore J_2 = S_1 I_1 + S_0 I_1 I_0 + S_1\bar{S}_0 I_0$$

$$J_2 = I_1(S_1 + S_0 I_0) + S_1\bar{S}_0 I_0$$

J_1

0	0	0	0	0	1	0	1
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
0	0	0	0	0	0	0	0
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
0	0	0	1	0	1	0	0

Essential prime implicants: $\bar{S}_2 S_0\bar{I}_1 I_0,\ \bar{S}_2 S_0 I_1\bar{I}_0,\ S_3\bar{S}_0 I_1\bar{I}_0,\ S_3 S_0\bar{I}_1 I_0$

$$\therefore J_1 = \bar{S}_2 S_0\bar{I}_1 I_0 + \bar{S}_2 S_0 I_1\bar{I}_0 + S_3\bar{S}_0 I_1\bar{I}_0 + S_3 S_0\bar{I}_1 I_0$$

$$J_1 = \bar{S}_2 S_0(I_0 \oplus I_1) + S_3(\bar{S}_0 I_1\bar{I}_0 + S_0\bar{I}_1 I_0)$$

J_0

1	0	0	0	X	X	X	X
0	0	0	1	X	X	X	X
0	1	1	1	X	X	X	X
0	0	0	1	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
0	1	0	1	X	X	X	X

Essential prime implicants: $S_1 I_1\bar{I}_0,\ S_2 I_1\bar{I}_0,\ S_2 S_1 I_0,\ S_3\bar{I}_1 I_0,\ \bar{S}_2\bar{S}_1\bar{I}_1\bar{I}_0$

$$\therefore J_0 = S_1 I_1\bar{I}_0 + S_2 I_1\bar{I}_0 + S_2 S_1 I_0 + S_3\bar{I}_1 I_0 + \bar{S}_2\bar{S}_1\bar{I}_1\bar{I}_0$$

$$J_0 = \bar{I}_0(I_1(S_1 + S_2) + \bar{S}_2\bar{S}_1\bar{I}_1) + I_0(S_2 S_1 + S_3\bar{I}_1)$$

K_3

X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	0	1	1	1
0	0	1	0	0	0	1	1

Essential prime implicants: $I_1 I_0,\ S_0 I_1,\ S_1 I_0$

$$\therefore K_3 = I_1 I_0 + S_0 I_1 + S_1 I_0$$

$$K_3 = I_0(I_1 + S_1) + S_0 I_1$$

K_2

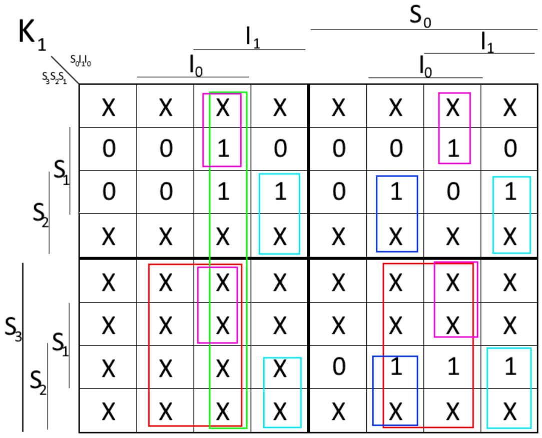
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
X	X	X	X	X	X	X	X
X	X	X	X	X	X	X	X
X	X	X	X	0	1	1	1
0	0	1	0	0	0	1	1

Essential prime implicants: $\bar{S}_1 I_1 I_0,\ S_3 S_0 I_1,\ S_3 S_1 I_0$

$$\therefore K_2 = \bar{S}_1 I_1 I_0 + S_3 S_0 I_1 + S_3 S_1 I_0$$

$$K_2 = I_1(\bar{S}_1 I_0 + S_3 S_0) + S_3 S_1 I_0$$

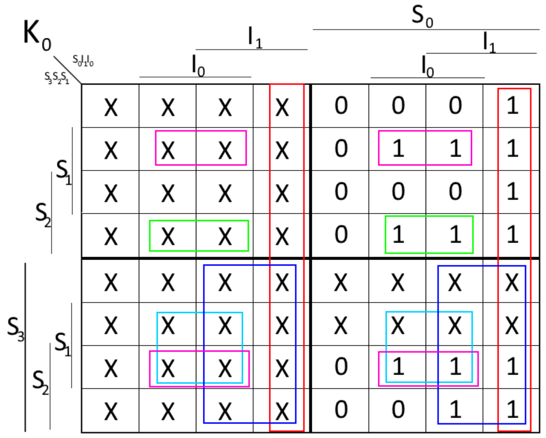
Zachary Hamid
z5059915



Essential prime implicants: $S_3I_0,\ \bar{S}_2I_1I_0,\ \bar{S}_0I_1I_0,\ S_2I_1\bar{I_0},\ S_2S_0\bar{I_1}I_0$

