REPÚBLICA DE CHILE
UNIVERSIDAD DEL BIO-BIO
FACULTAD DE CIENCIAS EMPRESARIALES
INGENIERÍA CIVIL EN INFORMÁTICA

Tarea 1

NOMBRES: Fredy Moncada
Sergio Nova

ASIGNATURA: Arquitectura de computadores

PROFESOR: L. Gajardo Díaz.

Chillán, 05 de noviembre del 2018.

[Fecha]

INDICE

Contenido

INTRODUCCION

En un mundo ampliamente globalizado donde hay una alta cantidad de interacciones y transacciones, los seres humanos tienden a omitir o desconocer la gran cantidad de operaciones que se realizan en su cotidianeidad debido a que, en algunos casos, dichas acciones se esperan con inmediatez, independiente a su complejidad o cantidad de suboperaciones que abarca y mucho menos van a considerar las operaciones y arquitectura necesaria para el correcto funcionamiento de acciones más complejas.

Comprender las operaciones de bajo nivel de arquitectura conlleva tener un amplio conocimiento respecto a la manipulación de componentes de diseño y que, con su correcta y certera combinación se puede obtener una gran cantidad de funcionalidad. A continuación, se presentará el análisis de la manipulación de componentes y como este nos ayuda a la correcta operatividad de información de manera sencilla y ordenada, tanto es su perspectiva operacional como en su respuesta funcional.

Objetivos:

- Analizar una situación problemática bajo la perspectiva de la arquitectura de bajo nivel, con el fin de comprender y aplicar los conocimientos obtenidos a lo largo la asignatura de arquitectura de computadores y anteriores.

- Realizar un análisis comparativo de soluciones, consolidando la que sea más optima según la problemática.

- Adquirir las capacidades necesarias al momento de analizar una problemática, de manera de conseguir adaptarse a las herramientas a disposición, con el propósito de dar una solución.

.

PROBLEMA

Se solicita diseñar, implementa y simular en hades un circuito que normaliza números de punto flotante. Por simplicidad se considerarán números de 12 bits: 8 bits de mantisa ($m_0 \ldots m_{-7}$) y 4 bits para el exponente ($e_3 \ldots e_0$). Todos los números son positivos y mayores que ceo. La función F permite

$$F(m_0 \ldots m_{-7} e_3 \ldots e_0) = \left(\sum_{i=-7}^{0} m_i 2^i\right) \cdot 2^{valor\,\mathrm{sin}\,signo(e_3 \ldots e_0)-8}$$

obtener el valor representado por una palabra de punto flotante:

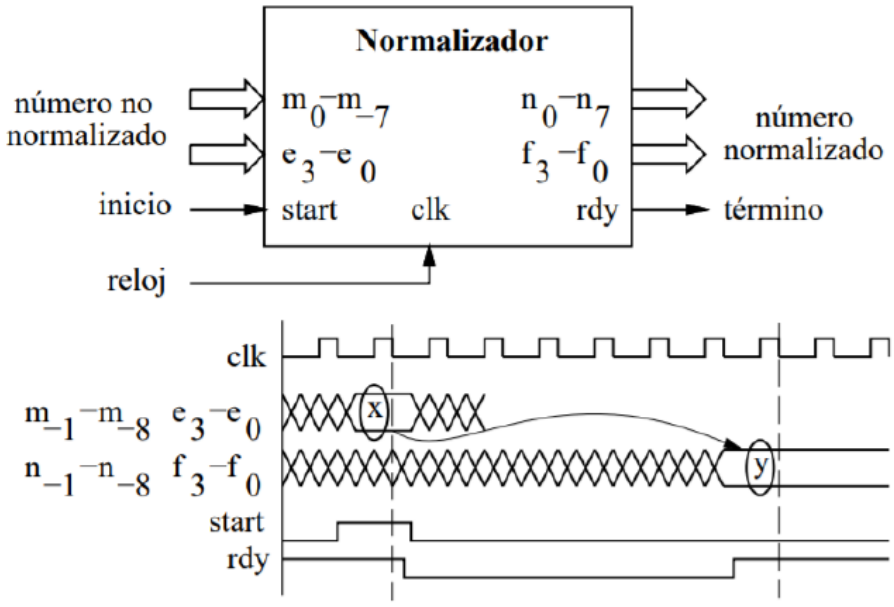
La siguiente tabla muestra algunos ejemplos de números en esta representación:

bits	valor	bits	valor
10000000 1000	1.0	10100000 1000	1.25
10000000 1001	2.0	*01000000 1001	1.0
10000000 0111	0.5	*01000000 0110	0.5
10000000 1111	2^7	11111111 1111	$\approx 2^8$
10000000 0000	2^{-8}	*00000001 1000	2^{-7}

Observe en la tabla que existen varias representaciones para un mismo número. Se dice que un número esta normalizado si su primer bit es 1. En la tabla, los números marcados con * son números no normalizados. Para normalizar un número, su mantisa se debe desplazar a la izquierda hasta que aparezca un uno en el primer bit. En seguida se resta al exponente un número equivalente a la cantidad de bits desplazados en la mantisa.

