Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica

IE0411 - Microelectrónica: Sistemas en Silicio I

Proyecto

Prof. Javier Pacheco Brito

Por:
Marco Chacon Soto, B61868

8 de diciembre de 2020

Índice

Índice de figuras

1. Introducción

El siguiente reporte contiene los resultados obtenidos del anális de los módulos arbiter.v y uart.v con la herramienta Qflow. Entre los datos obtenidos se obtienen el área, la cantidad de compuertas lógicas utilizadas y el timing de Flop to Flop y pin to Flop.

Se incluyen también las observaciones de los resultados obtenidos, el repositorio del git donde se encuentran los archivos y las recomendaciones.

2. Resultados Obtenidos

2.1. Parte A

2.1.1. Contador de 4 bits

Para este módulo se re utilizó el contador de 4 bits realizado anteriormente en la tarea 1. A dicho se módulo se le agregó un bloque de always @(negedge clk) en el cuál se baja la señal rco y se cambió la lógica de los modos con el fin de que funcione tal y como se pide en el enunciado. Además, se agregó la estructura de testbench utilizada en el curso, con los mismos drivers utilizados anteriormente.
En la siguiente figura se observan los resultados de realizar una prueba en modo aleatorio

Figura 1: Prueba random para el contador de 4 bits. Simulado en GTKwave

Como se observa en la figura 1, el contador de 4 bits y el scoreboard tienen el comportamiento solicitado. Los resultados, el makefile y las pruebas para este contador se encuentran dentro de la carpeta part_A.

2.2. Parte B

Se crea el módulo tt_counter, en el cuál se instancian 8 contadores de 4 bits para realizar un contador de 32 bits. Además, se utiliza un operador ternario para asignar los relojes a partir de las señales de rco para todas las etapas, lo cuál indica a los contadores de 4 bits cuándo funcionar. Además se asigna una nueva señal de modo para los bits 3 en adelante de los contadores, con el fin

de que las cuentas se lleven bien si se está en el modo de Q = Q - 3. A continuación se muestran los resultados de la prueba para modos aleatorios del contador de 32 bits.

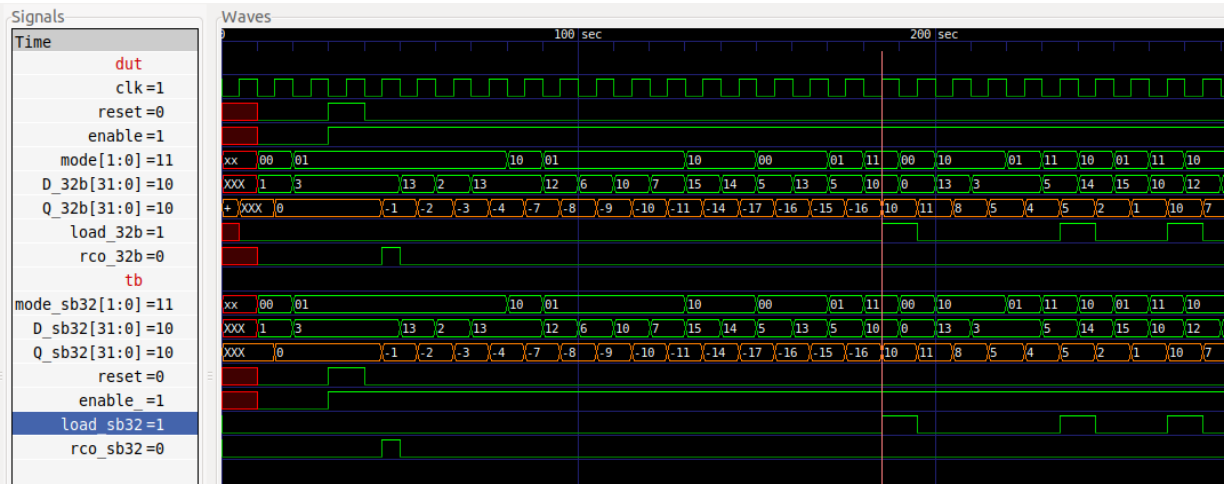

Figura 2: Prueba random para el contador de 32 bits. Simulado en GTKwave

En esta prueba se observan los cambios de modo 00, 01 y 10, donde se obtienen los resultados esperados en las salidas del contador para cada uno de estos modos. También se incluyen los resultados de las pruebas para las señales de reset y enable, así como las pruebas del modo 11 a continuación

