Proyecto de Arquitectura de Computadoras: Etapa 2

Materia: Arquitectura de Computadoras

Etapa del Proyecto: Etapa 2

Fecha: 20/05/2025

Comisión: Número 7

Integrante 1: Bassi, Juan Sebastián - 133646

Integrante 2: Cristobo, Juan Manuel - 137390

Integrante 3: Mosqueira, Juan Segundo - 131415

Integrante 4: Vilas, Santiago - 135163

Logisim-Evolution: v3.9.0

ÍNDICE

Introducción

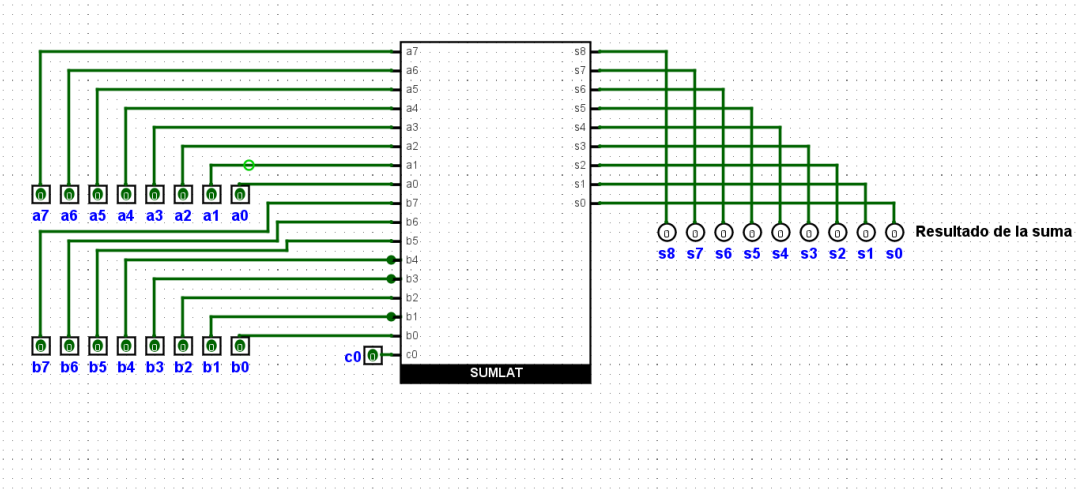
En este informe realizado a partir de la etapa uno del proyecto práctico de la materia "Arquitectura de Computadoras" se detallará el funcionamiento del sumador Lookahead Tree Adder

Etapa 1

Descripción:

En principio usamos un circuito denominado "FAGP" que implementa un sumador Full Adder, el cual realiza la suma de 2 bits y calcula los valores de propagación y generación para los bits de entrada al circuito. Por otro lado, implementamos un circuito "Bloque B" que permite calcular los valores PHL, GHL, CH y CL a partir de los valores de entrada: PH, GH, PL, GL y del carry in del bloque. Para esto, utilizamos compuertas del tipo AND, OR y NOT. Finalmente combinamos "FAGP" y "Bloque B" para implementar un circuito que permite sumar números de 8 bits basándonos en el diseño presentado por la cátedra del Look Ahead Tree Adder.

Circuito Main

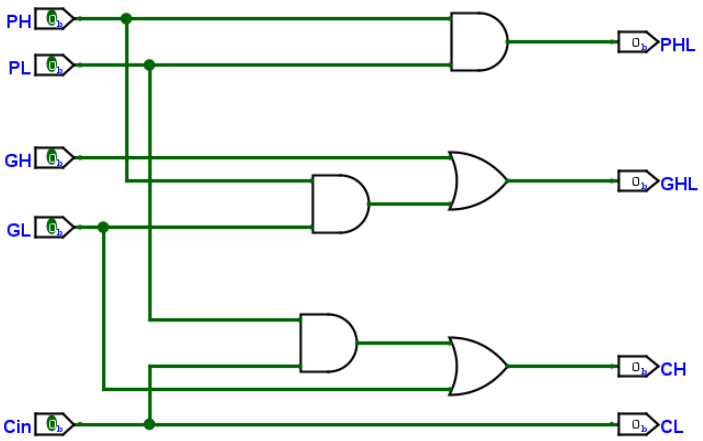


Funcionamiento del circuito Main:

El circuito main utiliza al circuito sumador "SUMLAT" como se pide en el enunciado con bits de entrada a7,a6,a5,a4,a3,a2,a1 y a0 correspondientes al número "a" (a7 dígito más significativo y a0 dígito menos significativo) y b7,b6,b5,b4,b3,b2,b1 y b0 los cuales pertenecen al número "b" (b7 dígito más significativo y b0 dígito menos significativo).

Las salidas s8 a s0 son los bits del resultado de la suma, s8 es el noveno bit (dígito más significativo) en caso de que haya overflow, s7 es el bit del dígito más significativo de la suma en caso de que no haya overflow y s0 el bit del dígito menos significativo.

Circuito del Bloque B



Valores de Entrada y Salida del Bloque B

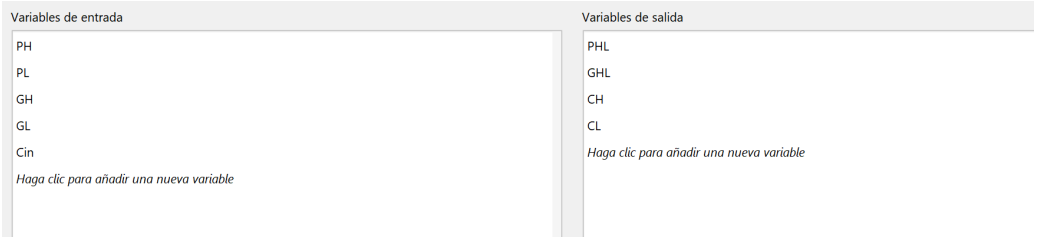


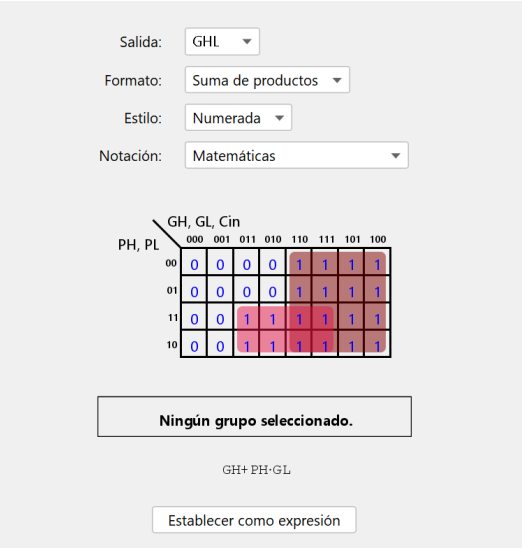
Tabla de Verdad del Bloque B

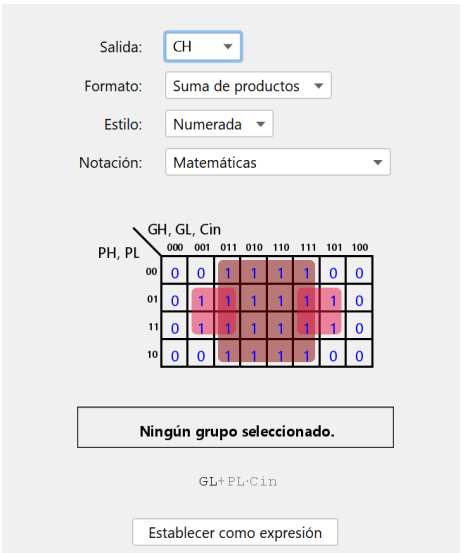
- 1 0 Ocultar líneas duplicadas Mostrar todas las línea

