Instituto Tecnologico de Buenos Aires

Ingeniera Electronica

Electronica III

Implementación de circuitos logicos

Autores:
Martín Rodriguez Turco
Tobias Scala
Guido Panaggio
Juan Martin Laguinge

Profesores:
Kevin DeWald
Pablo Wundes
Sebastian Falconaro

15 de noviembre de 2018

Índice general

Capítulo 1

1-Control de la Activación de 2 Bombas de Agua

1.1. Objetivo

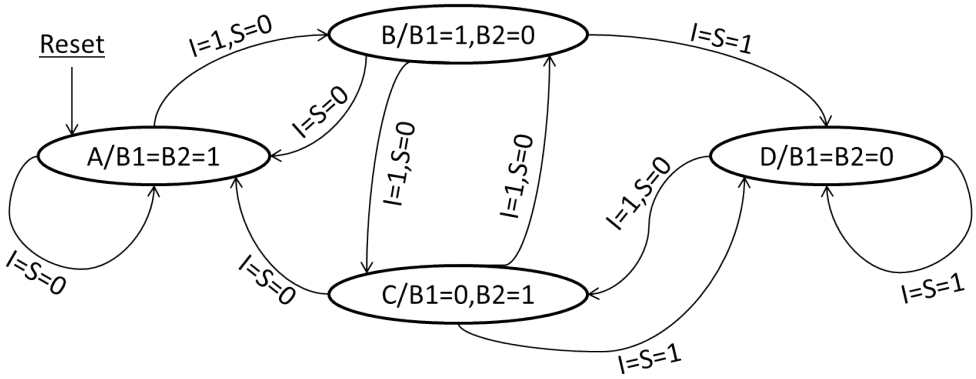
El objetivo es implementar una máquina de estados el cual controle la activación de 2 bombas de agua denominados com B1 y B2 que controlen el nivel de agua de un depósito el cual dispone de 2 sensores, uno en la parte inferior (I) y otro en la parte superior (S). Esta máquina de estados debe prender las bombas en función del nivel del agua en el depósito. Si el depósito esta vacío (I = S = 0), entonces se prenden las 2 bombas. Si el depósito esta lleno por la mitad (I = 1, S = 0), entonces las bombas se alternan en su trabajo. Si el depósito esta lleno (I = S = 1), entonces se apagan las 2 bombas.

1.2. Explicación y Desarrollo

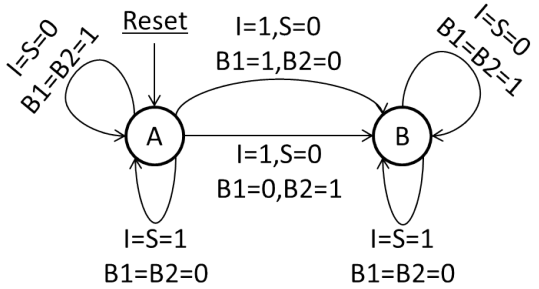
Teniéndo en cuenta las especificaciones de la máquina de estados, se procede a realizar el diagrama de estados del mismo. Sea realizará el mismo desarrollo implementando tanto una máquina de Moore como de Mealy.

1.2.1. Diagrama de Estados

El diagrama de estados para la máquina de estados de Moore queda de la siguiente forma:



y la máquina de estados de Mealy queda de la siguiente forma:



Se puede ver que la mmáquina de estados de Mealy tiene menos estados que la de Moore ya que la salida depende de la entrada además del estado.

1.2.2. Tabla de Estados

A partir de los diagramas de estados, se procede a armar las tablas de estados de cada máquina. Asignando a los estados como: A = 00, B = 01, C = 10 y D = 11, la tabla de estados de Moore queda de la siguiente forma:

Cuadro 1.1: Tabla de Estados de Moore

Present State	Next State			Output
	I=S=0	I=1,S=0	I=S=1	
y2 y1	Y2 Y1	Y2 Y1	Y2 Y1	B2 B1
0 0	0 0	0 1	x x	1 1
0 1	0 0	1 0	1 1	0 1
1 0	0 0	0 1	1 1	1 0
1 1	x x	1 0	1 1	0 0

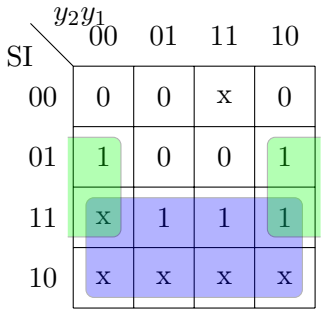
Y asignando a los estados como: A = 0 y B = 1, la tabla de estados de Mealy queda de la siguiente forma:

Cuadro 1.2: Tabla de Estados de Mealy

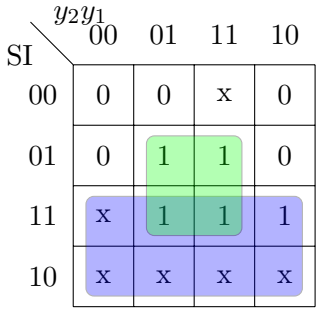
Present State	Next State			Output		
	I=S=0	I=1,S=0	I=S=1	I=S=0	I=1,S=0	I=S=1
y	Y	Y	Y	B2 B1	B2 B1	B2 B1
0	0	1	0	1 1	0 1	0 0
1	1	0	1	1 1	1 0	0 0