Las entradas y salidas del circuito que se pide en esta tarea y l diagrama de tiempo se muestran en la figura siguiente. La siguiente tabla muestra ejemplos de entradas y salidas para este circuito.

número *no* normalizado (x)	número normalizado (y)	bits desplazados en la mantisa
01011000 1001	10110000 1000	1
00000001 1001	10000000 0010	7
10111011 0101	10111011 0101	0
01001101 0000	*no especificado*	



Para simplificar la tarea no se especifica un resultado cuando el número no es normaliza o cuando es cero. Observe que su circuito puede tomar todos los ciclos que estime conveniente para normaliza el número.

Concretamente se pide diseñar, implementar y simular este circuito usando Hades. El circuito que implemente debe representar el diagrama de tiempo de la figura anterior.

SOLUCIÓN

Después de realizar un extenso y arduo análisis de la problemática y con el uso de una larga lluvia de ideas, se ha llegado a una solución (entre otras) de compleja explicación, debido a la implementación netamente de subdiseños, a complejidad radica en que la solución a sido implementada con el programa de construcción y diseño de circuitos hades, el cual, al mover un subdiseño, al compartir dicho archivo a otra computadora, se perdían los artefactos utilizados en un diseño como tal previamente utilizados. Frente a esta situación y con la esperanza de que esto no suceda en todos los casos de igual manera se adjunta una foto con la ubicación de cada componente en el circuito completo y de la tabla de verdad correspondiente (también adjuntaremos un autómata que diseñamos en la lluvia de ideas).

Una vez explicado el problema con el programa hades, se explicará el funcionamiento del circuito:

Paso a paso del circuito:

- La señal de **Start** es la encargada de iniciar o reiniciar el circuito cargando la **Mantisa IN** y el **Exponente IN**. Esta señal también alimenta a **Ready** para informar que se cargó la información, una vez apagada la señal, este comienza a trabajar.

- Una vez apagada la señal **Start,** los multiplexores desactivan la entrada a la **Mantisa IN** y **Exponente IN** cesando la transmisión de los datos ya trabajados que se explicaran más adelante.

- **La Mantisa IN**: Datos de entrada, son almacenados en el registro de 8 bits y que posteriormente son transportados para alimentar al componente shifter, también son trasportados hasta ser vistos en la salida "Mantisa OUT", donde se puede observar el estado actual de la mantisa. De igual manera esta información es ingresada al segundo multiplexor, este se encarga de verificar si existe un "1" en el primer bit de la mantisa, si es encontrado se deja pasar el valor del registro directamente, si no lo encuentra, permite avanzar la salida del componente shifter y admite que vuelva a entrar en el primer multiplexor. Esto se repite hasta que encuentra un "1" en la primera posición de la mantisa.

- **El exponente IN**: Datos de entrada, son almacena en el registro de 4 bits y que luego es transmitido para realizar una cadena de acciones consecutivas, la primera es mostrar el estado actual del exponente e ingresar dichos datos al restador. Si todavía no encuentra un "1" en la primera posición de la mantisa, este debe restar consecutivamente 0001, hasta que aparezca dicho "1", cada vez que resta se vuelve a guardar en el registro, volviendo a realizar toda la operación otra vez.

- Una vez concluida la operación de normalización, este debe avisar, activando la señal **Ready** y mostrar los resultados obtenidos de la resta.

- **Casos especiales**: En el caso de ingresar un exponente menor que la cantidad de bits desplazados este entregara un valor sin importancia (basura).

Tablas de verdad

A continuación, se presentarán las tablas de verdad pertinentes en nuestra implementación.

SEMI RESTADOR

A	B	CARRY	D
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

REST_X4

A	B	CARRY_IN	CARRY_OUT	D
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

shifter (1bit)

in	out
n	n+1

decoder 1x2

IN	SIGNAL_1	SIGNAL_0
0	0	1
1	1	0

multiplexor 2x1 (8bits , 4bits)

E1	E2	C	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Mapas de Karnaugh

A continuación, se presentarán los mapas obtenidos.