32 de 32 filas mostradas

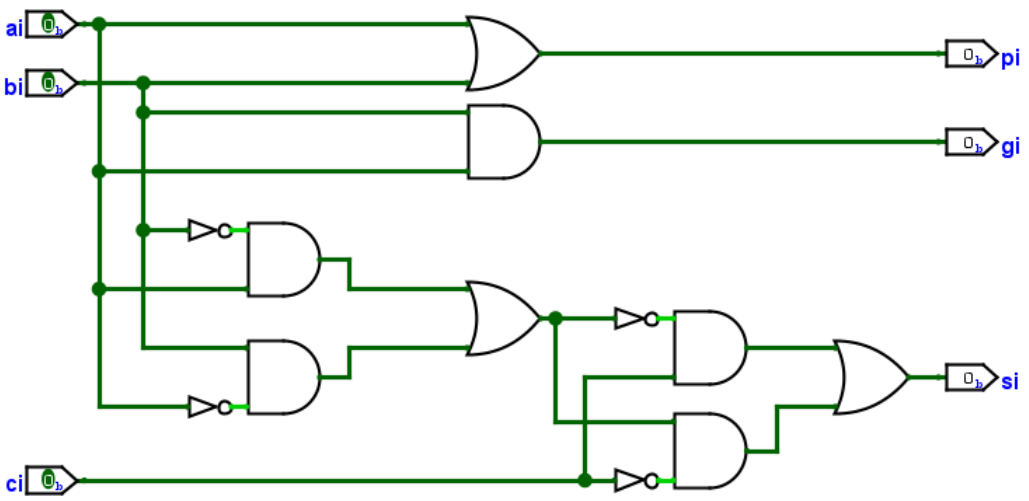
PH	PL	GH	GL	Cin	PHL	GHL	CH	CL
0	0	1	0	0	0	1	0	0
0	0	1	0	1	0	1	0	1
0	0	1	1	0	0	1	1	0
0	0	1	1	1	0	1	1	1
0	1	0	0	0	0	0	0	0
0	1	0	0	1	0	0	1	1
0	1	0	1	0	0	0	1	0
0	1	0	1	1	0	0	1	1
0	1	1	0	0	0	1	0	0
0	1	1	0	1	0	1	1	1
0	1	1	1	0	0	1	1	0
0	1	1	1	1	0	1	1	1
1	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
1	0	0	1	0	0	1	1	0
1	0	0	1	1	0	1	1	1
1	0	1	0	0	0	1	0	0
1	0	1	0	1	0	1	0	1
1	0	1	1	0	0	1	1	0
1	0	1	1	1	0	1	1	1
1	1	0	0	0	1	0	0	0
1	1	0	0	1	1	0	1	1
1	1	0	1	0	1	1	1	0
1	1	0	1	1	1	1	1	1
1	1	1	0	0	1	1	0	0
1	1	1	0	1	1	1	1	1
1	1	1	1	0	1	1	1	0
1	1	1	1	1	1	1	1	1

Mapas de Karnaugh del Bloque B para cada Salida





Sumador Full Adder “FAGP”



Valores de Entrada y Salida del Sumador FAGP

Variables de entrada	Variables de salida
ai	pi
bi	gi
ci	si
Haga clic para añadir una nueva variable	*Haga clic para añadir una nueva variable*

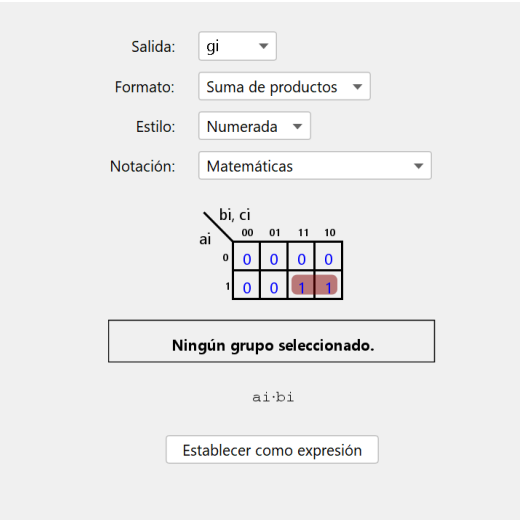
Tabla de Verdad del Sumador FAGP

- 1 0 Ocultar líneas duplicadas Mostrar todas las líneas

8 de 8 filas mostradas

ai	bi	ci	pi	gi	si
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	1	0	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	0	0
1	1	0	1	1	0
1	1	1	1	1	1

Mapas de Karnaugh de las salidas del Sumador FAGP



Circuito Sumador Lookahead Tree Adder “SUMLAT”

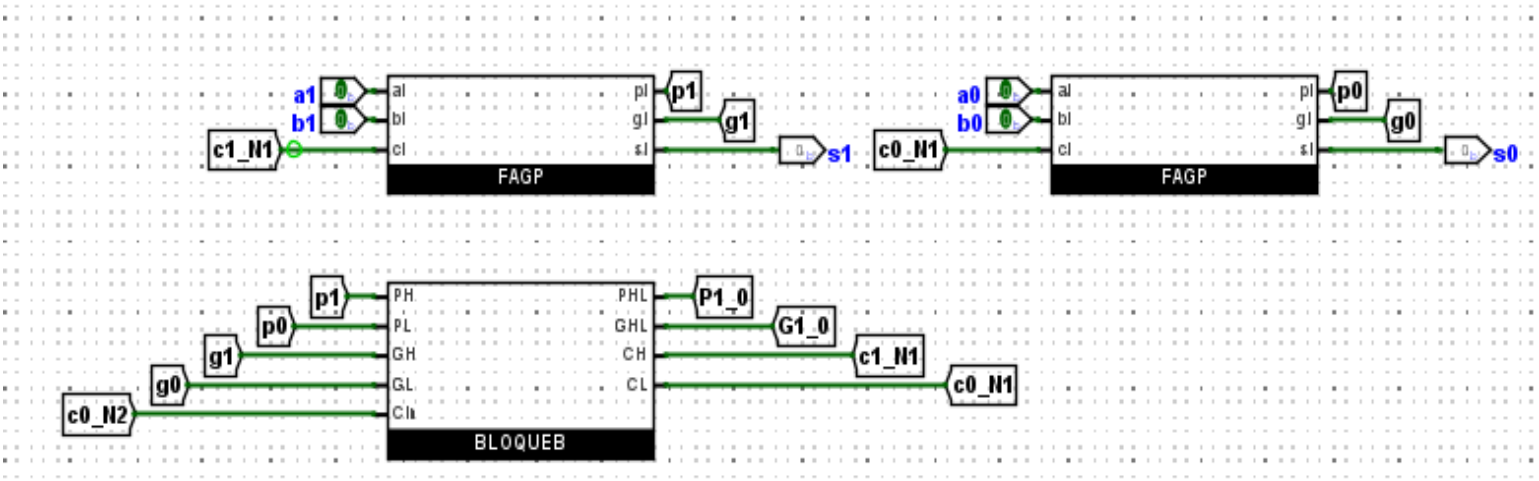
Introducción al circuito:

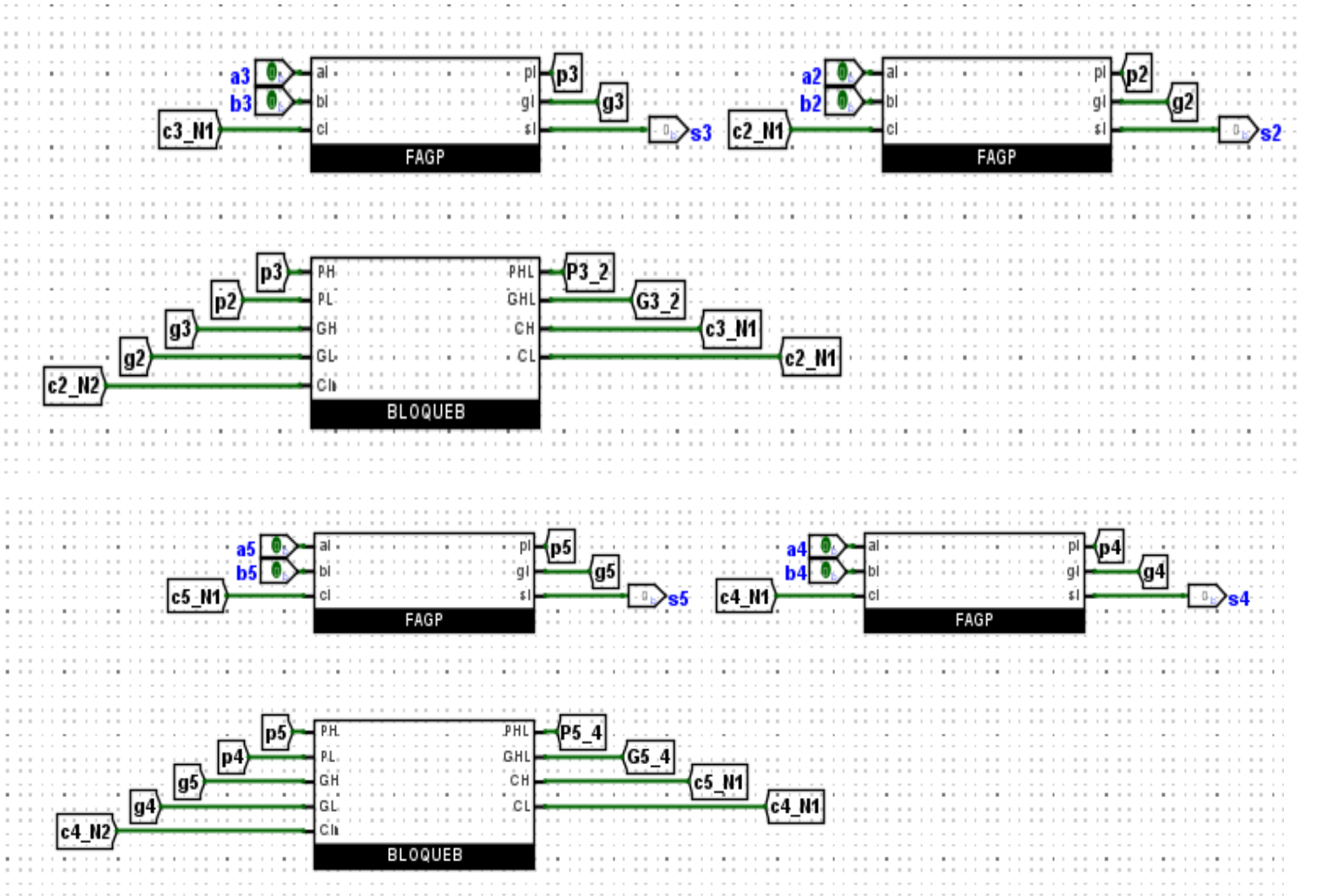
- Cada pin nombre a_i siendo i >= 0 e i <= 7, en donde 0 representa al dígito menos significativo del número “a” y 7 corresponde al dígito más significativo del número “a”. Lo mismo ocurre con el número “b”.
- c_i es el carry de entrada del sumador correspondiente al dígito i.
- p_i es el bit que propaga el sumador FAGP y g_i es el bit que genera el mismo sumador.
- El Bloque B se encarga de recibir el valor propagado alto (P_H ó P_{HIGH}), valor propagado bajo (P_L ó P_{LOW}), valor generado alto (G_H ó G_{HIGH}), valor generado bajo (G_L ó G_{LOW}) y el carry de entrada (C_{in}).