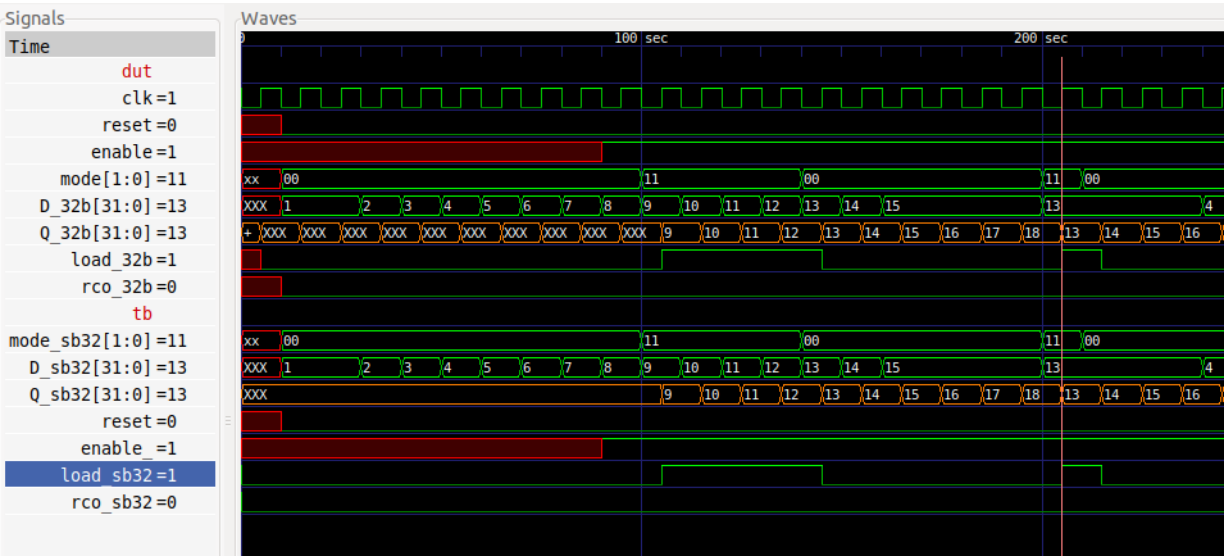

Figura 3: Prueba de enable para el contador de 32 bits. Simulado en GTKwave

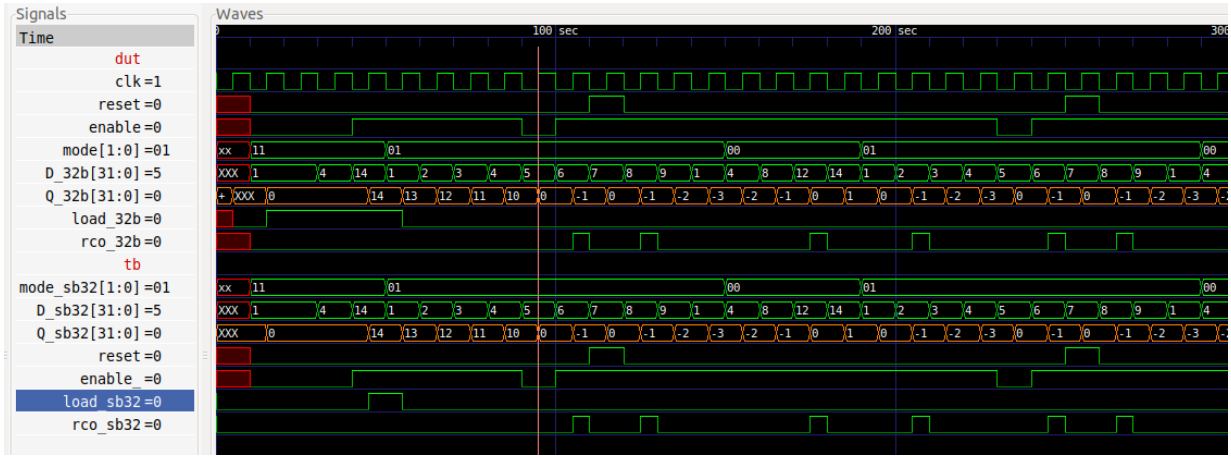

Figura 4: Prueba de reset para el contador de 32 bits. Simulado en GTKwave

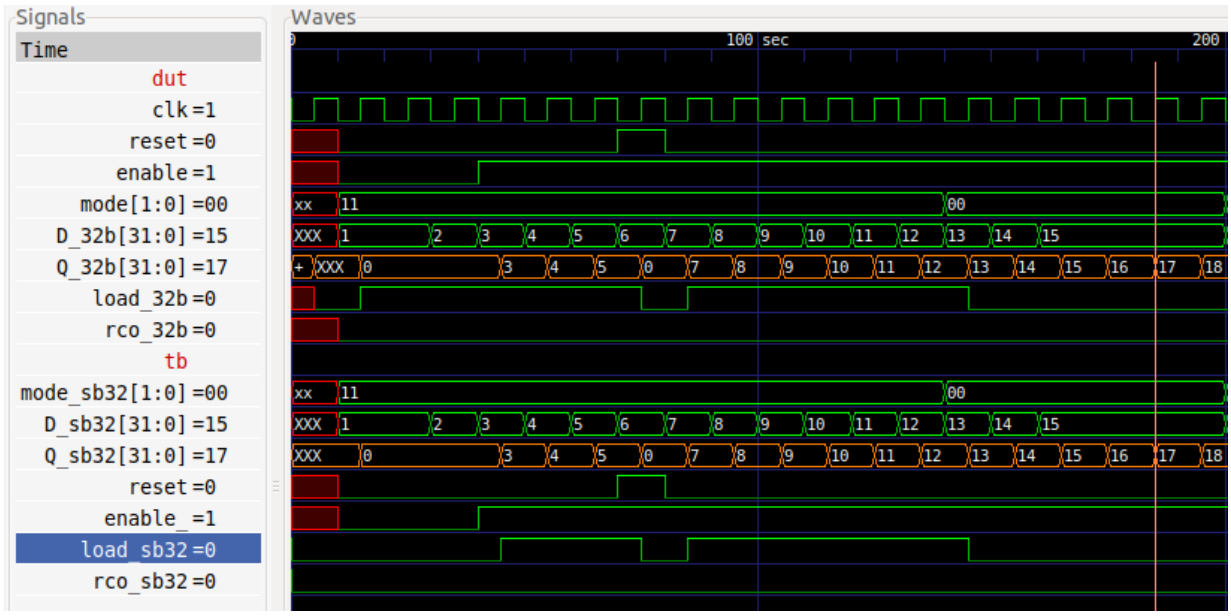

Figura 5: Prueba del modo 11 para el contador de 32 bits. Simulado en GTKwave

En las figuras 3 y 4 se observan los resultados de las pruebas de las señales de enable y reset, respectivamente. En estas pruebas se observa que al hacer toggle de estas señales, el contador de 32 bits continúa con un funcionamiento óptimo. En el modelo de referencia del scoreboard se presenta una complicación con la señal load_sb32, la cuál no se levanta si enable no se encuentra levantada, por motivos de tiempo no fue posible arreglar el funcionamiento de dicha señal, sin embargo se observa que para enable = 1, el comportamiento de esta señal y la del dut son el mismo.
En la figura 5 se muestra el funcionamiento del dut en el modo 11, donde se encuentra en modo de carga y se espera que la señal de load_32b se encuentre en 1. Para obtener dicho comportamiento fue necesario realizar una operación AND entre todas las señales de load de los 8 contadores de 4 bits instanciados. También se observa que cuando hay un reset, esta señal se pone en 0 ya que es una salida.

Los resultados de estas pruebas se encuentran en la carpeta part_B.

2.3. Parte C

Para esta sección se quitó el bloque de always secuencial utilizado en la sección A del módulo counter, con esto la señal rco del contador de 4 bits permanece en alto un ciclo completo de reloj. A continuación se presentan las dos figuras a analizar para esta sección

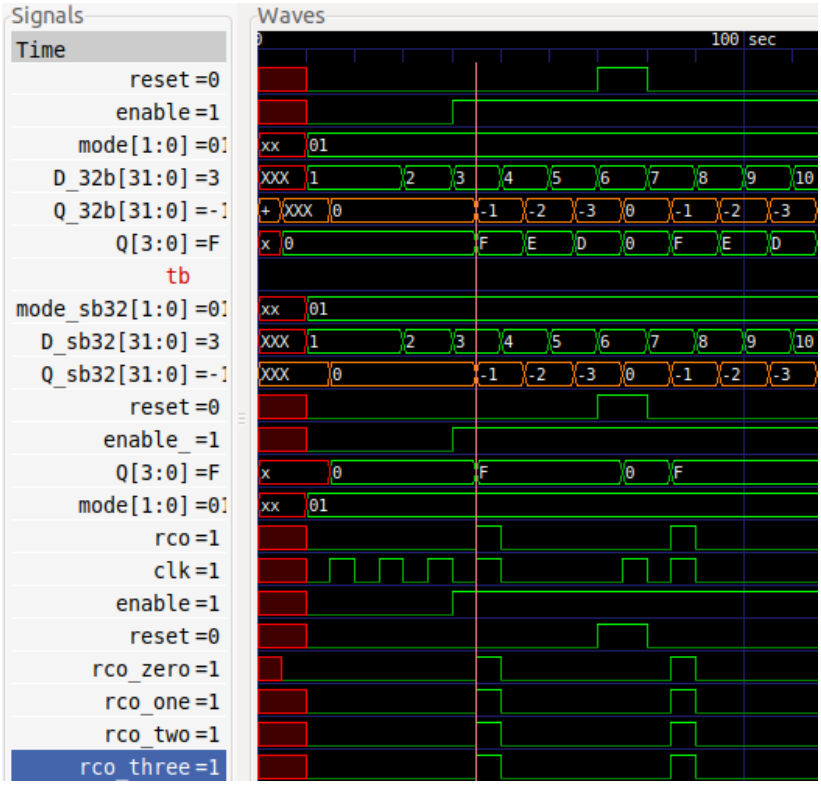

Figura 6: Prueba modo 01 para el contador de 32 bits con el bloque secuencial. Simulado en GTKwave

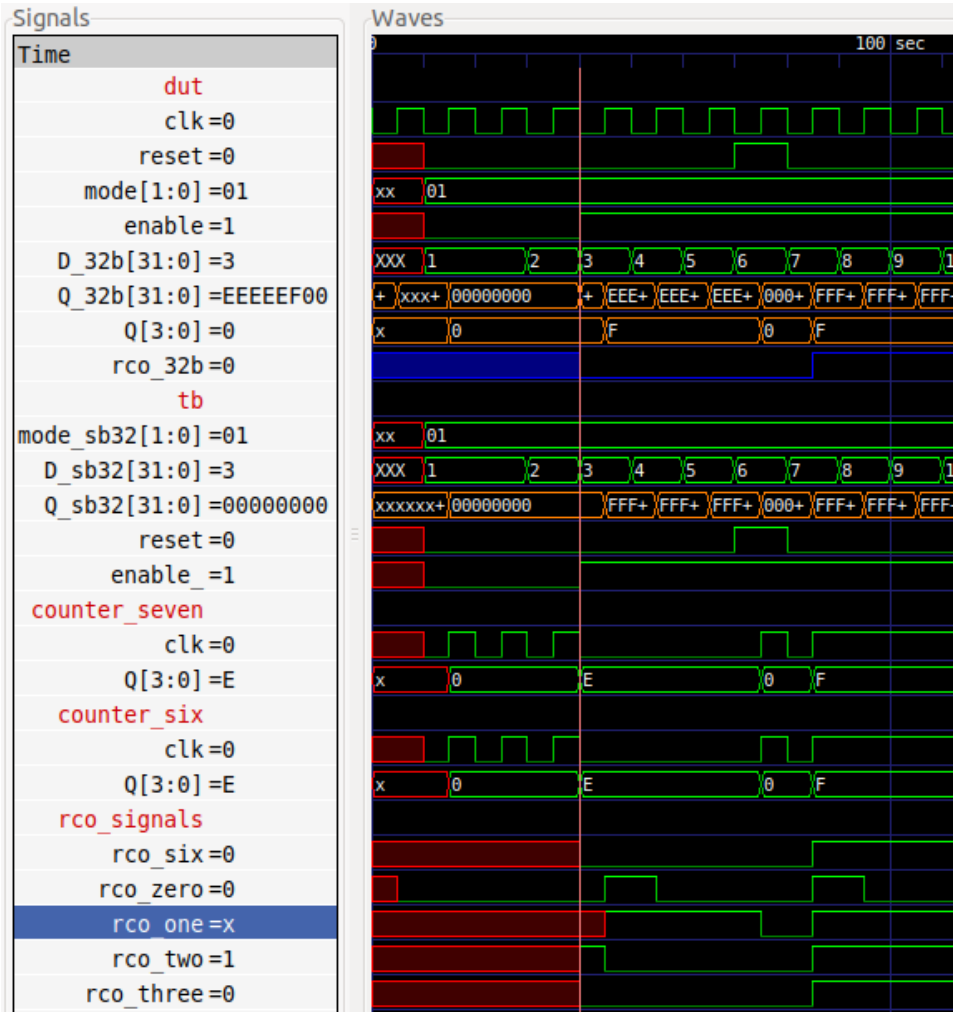

Figura 7: Prueba modo 01 para el contador de 32 bits sin el bloque secuencial. Simulado en GTKwave

Primero se observa en la figura 6 el comportamiento esperado para las señales rco_zero, rco_one, etc. La señal rco_zero es la señal de salida del contador de 4 bits para los 4 bits menos significativos (3:0), esta señal se levanta únicamente medio ciclo de reloj y además se asigna como señal de clk al contador counter_one en el momento que la señal de enable se levante con el fin de sincronizar los contadores y obtener un funcionamiento óptimo. Como se observa, las señales rco de los contadores funcionan de manera adecuada.
Por otra parte se tiene la figura 7, en la cuál se presentan las señales rco_zero, rco_one, rco_two, rco_three y rco_six. Esta vez se observa que, al levantarse la señal rco_zero por un ciclo completo de reloj, los otros contadores van a tener relojes desincronizados, y algunos de estos se van a mantener en alto por más tiempo del que deberían obteniendo así una salida Q_32b incorrecta (EEEEEF00).

Los resultados obtenidos para estas pruebas junto con los divers y el makefile se encuentran dentro de la carpeta part_C.

2.4. Parte D: Contador de 4 bits, librería osu035

Los resultados obtenidos para el módulo counter.v se muestran a continuación.