1.2.3. Mapas de Karnaugh

A continuación se procede a obtener las ecuaciones lógicas de cada tabla de estados. Empezando con la tabla de estados de Moore, se obtiene:
Para Y1:



Donde se extrae que Y1 = S + y1*I.
Para Y2:



Donde se extrae que Y2 = S + $\overline{y1}$*I.
Para B1:

y_2 \ y_1	0	1
0	1	1
1	0	0

Donde se extrae que B1 = $\overline{y2}$.
Para B2:

y_2 \ y_1	0	1
0	1	0
1	1	0

Donde se extrae que B2 = $\overline{y1}$.
Y con estas ecuaciones ya es posible diseñar el circuito de Moore. Ahora se obtendrán las ecuaciones lógicas a partir de la tabla de estados de Mealy:

Para Y:

y \ SI	00	01	11	10
0	0	1	0	x
1	1	0	1	x

Donde se extrae que Y = $\overline{S}$*I*$\overline{y}$ + y*(S + $\overline{I}$).
Para B1:

y \ SI	00	01	11	10
0	1	1	0	x
1	1	0	0	x

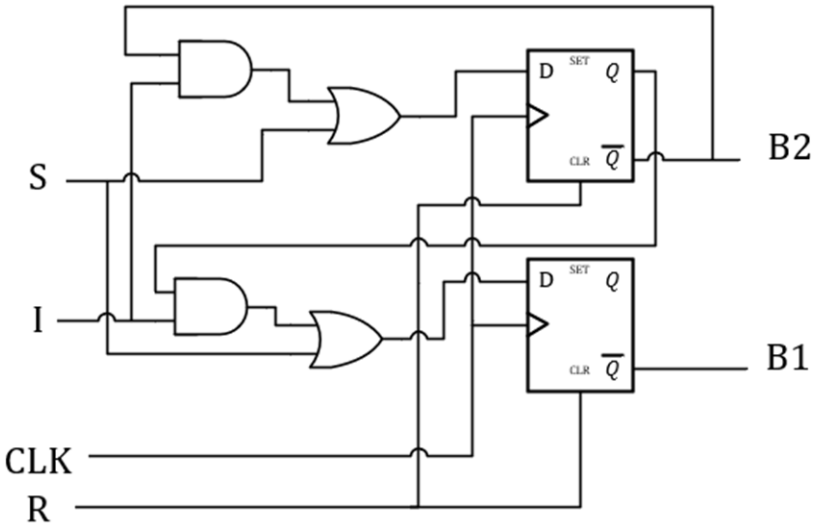
Donde se extrae que B1 = $\overline{y}$*$\overline{s}$ + $\overline{I}$.
Para B2:

y \ SI	00	01	11	10
0	1	0	0	x
1	1	1	0	x

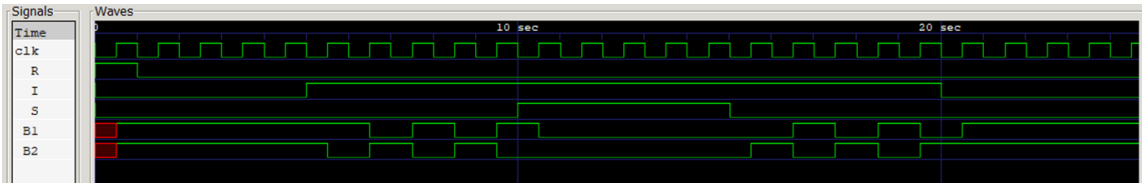
Donde se extrae que B2 = y*$\overline{s}$ + $\overline{I}$. Y con estas ecuaciones ya es posible diseñar el circuito de Mealy.

1.3. Circuitos Lógicos y Simulación en GTKWave

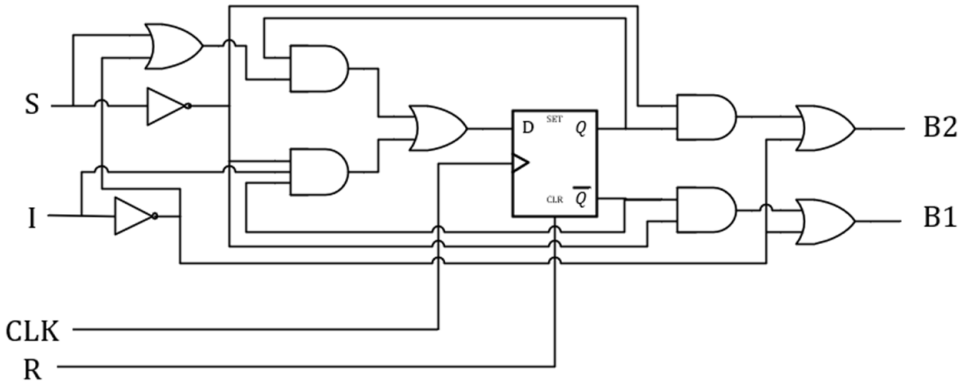
Como ya se tiene lo necesario para diseñar y simular los circuitos, se los diseña quedando de la siguiente forma: Circuito utilizando la máquina de estados de Moore:



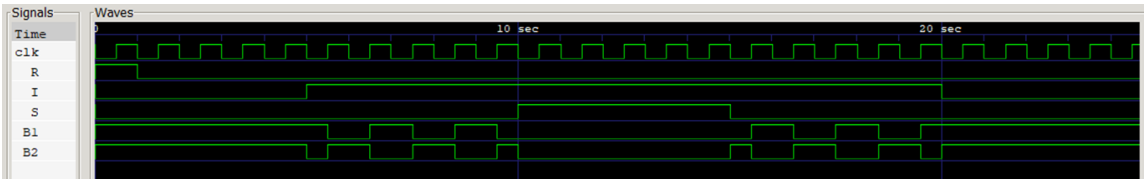
Al simular el circuito en GTKWave se obtiene lo siguiente:



Circuito utilizando la máquina de estados de Mealy:



Al simular el circuito en GTKWave se obtiene lo siguiente:



La gran diferencia que se puede observar en la simulación es que la salidas están desplasadas entre sí exactamente un ciclo de clock. Eso se debe a que el circuito el cual fue implimentado con Mealy responde más rápido ya que su salida depende tanto de la entrada como del estado. En cambio el circuito el cual fue implementado con Moore solo depende del estado.

Capítulo 2

Ejercicio 2

Se desea diseñar una máquina de estados que, al recibir la siguiente secuencia de bits en forma sincrónica 1-1-0-1 encienda una salida y en caso contrario, la mantiene apagada. Se obtienen 5 estados para la misma, en los cuales va a haber un default que va a ser el estado al cual todos los demás estados van a volver en caso de no recibir los deseados además de ser el estado por el cual va a empezar la máquina de estados.
Podemos representar los mismos en el siguiente diagrama de estados:
En donde Z es la salida dada por la máquina de estados al encontrarse en el estado corres-

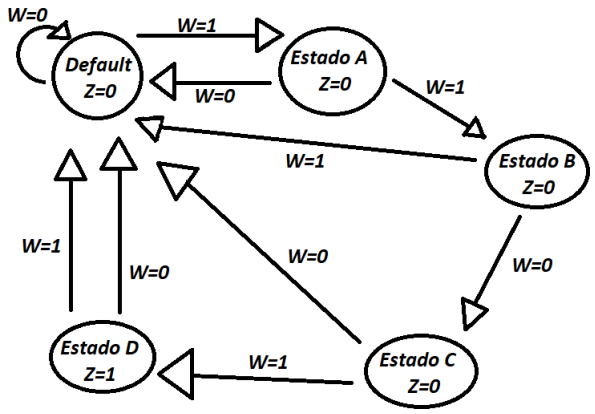


Figura 2.1: Diagrama de estados

pondiente y W es la entrada necesaria para que transicione al siguiente estado y la flecha es la encargada de indicar el sentido de la transición.
En este mismo esquema también queda encapsulado en la siguiente tabla de estados:

Cuadro 2.1: Tabla de estados

Estado actual	Estado siguiente **W=0**	**W=1**	**Salida Z**
Default	**Default**	**A**	0
A	**Default**	**B**	0
B	**C**	**Default**	0
C	**Default**	**D**	0
D	**Default**	**Default**	1

Para la implementación de esté falta realizar la asignación de valores de estado, lo cual nos lleva cambiar la tabla anterior por la siguiente:

Cuadro 2.2: Tabla de estados asignados

Estado actual	**Asignacion del Estado actual**	Estado siguiente **W=0**	**W=1**	**Salida Z**
Default	000	000	001	0
A	001	000	010	0
B	010	011	000	0
C	011	000	100	0
D	100	000	000	1

2.1. Máquina de Moore

Para nuestro caso tenemos el siguiente circuito secuencial genérico:

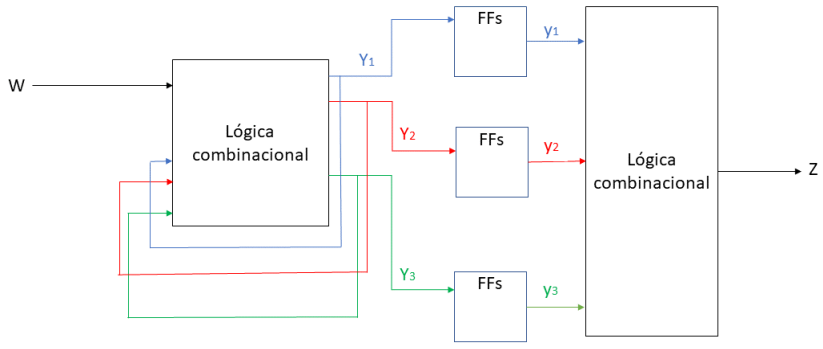


Figura 2.2: Circuito generico

Donde vamos a utilizar Flip-Flops D dado que la entrada D de estos va a corresponder

con el estado siguiente Y_i y van a estar seteados por el clock para que esta salida luego cambia la variable y_i a Y_i, dado que y_i son las variables de estado actual.De la tabla 2, obtenemos los siguientes mapas de Karnaugh:

Figura 2.3: Mapa de Karnaugh para Y_1

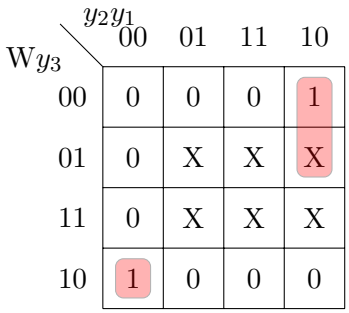


Figura 2.4: Mapa de Karnaugh para Y_2

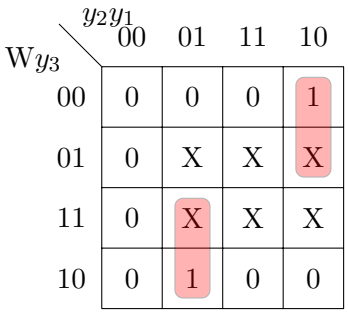


Figura 2.5: Mapa de Karnaugh para Y_3

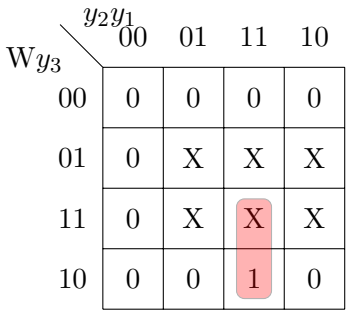
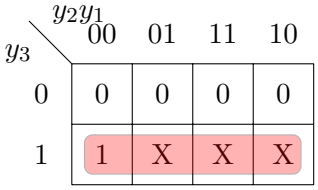


Figura 2.6: Mapa de Karnaugh para Z



En donde las X representan los don't care y se les decidió dar un valor acorde al cual permiten la simplificación del circuito. Dando como resultado las siguientes ecuaciones:

$$Y_1 = W \cdot \overline{y_3} \cdot \overline{y_2} \cdot \overline{y_1} + \overline{W} \cdot \overline{y_1} \cdot y_2$$
$$Y_2 = W \cdot \overline{y_2} \cdot y_1 + \overline{W} \cdot \overline{y_1} \cdot y_2$$
$$Y_3 = W \cdot y_2 \cdot y_1$$
$$Z = y_3$$

Se realizo la correspondiente simulación en verilog, el cual nos da un comportamiento ideal del circuito, obteniendo el siguiente resultado:

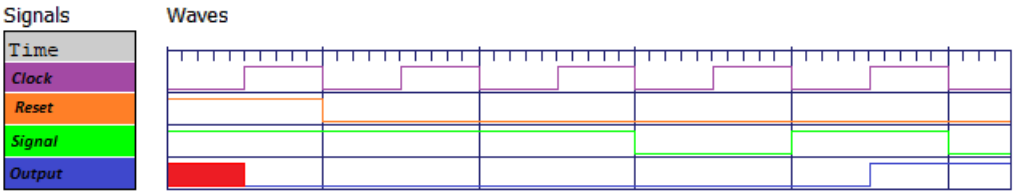


Figura 2.7: Simulación

Luego si se quiere observar el comportamiento del diseño, el cual coincidió con el previsto por la simulación, solo necesita ir al siguiente enlace:
`https://www.youtube.com/watch?v=Zg6114gPW9U&feature=youtu.be`

2.2. Máquina de Mealy

Para poder realizar la máquina de estados con el modelo de Mealy es necesario que la salida dependa tanto de los estados como de la entrada de esta, con lo cual van a ser necesarios realizar cambios a la actual máquina de estados.
Quitando el último estado del diagrama 2 y expresando la salida junto con la entrada podemos obtener el siguiente diagrama:

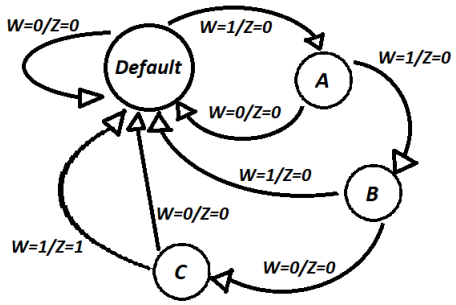


Figura 2.8: Diagrama de estados

El diagrama tiene un estado menos ahorra, esto provoca los siguientes cambio en la tabla de asignación:

Cuadro 2.3: Tabla de estados asignados

Estado actual	**Asignacion del Estado actual**	Estado siguiente		Salida Z	
		W=0	**W=1**	**W=0**	**W=1**
Default	00	00	01	0	0
A	01	00	10	0	0
B	10	11	00	0	0
C	11	00	00	0	1

La cual nos permite obtener los siguientes mapas de Karnaugh:

Figura 2.9: Mapa de Karnaugh para Y_1

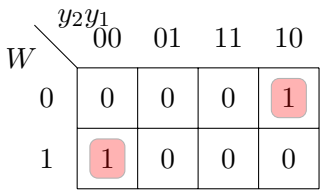


Figura 2.10: Mapa de Karnaugh para Y_2

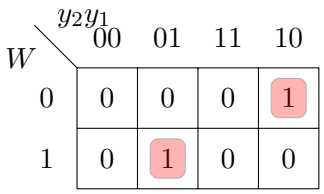
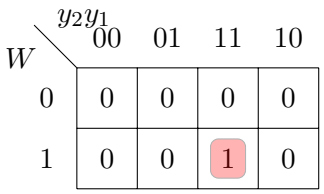


Figura 2.11: Mapa de Karnaugh para Z



Dando como resultado de las simplificaciones las siguientes ecuaciones:

$$Y_1 = W \cdot \overline{y_2} \cdot \overline{y_1} + \overline{W} \cdot \overline{y_1} \cdot y_2$$
$$Y_2 = W \cdot \overline{y_2} \cdot y_1 + \overline{W} \cdot \overline{y_1} \cdot y_2$$
$$Z = W \cdot y_2 \cdot y_1$$

Se realizo la correspondiente simulación en verilog, el cual nos da un comportamiento ideal del circuito, obteniendo el siguiente resultado:

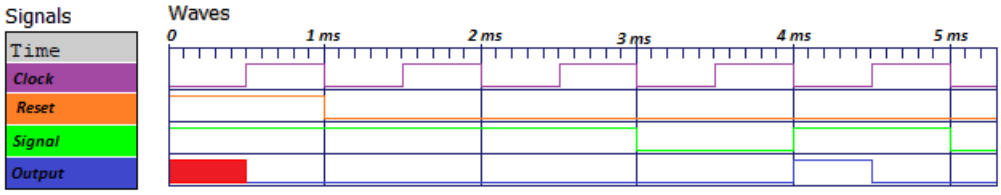


Figura 2.12: Simulación

Luego si se quiere observar el comportamiento del diseño, el cual coincidió con el previsto por la simulación, solo necesita ir al siguiente enlace:
`https://www.youtube.com/watch?v=OcXzGi5TAnY&feature=youtu.be`

Capítulo 3

Ejercicio 3

3.1. Consigna

En este ejercicio se implementará una maquina de estados de Moore siguiendo lo solicitado por el trabajo y luego se hará una máquina de estados equivalente en su versión Mealy. Esta máquina es la que se muestra a continuación:

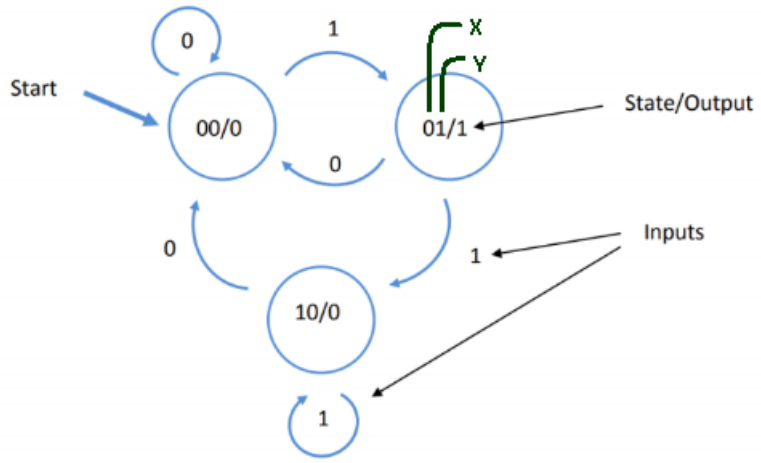


Figura 3.1: Maquina de estados solicitada

3.2. Maquina de estado de Moore

Como se sabe las maquinas de estado de Moore se caracterizan por tener una salida unicamente dependiente del estado actual del sistema, sin depender de forma instantánea de la entrada.

Como primera instancia para resolución de esta máquina se define que valores para los bits denominados como X e Y , ilustrados en la figura anterior, caracterizan a cada uno de los estados, se puede observar ademas que cuando ambos de estos bits valen 1 la maquina no se encuentra en ningún estado relevante por lo que este se tratara como un estado de *don´t care*. En la siguiente tabla se resume todo esto:

Estados	Condiciones	
	X	Y
A	0	0
B	0	1
C	1	0
don´t care	1	1

Figura 3.2: Condiciones de estados

Una vez definido que valores identifican a cada uno de los estados lo que sigue es recopilar la información sobre como evoluciona el sistema según las entradas a este y como también cada estado determina la salida obtenida del sistema, todo esto se resume en la siguiente tabla:

Estado Actual		Estado futuro				Output
x	y	input = 0		input = 1		
		X	Y	X	Y	
0	0	0	0	0	1	0
0	1	0	0	1	0	1
1	0	0	0	1	0	0
1	1	*d*	*d*	*d*	*d*	*d*

Figura 3.3: State assigned table

Nótese que se utilizaran letras minúsculas para designar estados actuales y letras mayúsculas para estados futuros.