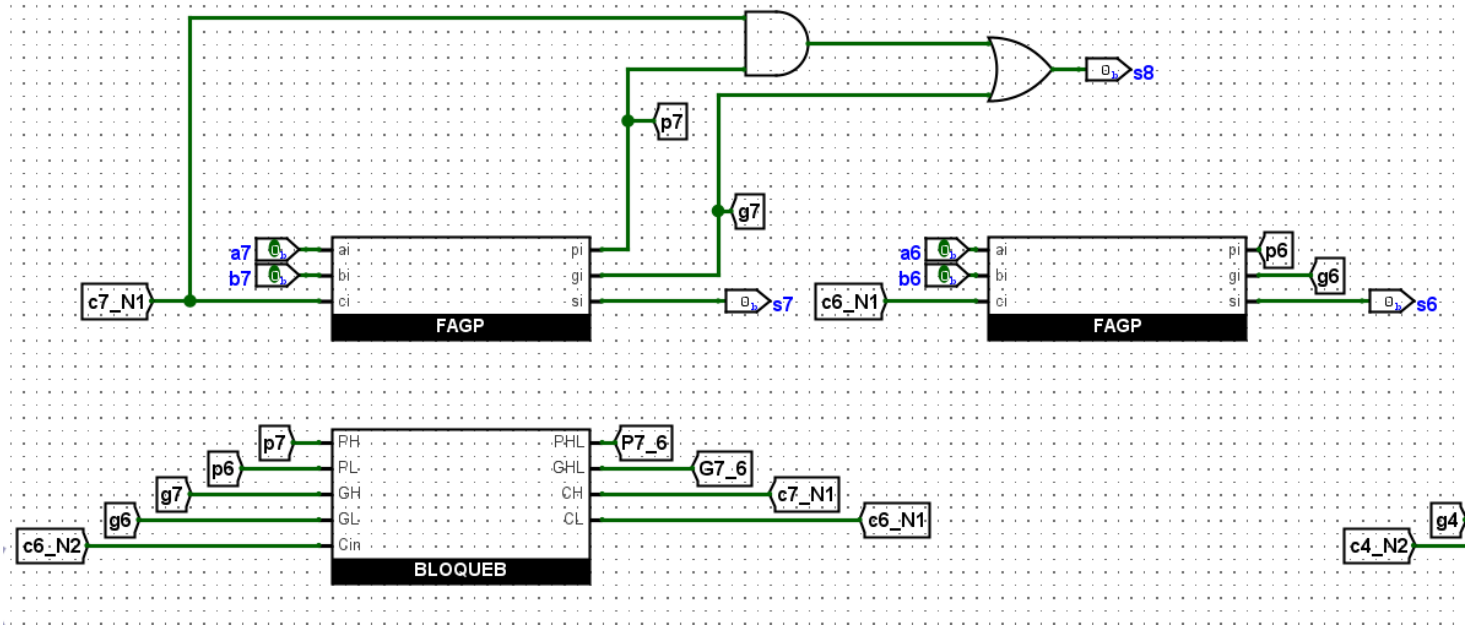
Niveles del circuito

Nivel 1

En la primera imagen vemos la suma entre los bits a0 y b0 en el sumador FAGP de la derecha y la suma entre los bits a1 y b1 en el sumador FAGP de la izquierda. En las siguientes imágenes correspondientes al nivel 1 podemos observar que preservan el mismo comportamiento menos en la última del mismo nivel en donde podemos notar que en el sumador de los bits más significativos (a7 y b7) hay un pin de salida “s8” el cual recibirá un valor en caso de que haya overflow y esa suma genere un nuevo número de 9 bits, por ejemplo: 1111 1111 (8bits) + 1111 1111 (8bits) = 1 1111 1110 (9 bits).

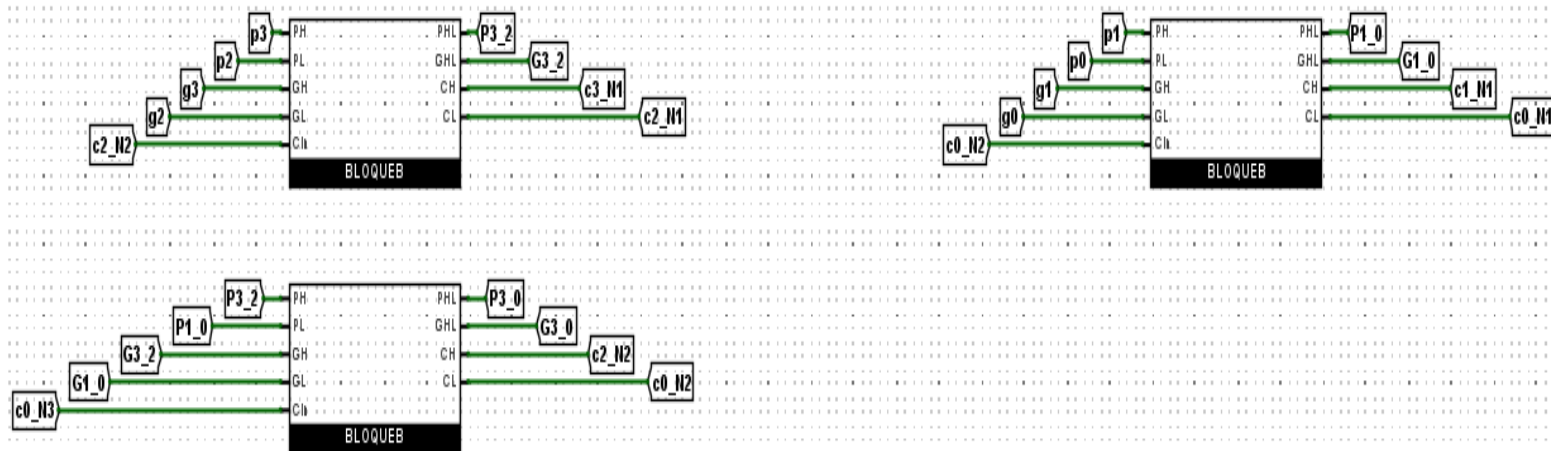

a1
b1
c1_N1
al
bl
cl
FAGP
pl
gl
sl
p1
g1
s1
a0
b0
c0_N1
p0
g0
s0
p1
p0
g1
g0
c0_N2
PH
PL
GH
GL
CH
BLOQUEB
PHL
GHL
CH
CL
P1_0
G1_0
c1_N1
c0_N1