2.4.1. Compuertas

Las siguientes compuertas se obtuvieron al analizar el archivo counter.rtl.v, el cuál se encuentra dentro de la carpeta synthesis

- **INVX1**: 7
- **NAND2X1**: 6
- **INVX2**: 1
- **NOR2X1**: 3
- **OAI21X1**: 4
- **AOI21X1**: 10
- **XNOR2X1**: 3
- **NAND3X1**: 2
- **AND2X2**: 2
- **BUFX2**: 6
- **DFFPOSX**: 5

2.4.2. Paths y timings

A continuación se presentan los resultados de los paths con mayor y menor retardo, así como la frecuencia máxima del circuito para el módulo counter.v

```
Top 15 maximum delay paths:
Path DFFPOSX1_1/CLK to DFFPOSX1_3/D delay 1419.81 ps
Path DFFPOSX1_1/CLK to DFFPOSX1_4/D delay 1406.99 ps
Path DFFPOSX1_2/CLK to DFFPOSX1_3/D delay 1398.78 ps
Path DFFPOSX1_2/CLK to DFFPOSX1_4/D delay 1377.61 ps
Path DFFPOSX1_3/CLK to DFFPOSX1_3/D delay 1348.05 ps
Path DFFPOSX1_3/CLK to DFFPOSX1_4/D delay 1313.02 ps
Path DFFPOSX1_4/CLK to DFFPOSX1_4/D delay 1258.51 ps
Path DFFPOSX1_2/CLK to DFFPOSX1_2/D delay 1215.32 ps
Path DFFPOSX1_1/CLK to DFFPOSX1_2/D delay 1188.31 ps
Path DFFPOSX1_1/CLK to DFFPOSX1_1/D delay 947.94 ps
Path DFFPOSX1_1/CLK to output pin Q[0] delay 517.321 ps
Path DFFPOSX1_2/CLK to output pin Q[1] delay 515.982 ps
Path DFFPOSX1_4/CLK to output pin Q[3] delay 488.687 ps
Path DFFPOSX1_3/CLK to output pin Q[2] delay 483.851 ps
Path DFFPOSX1_5/CLK to output pin load delay 367.07 ps
Computed maximum clock frequency (zero slack) = 704.321 MHz
```

Figura 8: Paths con mayor retardo, Flop to Flop y frecuencia máxima. Simulado en Qflow

```
Top 15 minimum delay paths:
Path DFFPOSX1_5/CLK to output pin load delay 252.484 ps
Path DFFPOSX1_3/CLK to output pin Q[2] delay 348.388 ps
Path DFFPOSX1_4/CLK to output pin Q[3] delay 353.626 ps
Path DFFPOSX1_2/CLK to output pin Q[1] delay 383.141 ps
Path DFFPOSX1_1/CLK to output pin Q[0] delay 384.583 ps
Path DFFPOSX1_1/CLK to DFFPOSX1_1/D delay 621.68 ps
Path DFFPOSX1_4/CLK to DFFPOSX1_4/D delay 665.467 ps
Path DFFPOSX1_3/CLK to DFFPOSX1_3/D delay 752.994 ps
Path DFFPOSX1_3/CLK to DFFPOSX1_4/D delay 757.266 ps
Path DFFPOSX1_1/CLK to DFFPOSX1_2/D delay 786.435 ps
Path DFFPOSX1_2/CLK to DFFPOSX1_4/D delay 799.31 ps
Path DFFPOSX1_1/CLK to DFFPOSX1_4/D delay 801.546 ps
Path DFFPOSX1_2/CLK to DFFPOSX1_3/D delay 803.578 ps
Path DFFPOSX1_2/CLK to DFFPOSX1_2/D delay 804.829 ps
Path DFFPOSX1_1/CLK to DFFPOSX1_3/D delay 805.128 ps
Design meets minimum hold timing.
```

Figura 9: Paths con menor retardo, Flop to Flop. Simulado en Qflow

```
Top 20 maximum delay paths:
Path input pin mode[1] to DFFPOSX1_1/D delay 705.764 ps
Path input pin mode[1] to DFFPOSX1_2/D delay 698.825 ps
Path input pin mode[1] to DFFPOSX1_4/D delay 694.342 ps
Path input pin mode[1] to DFFPOSX1_3/D delay 673.059 ps
Path input pin mode[0] to DFFPOSX1_1/D delay 666.932 ps
Path input pin mode[0] to DFFPOSX1_2/D delay 659.974 ps
Path input pin mode[0] to DFFPOSX1_4/D delay 655.75 ps
Path input pin mode[1] to DFFPOSX1_5/D delay 646.691 ps
Path input pin mode[0] to DFFPOSX1_3/D delay 634.501 ps
Path input pin mode[0] to DFFPOSX1_5/D delay 607.851 ps
Path input pin reset to DFFPOSX1_1/D delay 415.133 ps
Path input pin reset to DFFPOSX1_2/D delay 415.133 ps
Path input pin reset to DFFPOSX1_3/D delay 415.133 ps
Path input pin reset to DFFPOSX1_4/D delay 415.133 ps
Path input pin enable to DFFPOSX1_1/D delay 316.672 ps
Path input pin enable to DFFPOSX1_2/D delay 316.672 ps
Path input pin enable to DFFPOSX1_3/D delay 316.672 ps
Path input pin enable to DFFPOSX1_4/D delay 316.672 ps
Path input pin D[3] to DFFPOSX1_4/D delay 219.663 ps
Path input pin D[2] to DFFPOSX1_3/D delay 207.47 ps
```

Figura 10: Paths con mayor retardo, pin to Flop. Simulado en Qflow

```
Top 20 minimum delay paths:
Path input pin clk to DFFPOSX1_5/CLK delay 0 ps
Path input pin clk to DFFPOSX1_4/CLK delay 0 ps
Path input pin clk to DFFPOSX1_3/CLK delay 0 ps
Path input pin clk to DFFPOSX1_2/CLK delay 0 ps
Path input pin clk to DFFPOSX1_1/CLK delay 0 ps
Path input pin D[1] to DFFPOSX1_2/D delay 174.524 ps
Path input pin D[0] to DFFPOSX1_1/D delay 174.545 ps
Path input pin reset to DFFPOSX1_5/D delay 195.353 ps
Path input pin D[2] to DFFPOSX1_3/D delay 200.215 ps
Path input pin D[3] to DFFPOSX1_4/D delay 211.294 ps
Path input pin mode[1] to DFFPOSX1_2/D delay 278.866 ps
Path input pin mode[0] to DFFPOSX1_1/D delay 305.109 ps
Path input pin enable to DFFPOSX1_4/D delay 314.623 ps
Path input pin enable to DFFPOSX1_3/D delay 314.623 ps
Path input pin enable to DFFPOSX1_2/D delay 314.623 ps
Path input pin enable to DFFPOSX1_1/D delay 314.623 ps
Path input pin mode[1] to DFFPOSX1_1/D delay 345.313 ps
Path input pin mode[0] to DFFPOSX1_2/D delay 346.635 ps
Path input pin mode[0] to DFFPOSX1_3/D delay 346.681 ps
Path input pin reset to DFFPOSX1_4/D delay 347.841 ps
```

Figura 11: Paths con menor retardo, pin to Flop. Simulado en Qflow

2.4.3. Layout y área

Por último se obtienen los datos de área y se presenta el layout obtenido a partir de utilizar la herramienta Qflow

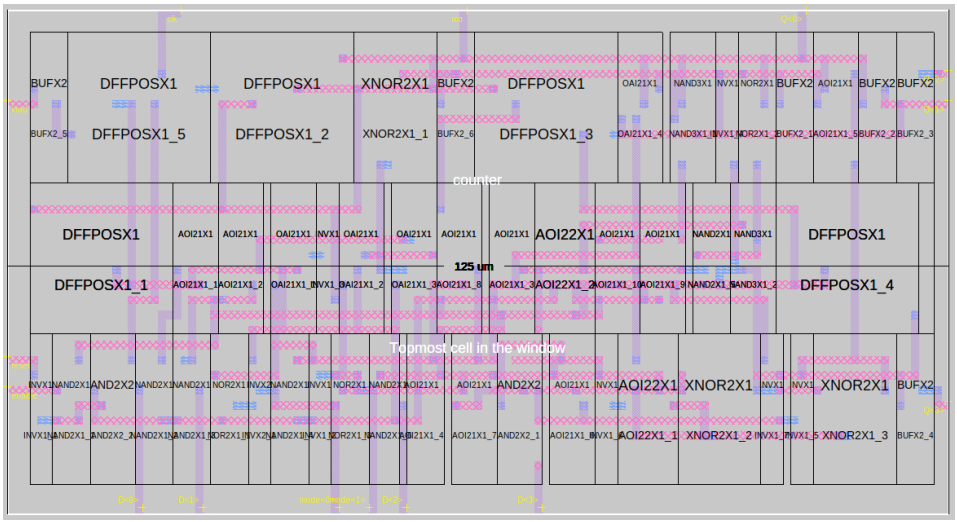

Figura 12: Longitud horizontal del layout del módulo counter. Simulado en Qflow

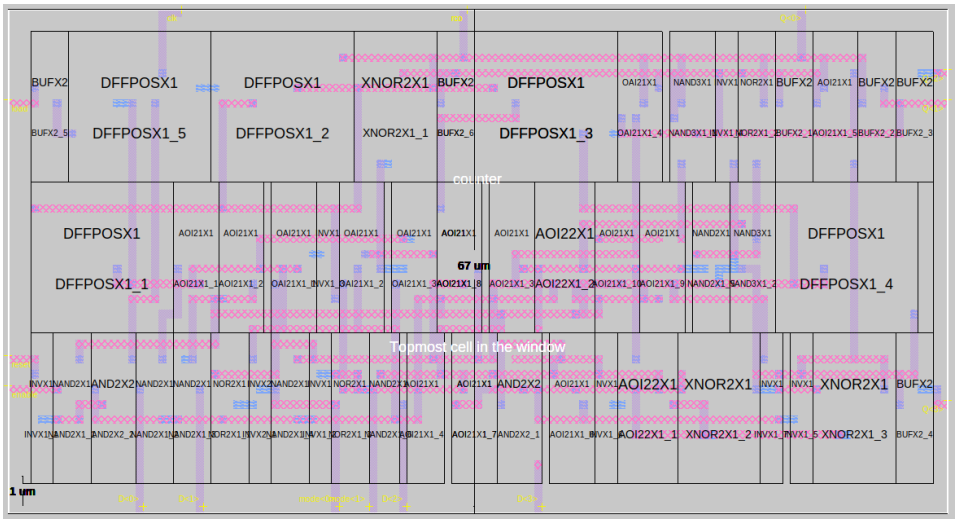

Figura 13: Longitud vertical del layout del módulo counter. Simulado en Qflow

2.5. Parte D: Contador de 4 bits, librería osu050.

Para esta sección fue necesario modificar el archivo qflow_varsh.sh y volver a correr el código utilizado en la parte anterior, sin embargo, esta vez no se utiliza el comando route ya que presenta errores.

