Instituto Tecnológico de Buenos Aires

14 de Noviembre de 2018

Trabajo de Laboratorio $N°$ 3

Electrónica III

Grupo 7

Porfesores:

Kevin Dewald
Pablo Wundes

Alumnos:

Nombre	*Legajo*
Ariel Nowik	58309
Joaquín Mestanza	58288
Marcelo Regueira	58300
Martina Máspero	57120

Task 1

In this section, a state machine will be developed for controlling the switching on and off of two pumps, to fill a tank. The are controlled by two sensors from the upper part of the tank (S) and the lower part of the tank (I). The actions to take are as follows:

- Tank full: S = I = 1 - Pumps OFF
- Tank empty: S = I = 0 - Pumps ON
- Half full tank: S = 0 & I = 1 - Pumps alternate

With this in mind, a Moore machine is developed as follows.

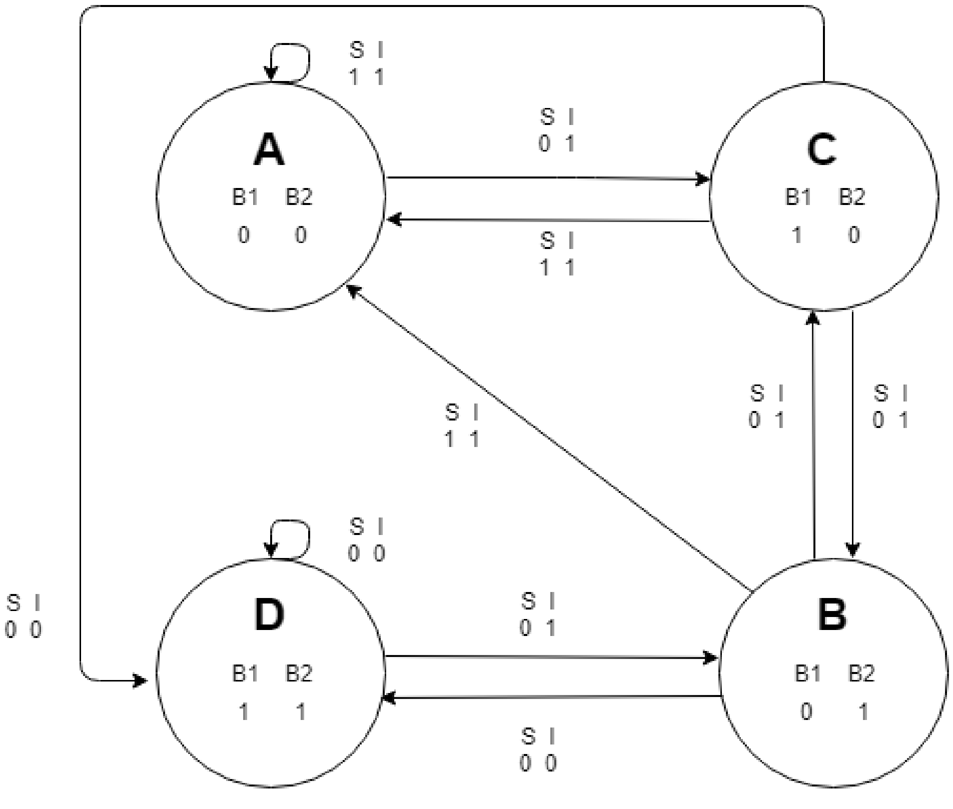


Figure 1: Moore state machine diagram

Using two bits to asign the states, a table of transitions is made, as shown below.

Estado Actual		Estado Siguiente				Salidas	
	y2 - y1	S - I				B1	B2
		00	01	10	11		
A	00	X	C	X	A	0	0
B	01	D	C	X	A	0	1
C	10	D	B	X	A	1	0
D	11	D	B	X	X	1	1

Figure 2: Moore state machine - Transitions

From the table, using Karnaugh's maps (see resolution in *Annex*) the functions for the state variables and the two pumps outputs result as shown below.

$$Y_2 = \overline{I} + \overline{S} \cdot \overline{y_2}$$

$$Y_1 = \overline{I} + \overline{S} \cdot y_2$$

For the outputs, $B_1 = y_2$ and $B_2 = y_1$. Finally, the state machine is implemented using D Flip Flops:

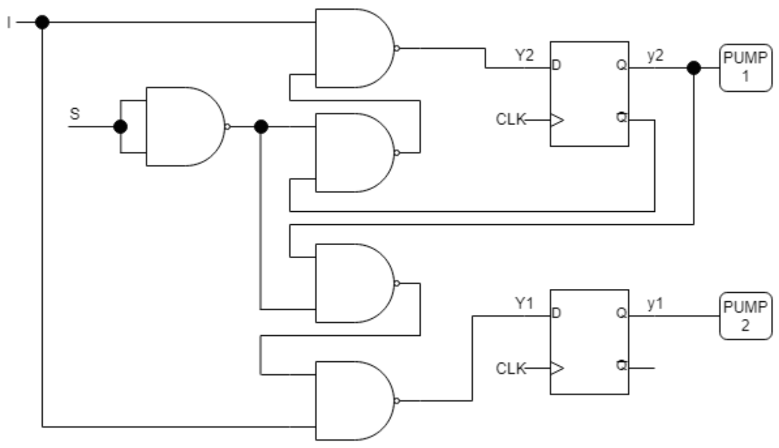


Figure 3: Moore state machine - Circuit implementation

On the other side, the same system is implemented now using a Mealy state machine, as shown below.