Tanto la información sobre los estados futuros como de la salida se pueden contener en una tabla de verdad teniendo como variables los estados actuales x e y y el valor de entrada:

input	x	y	X	Y
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	*d*	*d*

x	y	Output
0	0	0
0	1	1
1	0	0
1	1	*d*

Figura 3.4: Tablas de verdad

Como se puede ver queda implícito en la tabla anterior que esta será una maquina de estado de Moore ya que la salida esta unicamente determinada por el estado actual.

Teniendo ya estas tablas de verdad se procede a utilizar el método de resolución con min-terminos del mapa de Karnaugh para obtener las expresiones lógicas de X e Y y de la salida, se utilizarán los estados de *don´t care* para lograr una expresión lo mas compacta posible, se aclara que *y1* y *y2* representan a x e y respectivamente mientras que W denota el valor de entrada:

W \ y_2y_1	00	01	11	10
0	0	0	X	0
1	0	1	X	1

Figura 3.5: Mapa de Karnaugh de X

W \ y_2y_1	00	01	11	10
0	0	0	X	0
1	1	0	X	0

Figura 3.6: Mapa de Karnaugh de Y

y_2 \ y_1	0	1
0	0	0
1	1	X

Figura 3.7: Mapa de Karnaugh de Output

Las expresiones obtenidas son las siguientes:

- $X = inp \cdot x + inp \cdot y = inp \cdot (x + y)$
- $Y = inp \cdot \overline{x} \cdot \overline{y}$
- $Output = y$

Las cuales describen el comportamiento del siguiente circuito:

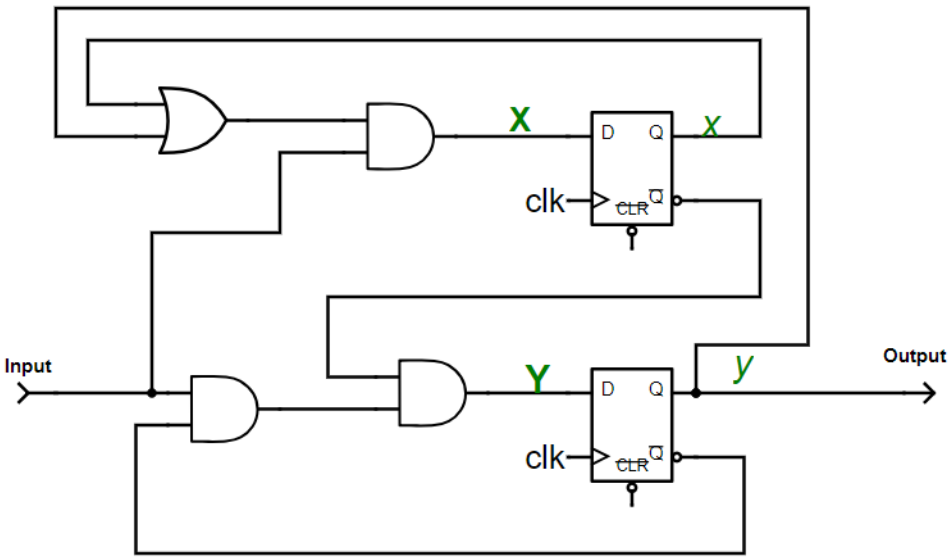


Figura 3.8: Circuito de Moore

3.3. Maquina de estado de Mealy

Mientras que la maquina de estados de Moore depende exclusivamente del estado actual del sistema la de Mealy no solo depende de este sino también del que se tiene a la entrada del sistema, de allí una de las principales diferencias con respecto a las maquinas de estado

de Moore, un cambio a la entrada se verá reflejado *instantáneamente* a la salida.

Para llegar a que la salida esté en relación directa con la entrada se analizó paso a paso el comportamiento del sistema. Se entiende que la maquina verá un 1 a su salida cuando se tenga un 1 a la entrada y simultaneamente se esté en un estado latente o inicial, una vez muestreado este valor se pasará a un nuevo estado en el cual el sistema arrojará 0 ante cualquier salida. En este estado se permanecerá mientras se siga leyendo un 1 a la entrada y se regresará al estado inicial en caso contrario. A continuación un ejemplo:

	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7
Input	0	1	0	1	1	1	0	1
Output	0	1	0	1	0	0	0	1

Figura 3.9: Ejemplo de funcionamiento de la maquina

Visto el comportamiento de la máquina se propone el siguiente esquema para este:

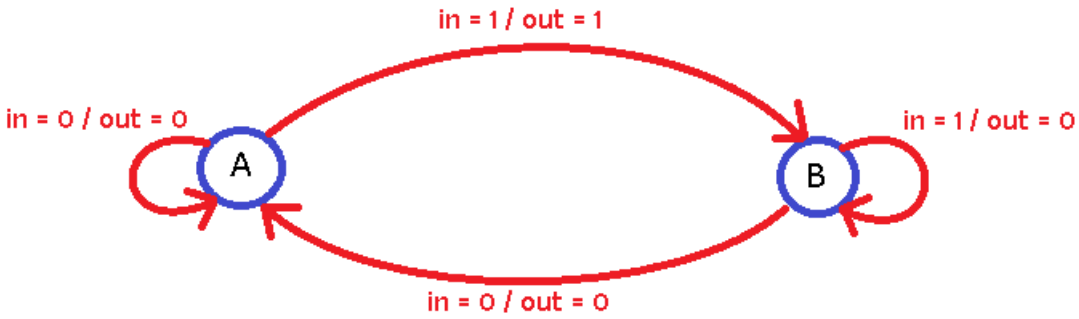


Figura 3.10: Esquema de la maquina de Mealy

Del mismo modo que antes se puede representar el funcionamiento de la maquina en la siguiente tabla:

Estado actual	Estado futuro		Output	
	input = 0	input = 1	input = 0	input = 1
A	A	B	0	1
B	A	B	0	0

Figura 3.11: State assigned table

Como se ha planteado hasta ahora solo existen 2 estados posibles por lo tanto si se considera la variable *estados* (la cual se llamara x para los presentes y X para los futuros) se puede decir que esta nueva variable posee un valor binario, 0 para el estado A y 1 para el estado B. Desde aquí se puede implementar la siguiente tabla de verdad:

x	input	X	Output
0	0	0	0
0	1	1	1
1	0	0	0
1	1	1	0

Figura 3.12: Tablas de verdad

Desde aquí es fácil observar las siguientes implicaciones lógicas:

- $X = inp$
- $Output = inp \cdot \overline{x}$

Las cuales se representan el funcionamiento del siguiente circuito:

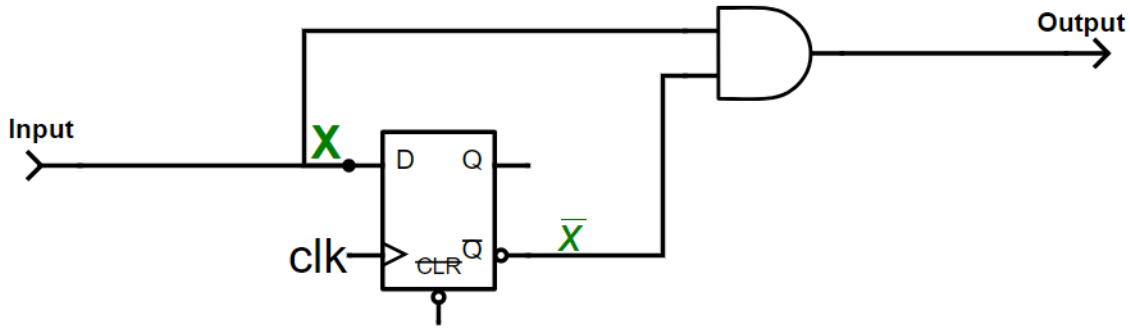


Figura 3.13: Circuito de Mealy

3.4. Conversión de tensiones

Una condición para el diseño de los circuitos es que estos reciban y devuelvan tensiones de 0 o 5V mientras que la lógica interna de esta debe operar a 3.3V, por lo que se deberá implementar tecnología que trabaje con este nivel de tensión realizando la respectiva conversión de voltaje tanto a la entrada como a la salida.

Para esto último se utilizó el circuito integrado SN7407N el cual permite realizar cómodamente una conmutación de tensión dado que se le permite al usuario operar con el colector del transistor ubicado a la salida de este, es decir, este es un dispositivo *open-colector*. Un diagrama simplificado de su funcionamiento se muestra a continuación:

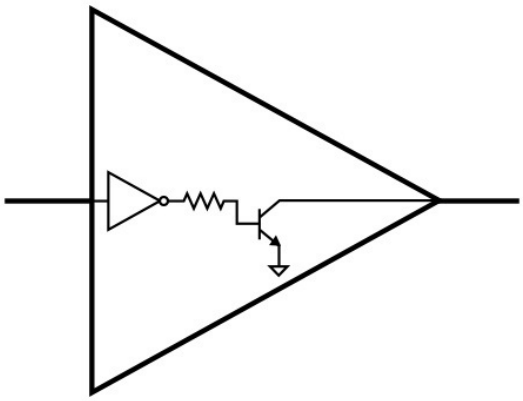


Figura 3.14: Diagrama del SN7407N

Como se puede ver un 1 lógico a la entrada dejara en alta impedancia la salida del integrado, dado que el transistor estará en estado de corte, con lo cual implementado una resistencia de pull-up sobre el colector a la salida se tendrá la tensión deseada.