2.5.1. Compuertas

La cantidad de compuertas que requiere el diseño obtenidas a partir del archivo arbiter.rtl.v se muestran a continuación

- **INVX1**: 7
- **NAND2X1**: 9
- **INVX2**: 1
- **NOR2X1**: 3
- **OAI21X1**: 4
- **AOI21X1**: 10
- **XNOR2X1**: 1
- **NAND3X1**: 3
- **AND2X2**: 2
- **BUFX2**: 6
- **DFFPOSX**: 5

2.5.2. Paths y timings

A continuación se presentan los resultados de los paths con mayor y menor retardo, así como la frecuencia máxima del circuito para el módulo counter.v con la libreria osu050

```
Top 9 maximum delay paths:
Path DFFPOSX1_1/CLK to DFFPOSX1_4/D delay 1406.79 ps
Path DFFPOSX1_2/CLK to DFFPOSX1_4/D delay 1384.48 ps
Path DFFPOSX1_3/CLK to DFFPOSX1_4/D delay 1367.53 ps
Path DFFPOSX1_1/CLK to DFFPOSX1_3/D delay 1349.29 ps
Path DFFPOSX1_2/CLK to DFFPOSX1_3/D delay 1336.07 ps
Path DFFPOSX1_1/CLK to output pin Q[0] delay 602.118 ps
Path DFFPOSX1_2/CLK to output pin Q[1] delay 586.959 ps
Path DFFPOSX1_3/CLK to output pin Q[2] delay 575.815 ps
Path DFFPOSX1_4/CLK to output pin Q[3] delay 573.174 ps
Computed maximum clock frequency (zero slack) = 710.839 MHz
```

Figura 14: Paths con mayor retardo, Flop to Flop y frecuencia máxima (librería osu050). Simulado en Qflow

```
Top 9 minimum delay paths:
Path DFFPOSX1_4/CLK to output pin Q[3] delay 368.491 ps
Path DFFPOSX1_3/CLK to output pin Q[2] delay 439.923 ps
Path DFFPOSX1_2/CLK to output pin Q[1] delay 467.867 ps
Path DFFPOSX1_1/CLK to output pin Q[0] delay 484.838 ps
Path DFFPOSX1_2/CLK to DFFPOSX1_3/D delay 974.83 ps
Path DFFPOSX1_1/CLK to DFFPOSX1_3/D delay 990.488 ps
Path DFFPOSX1_3/CLK to DFFPOSX1_4/D delay 1000.44 ps
Path DFFPOSX1_2/CLK to DFFPOSX1_4/D delay 1020.68 ps
Path DFFPOSX1_1/CLK to DFFPOSX1_4/D delay 1036.33 ps
Design meets minimum hold timing.
```

Figura 15: Paths con menor retardo, Flop to Flop (librería osu050). Simulado en Qflow

```
Top 20 maximum delay paths:
Path input pin mode[1] to DFFPOSX1_2/D delay 972.259 ps
Path input pin mode[1] to DFFPOSX1_1/D delay 971.576 ps
Path input pin mode[1] to DFFPOSX1_3/D delay 948.451 ps
Path input pin reset to DFFPOSX1_4/D delay 916.847 ps
Path input pin mode[1] to DFFPOSX1_5/D delay 916.183 ps
Path input pin enable to DFFPOSX1_4/D delay 901.982 ps
Path input pin mode[0] to DFFPOSX1_1/D delay 869.697 ps
Path input pin mode[0] to DFFPOSX1_2/D delay 869.424 ps
Path input pin mode[0] to DFFPOSX1_3/D delay 816.461 ps
Path input pin mode[0] to DFFPOSX1_5/D delay 813.847 ps
Path input pin reset to DFFPOSX1_1/D delay 465.266 ps
Path input pin reset to DFFPOSX1_2/D delay 465.266 ps
Path input pin enable to DFFPOSX1_1/D delay 396.906 ps
Path input pin enable to DFFPOSX1_2/D delay 396.906 ps
Path input pin mode[1] to DFFPOSX1_4/D delay 395.989 ps
Path input pin mode[0] to DFFPOSX1_4/D delay 378.409 ps
Path input pin reset to DFFPOSX1_5/D delay 230.869 ps
Path input pin D[0] to DFFPOSX1_1/D delay 207.911 ps
Path input pin D[1] to DFFPOSX1_2/D delay 203.817 ps
Path input pin clk to DFFPOSX1_1/CLK delay 0 ps
```

Figura 16: Paths con mayor retardo, pin to Flop (librería osu050). Simulado en Qflow

```
Top 20 minimum delay paths:
Path input pin clk to DFFPOSX1_5/CLK delay 0 ps
Path input pin clk to DFFPOSX1_4/CLK delay 0 ps
Path input pin clk to DFFPOSX1_3/CLK delay 0 ps
Path input pin clk to DFFPOSX1_2/CLK delay 0 ps
Path input pin clk to DFFPOSX1_1/CLK delay 0 ps
Path input pin mode[1] to DFFPOSX1_3/D delay 168.114 ps
Path input pin D[1] to DFFPOSX1_2/D delay 192.717 ps
Path input pin D[0] to DFFPOSX1_1/D delay 195.106 ps
Path input pin reset to DFFPOSX1_5/D delay 230.869 ps
Path input pin mode[0] to DFFPOSX1_4/D delay 307.66 ps
Path input pin mode[1] to DFFPOSX1_4/D delay 325.058 ps
Path input pin mode[0] to DFFPOSX1_3/D delay 333.894 ps
Path input pin enable to DFFPOSX1_4/D delay 347.077 ps
Path input pin enable to DFFPOSX1_2/D delay 347.077 ps
Path input pin enable to DFFPOSX1_1/D delay 347.077 ps
Path input pin mode[0] to DFFPOSX1_1/D delay 363.956 ps
Path input pin mode[1] to DFFPOSX1_2/D delay 382.838 ps
Path input pin reset to DFFPOSX1_4/D delay 403.873 ps
Path input pin reset to DFFPOSX1_2/D delay 403.873 ps
Path input pin reset to DFFPOSX1_1/D delay 403.873 ps
```

Figura 17: Paths con menor retardo, pin to Flop (librería osu050). Simulado en Qflow

2.5.3. Layout y área

Por último se obtienen los datos de área y se presenta el layout obtenido a partir de utilizar la herramienta Qflow

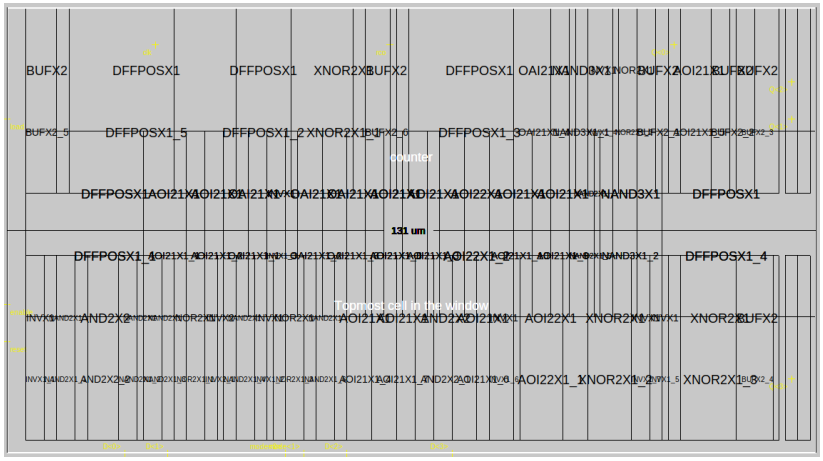

Figura 18: Longitud horizontal del layout del módulo counter (librería osu050). Simulado en Qflow

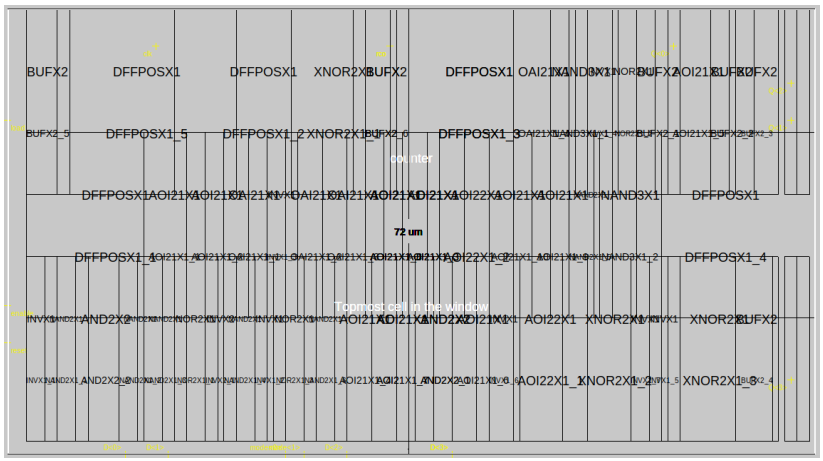

Figura 19: Longitud vertical del layout del módulo counter (librería osu050). Simulado en Qflow

2.6. Parte D: Contador de 32 bits, librería osu035

Los resultados obtenidos para el módulo tt_counter.v se muestran a continuación.
Estos se encuentran dentro de la carpeta part_D32.