$$\therefore K_1 = S_3I_0 + \bar{S}_2I_1I_0 + \bar{S}_0I_1I_0 + S_2I_1\bar{I_0} + S_2S_0\bar{I_1}I_0$$

$$K_1 = I_0(S_3 + I_1(\bar{S_2} + \bar{S_0}) + S_2S_0\bar{I_1}) + S_2I_1\bar{I_0}$$

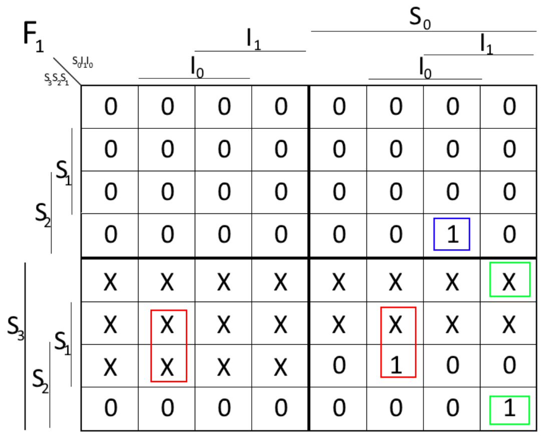


Essential prime implicants: $I_1\bar{I_0},\ S_3I_1,\ \bar{S}_2S_1I_0,\ S_3S_1I_0,\ \bar{S}_3S_2\bar{S}_1I_0$

$$\therefore K_0 = I_1\bar{I_0} + S_3I_1 + \bar{S}_2S_1I_0 + S_3S_1I_0 + \bar{S}_3S_2\bar{S}_1I_0$$

$$K_0 = I_1\bar{I_0} + S_3I_1 + I_0(S_1(\bar{S_2} + S_3) + \bar{S}_3S_2\bar{S}_1)$$

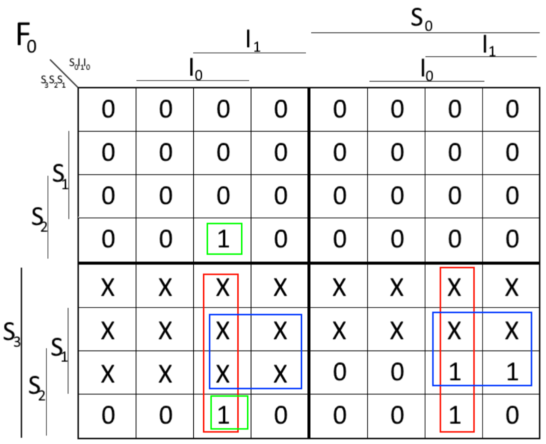
Outputs



Essential prime implicants: $S_3S_1\bar{I_1}I_0,\ S_3\bar{S}_1S_0I_1\bar{I_0},\ \bar{S}_3S_2\bar{S}_1S_0I_1I_0$

$$\therefore F_1 = S_3S_1\bar{I_1}I_0 + S_3\bar{S}_1S_0I_1\bar{I_0} + \bar{S}_3S_2\bar{S}_1S_0I_1I_0$$

$$F_1 = S_3S_1\bar{I_1}I_0 + S_0I_1(S_3\bar{S}_1\bar{I_0} + \bar{S}_3S_2\bar{S}_1I_0)$$



Essential prime implicants: $S_3I_1I_0,\ S_3S_1I_1,\ S_2\bar{S}_1\bar{S}_0I_1I_0$

$$\therefore F_0 = S_3I_1I_0 + S_3S_1I_1 + S_2\bar{S}_1\bar{S}_0I_1I_0$$

$$F_0 = I_1(S_3(I_0 + S_1) + S_2\bar{S}_1\bar{S}_0I_0)$$

From the above Boolean expression derivations, the logic diagrams for each implementation (D, T and JK implementations) can be drawn (note that most wires were omitted and replaced with labels in order to preserve readability of the diagrams, also the flip-flops were assumed to have an active-low clear input, hence why they are connected to Vcc). The labels are assigned to mean the following:

Label	Meaning
X_NOT	The complement of some signal X
IN_1/0	The input bits
F1/0	The output bits
S3/2/1/0	State bits

Zachary Hamid
z5059915

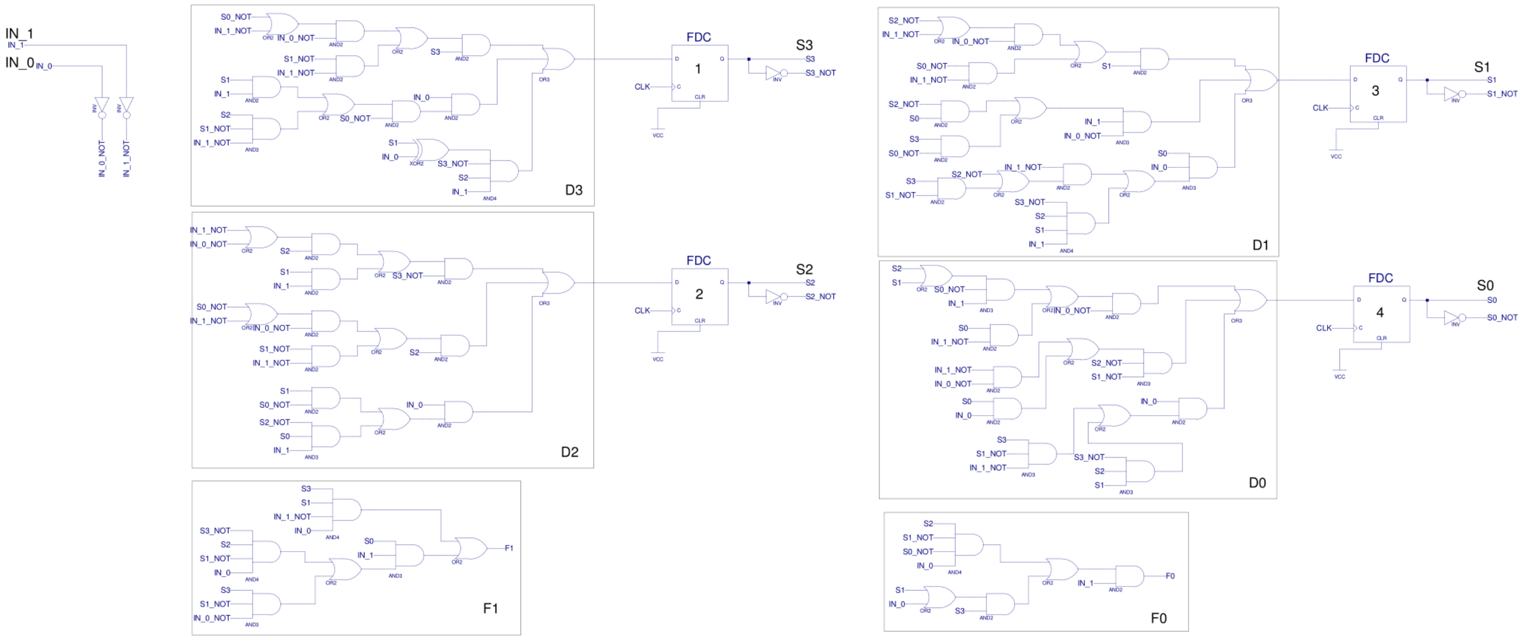


Fig. 3 – D flip-flop implementation of design

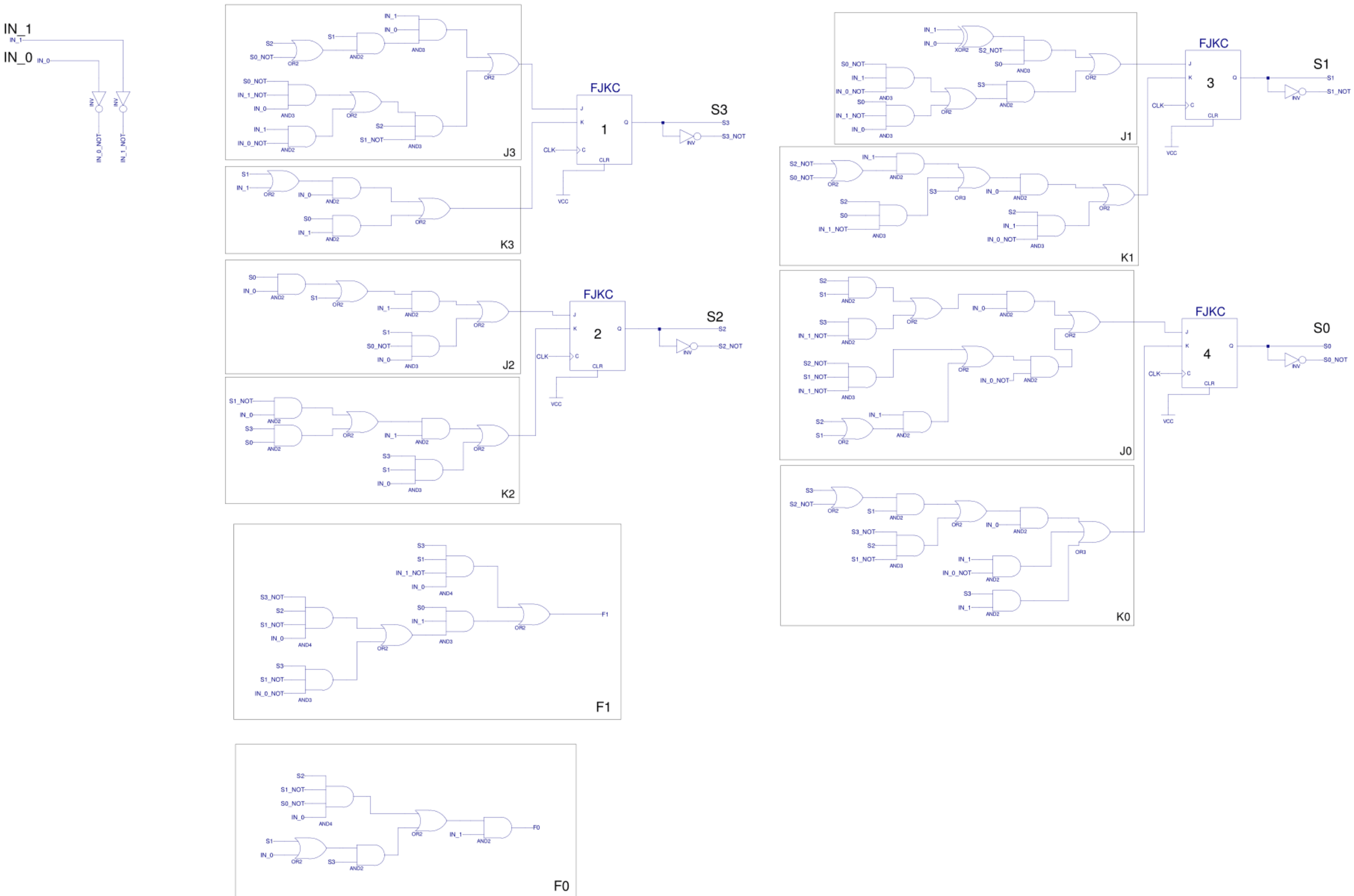


Fig. 4 – JK flip-flop implementation of design

Zachary Hamid
z5059915

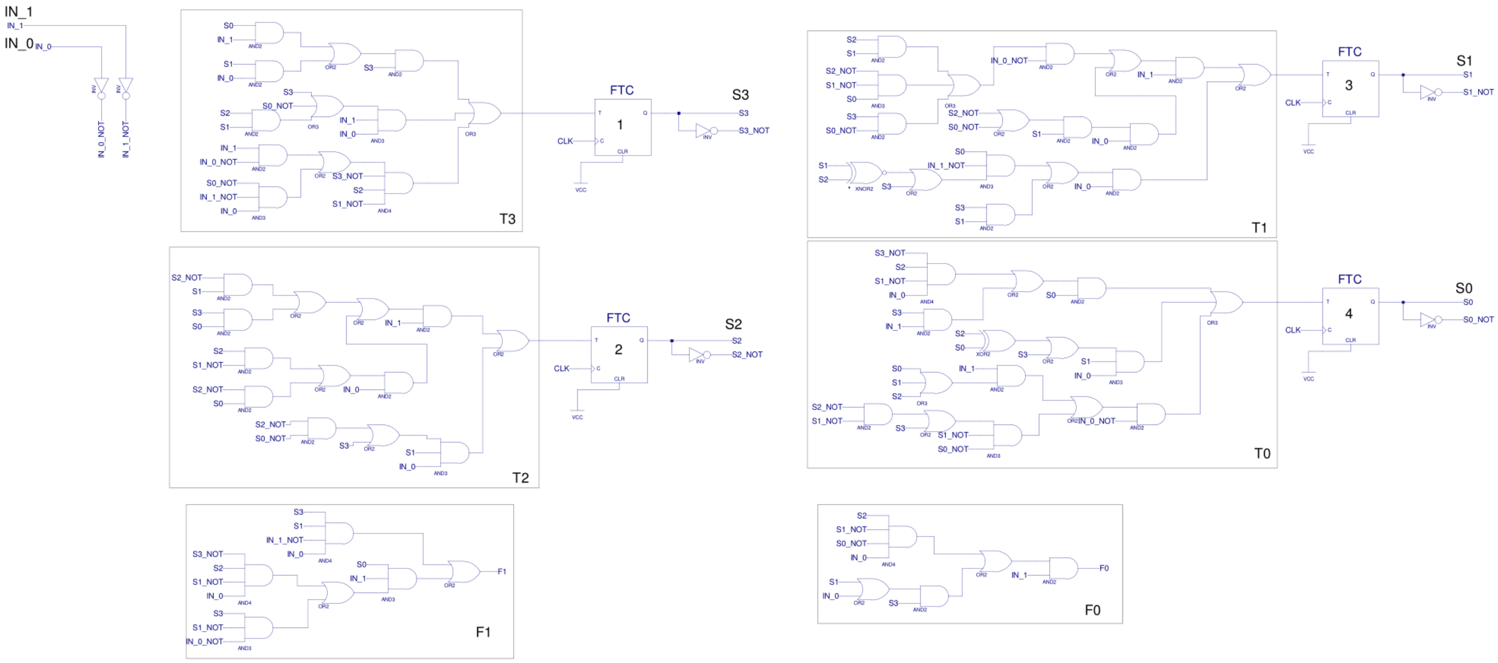


Fig. 5 – T flip-flop implementation of design

Gate input cost

By simply counting the gates in the above logic diagrams, the GIC for the inputs into the flip-flops and outputs of each implementation was found. An assumption was made that the flip-flops used didn't have a "free" complement of the output from the flip-flop and instead had to be manually inverted, adding a GIC of 4 (GIC(State inverters) = 4) to each implementation. It was also assumed that XOR/XNOR gates had a GIC of 2, and that the flip-flops themselves did not have a