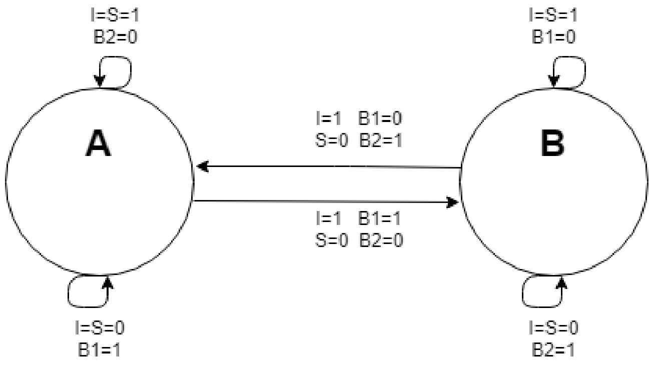


Figure 4: Mealy state machine diagram

Notice that the direct connection between the input and the pumps outputs reduces the number of states from four to two, in comparison with the Moore machine. Using one bit for the states, a table of transitions is made as follows.

Estado Actual		Estado Siguiente				Salidas			
	y	S - I				S - I			
		00	01	10	11	00	01	10	11
		y				B1 - B2	B1 - B2	B1 - B2	B1 - B2
A	0	A	B	A	X	11	10	XX	00
B	1	B	A	B	X	11	01	XX	00

Figure 5: Mealy state machine - Transitions

Using again Karnaugh's maps (see resolution in *Annex*), the resulting functions for the state variable and the two pumps outputs are shown below.

$$Y = \overline{y}\cdot\overline{S}\cdot I + y\cdot\overline{I} + y\cdot S$$
$$B_1 = \overline{y}\cdot\overline{S} + \overline{S}\cdot\overline{I}$$
$$B_2 = \overline{S}\cdot\overline{I} + y\cdot\overline{S}$$

Finally, the state machine is implemented using one D Flip Flop.

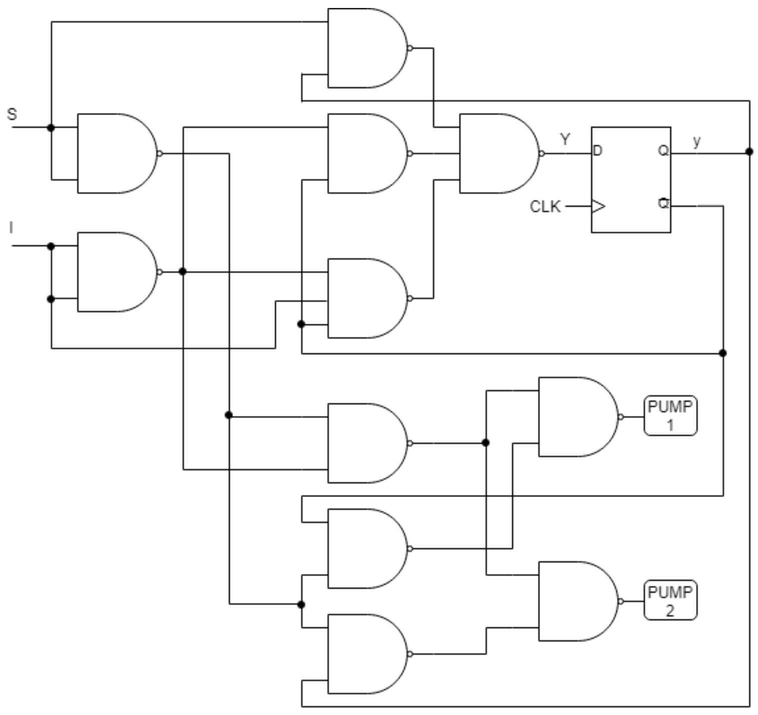


Figure 6: Mealy state machine - Circuit implementation

Verilog tests

This task was tested using verilog tests benchs. Here we show the results of simulating Moore's and Mealy's implementations. For extra support, it was also simulated on Proteus.

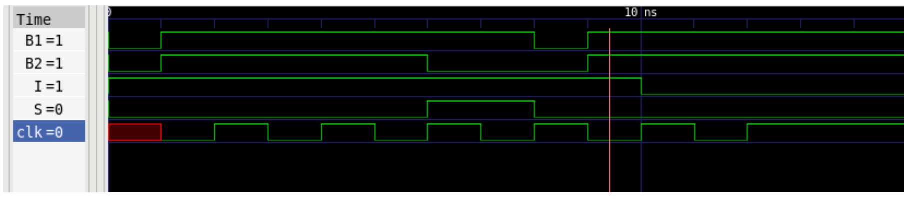


Figure 7: Moore state machine - Verilog test results

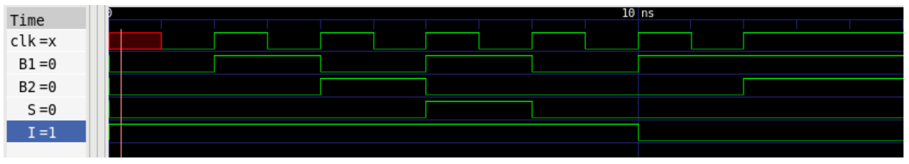


Figure 8: Mealy state machine - Verilog test results

Task 2

In this section, the objetive is to recognize a sequence of 4 bits that come in a synchronized way. If the sequence is recognized, an output is turned on. Using a Moore's state machine, the resulting diagram is as shown below.

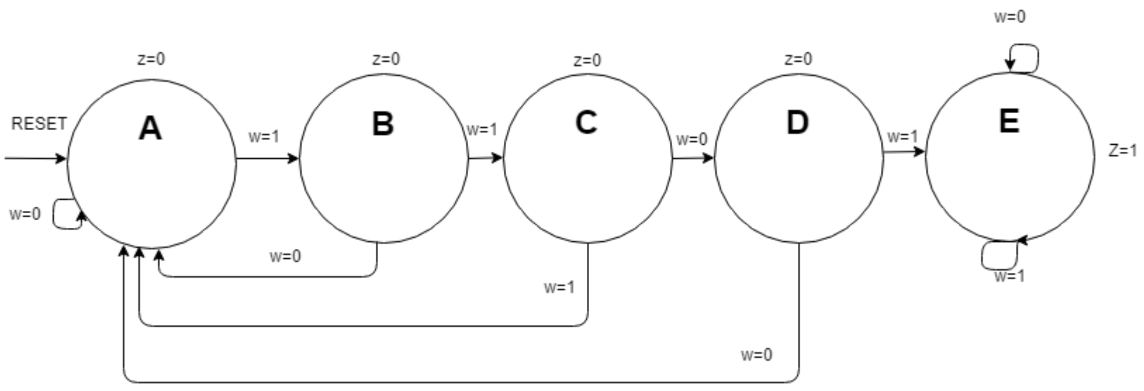


Figure 9: Moore state machine diagram

Notice that when the sequence is recognized, the machine needs to be reseted to detect a new combination. With the diagram, the following transition table is made.