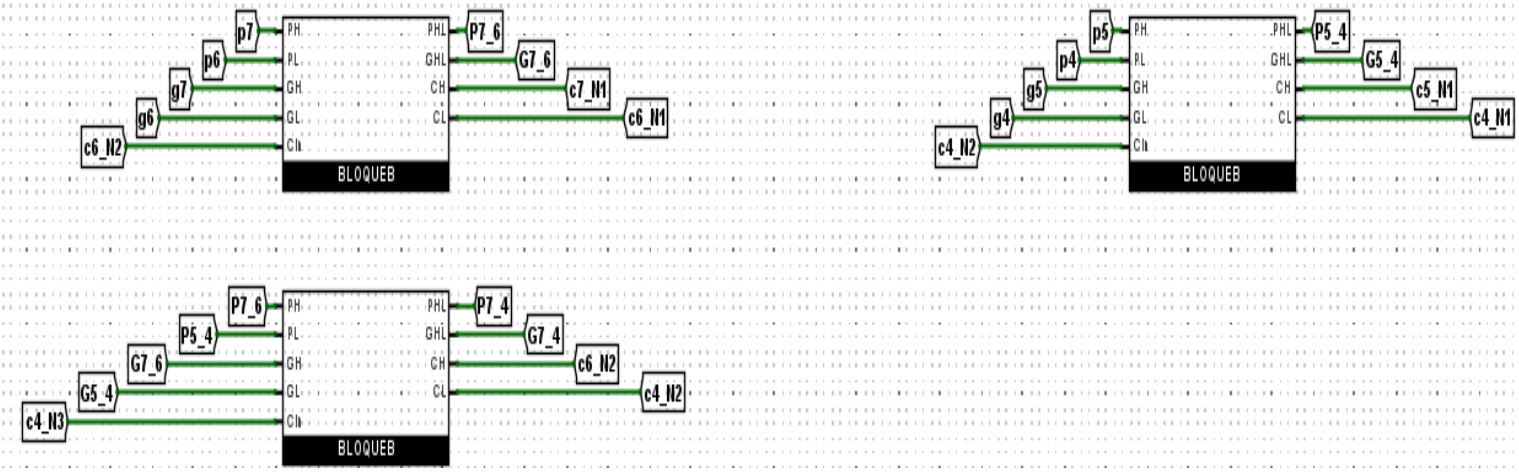

a3
b3
c3_N1
FAGP
p3
g3
s3
a2
b2
c2_N1
p2
g2
s2
p3
p2
g3
g2
c2_N2
BLOQUEB
P3_2
G3_2
c3_N1
c2_N1
a5
b5
c5_N1
p5
g5
s5
a4
b4
c4_N1
p4
g4
s4
p5
p4
g5
g4
c4_N2
P5_4
G5_4
c5_N1
c4_N1



Nivel 2

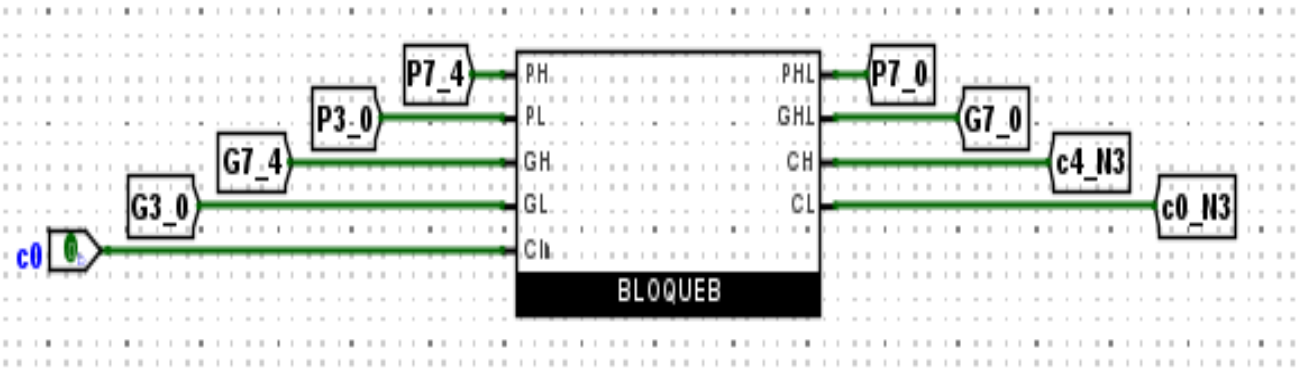
En los Bloques B del nivel 2, las entradas ya no van a ser las salidas de los circuitos FAGP, si no que serán de los Bloques B inmediatos previos. Por ejemplo; en el lado derecho de SUMLAT las entradas de propagación y generación menos significativas de este nivel ahora provienen del PHL y GHL generados por el Bloque B que trabaja con los dos bits menos significativos, es decir, el PL nuevo será "P1_0" pues de las dos propagaciones recibidas, "P1_0" es menos significativa que "P2_3", que será la ingresada como PH. Lo mismo ocurrirá con la selección de entradas para las variables de generación.





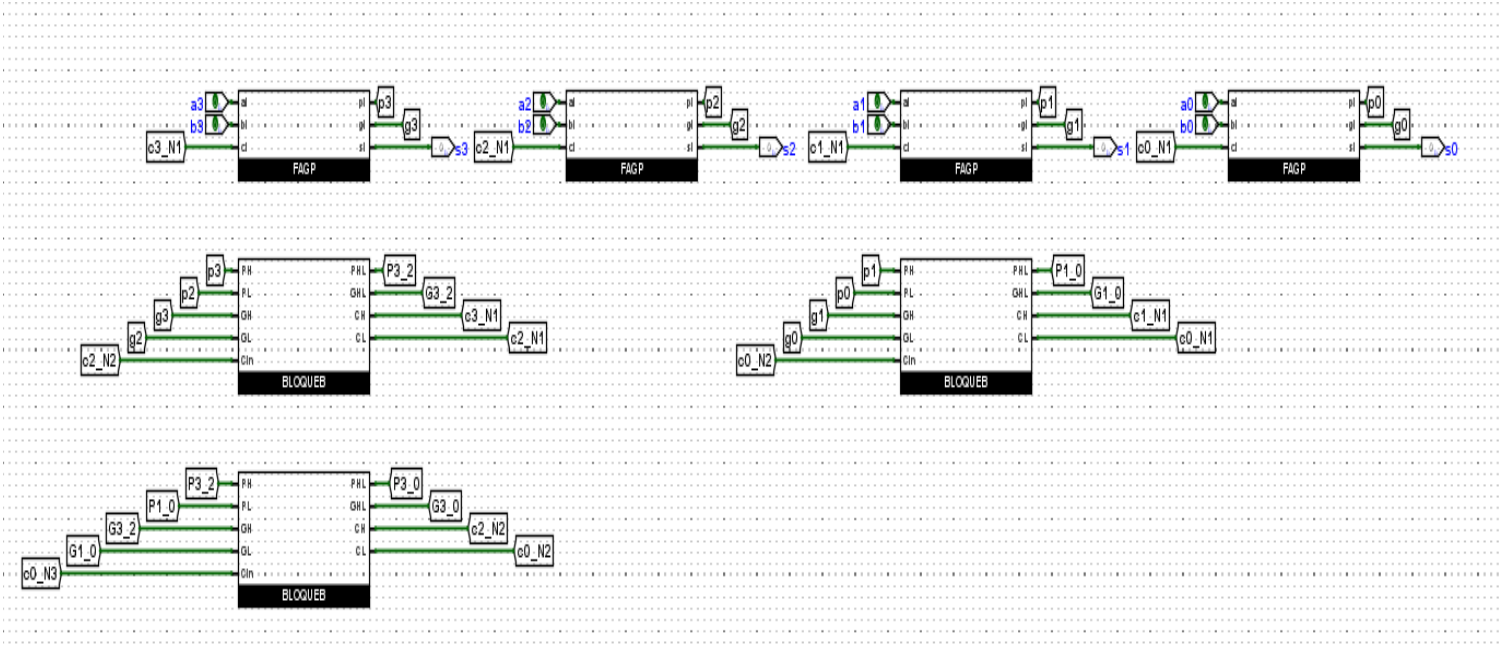
Nivel 3

Similar al nivel dos, las entradas de propagaciones y generaciones menos significativas van a corresponder a las salidas del Bloque B que trabaja las generaciones y propagaciones menos significativas, y análogamente ocurre para las más significativas.



Imágenes del circuito entero sumador “SUMLAT”

La primera imagen corresponde a la primera mitad del lado derecho del circuito, y la segunda imagen a la mitad del lado izquierdo.