2.6.1. Compuertas

Las siguientes compuertas se obtuvieron al analizar el archivo tt_counter.rtl.v, el cuál se encuentra dentro de la carpeta synthesis

- **BUFX4**: 9
- **AND2X2**: 18
- **NAND3X1**: 26
- **NOR2X1**: 34
- **INVX1**: 44

- **NAND2X1**: 58
- **INVX8**: 1
- **BUFX2**: 34
- **INVX2**: 16
- **OAI21X1**: 32
- **AOI21X1**: 64
- **XNOR2X1**: 24
- **DFFPOSX**: 40

2.6.2. Paths y timings

A continuación se presentan los resultados de los paths con mayor y menor retardo, así como la frecuencia máxima del circuito para el módulo tt_counter.v

```
Top 20 maximum delay paths:
Path DFFPOSX1_16/CLK to DFFPOSX1_19/D delay 1384.23 ps
Path DFFPOSX1_21/CLK to DFFPOSX1_24/D delay 1384.23 ps
Path DFFPOSX1_1/CLK to DFFPOSX1_4/D delay 1384.23 ps
Path DFFPOSX1_6/CLK to DFFPOSX1_9/D delay 1384.23 ps
Path DFFPOSX1_26/CLK to DFFPOSX1_29/D delay 1384.23 ps
Path DFFPOSX1_31/CLK to DFFPOSX1_34/D delay 1384.23 ps
Path DFFPOSX1_11/CLK to DFFPOSX1_14/D delay 1384.23 ps
Path DFFPOSX1_36/CLK to DFFPOSX1_39/D delay 1384.23 ps
Path DFFPOSX1_16/CLK to DFFPOSX1_18/D delay 1372.67 ps
Path DFFPOSX1_21/CLK to DFFPOSX1_23/D delay 1372.67 ps
Path DFFPOSX1_1/CLK to DFFPOSX1_3/D delay 1372.67 ps
Path DFFPOSX1_6/CLK to DFFPOSX1_8/D delay 1372.67 ps
Path DFFPOSX1_26/CLK to DFFPOSX1_28/D delay 1372.67 ps
Path DFFPOSX1_31/CLK to DFFPOSX1_33/D delay 1372.67 ps
Path DFFPOSX1_11/CLK to DFFPOSX1_13/D delay 1372.67 ps
Path DFFPOSX1_36/CLK to DFFPOSX1_38/D delay 1372.67 ps
Path DFFPOSX1_17/CLK to DFFPOSX1_19/D delay 1354.88 ps
Path DFFPOSX1_22/CLK to DFFPOSX1_24/D delay 1354.88 ps
Path DFFPOSX1_2/CLK to DFFPOSX1_4/D delay 1354.88 ps
Path DFFPOSX1_7/CLK to DFFPOSX1_9/D delay 1354.88 ps
Computed maximum clock frequency (zero slack) = 722.424 MHz
```

Figura 20: Paths con mayor retardo, Flop to Flop y frecuencia máxima. Simulado en Qflow

```
Top 20 minimum delay paths:
Path DFFPOSX1_38/CLK to output pin Q_32b[2] delay 348.388 ps
Path DFFPOSX1_13/CLK to output pin Q_32b[6] delay 348.388 ps
Path DFFPOSX1_33/CLK to output pin Q_32b[10] delay 348.388 ps
Path DFFPOSX1_28/CLK to output pin Q_32b[14] delay 348.388 ps
Path DFFPOSX1_8/CLK to output pin Q_32b[18] delay 348.388 ps
Path DFFPOSX1_3/CLK to output pin Q_32b[22] delay 348.388 ps
Path DFFPOSX1_23/CLK to output pin Q_32b[26] delay 348.388 ps
Path DFFPOSX1_18/CLK to output pin Q_32b[30] delay 348.388 ps
Path DFFPOSX1_39/CLK to output pin Q_32b[3] delay 353.626 ps
Path DFFPOSX1_14/CLK to output pin Q_32b[7] delay 353.626 ps
Path DFFPOSX1_34/CLK to output pin Q_32b[11] delay 353.626 ps
Path DFFPOSX1_29/CLK to output pin Q_32b[15] delay 353.626 ps
Path DFFPOSX1_9/CLK to output pin Q_32b[19] delay 353.626 ps
Path DFFPOSX1_4/CLK to output pin Q_32b[23] delay 353.626 ps
Path DFFPOSX1_24/CLK to output pin Q_32b[27] delay 353.626 ps
Path DFFPOSX1_19/CLK to output pin Q_32b[31] delay 353.626 ps
Path DFFPOSX1_37/CLK to output pin Q_32b[1] delay 383.141 ps
Path DFFPOSX1_12/CLK to output pin Q_32b[5] delay 383.141 ps
Path DFFPOSX1_32/CLK to output pin Q_32b[9] delay 383.141 ps
Path DFFPOSX1_27/CLK to output pin Q_32b[13] delay 383.141 ps
Design meets minimum hold timing.
```

Figura 21: Paths con menor retardo, Flop to Flop. Simulado en Qflow

```
Top 20 maximum delay paths:
Path input pin mode[1] to DFFPOSX1_11/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_12/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_13/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_14/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_15/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_31/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_32/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_33/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_34/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_35/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_26/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_27/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_28/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_29/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_30/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_6/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_7/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_8/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_9/CLK delay 1747.9 ps
Path input pin mode[1] to DFFPOSX1_10/CLK delay 1747.9 ps
```

Figura 22: Paths con mayor retardo, pin to Flop. Simulado en Qflow

```
Top 20 minimum delay paths:
Path input pin clk to DFFPOSX1_40/CLK delay 0 ps
Path input pin clk to DFFPOSX1_39/CLK delay 0 ps
Path input pin clk to DFFPOSX1_38/CLK delay 0 ps
Path input pin clk to DFFPOSX1_37/CLK delay 0 ps
Path input pin clk to DFFPOSX1_36/CLK delay 0 ps
Path input pin D_32b[29] to DFFPOSX1_17/D delay 174.524 ps
Path input pin D_32b[25] to DFFPOSX1_22/D delay 174.524 ps
Path input pin D_32b[21] to DFFPOSX1_2/D delay 174.524 ps
Path input pin D_32b[17] to DFFPOSX1_7/D delay 174.524 ps
Path input pin D_32b[13] to DFFPOSX1_27/D delay 174.524 ps
Path input pin D_32b[9] to DFFPOSX1_32/D delay 174.524 ps
Path input pin D_32b[5] to DFFPOSX1_12/D delay 174.524 ps
Path input pin D_32b[1] to DFFPOSX1_37/D delay 174.524 ps
Path input pin D_32b[28] to DFFPOSX1_16/D delay 174.545 ps
Path input pin D_32b[24] to DFFPOSX1_21/D delay 174.545 ps
Path input pin D_32b[20] to DFFPOSX1_1/D delay 174.545 ps
Path input pin D_32b[16] to DFFPOSX1_6/D delay 174.545 ps
Path input pin D_32b[12] to DFFPOSX1_26/D delay 174.545 ps
Path input pin D_32b[8] to DFFPOSX1_31/D delay 174.545 ps
Path input pin D_32b[4] to DFFPOSX1_11/D delay 174.545 ps
```

Figura 23: Paths con menor retardo, pin to Flop. Simulado en Qflow

2.6.3. Layout y área

Por último se obtienen los datos de área y se presenta el layout obtenido a partir de utilizar la herramienta Qflow