Estado Actual		Estado Siguiente		Salidas
	y3 - y2 - y1	W		Z
		0	1	
A	000	A	B	0
B	001	A	C	0
C	010	D	A	0
D	011	A	E	0
E	100	E	E	1

Figure 10: Moore state machine - Transitions

Using Karnaugh's maps (see resolution in *Annex*), the functions for the diferent states and the output are as follows:

$$Y_3 = y_3 + y_2 \cdot y_1 \cdot W$$

$$Y_2 = y_2 \cdot \overline{y_1} \cdot \overline{W} + \overline{y_2} \cdot y_1 \cdot W$$

$$Y_1 = \overline{y_3} \cdot \overline{y_2} \cdot \overline{y_1} \cdot W + y_2 \cdot \overline{y_1} \cdot \overline{W}$$

From the table of transitions, $Z = y_3$.

With the functions, the state machine is implemented using three D Flip Flops as shown below.

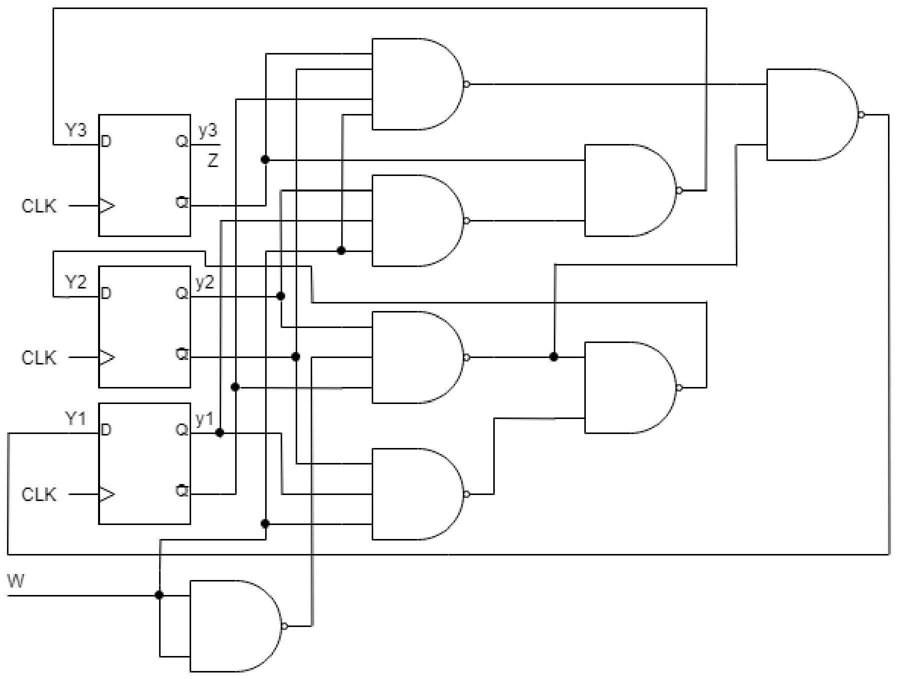


Figure 11: Moore state machine - Circuit implementation

Now, the same system is implemented using a Mealy's state machine, wich resulting diagram is shown below.

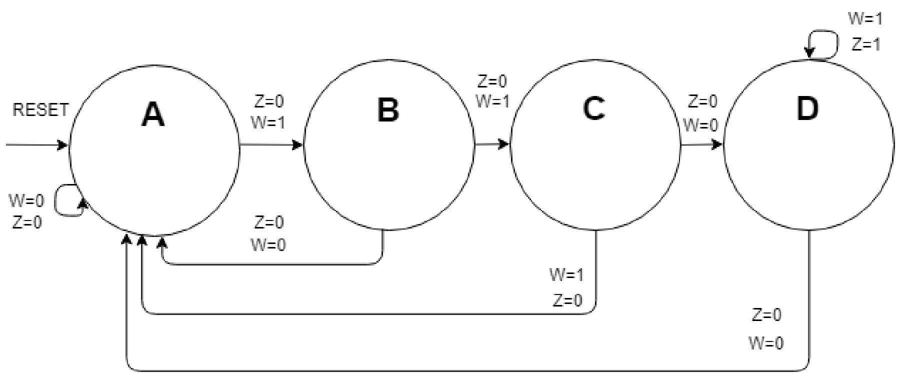


Figure 12: Mealy state machine diagram

Using the diagram, a table with the state transitions is made.

Estado Actual		Estado Siguiente		Salidas	
	y2 - y1	W		Z	
		0	1	W=0	W=1
A	00	A	B	0	0
B	01	A	C	0	0
C	10	D	A	0	0
D	11	A	D	0	1

Figure 13: Mealy state machine - Transitions

Using Karnaugh's maps (see resolution in *Annex*), the functions for the states and the output are as below.

$$Y_2 = W \cdot y_1 + \overline{W} \cdot y_2 \cdot \overline{y_1}$$

$$Y_1 = W \cdot \overline{y_2} \cdot \overline{y_1} + W \cdot y_2 \cdot y_1 + \overline{W} \cdot y_2 \cdot \overline{y_1}$$

$$Z = W \cdot y_2 \cdot y_1$$

With the defined functions, the state machine is implemented with 2 D Flip Flops. In this case is used one less flip flop, and the machine can used again without reset it.