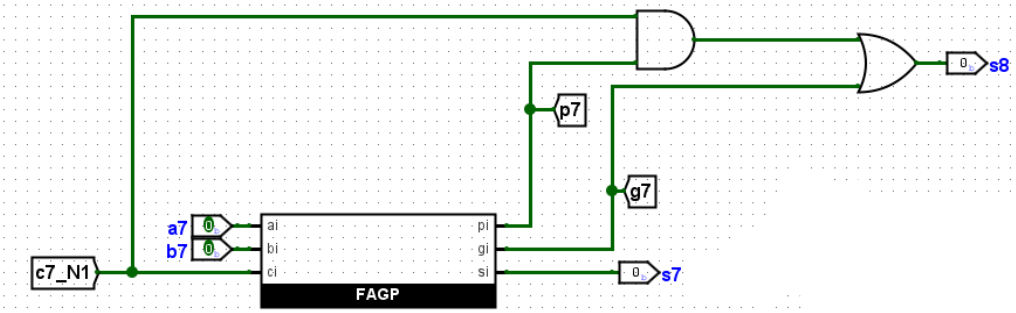


Aclaraciones

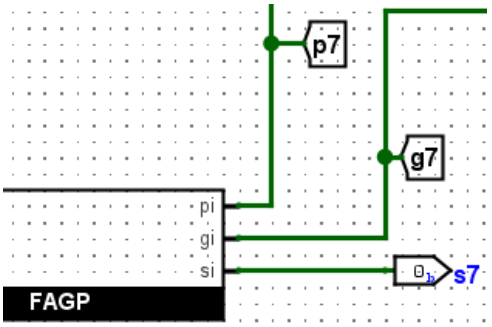
- c_i_N_j es el túnel correspondiente a cada sumador, en donde c_i representa al carry de entrada y la i al igual que en a_i y b_i el orden ya mencionado del dígito en el que se encuentra y N_j corresponde al nivel, siendo j >= 1 y j <= 3, debido a que son 3 niveles para poder llegar al valor de c0 que se encuentra en el último bloque. Por ejemplo, 

va a corresponder al dígito más significativo del nivel 1.

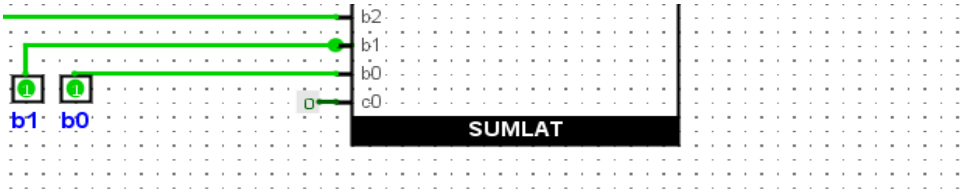
- En el último full adder FAGP del circuito "SUMLAT" la salida s8 que corresponde al noveno bit (carry out) en caso de que haya overflow, recibirá la siguiente fórmula C_{out} = (c7_N1*p7) + g7



- Los valores de propagación "p7" y valor de generación "g7" no se utilizan y podrían no ir en el circuito implementado, pero sin embargo adoptamos la decisión de dejarlos en caso de que se agreguen más bits para sumar y se necesiten más sumadores.



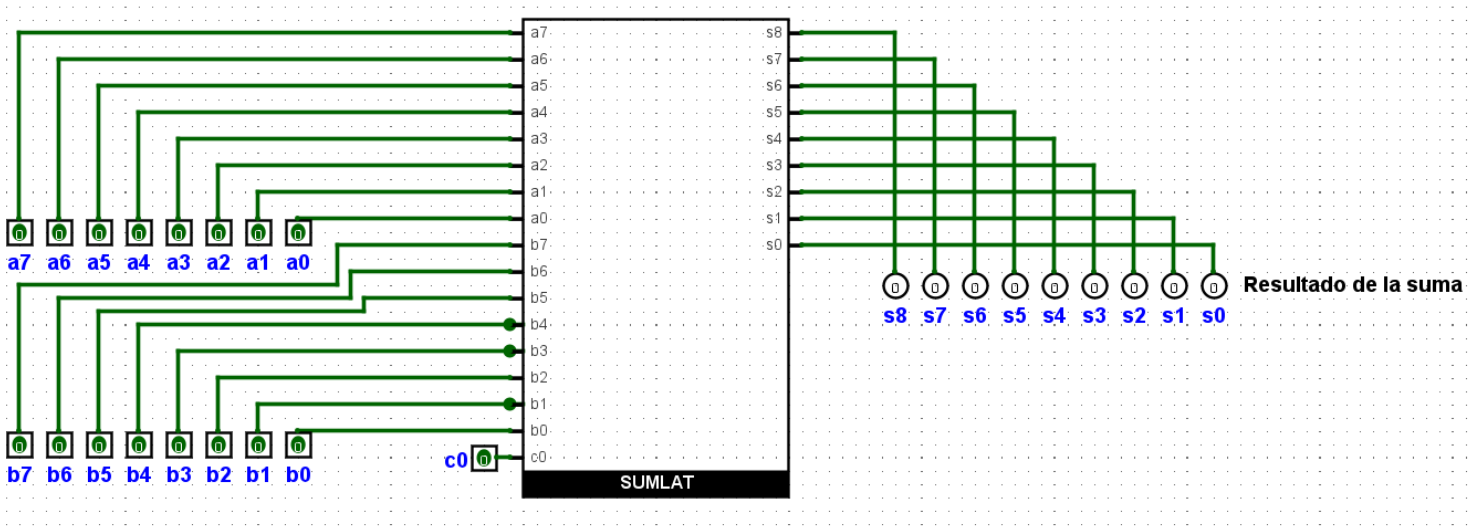
- Como decisión de diseño hemos utilizado como valor de entrada “c0” a la constante 0 ya que asumimos de que la suma siempre la comenzamos sin ningún carry de entrada en el menos significativo.



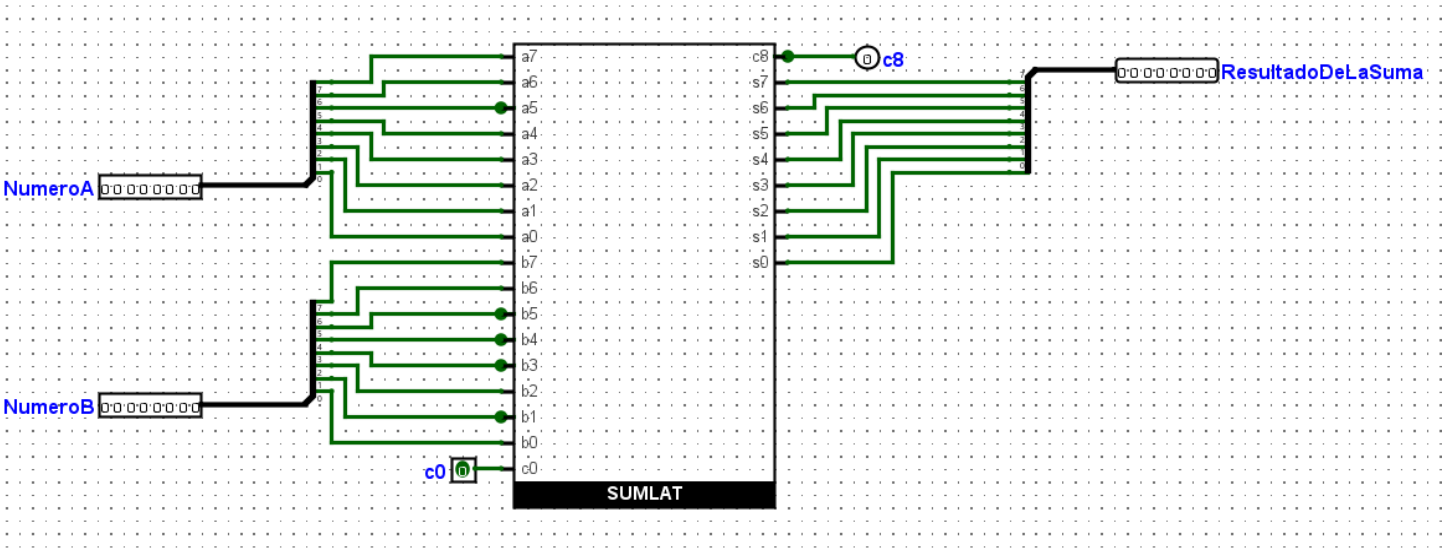
Correcciones Etapa 1

- Se modificó el Circuito Main; se pasó de tener de entradas y salida 8 pines individuales de 1 bit, a tener pins de 8 bits.
- El s8 se colocó como un carry a parte llamado s8, y no como parte del resultado de la suma.