clear (it is not used in this implementation and so in real application a flip-flop without clear would be used in it's place), therefore the flip-flops had GICs of:

GIC(D FF) = 9, GIC(T FF) = 10, GIC(JK FF) = 10

The output circuit and inputs are common to all implementations and they have a GIC of:

GIC(F_1) = 18, GIC(F_0) = 12, GIC(Input) = 2

D Flip-flop implementation

GIC(D_3) = 30, GIC(D_2) = 32, GIC(D_1) = 37, GIC(D_0) = 33

So the GIC of the flip-flop input equations and outputs is:

GIC(D_3) + GIC(D_2) + GIC(D_1) + GIC(D_0) + GIC(F_1) + GIC(F_0) + GIC(Input) + GIC(State inverters) + 4xGIC(D FF)

= 204

T Flip-flop implementation

GIC(T_3) = 30, GIC(T_2) = 27, GIC(T_1) = 37, GIC(T_0) = 36

So the GIC of the flip-flop input equations and outputs is:

GIC(T_3) + GIC(T_2) + GIC(T_1) + GIC(T_0) + GIC(F_1) + GIC(F_0) + GIC(Input) + GIC(State inverters) + 4xGIC(T FF)

= 206

Zachary Hamid
z5059915

JK Flip-flop implementation

GIC(J_3) = 19, GIC(J_2) = 11, GIC(J_1) = 17, GIC(J_0) = 21, GIC(K_3) = 8, GIC(K_2) = 13, GIC(K_1) = 17, GIC(K_0) = 18

So the GIC of the flip-flop input equations and outputs is:

GIC(J_3) + GIC(J_2) + GIC(J_1) + GIC(J_0) + GIC(K_3) + GIC(K_2) + GIC(K_1) + GIC(K_0) + GIC(F_1) + GIC(F_0) + GIC(Input) + GIC(State inverters) + 4xGIC(JK FF)

= 200

So, it can be seen that the GIC of each circuit is incredibly similar, with JK being the most optimal implementation by a small margin (however, the original Boolean expressions may not have been completely optimally reduced and so this result could change, since the difference in GIC between implementations is so small). Since the gate input costs are only marginally different, it doesn't particularly matter which implementation is chosen for this design, however JK flip-flops have 8 less complex sub-circuits and so have more modularity than the other designs.