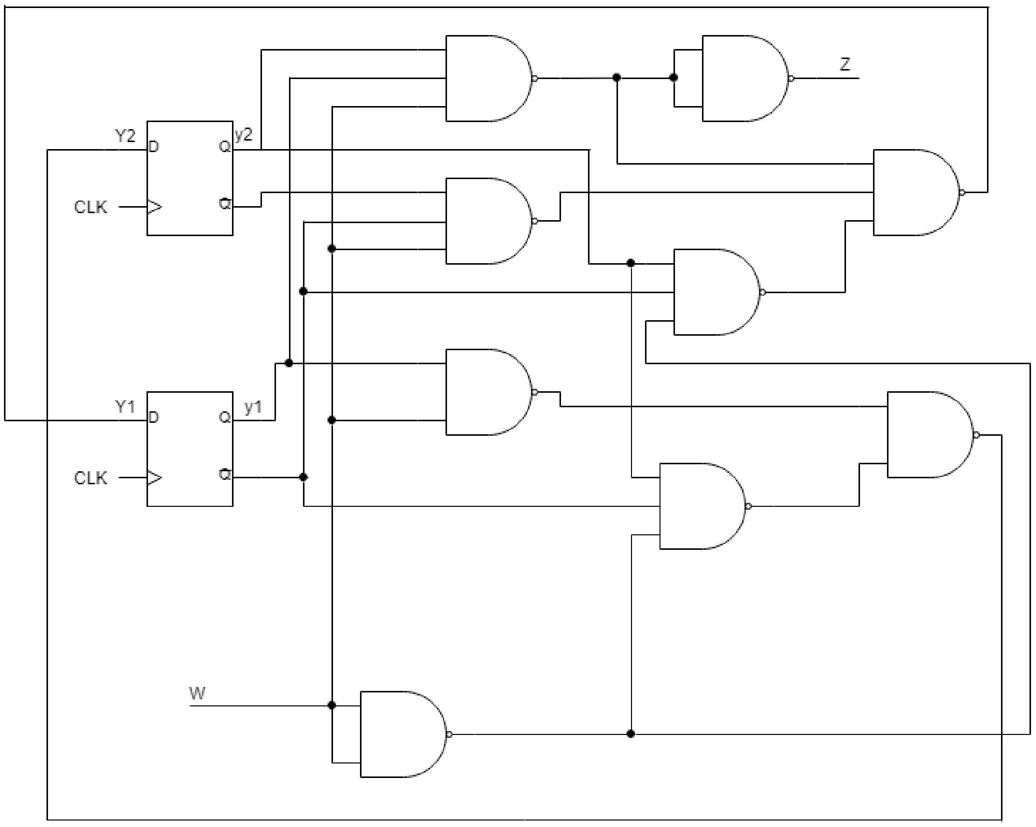


Figure 14: Mealy state machine - Circuit implementation

Verilog tests

This task was tested using verilog tests. Here we show the results of Moore and Mealy's implementations. For extra support, it was always simulated on Proteus.

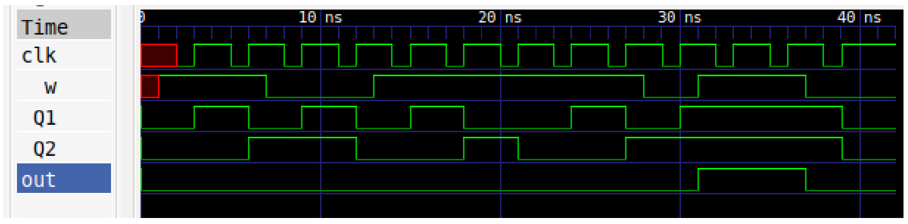


Figure 15: Moore state machine - Verilog test results

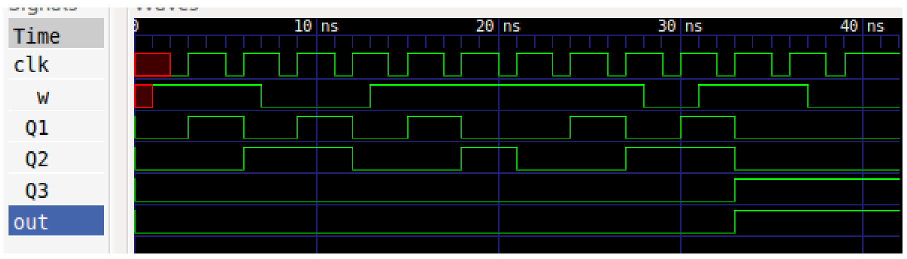


Figure 16: Mealy state machine - Verilog test results

Task 3

In this section, is implemented a Moore's state machine already defined, as shown below.