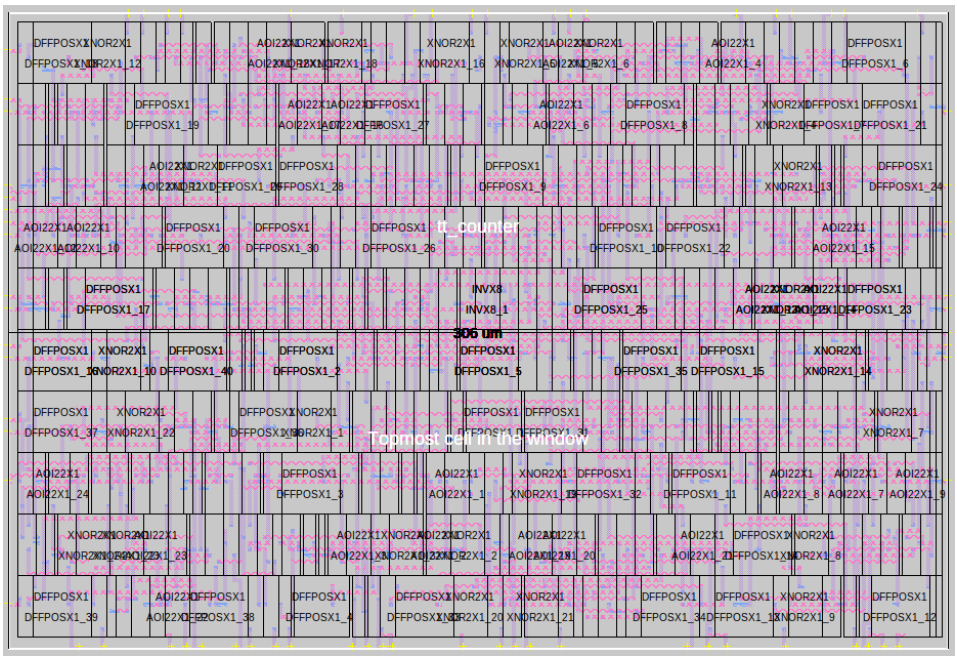

Figura 24: Longitud horizontal del layout del módulo counter. Simulado en Qflow

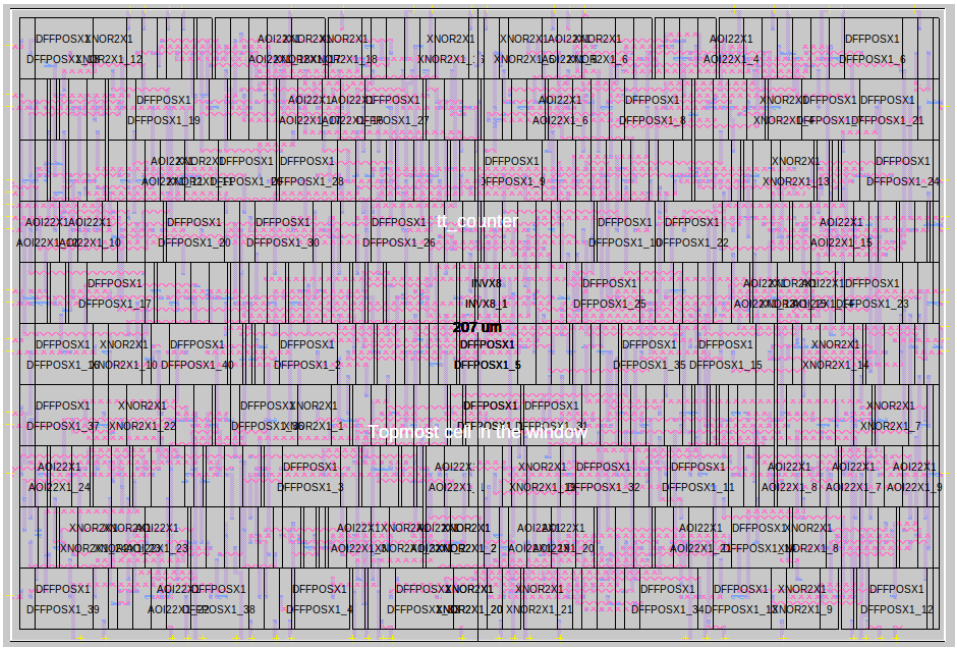

Figura 25: Longitud vertical del layout del módulo counter. Simulado en Qflow

2.7. Parte D: Contador de 32 bits, librería osu050.

Para esta sección nuevamente fue necesario modificar el archivo qflow_varsh.sh y volver a correr el código utilizado en la parte anterior, sin embargo, esta vez no se utiliza el comando route ya que presenta errores.

2.7.1. Compuertas

La cantidad de compuertas que requiere el diseño obtenidas a partir del archivo arbiter.rtl.v se muestran a continuación

- **AND2X2**: 18
- **NAND3X1**: 35
- **NOR2X1**: 33
- **INVX1**: 74
- **INVX2**: 2
- **XNOR2X1**: 16
- **NAND2X1**: 51
- **BUFX2**: 43
- **OAI21X1**: 39
- **AOI21X1**: 64
- **DFFPOSX**: 40

2.7.2. Paths y timings

A continuación se presentan los resultados de los paths con mayor y menor retardo, así como la frecuencia máxima del circuito para el módulo tt_counter.v con la libreria osu050

```
Top 9 maximum delay paths:
Path DFFPOSX1_1/CLK to DFFPOSX1_4/D delay 1406.79 ps
Path DFFPOSX1_2/CLK to DFFPOSX1_4/D delay 1384.48 ps
Path DFFPOSX1_3/CLK to DFFPOSX1_4/D delay 1367.53 ps
Path DFFPOSX1_1/CLK to DFFPOSX1_3/D delay 1349.29 ps
Path DFFPOSX1_2/CLK to DFFPOSX1_3/D delay 1336.07 ps
Path DFFPOSX1_1/CLK to output pin Q[0] delay 602.118 ps
Path DFFPOSX1_2/CLK to output pin Q[1] delay 586.959 ps
Path DFFPOSX1_3/CLK to output pin Q[2] delay 575.815 ps
Path DFFPOSX1_4/CLK to output pin Q[3] delay 573.174 ps
Computed maximum clock frequency (zero slack) = 710.839 MHz
```

Figura 26: Paths con mayor retardo, Flop to Flop y frecuencia máxima (librería osu050). Simulado en Qflow

```
Top 20 minimum delay paths:
Path DFFPOSX1_38/CLK to output pin Q_32b[2] delay 458.904 ps
Path DFFPOSX1_13/CLK to output pin Q_32b[6] delay 458.904 ps
Path DFFPOSX1_33/CLK to output pin Q_32b[10] delay 458.904 ps
Path DFFPOSX1_28/CLK to output pin Q_32b[14] delay 458.904 ps
Path DFFPOSX1_8/CLK to output pin Q_32b[18] delay 458.904 ps
Path DFFPOSX1_3/CLK to output pin Q_32b[22] delay 458.904 ps
Path DFFPOSX1_23/CLK to output pin Q_32b[26] delay 458.904 ps
Path DFFPOSX1_18/CLK to output pin Q_32b[30] delay 458.904 ps
Path DFFPOSX1_39/CLK to output pin Q_32b[3] delay 484.266 ps
Path DFFPOSX1_14/CLK to output pin Q_32b[7] delay 484.266 ps
Path DFFPOSX1_34/CLK to output pin Q_32b[11] delay 484.266 ps
Path DFFPOSX1_29/CLK to output pin Q_32b[15] delay 484.266 ps
Path DFFPOSX1_9/CLK to output pin Q_32b[19] delay 484.266 ps
Path DFFPOSX1_4/CLK to output pin Q_32b[23] delay 484.266 ps
Path DFFPOSX1_24/CLK to output pin Q_32b[27] delay 484.266 ps
Path DFFPOSX1_19/CLK to output pin Q_32b[31] delay 484.266 ps
Path DFFPOSX1_36/CLK to output pin Q_32b[0] delay 490.113 ps
Path DFFPOSX1_11/CLK to output pin Q_32b[4] delay 490.113 ps
Path DFFPOSX1_31/CLK to output pin Q_32b[8] delay 490.113 ps
Path DFFPOSX1_26/CLK to output pin Q_32b[12] delay 490.113 ps
Design meets minimum hold timing.
```

Figura 27: Paths con menor retardo, Flop to Flop (librería osu050). Simulado en Qflow

```
Top 20 maximum delay paths:
Path input pin mode[1] to DFFPOSX1_11/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_12/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_13/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_14/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_15/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_31/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_32/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_33/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_34/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_35/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_26/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_27/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_28/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_29/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_30/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_6/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_7/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_8/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_9/CLK delay 1964.2 ps
Path input pin mode[1] to DFFPOSX1_10/CLK delay 1964.2 ps
```

Figura 28: Paths con mayor retardo, pin to Flop (librería osu050). Simulado en Qflow

```
Top 20 minimum delay paths:
Path input pin clk to DFFPOSX1_40/CLK delay 0 ps
Path input pin clk to DFFPOSX1_39/CLK delay 0 ps
Path input pin clk to DFFPOSX1_38/CLK delay 0 ps
Path input pin clk to DFFPOSX1_37/CLK delay 0 ps
Path input pin clk to DFFPOSX1_36/CLK delay 0 ps
Path input pin D_32b[29] to DFFPOSX1_17/D delay 192.717 ps
Path input pin D_32b[25] to DFFPOSX1_22/D delay 192.717 ps
Path input pin D_32b[21] to DFFPOSX1_2/D delay 192.717 ps
Path input pin D_32b[17] to DFFPOSX1_7/D delay 192.717 ps
Path input pin D_32b[13] to DFFPOSX1_27/D delay 192.717 ps
Path input pin D_32b[9] to DFFPOSX1_32/D delay 192.717 ps
Path input pin D_32b[5] to DFFPOSX1_12/D delay 192.717 ps
Path input pin D_32b[1] to DFFPOSX1_37/D delay 192.717 ps
Path input pin D_32b[28] to DFFPOSX1_16/D delay 195.106 ps
Path input pin D_32b[24] to DFFPOSX1_21/D delay 195.106 ps
Path input pin D_32b[20] to DFFPOSX1_1/D delay 195.106 ps
Path input pin D_32b[16] to DFFPOSX1_6/D delay 195.106 ps
Path input pin D_32b[12] to DFFPOSX1_26/D delay 195.106 ps
Path input pin D_32b[8] to DFFPOSX1_31/D delay 195.106 ps
Path input pin D_32b[4] to DFFPOSX1_11/D delay 195.106 ps
```

Figura 29: Paths con menor retardo, pin to Flop (librería osu050). Simulado en Qflow

2.7.3. Layout y área

Por último se obtienen los datos de área y se presenta el layout obtenido a partir de utilizar la herramienta Qflow

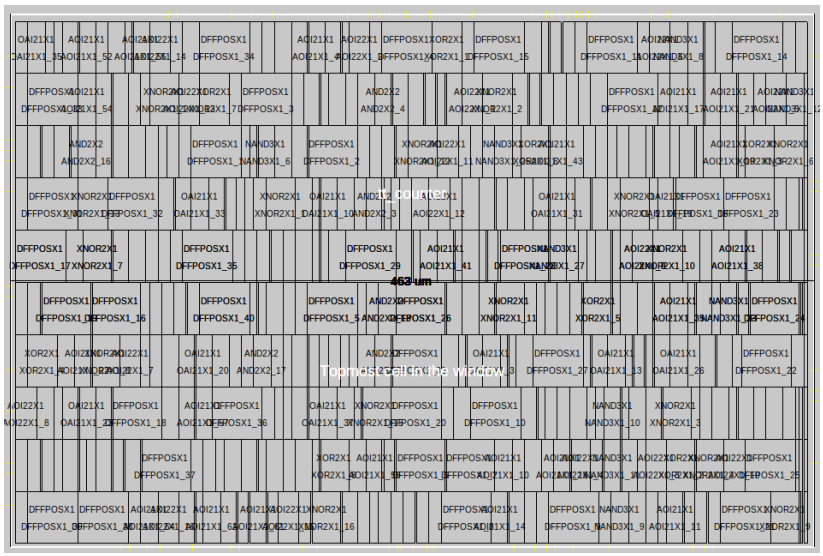

Figura 30: Longitud horizontal del layout del módulo counter (librería osu050). Simulado en Qflow