Circuito Main antes de la corrección:



Circuito Main después de la corrección:



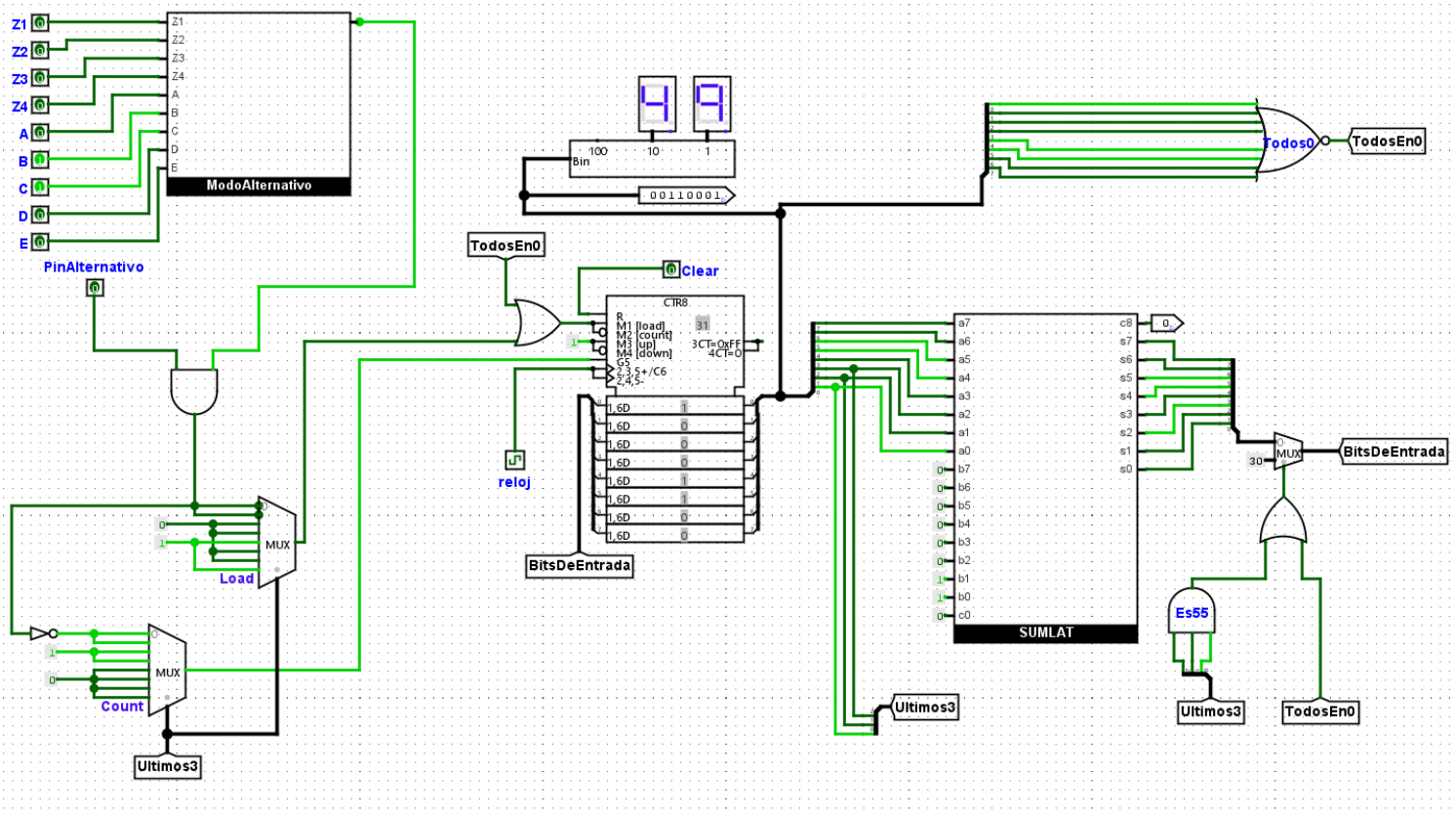
Etapa 2

Descripción:

Durante esta etapa diseñamos un circuito controlador de estados en Logisim-Evolution, que gestiona seis estados distintos codificados en 8 bits, usando un contador y el sumador SUMLAT (desarrollado en la etapa 1) para la transición de algunos estados. En adición a esto, implementamos una función externa para modificar el estado sin utilizar el sumador.

Las transiciones de estado funcionan de forma que el modo alternativo también puede realizar los cambios de estado empleando el sumador y el estado actual. Se utilizó un contador de 8 bits, y el número de estado interno del contador corresponde con los estados del sistema: del 48 al 55.

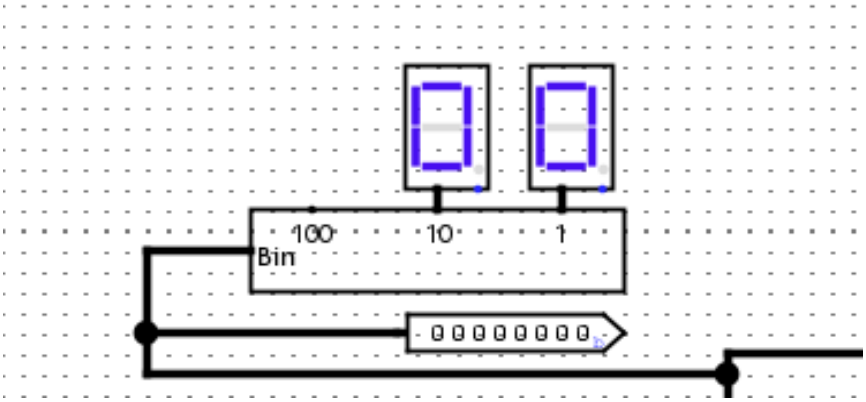
mainEtapa2



Introducción al circuito:

Sobre el lado derecho del contador se encuentra un bus de 8 bits el cual provee como salida el estado actual interno del contador y envía los mismos:

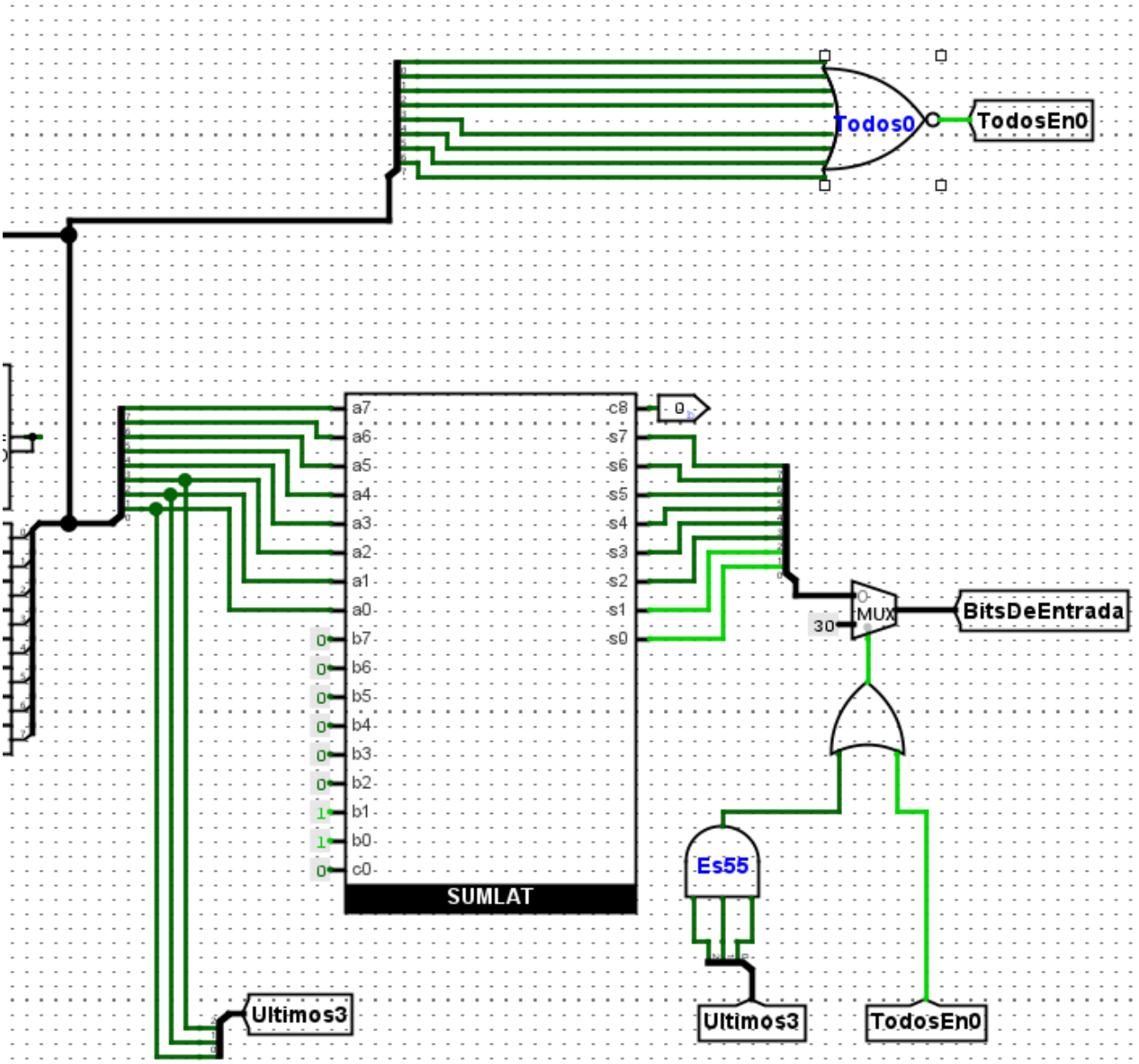
- Al sumador "SUMLAT".
- A un separador de 8 bits que a su vez se conecta con una puerta 'NOR' llamada"TodosO" y verifica que los 8 sean 0 donde su salida es el túnel "TodosEnO".
- Al túnel "Ultimos3" solo los últimos 3 bits menos significativos.
- Y a un display que nos permite como usuarios detectar visualmente en qué estado nos encontramos.