Zachary Hamid
z5059915

Verification

The schematics for the D implementation in Xilinx can be seen above in Implementation. This implementation was tested using the behavioural Verilog HDL rather than using the schematic. The test bench is shown below, the input was set as always having nickels inserted into the system with a start signal input to go from state START -> 0c, and it sequentially goes through each state.

```verilog
`timescale 1ns / 1ps
module beh_tbw;

   // Inputs
   reg [1:0] IN;
   reg CLK;

   // Outputs
   wire [1:0] OUT;
   wire [3:0] state;
   wire [3:0] next_state;

   // Instantiate the Unit Under Test (UUT)
   DFF_Beh uut (
      .IN(IN),
      .CLK(CLK),
      .OUT(OUT),
      .state(state),
      .next_state(next_state)
   );

   initial begin
      // Initialize Inputs
      IN = 2'b00;
      CLK = 1'b0;

      // Wait 100 ns for global reset to finish
      #100; // Hold clock off for 100ns so 0000 state stays on for same time as the other states
      #100;
      CLK = ~CLK;
      #100
      CLK = ~CLK;
      IN = 2'b01; //After initial input of 00 to go from START -> 0c states, input nickels, this ensures every state is visited
                  //sequentially
      forever
      begin
      #100;
      CLK = ~CLK;     //alternate clock every 100ns
      end
   end

endmodule
```

Fig. 6 – Test bench for verification of the D implementation of the FSM

This test bench produced the following simulation:

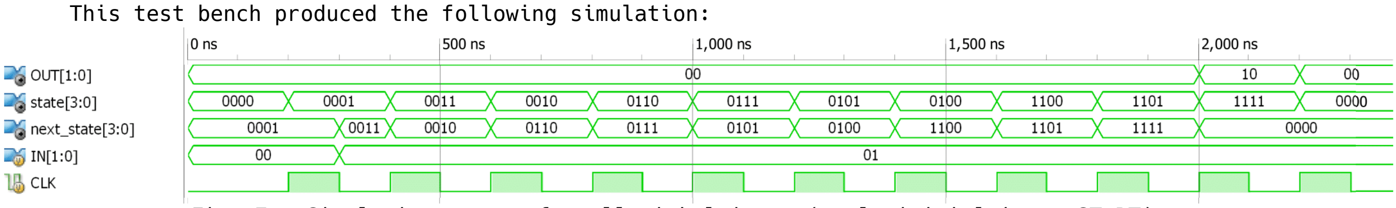


Fig. 7 – Simulation output for all nickel input (excl. initial input START)

Zachary Hamid
z5059915

By changing line 33 in the test bench to give a different constant input of dimes and quarters these simulations were generated:

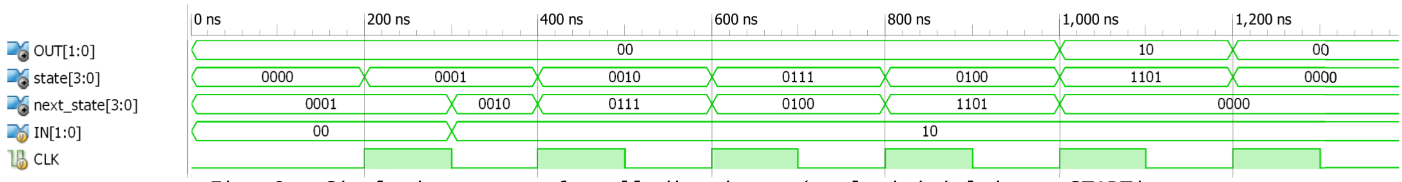


Fig. 8 – Simulation output for all dime input (excl. initial input START)