SEMI RESTADOR				
C-in\E1 E2	0 0	0 1	11	10
0	0	1	0	1
1	1	0	1	0

REST_X4				
C-in\E1 E2	0 0	0 1	11	10
0	0	0	1	0
1	0	1	1	1

multiplexor 2x1 (8bits , 4bits)		
E2 C\E1	0	1
0 0	0	0
0 1	0	1
11	0	1
10	1	1

Componentes

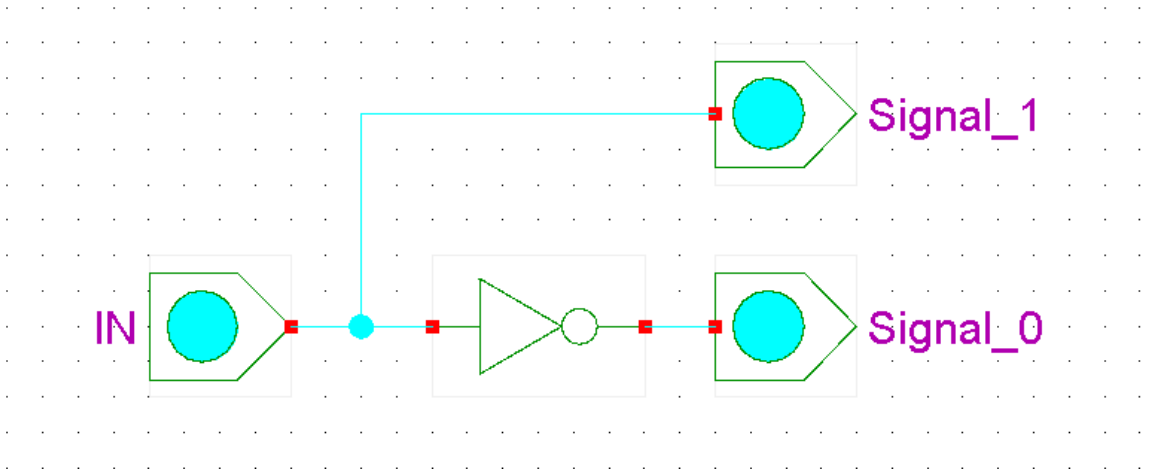


Ilustración 1:Circuito de decodificador 2x1

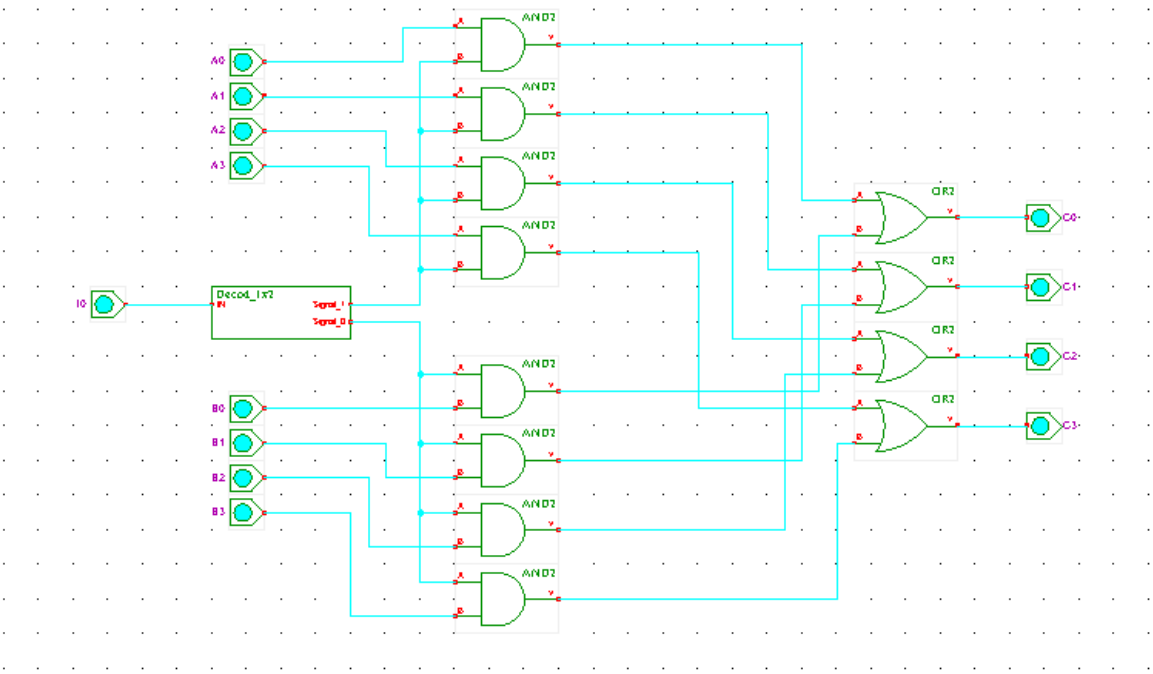


Ilustración 2:Circuito multiplexor 2x1 (4 bits)