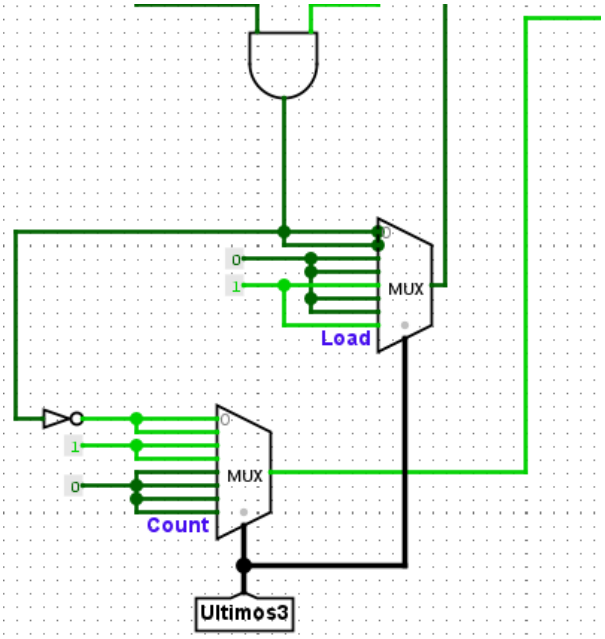
El túnel "Ultimos3" usa a los mismos como bits de selección con los multiplexores que controlan al LOAD y al Count. Solamente se usan esos 3 porque son los únicos que varían entre los 6 estados y nos simplifica la cantidad de entradas de ambos multiplexores ya que nos quedan 2^3 bits de entrada y no 2^8 bits de entrada en el peor de los casos en caso de tomar los 8 bits del estado actual.

Por otra parte los 8 que ingresan como número A al sumador "SUMLAT" se suman con el número constante 3 en los casos mencionados por el enunciado (sumar 3 solo si estamos en el estado 52 en modo normal y sumar 3 si estamos en los estados 48, 49 y 52 en modo alternativo).

A su vez el resultado del sumador ingresa como entrada en la posición 1 al MUX que tiene un solo bit de selección y es una combinación 'OR' entre el túnel "TodosEn0" y el AND "Es55" que determina que los bits del túnel "Ultimos3" sean '111' (Estado 55), en ese caso recibe como entrada en la posición 0 a la constante de valor 48 en base decimal (30 en base hexadecimal). En ambos casos la salida del MUX es el túnel "BitsDeEntrada" el cual se conecta al bus de 8 bits de entrada de datos del contador y se cargan los valores correspondientes cuando se activa la función LOAD.



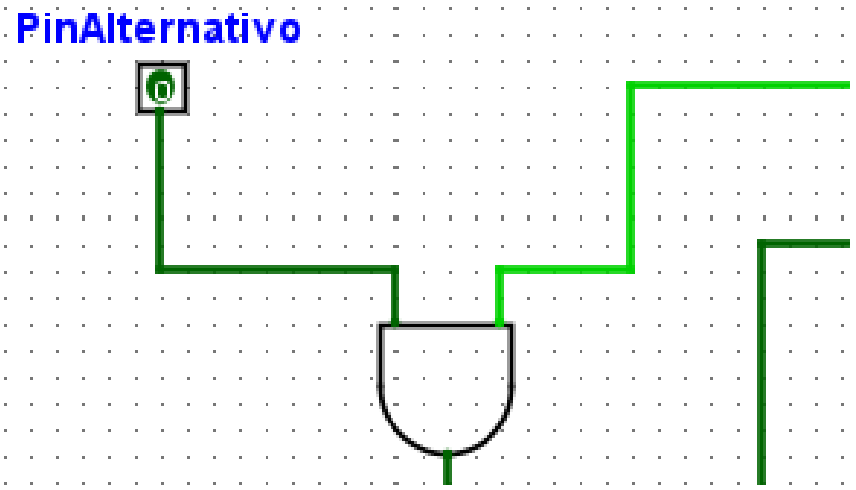
En Modo Normal



- ➔ En el multiplexor correspondiente a la entrada "Count" si la combinación de los últimos 3 bits del estado actual corresponden a 000 (48), 001 (49), 010 (50) o 011 (51), recibe como entrada el valor '1', se activa el mismo y pasará al próximo estado con el pulso de reloj. En cambio si la combinación de los últimos 3 bits del estado actual corresponden a 100 (52), 101 (53), 110 (54) o 111 (55) el multiplexor recibe como entrada el valor '0', no se activa y no hace nada.
- ➔ En el Multiplexor correspondiente a la entrada "Load" si la combinación de los últimos 3 bits del estado actual corresponden a 100 (52) o 111 (55) el multiplexor recibe como entrada el valor '1', se activa junto al túnel "TodosEn0" mediante la puerta 'OR' y carga los valores de entrada de datos correspondientes como próximo estado con el pulso de reloj. En cambio si la combinación de los últimos 3 bits del estado actual corresponden a 000 (48), 001 (49), 010 (50) o 011 (51), 101 (53) o 110 (54) recibe como entrada el valor '0', no se activa y no hace nada.

En Modo Alternativo

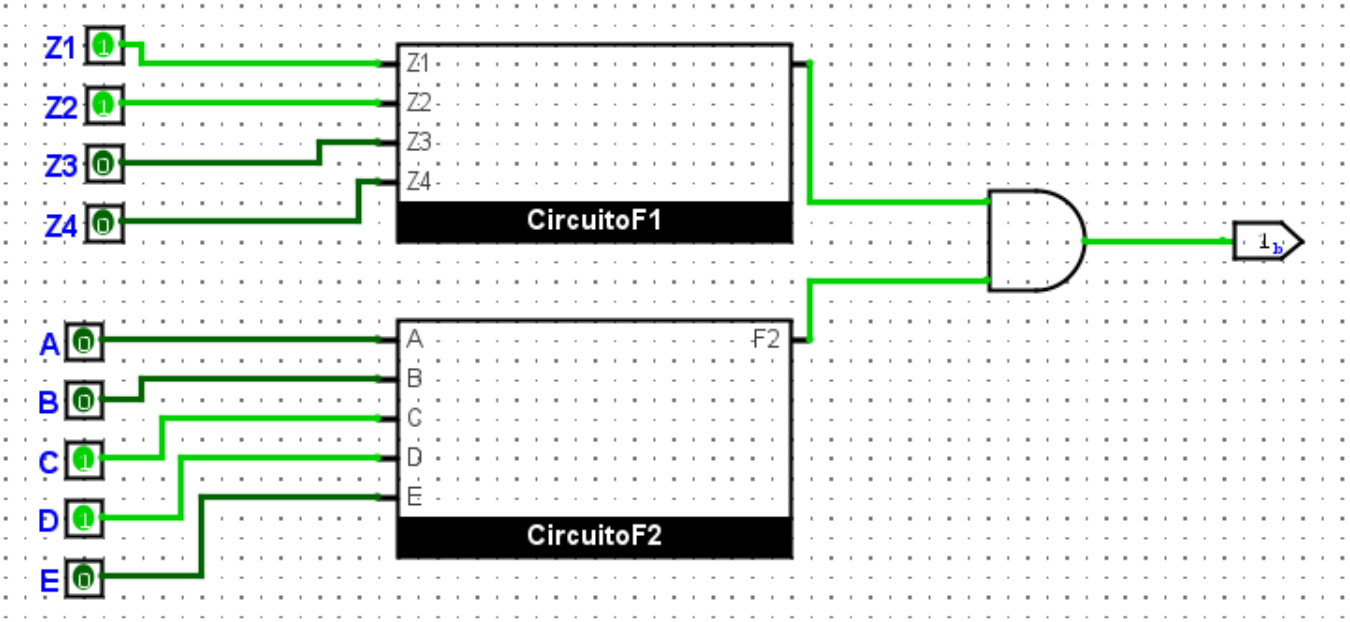
En caso de que el usuario active el "PinAlternativo" y las entradas correspondientes a las tablas de verdad de los circuitos "F1" y "F2" son las indicadas, el circuito ModoAlternativo también se activa y la puerta 'AND' envía los valores de entrada a ambos multiplexores