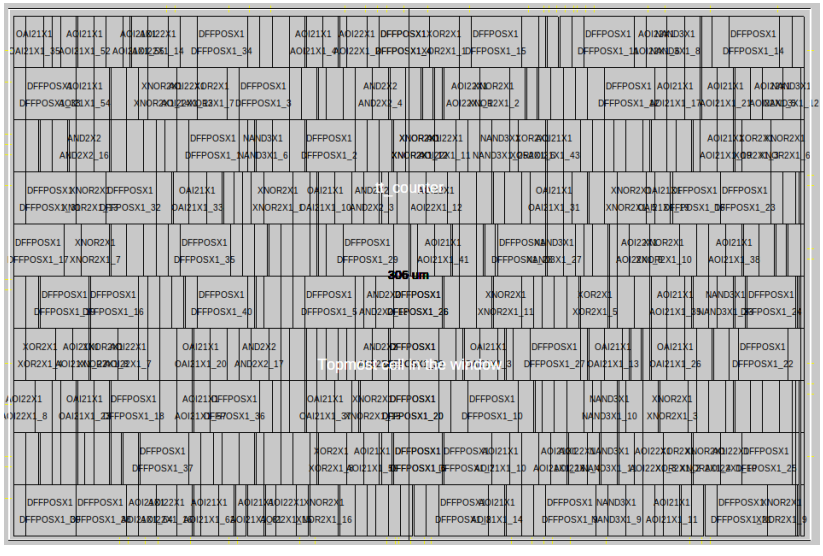

Figura 31: Longitud vertical del layout del módulo counter (librería osu050). Simulado en Qflow

2.8. Parte E

2.8.1. 3 INPUT NOR

El layout de la compuerta nor de 3 entradas se muestra a continuación

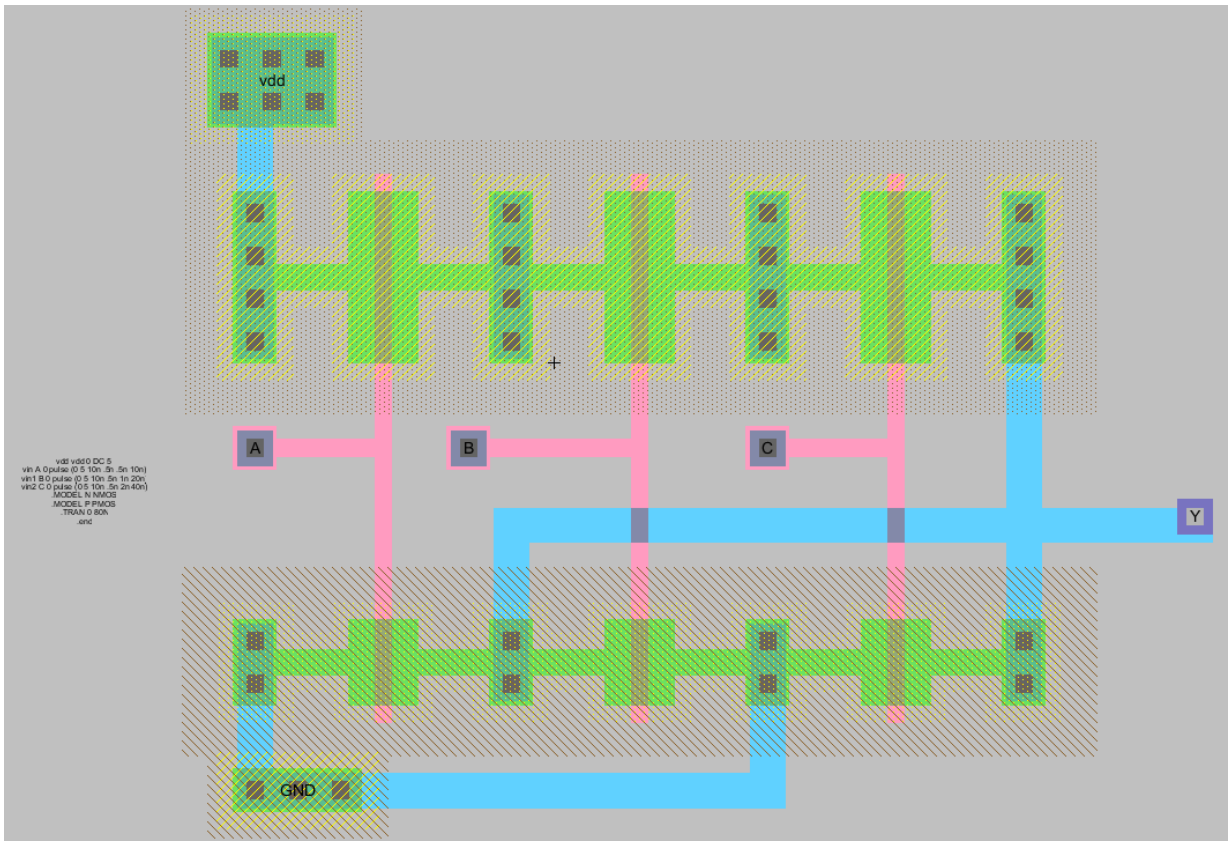

Figura 32: Layout de la compuerta NOR de 3 entradas. Simulado en Electric

En esta compuerta se observan 3 inputs: a, b y c, una salida "y", una fuente vdd y ground. A partir de este layout se escribe el siguiente código de spice y este es implementado en LTspice, produciendo el waveform de la figura 33

```
vdd vdd 0 DC 5
vin A 0 pulse (0 5 10n .5n .5n 10n)
vin1 B 0 pulse (0 5 10n .5n 1n 20n)
vin2 C 0 pulse (0 5 10n .5n 2n 40n)
.MODEL N NMOS
.MODEL P PMOS
.TRAN 0 80N
.end
```

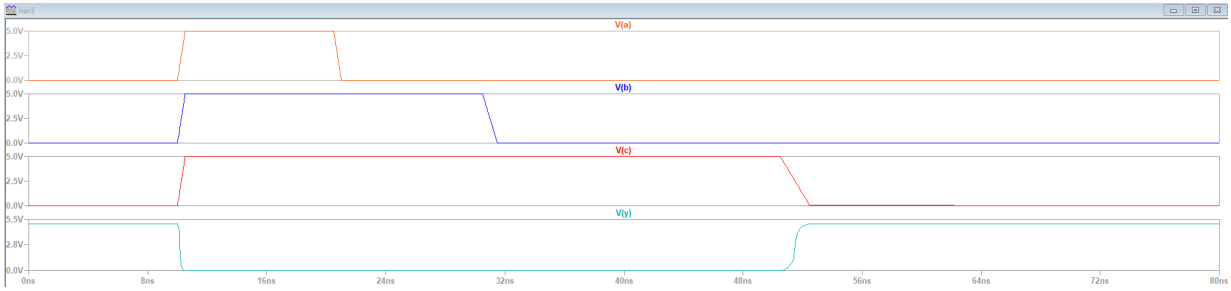

Figura 33: Waveform de la compuerta NOR de 3 entradas. Simulado en LTspice

En la figura anterior se puede observar el comportamiento esperado para una compuerta NOR de 3 entradas, donde su salida únicamente se pone en 1 cuando sus 3 entradas tienen un valor de 0.

2.8.2. 3 INPUT NAND

El layout de la compuerta nand de 3 entradas se muestra a continuación

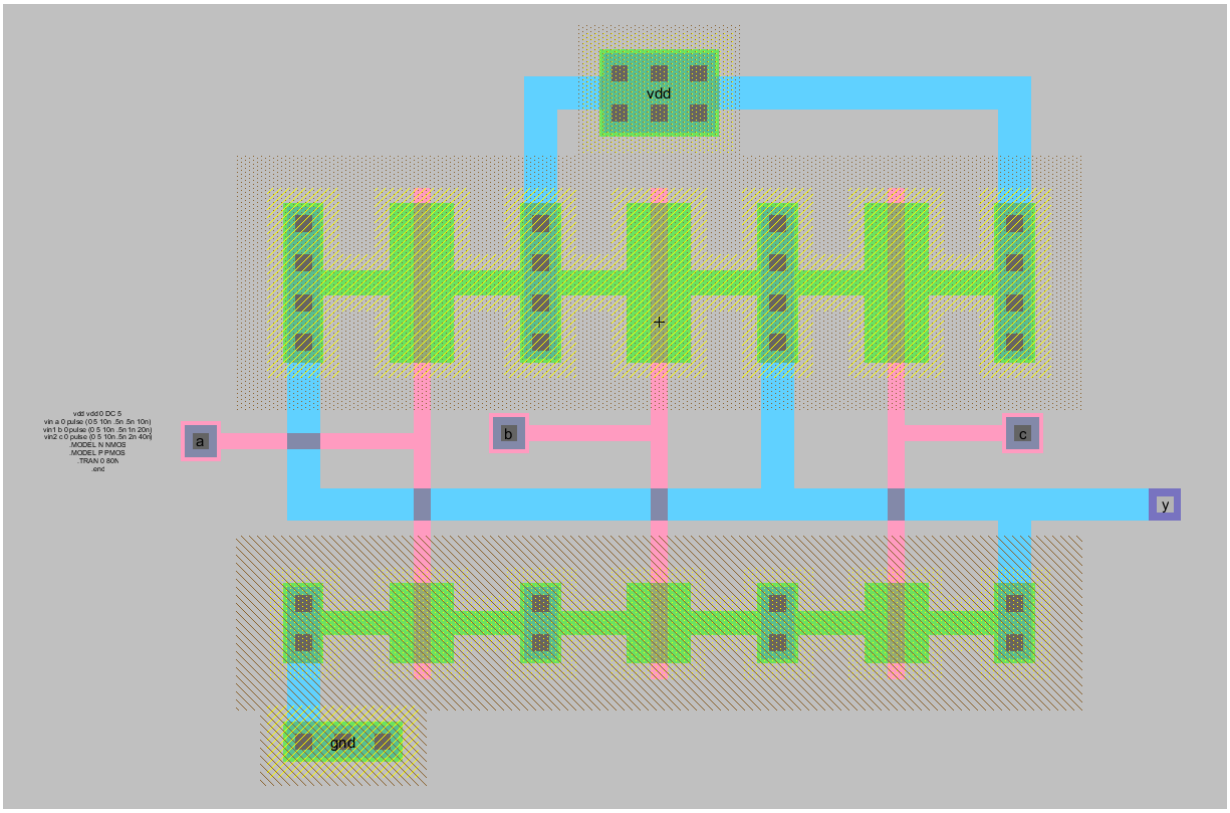

Figura 34: Layout de la compuerta NAND de 3 entradas. Simulado en Electric

En esta compuerta se observan 3 inputs: a, b y c, una salida "y", una fuente vdd y ground. A partir de este layout se escribe el siguiente código de spice y este es implementado en LTspice, produciendo el waveform de la figura 33

```
vdd vdd 0 DC 5
vin a 0 pulse (0 5 10n .5n .5n 10n)
vin1 b 0 pulse (0 5 10n .5n 1n 20n)
vin2 c 0 pulse (0 5 10n .5n 2n 40n)
.MODEL N NMOS
.MODEL P PMOS
.TRAN 0 80N
.end
```