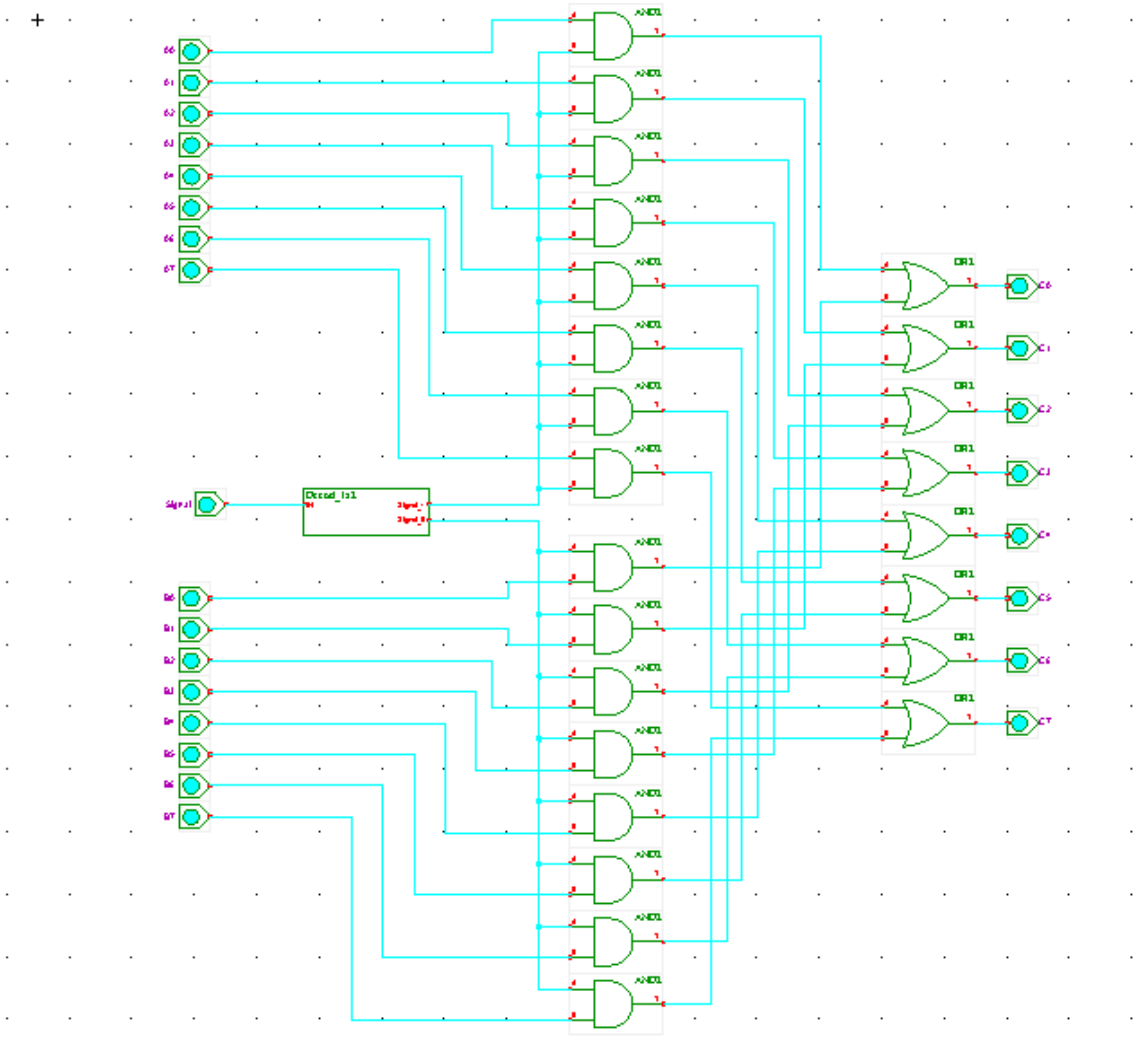


Ilustración 3:Circuito multiplexor 2x1(8 bits)