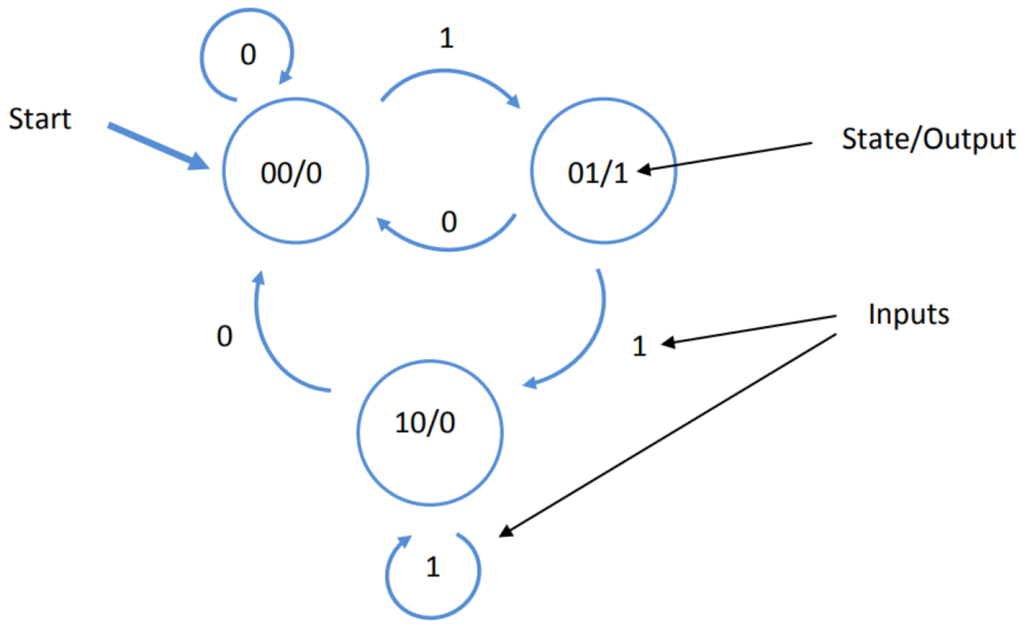


Figure 17: Moore state machine diagram

Using the diagram, the following table of transitions is made.

Estado Actual		Estado Siguiente		Salidas
	y2 - y1	W		Z
		0	1	
A	00	A	B	0
B	01	A	C	1
C	10	A	C	0

Figure 18: Moore state machine - Transitions

With the transitions, using Karnaugh's maps (see resolution in *Annex*), the functions for the states and the output result as follows: $Y_2 = W \cdot y_1 + W \cdot y_2$, and $Y_1 = W \cdot \overline{y_2} \cdot \overline{y_1}$.

From the transitions table, it is simple to see that $Z = y_1$. With the functions, the state machine is implemented using two D Flip Flops as follows.

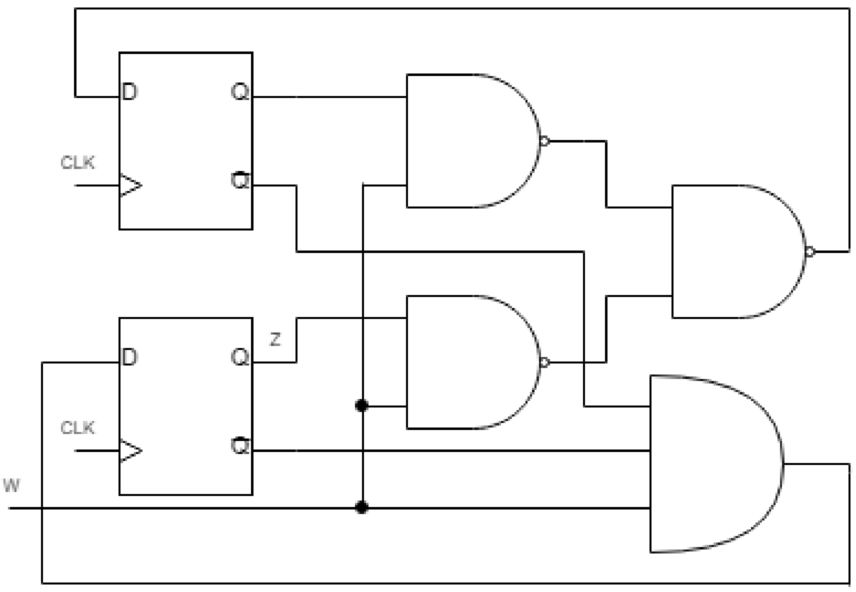


Figure 19: Moore state machine - Circuit implementation

Now the same system is implemented using a Mealy's state machine, wich resulting diagram is shown below.

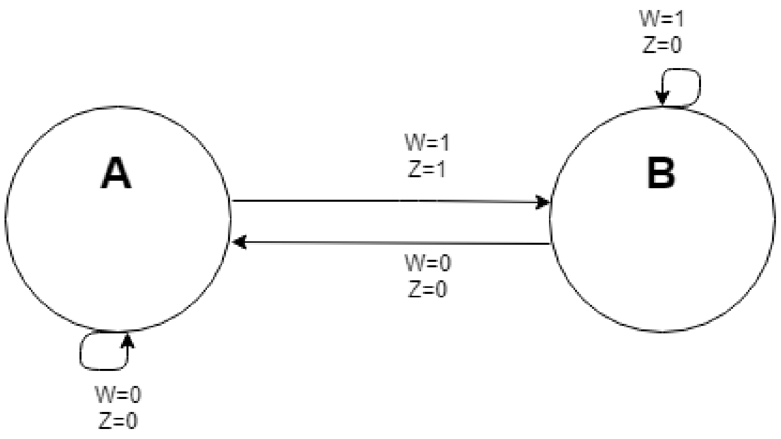


Figure 20: Mealy state machine diagram

Notice that it requires one less state than Moore's machine because of the direct connection of the from the input to the output. The following transition table is made using the diagram.

Estado Actual		Estado Siguiente		Salidas	
	y	W		Z	
		0	1	W=0	W=1
A	0	A	B	0	1
B	1	A	B	0	0

Figure 21: Mealy state machine - Transitions

With the table, using Karnaugh's maps (see resolution in *Annex*), are made the functions for the states and the output as follows: $Y = W$, and $Z = \overline{y} \cdot W$. With the defined functions, the state machine is implemented using one D Flip Flop as shown below.

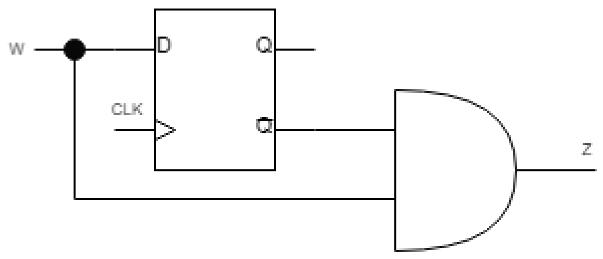


Figure 22: Mealy state machine - Circuit implementation

Since the internal logic works with 3.3V power supply, and the external signals work with 5V, level shifters are implemented using BJT transistors. For adapting the inputs of CLK and W, the circuits are shown below.