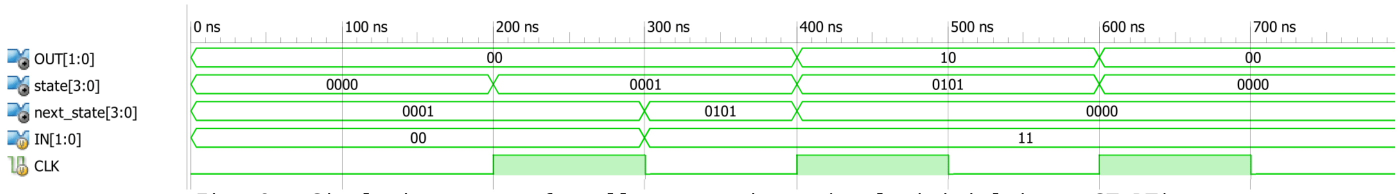


Fig. 9 – Simulation output for all quarter input (excl. initial input START)

Zachary Hamid
z5059915

Verilog HDL

The behavioural Verilog HDL that was used to verify the D implementation of the design is shown below:

```
`timescale 1ns / 1ps
module DFF_Beh(
    input [1:0] IN,
    input CLK,
    output reg [1:0]OUT,
    output reg [3:0] state, next_state
    );
    // Possible states
    parameter S0 = 4'b0000, S1 = 4'b0001, S2 = 4'b0011, S3 = 4'b0010,
            S4 = 4'b0110, S5 = 4'b0111, S6 = 4'b0101, S7 = 4'b0100,
            S8 = 4'b1100, S9 = 4'b1101, S10 = 4'b1111;
   //Possible inputs
    parameter I0 = 2'b00, I1 = 2'b01, I2 = 2'b10, I3 = 2'b11;
    //Possible outputs
    parameter NS = 2'b00, REF = 2'b01, REL = 2'b10;
    initial state = S0;
    initial next_state = S0;
    initial OUT = NS;
    always @(posedge CLK)
      state <= next_state;
    always @(state, IN)
      case(state)
         S0: case (IN)
               I0: next_state <= S1;
               I1: next_state <= S0;
               I2: next_state <= S0;
               I3: next_state <= S0;
             endcase
         S1: case (IN)
               I0: next_state <= S1;
               I1: next_state <= S2;
               I2: next_state <= S3;
               I3: next_state <= S6;
             endcase
         S2: case (IN)
               I0: next_state <= S2;
               I1: next_state <= S3;
               I2: next_state <= S4;
               I3: next_state <= S7;
             endcase
         S3: case (IN)
               I0: next_state <= S3;
               I1: next_state <= S4;
               I2: next_state <= S5;
               I3: next_state <= S8;
             endcase
         S4: case (IN)
               I0: next_state <= S4;
               I1: next_state <= S5;
               I2: next_state <= S6;
               I3: next_state <= S9;
             endcase
         S5: case (IN)
               I0: next_state <= S5;
               I1: next_state <= S6;
               I2: next_state <= S7;
               I3: next_state <= S10;
```

```
             endcase
         S6: case (IN)
               I0: next_state <= S6;
               I1: next_state <= S7;
               I2: next_state <= S8;
               I3: next_state <= S0;
             endcase
         S7: case (IN)
               I0: next_state <= S7;
               I1: next_state <= S8;
               I2: next_state <= S9;
               I3: next_state <= S0;
             endcase
         S8: case (IN)
               I0: next_state <= S8;
               I1: next_state <= S9;
               I2: next_state <= S10;
               I3: next_state <= S0;
             endcase
         S9: case (IN)
               I0: next_state <= S9;
               I1: next_state <= S10;
               I2: next_state <= S0;
               I3: next_state <= S0;
             endcase
         S10: case (IN)
               I0: next_state <= S10;
               I1: next_state <= S0;
               I2: next_state <= S0;
               I3: next_state <= S0;
             endcase
      endcase

   always @(state, IN)
      case (state)
         S0,S1,S2,S3,S4,S5: OUT <= NS;
         S6:      case(IN)
                     I0,I1,I2: OUT <= NS;
                     I3: OUT <= REL;
                  endcase
         S7,S8:   case(IN)
                     I0,I1,I2: OUT <= NS;
                     I3: OUT <= REF;
                  endcase
         S9:      case(IN)
                     I0,I1: OUT <= NS;
                     I2: OUT <= REL;
                     I3: OUT <= REF;
                  endcase
         S10:     case(IN)
                     I0: OUT <= NS;
                     I1: OUT <= REL;
                     I2,I3: OUT <= REF;
                  endcase
      endcase
endmodule
```