1. Ejercicio 3

1.1. Consigna

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

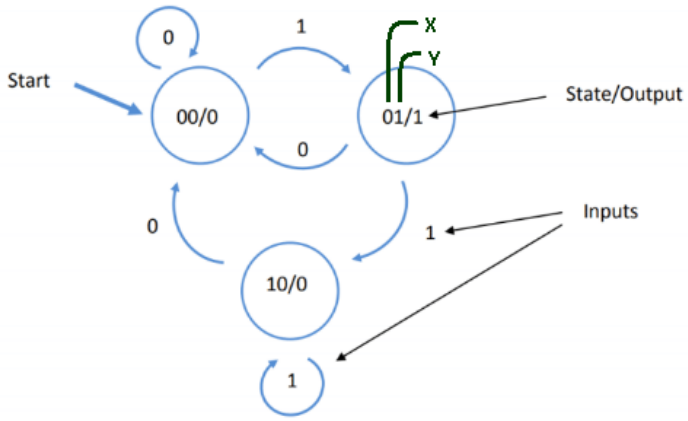

Figura 1: Maquina de estados solicitada

1.2. 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 maquina se define que valores para los bits denominados como X e Y caracterizan a cada uno de los estados, se puede observar ademas 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 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 observa 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: 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 4: Tablas de verdad

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

Teniendo ya estas tablas de verdad se procede 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, para se utilizan los estados de *don´t care* para lograr una expresión lo mas compacta posible:

Figura 5: Mapa de Karnaugh de X

Figura 6: Mapa de Karnaugh de Y

Figura 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:

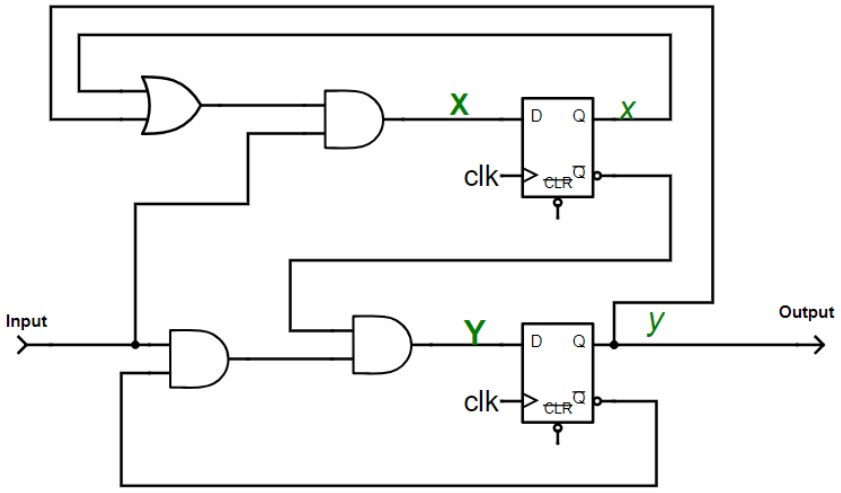

Figura 8: Circuito de Moore

1.3. 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 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 vera reflejado *instantáneamente* a la salida.

Para llegar a que la salida este en relación directa con la entrada se analizo 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 simultáneamente se este en estado latente o inicial, una vez muestreado este valor se pasara a un nuevo estado en el cual el sistema arrojara 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 9: Ejemplo de funcionamiento de la maquina

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

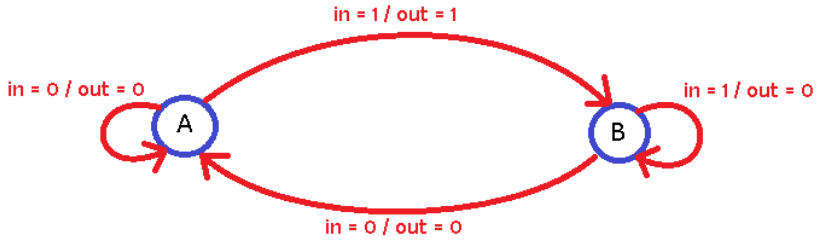

Figura 10: Esquema de la maquina de Mealy

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 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 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:

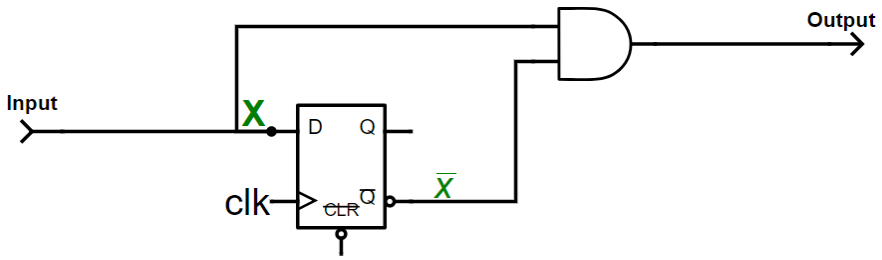

Figura 13: Circuito de Mealy

1.4. 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 utilizará 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:

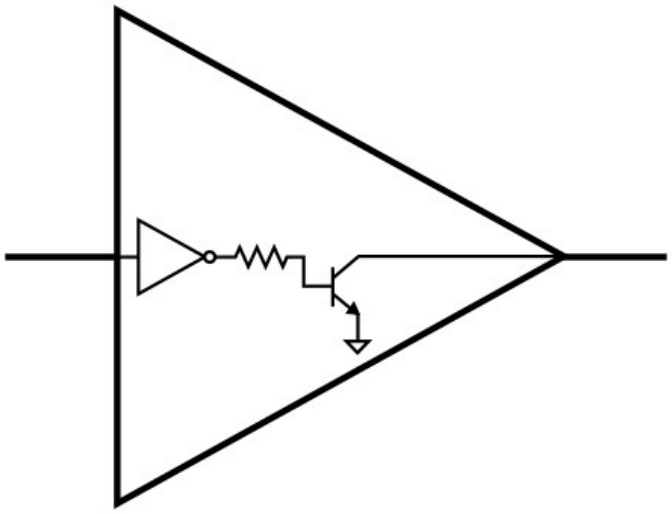

Figura 14: Diagrama del SN7407N

Como se puede ver un 1 lógico a la entrada dejara en alta impedancia la salida del integrado, dado que le 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.