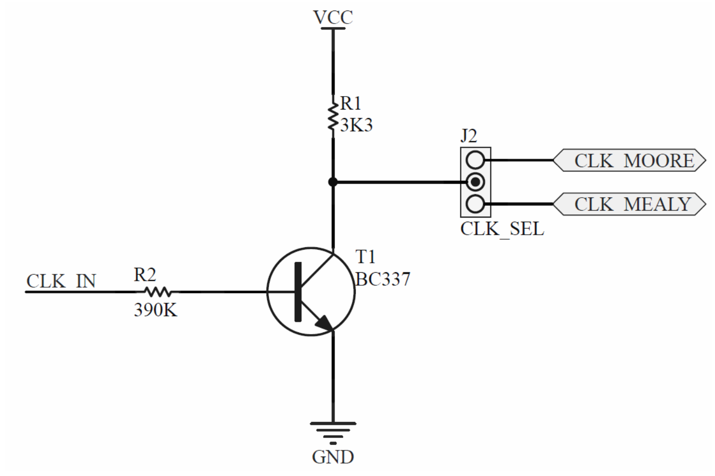


Figure 23: Level shifter for CLK from 5V to 3.3V (VCC)

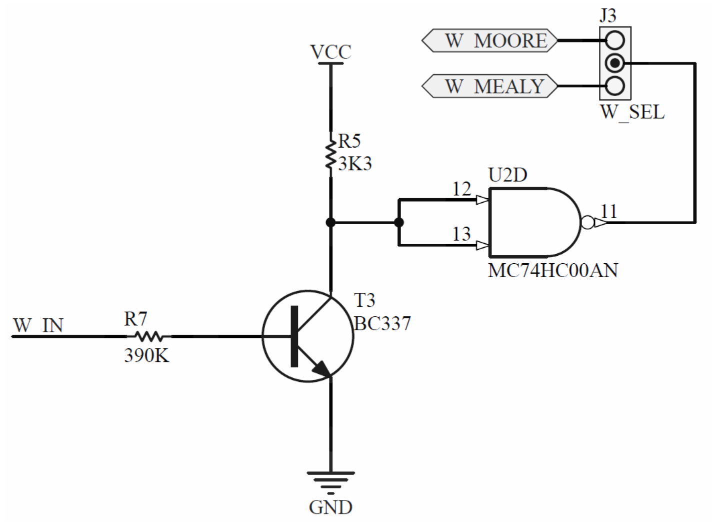


Figure 24: Level shifter for W from 5V to 3.3V (VCC). The inverting gate is to compense the transistor logic inversion.

And for the outputs (Moore and Mealy machines) the driver circuit is as shown below.

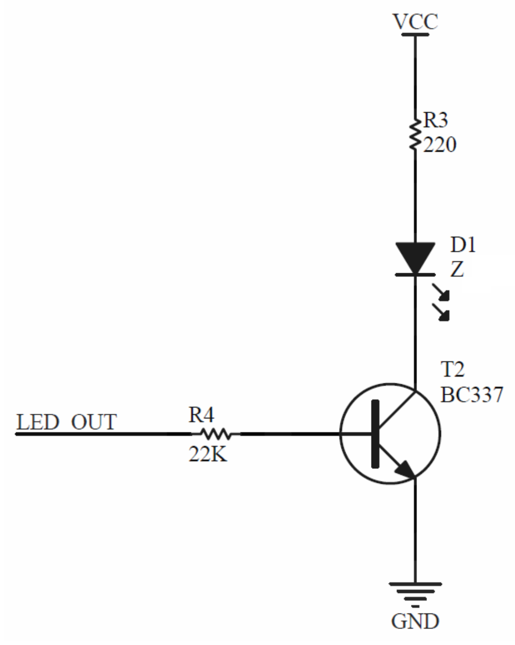


Figure 25: Driver for LED output.

The design of the circuit is described in the *Annex*.

Verilog tests

This task was tested using verilog tests. Here we show the results of Moore and Mealy's implementations. For extra support, it was always simulated on Proteus.

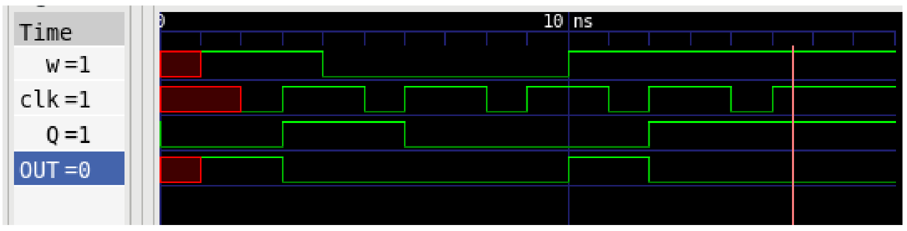


Figure 26: Moore state machine - Verilog test results

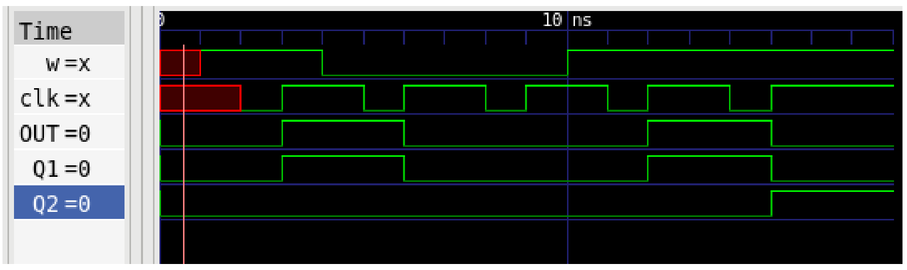


Figure 27: Mealy state machine - Verilog test results

Annex

Karnaugh's maps for Task 1

For the Moore's state machine implementation:

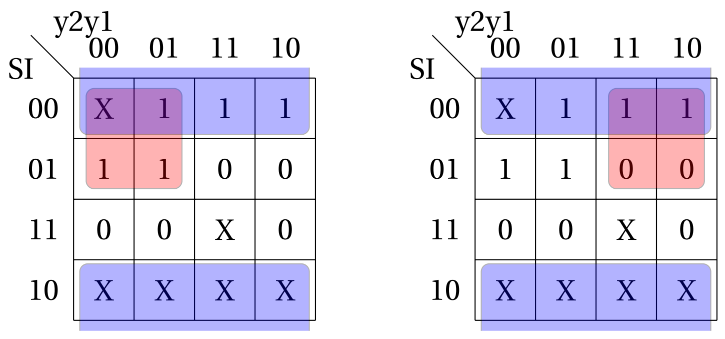


Figure 28: Maps for Y_2 (left) and Y_1 (right) functions.

Where from the left table $Y_2 = \overline{I} + \overline{S} \cdot \overline{y_2}$, and from the right $Y_1 = \overline{I} + \overline{S} \cdot y_2$. And for the outputs, $B_1 = y_2$ and $B_2 = y_1$.

For the Mealy's state machine implementation:

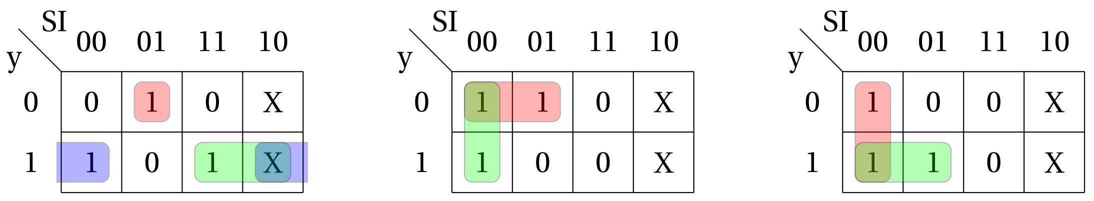
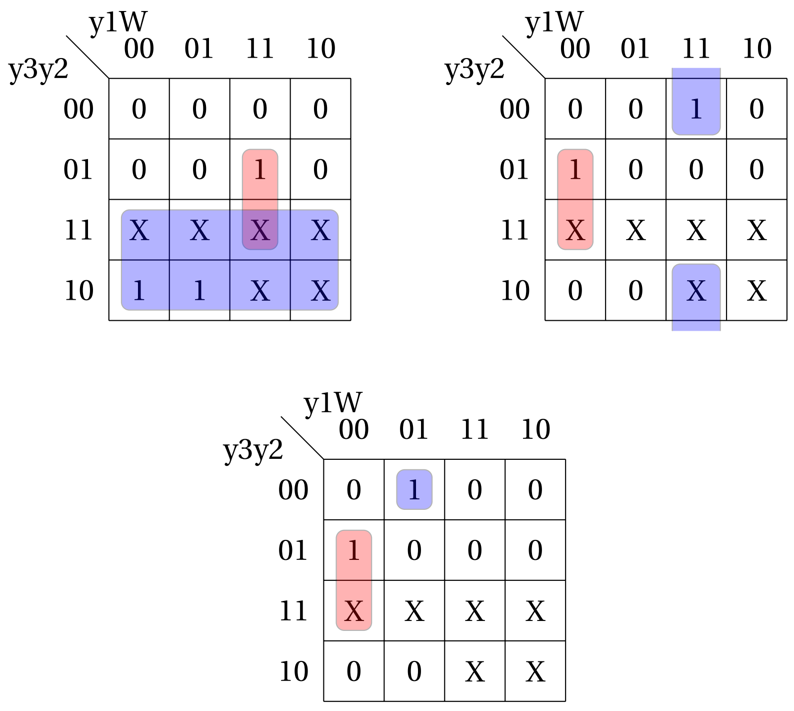


Figure 29: Maps for Y (left), B_1 (center) and B_2 (right) functions.

Where from the left table $Y = \overline{y} \cdot \overline{S} \cdot I + y \cdot \overline{I} + y \cdot S$. From the center table $B_1 = \overline{y} \cdot \overline{S} + \overline{S} \cdot \overline{I}$, and from the right table $B_2 = \overline{S} \cdot \overline{I} + y \cdot \overline{S}$.

Karnaugh's maps for Task 2

For the Moore's state machine implementation:



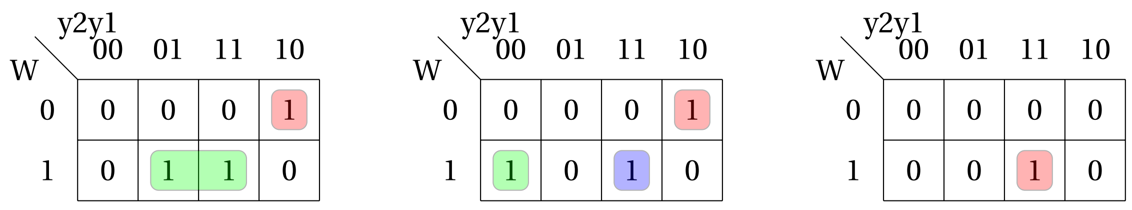
y3y2 \ y1W	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	X	X	X	X
10	1	1	X	X

y3y2 \ y1W	00	01	11	10
00	0	0	1	0
01	1	0	0	0
11	X	X	X	X
10	0	0	X	X

y3y2 \ y1W	00	01	11	10
00	0	1	0	0
01	1	0	0	0
11	X	X	X	X
10	0	0	X	X

Figure 30: Maps for Y_3 (left), Y_2 (right), and Y_1 (center) functions.