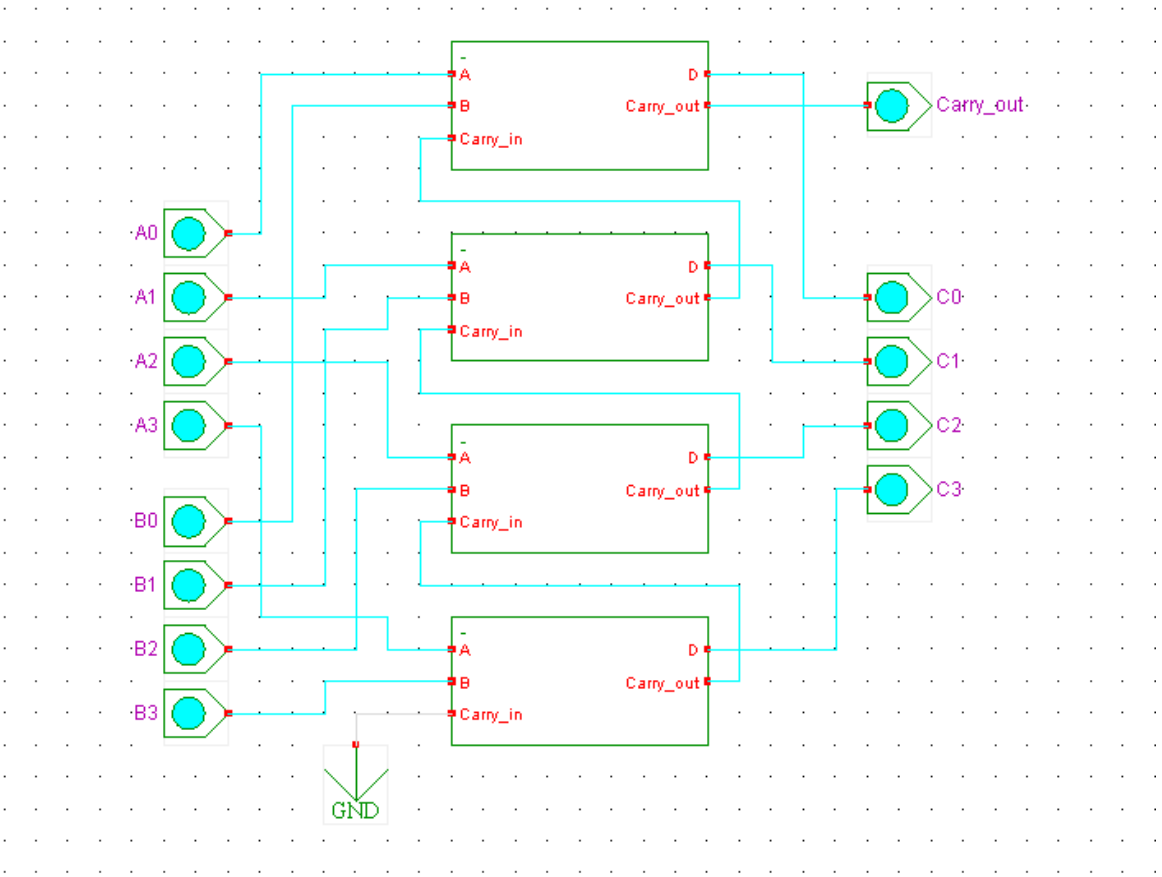


Ilustración 4:Circuito restador (4 bits)

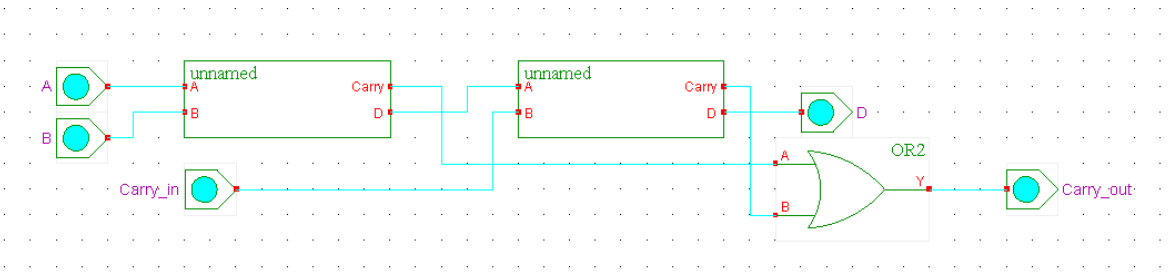


Ilustración 5:Circuito restador completo (1 bit)

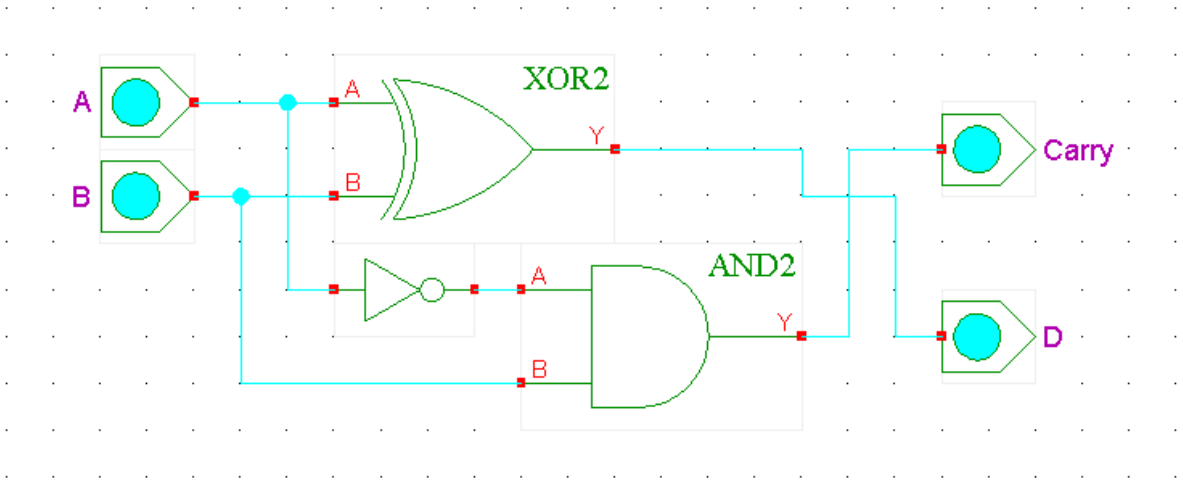


Ilustración 6:Circuito semi-restador (1bit)