3.5. Resultados obtenidos

Ya con los esquematicos y la estructura del circuito definida se procedio a realizar la implementación de esta. Finalmente el circuito se comportó conforme a lo esperado lo cual se muestra en las siguientes imagenes que se consideraron suficientemente representativas:

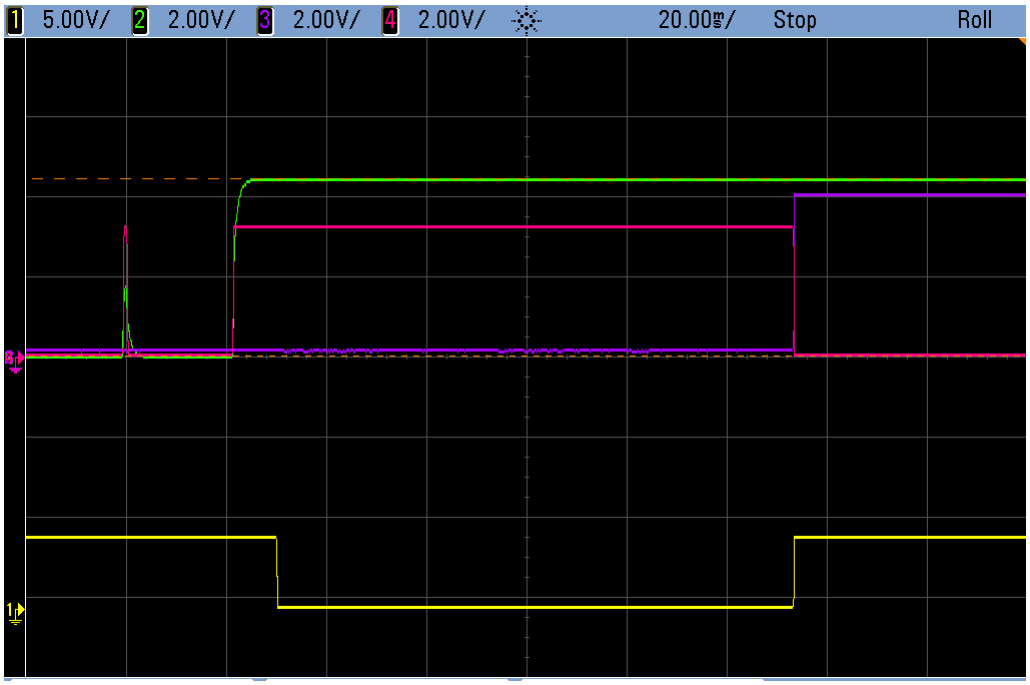


Figura 3.15: Transisión de estados

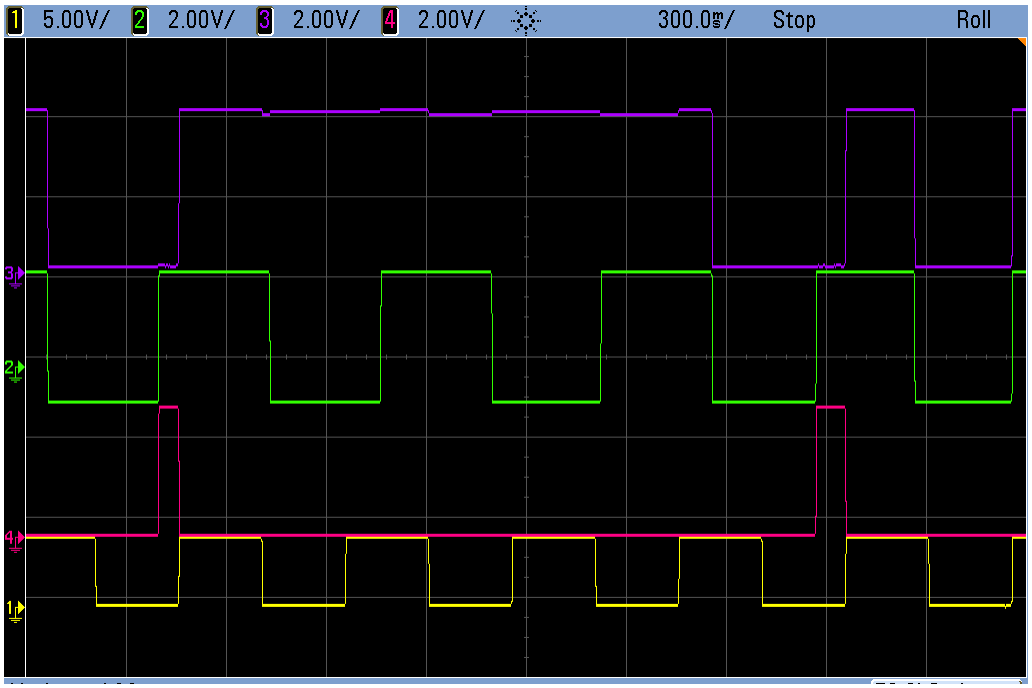


Figura 3.16: Transisión de estados

En las figuras anteriores, las cuales pertenecen al circuito de Mealy, se tiene el clock

como la señal amarilla, la señal de input como verde, la salida no negada del flip-flop como la violeta y finalmente la salida del circuito como rosada. Se observa al comnutar la entrada al valor HIGH en el intervalo de tiempo compendido este evento y el flanco positivo del clock se tiene que la salida del será también HIGH, regresando al valor LOW cuando se alcanza dicho el flanco positivo.

Appendix

Bibliografía

[1] Stephen Brown and Zvonko Vranesic. "Fundamentals of Digital Logic with Verilog Design"third edition.