Where $Y_3 = y_3 + y_2 \cdot y_1 \cdot W$, $Y_2 = y_2 \cdot \overline{y_1} \cdot \overline{W} + \overline{y_2} \cdot y_1 \cdot W$, and $Y_1 = \overline{y_3} \cdot \overline{y_2} \cdot \overline{y_1} \cdot W + y_2 \cdot \overline{y_1} \cdot \overline{W}$. From the table of transitions, $Z = y_3$. And for the Mealy's state machine implementation:



W \ y2y1	00	01	11	10
0	0	0	0	1
1	0	1	1	0

W \ y2y1	00	01	11	10
0	0	0	0	1
1	1	0	1	0

W \ y2y1	00	01	11	10
0	0	0	0	0
1	0	0	1	0

Figure 31: Maps for Y_2 (left), Y_1 (center) and Z (right) functions.

Where $Y_2 = W \cdot y_1 + \overline{W} \cdot y_2 \cdot \overline{y_1}$, $Y_1 = W \cdot \overline{y_2} \cdot \overline{y_1} + W \cdot y_2 \cdot y_1 + \overline{W} \cdot y_2 \cdot \overline{y_1}$, and $Z = W \cdot y_2 \cdot y_1$.

Karnaugh's maps for Task 3

For the Moore's state machine implementation:

W \ y2y1	00	01	11	10
0	0	0	X	0
1	0	1	X	1

W \ y2y1	00	01	11	10
0	0	0	0	1
1	1	0	1	0

Figure 32: Maps for Y_2 (left) and Y_1 (right) functions.

Where $Y_2 = W \cdot y_1 + W \cdot y_2$, and $Y_1 = W \cdot \overline{y_2} \cdot \overline{y_1}$. From the transitions table, it is simple to see that $Z = y_1$.

And for the Mealy's state machine implementation:

y \ W	0	1
0	0	1
1	0	1

y \ W	0	1
0	0	1
1	0	0

Figure 33: Maps for Y (left) and Z (right)

Where from the left map $Y = W$, and from the right table $Z = \overline{y} \cdot W$.

Level shifter for inputs

From the implemented circuit:

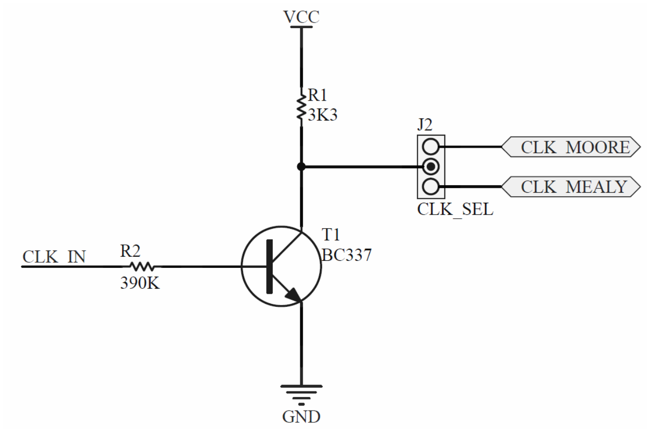


Figure 34: Level shifter for CLK from 5V to 3.3V (VCC)

Usign for $I_{SAT} = 1mA$, considerating $VCE_{SAT} = 0.2V$, the equation from the out mesh:

$$3.3V - VCE_{SAT} - I_{SAT}R_1 = 0$$
$$\frac{3.3V - VCE_{SAT}}{I_{SAT}} = R_1 = 3.1K\Omega$$

Normalizing we have $R_1 = 3.3K\Omega$. Considering $HFE_{MIN} = 100$, from the input mesh:

$$5V - VBE_{ON} - \frac{I_C}{HFE_{MIN}}R_2 = 0$$
$$\frac{5V - VBE_{ON}}{I_C}HFE_{MIN} = R_2 = 430K\Omega$$

Normalizing we have $R_2 = 390K\Omega$. The same circuit is for adapting W input.

Driver for output led

Taking the implemented circuit:

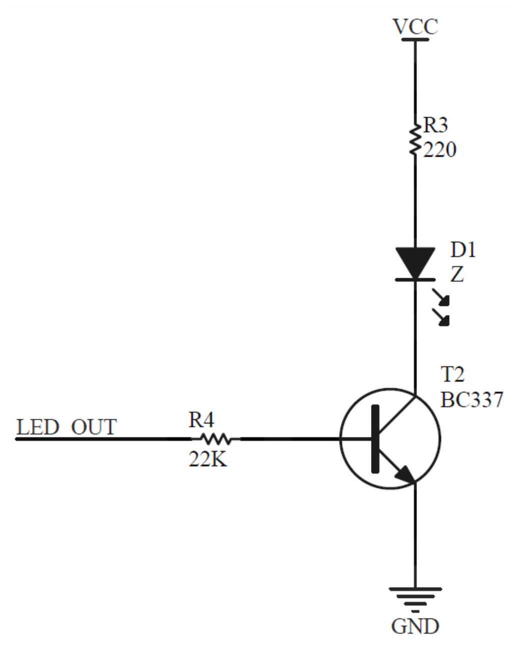


Figure 35: Driver for LED output.

Usign for $I_{LED} = 10mA$, considerating $VCE_{SAT} = 0.2V$ and $VLED = 2V$, the equation from the out mesh:

$$3.3V - V_{LED} - VCE_{SAT} - I_{LED}R_3 = 0$$
$$\frac{3.3V - V_{LED} - VCE_{SAT}}{I_{LED}} = R_3 = 110\Omega$$

Normalizing we have $R_3 = 220\Omega$. Considering $HFE_{MIN} = 100$, from the input mesh:

$$3.3V - VBE_{ON} - \frac{I_C}{HFE_{MIN}} R_4 = 0$$

$$\frac{3.3V - VBE_{ON}}{I_C} HFE_{MIN} = R_4 = 26K\Omega$$

Normalizing we have $R_4 = 22K\Omega$, to guarantee saturation.