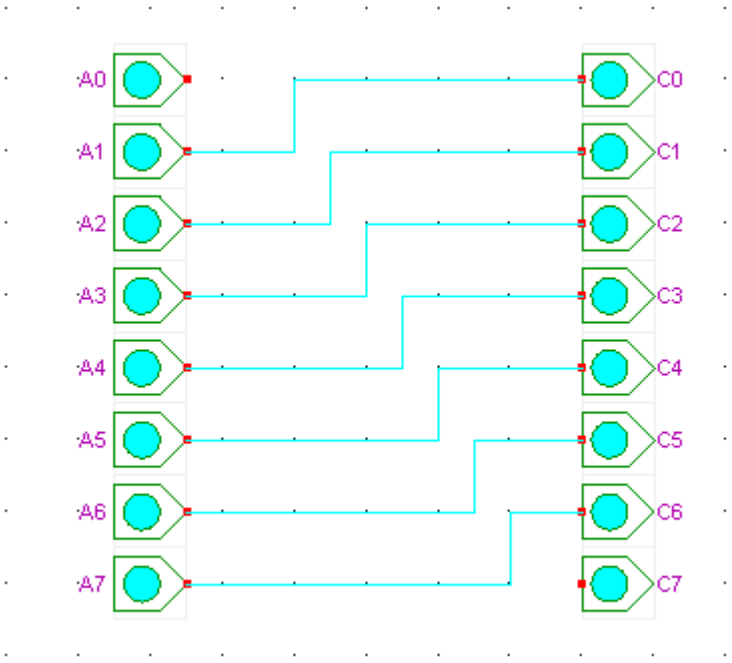


Ilustración 7:Circuito Shifter

Autómata (diagrama de estados)

A continuación, se presentará el diagrama de estados realizado.

Etiqueta del diagrama de estados:

N° de entrada, Exp de entrada, Inicio / N° de salida, Exp de salida, termino

$m_0m_{-1}m_{-2}m_{-3}m_{-4}m_{-5}m_{-6}m_{-7}$ $e_3e_2e_1e_0$ Strt / $n_0n_{-1}n_{-2}n_{-3}n_{-4}n_{-5}n_{-6}n_{-7}$ $f_3f_2f_1f_0$ Rdy

Notar que (X-1) simboliza la sustracción que se realiza al número del exponente en 1 (0001) por cada movimiento a la izquierda realizado (shifter). Esta notación está presente debido al desconocimiento del número en cuestión.

Notar que X* representa el resultado obtenido de la sustracción reiterada del exponente por cada movimiento a la izquierda realizado (shifter). Esta notación está presente debido al desconocimiento del número en cuestión.

Transiciones

Estado: Del 0 al 8 Destino: 0 Condición: Activación de inicio

XXXXXXXX XXXX 1 / XXXXXXXX XXXX 0

Estado: 0 Destino: 1 Condición: -

XXXXXXXX XXXX 0 / XXXXXXXX XXXX 0

Estado: 1 Destino: 2 Condición: Movimiento a la izquierda (Shifter) sin normalizar

0XXXXXX0 XXX(X-1) 0 / 0XXXXXX0 XXX(X-1) 0

Estado: 1 Destino: 9 Condición: Movimiento a la izquierda (Shifter) normalizado

1XXXXXX0 XXX(X-1) 0 / 1XXXXXX0 XXX(X-1) 0

Estado: 2 Destino: 3 Condición: Movimiento a la izquierda (Shifter) sin normalizar

0XXXXX00 XXX(X-1) 0 / 0XXXXX00 XXX(X-1) 0

Estado: 2 Destino: 9 Condición: Movimiento a la izquierda (Shifter) normalizado

1XXXXX00 XXX(X-1) 0 / 1XXXXX00 XXX(X-1) 0

Estado: 3 Destino: 4 Condición: Movimiento a la izquierda (Shifter) sin normalizar

0XXXX000 XXX(X-1) 0 / 0XXXX000 XXX(X-1) 0

Estado: 3 Destino: 9 Condición: Movimiento a la izquierda (Shifter) normalizado

1XXXX000 XXX(X-1) 0 / 1XXXX000 XXX(X-1) 0

Estado: 4 Destino: 5 Condición: Movimiento a la izquierda (Shifter) sin normalizar

0XXX0000 XXX(X-1) 0 / 0XXX0000 XXX(X-1) 0

Estado: 4 Destino: 0 Condición: Movimiento a la izquierda (Shifter) normalizado

1XXX0000 XXX(X-1) 0 / 1XXX0000 XXX(X-1) 0

Estado: 5 Destino: 6 Condición: Movimiento a la izquierda (Shifter) sin normalizar

0XX00000 XXX(X-1) 0 / 0XX00000 XXX(X-1) 0

Estado: 5 Destino: 9 Condición: Movimiento a la izquierda (Shifter) normalizado

1XX00000 XXX(X-1) 0 / 1XX00000 XXX(X-1) 0

Estado: 6 Destino: 7 Condición: Movimiento a la izquierda (Shifter) sin normalizar