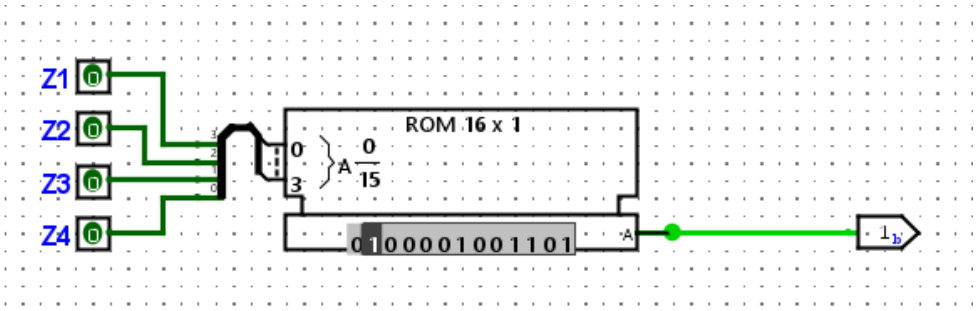
- ➔ En el multiplexor correspondiente a la entrada "Count" ocurre lo mismo que en el "Modo Normal" a diferencia que si la combinación de los últimos 3 bits del estado actual corresponden a 000 (48) y 001 (49) no hace nada y no se activa.
- ➔ En el multiplexor correspondiente a la entrada "Load" ocurre lo mismo que en el "Modo Normal" pero si los últimos 3 bits del estado actual son 000 (48) y 001 (49) también se activa.

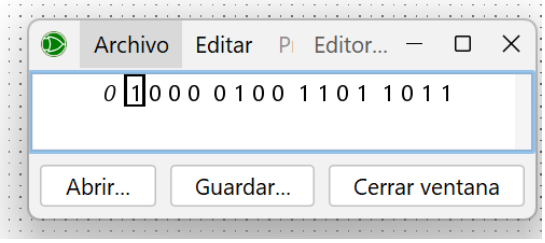
modoalterna

El modo alternativo es un circuito el cual está compuesto por las funciones F1 y F2 y su salida será F1 AND F2, es decir cuando las señales externas Z1,Z2,Z3,Z4 para F1 y A,B,C,D,E para F2 sean equivalentes a las indicadas por las tablas de verdad correspondientes la salida principal dará verdadero (1).



Circuito F1





Cada '1' dentro del editor representa la posición de la tabla de verdad en la que la función dada F1 se hace verdadera.

$$F1 = \overline{Z2}\cdot\overline{Z3}\cdot\overline{Z4}+\overline{Z1}\cdot Z2\cdot\overline{Z3}\cdot Z4+Z1\cdot\overline{Z2}\cdot Z4+Z1\cdot Z2\cdot\overline{Z4}+Z1\cdot Z3\cdot Z4$$

Tabla de Verdad de la Función F1

16 de 16 filas mostradas

Z1	Z2	Z3	Z4	x
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

Circuito F2

El circuito F2 fue creado con la herramienta de Logisim en base a la tabla de verdad provista por la cátedra.

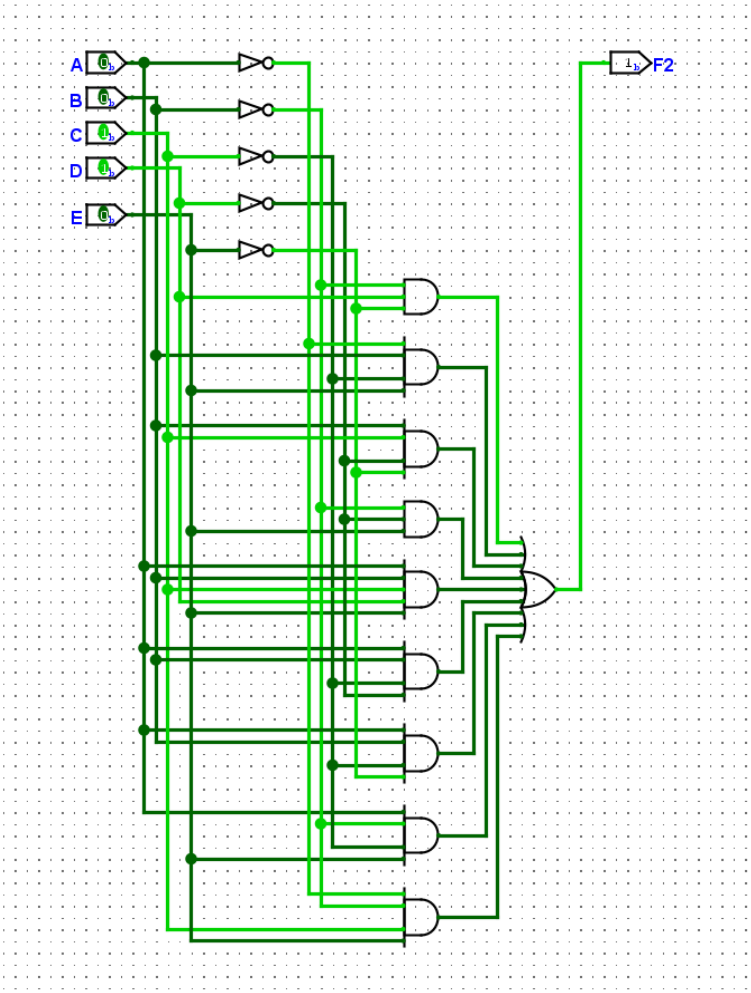


Tabla de Verdad de la Función F2

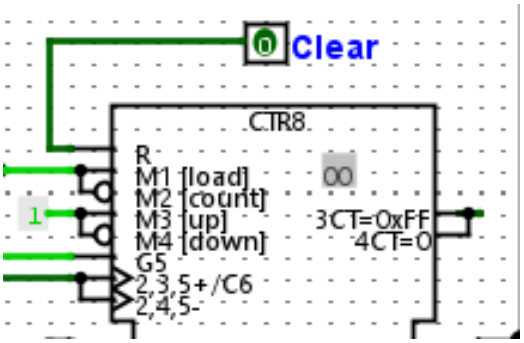
32 de 32 filas mostradas

A	B	C	D	E	F2
0	0	0	0	0	0
0	0	0	0	1	1
0	0	0	1	0	1
0	0	0	1	1	0
0	0	1	0	0	0
0	0	1	0	1	1
0	0	1	1	0	1
0	0	1	1	1	1
0	1	0	0	0	0
0	1	0	0	1	1
0	1	0	1	0	0
0	1	0	1	1	1
0	1	1	0	0	1
0	1	1	0	1	0
0	1	1	1	0	0
0	1	1	1	1	0
1	0	0	0	0	0
1	0	0	0	1	1
1	0	0	1	0	1
1	0	0	1	1	1
1	0	1	0	0	0
1	0	1	0	1	1
1	0	1	1	0	1
1	0	1	1	1	0
1	1	0	0	0	1
1	1	0	0	1	1
1	1	0	1	0	1
1	1	0	1	1	0
1	1	1	0	0	1
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	1

Aclaraciones

- El pin "Clear" se conecta a la señal asincrónica R (Reset) para resetear al circuito del Contador.

- En la entrada M3 ponemos una constante con valor 1 (uno), para contar hacia arriba con cada pulso de reloj correspondiente al "count".



Conclusión

Este proyecto de diseño e implementación de un circuito controlador de estados nos permitió comprender con mayor profundidad y utilizar los conocimientos obtenidos en la materia desde la conceptualización inicial hasta la implementación práctica.

En principio, aprendimos a formular funciones lógicas que representan el comportamiento deseado del sistema, para luego traducirlas en circuitos utilizando software como Logisim-Evolution. Esto nos ayudó a entender la relación entre la etapa del diseño lógico y su representación física en un circuito, así como la importancia de la precisión y claridad en la definición de las funciones.

Además, el realizar el diseño del circuito y seleccionar sus componentes, nos familiarizó con una variedad de componentes electrónicos (puertas, pines de entrada y salida, separadores, multiplexores, túneles, reloj, entre otros) y sus funciones específicas. Esto incluye no sólo comprender cómo funcionan los componentes individuales, sino también cómo interactúan entre sí para lograr el comportamiento deseado del sistema que integra a todos los circuitos implementados.

Finalmente, consideramos importante el uso de etiquetas, y de un cableado comprensible, ordenado, y organizado en túneles. El correcto uso de etiquetas y un cableado prolijo no solo mejoran la estética de la implementación, si no que promueven un entendimiento claro del diseño y funcionamiento para otros usuarios.