0X000000 XXX(X-1) 0 / 0X000000 XXX(X-1) 0

Estado: 6 Destino: 9 Condición: Movimiento a la izquierda (Shifter) normalizado

1X000000 XXX(X-1) 0 / 1X000000 XXX(X-1) 0

Estado: 7 Destino: 8 Condición: Movimiento a la izquierda (Shifter) sin normalizar

00000000 XXX(X-1) 0 / 00000000 XXX(X-1) 0

Estado: 7 Destino: 9 Condición: Movimiento a la izquierda (Shifter) normalizado

10000000 XXX(X-1) 0 / 10000000 XXX(X-1) 0

Estado: 8 Destino: 8 Condición: Movimiento a la izquierda (Shifter) sin normalizar, (loop)

00000000 XXX(X-1) 0 / 00000000 XXX(X-1) 0

Estado: 8 Destino: 9 Condición: Movimiento a la izquierda (Shifter) normalizado

00000000 XXXX* 0 / 00000000 XXXX* 0

Estado: 9 Destino: 10 Condición: reconocimiento de normalización

1XXXXXXX X*X*X*X* 0 / 1XXXXXXX X*X*X*X* 1

Estado: 9 Destino: 0 Condición: reconocimiento de normalización y activación inicio

1XXXXXXX X*X*X*X* 1 / 1XXXXXXX X*X*X*X* 1

Estado: 10 Destino: 10 Condición: reconocimiento de normalización (loop)

1XXXXXXX X*X*X*X* 0 / 1XXXXXXX X*X*X*X* 1

Estado: 10 Destino: 0 Condición: reconocimiento de normalización y activación de inicio

1XXXXXXX X*X*X*X* 1 / 1XXXXXXX X*X*X*X* 1

ELEMENTOS ADJUNTOS:

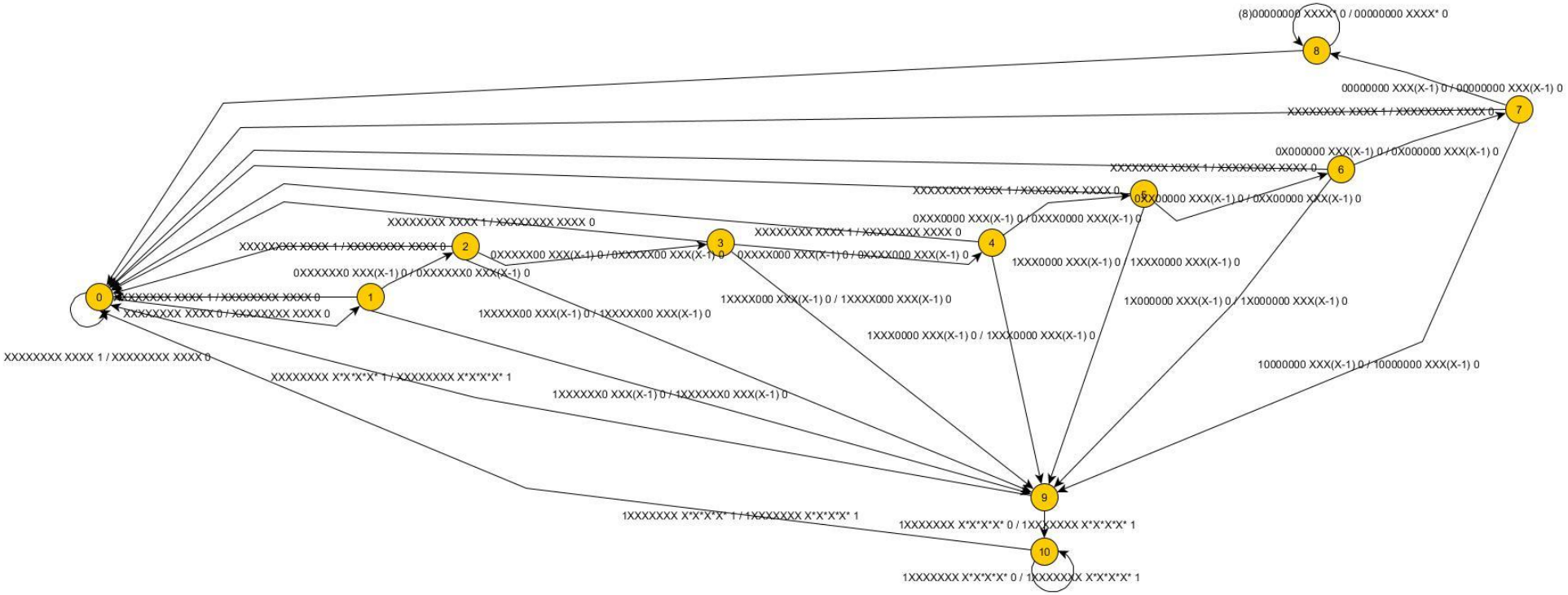


Ilustración 8: La imagen corresponde al diagrama de estados usado para dar solución a la problemática

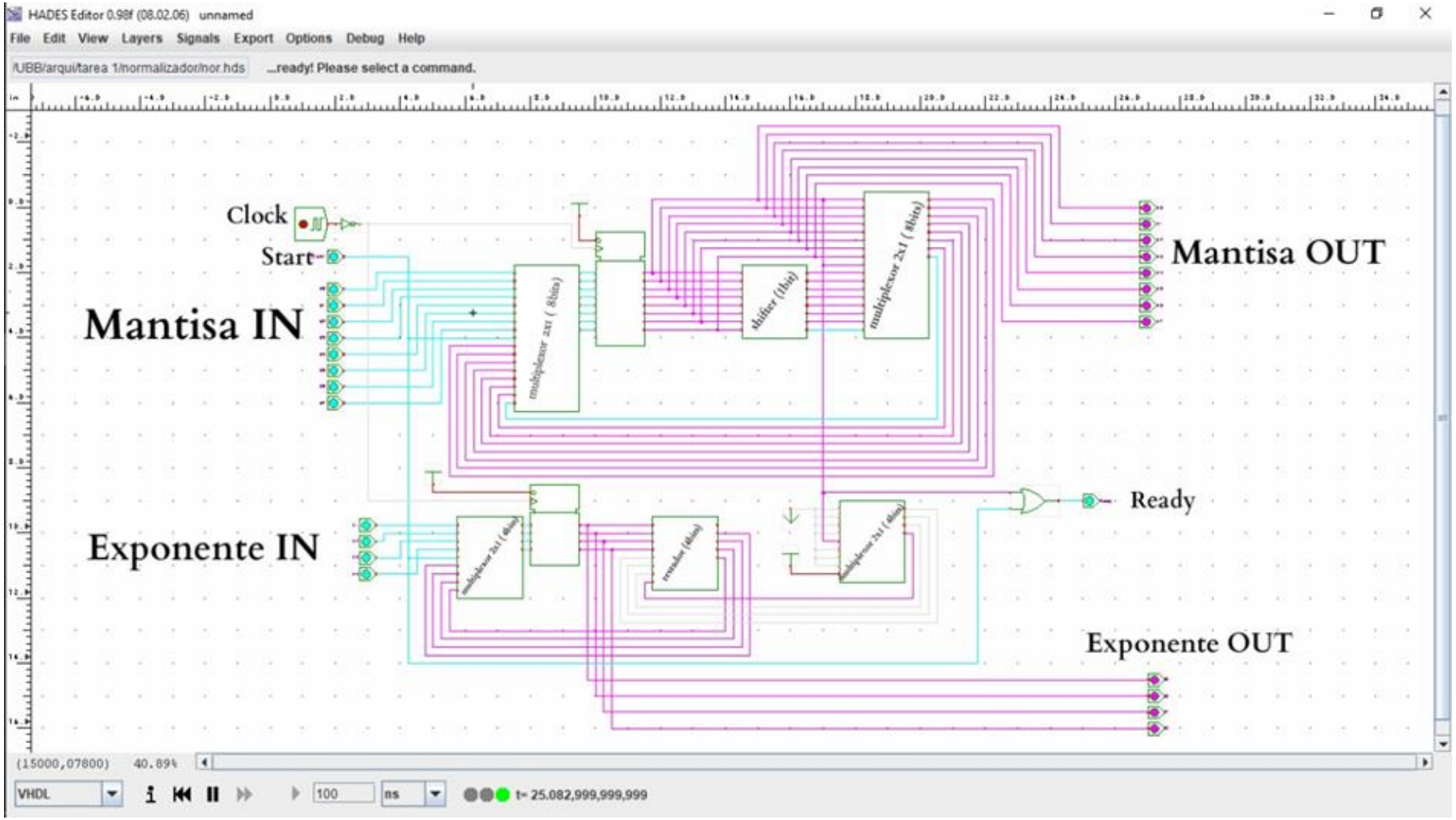


Ilustración 9:La imagen corresponde al circuito usado para dar solución a la problemática.

CONCLUSION

En retrospectiva, la operatividad de un sistema se radica principalmente en la cantidad de relaciones que sus componentes tienen y en cómo se relacionan con el fin de conseguir dar funcionalidades, optimizando la cantidad de requerimientos arquitectónicos y conseguir una alta compenetración operativa. Lo anteriormente mencionado no es una actividad sencilla, pues se debe tener un alto dominio en el área de trabajo, tanto de conocimiento como experimental.

Considerando la actividad realizada, se puede apreciar un cumplimento satisfactorio, tanto en la solución conseguida a la problemática planteada y su posterior implementación, así como en los objetivos considerados como grupo, apreciando un aprendizaje considerable en estas temáticas (análisis de problemas de arquitecturas), haciendo hincapié en la adaptabilidad que se debe dar a la hora de genera soluciones según las funcionalidades solicitadas.