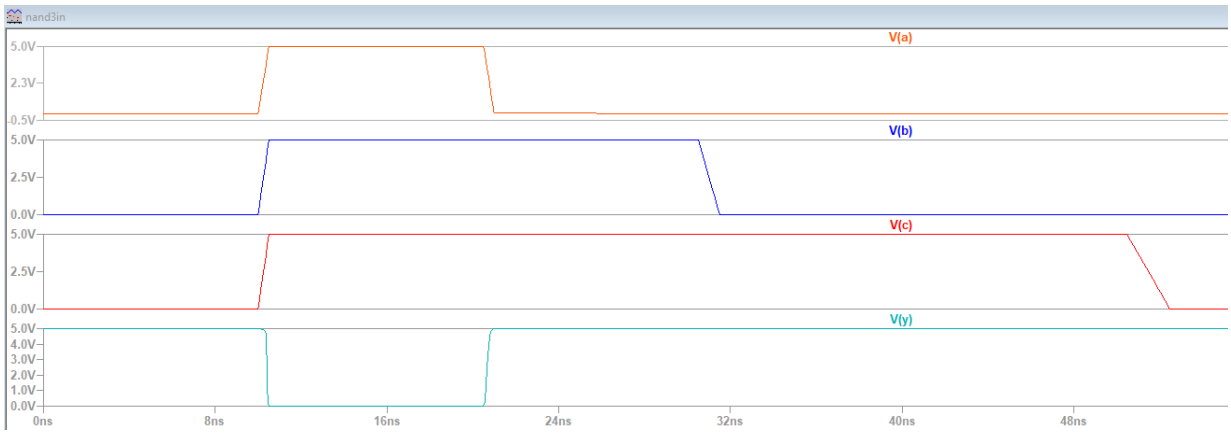

Figura 35: Waveform de la compuerta NAND de 3 entradas. Simulado en LTspice

En la figura anterior se puede observar el comportamiento esperado para una compuerta NAND de 3 entradas, donde su salida únicamente se pone en 0 cuando sus 3 entradas tienen un valor de 1.

2.8.3. AOI22

El layout de la compuerta aoi22 se muestra a continuación

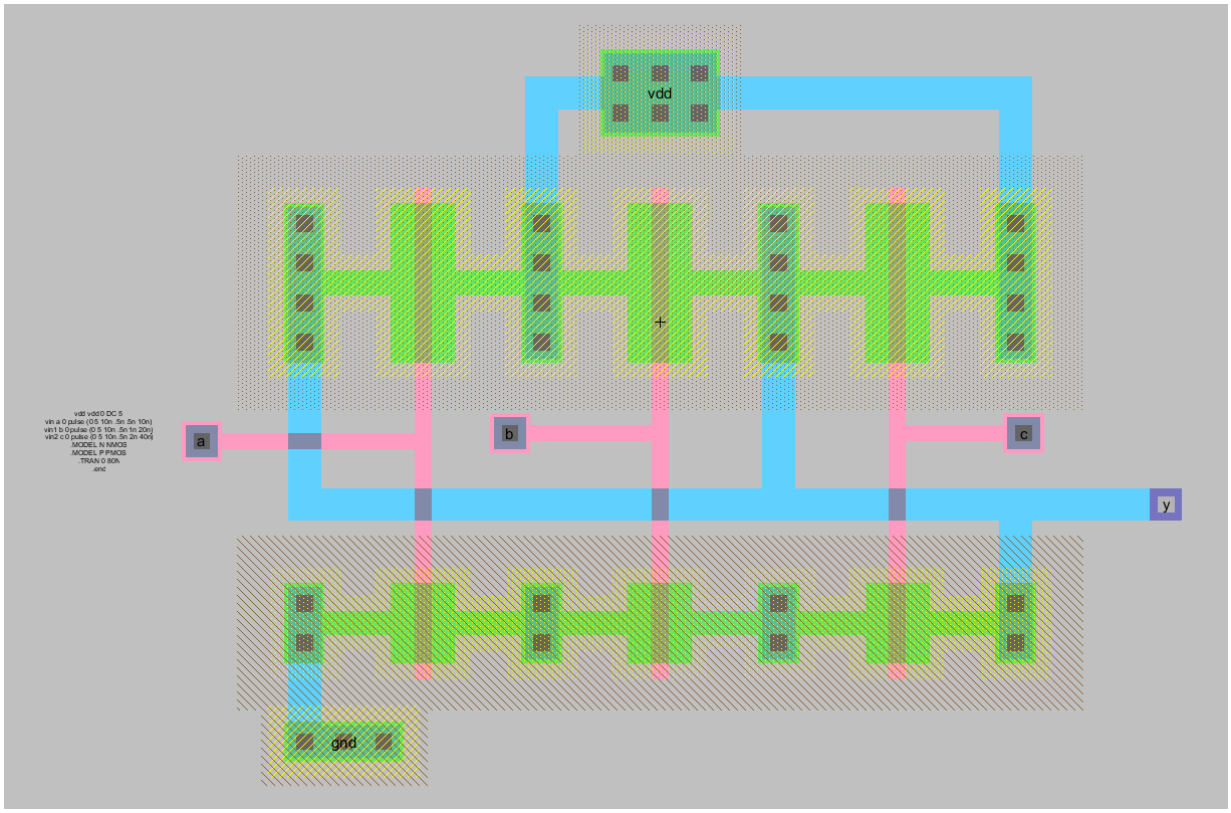

Figura 36: Layout de la compuerta AOI22. Simulado en Electric

En esta compuerta se observan 4 inputs: a, b, c y d, una salida “y”, una fuente vdd y ground. A partir de este layout se escribe el siguiente código de spice y este es implementado en LTspice,

produciendo el waveform de la figura 33. También se presenta la tabla de verdad para el layout implementado, es importante mencionar que dicha tabla es única para el orden en que las entradas fueron conectadas.

```
vdd vdd 0 DC 5
vin a 0 pulse (0 5 0.25p 0.25p 0.25p 0.125m 0.25m)
vin1 b 0 pulse (0 5 0.25p 0.25p 0.5p 0.25m 0.5m)
vin2 c 0 pulse (0 5 0.25p 0.25p 1p 0.5m 1m)
vin3 d 0 pulse (0 5 0.25p 0.25p 2p 1m 2m)
.MODEL N NMOS
.MODEL P PMOS
.TRAN 3m startup
.end
```

A	B	C	D	Y
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

Figura 37: Tabla de verdad de la compuerta AOI22. Simulado en LTspice

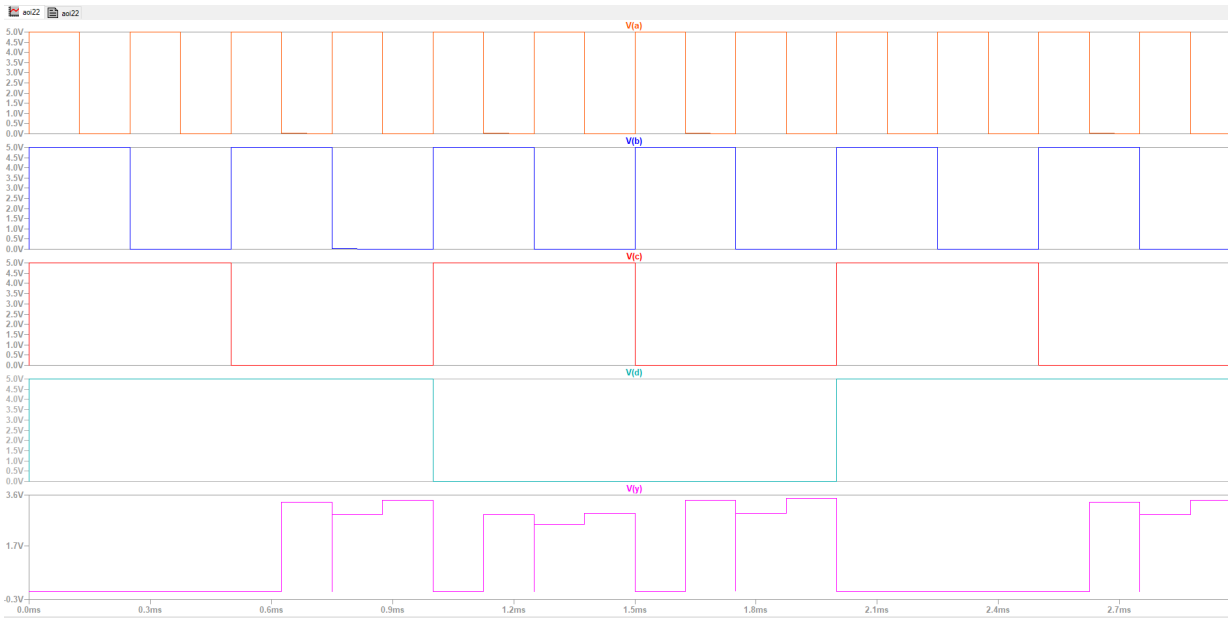

Figura 38: Waveform de la compuerta AOI22. Simulado en LTspice

Finalmente, se incluye el waveform de la compuerta, simulado en LTspice, con apuntes a los 16 posibles casos de la tabla de verdad de la figura 37.

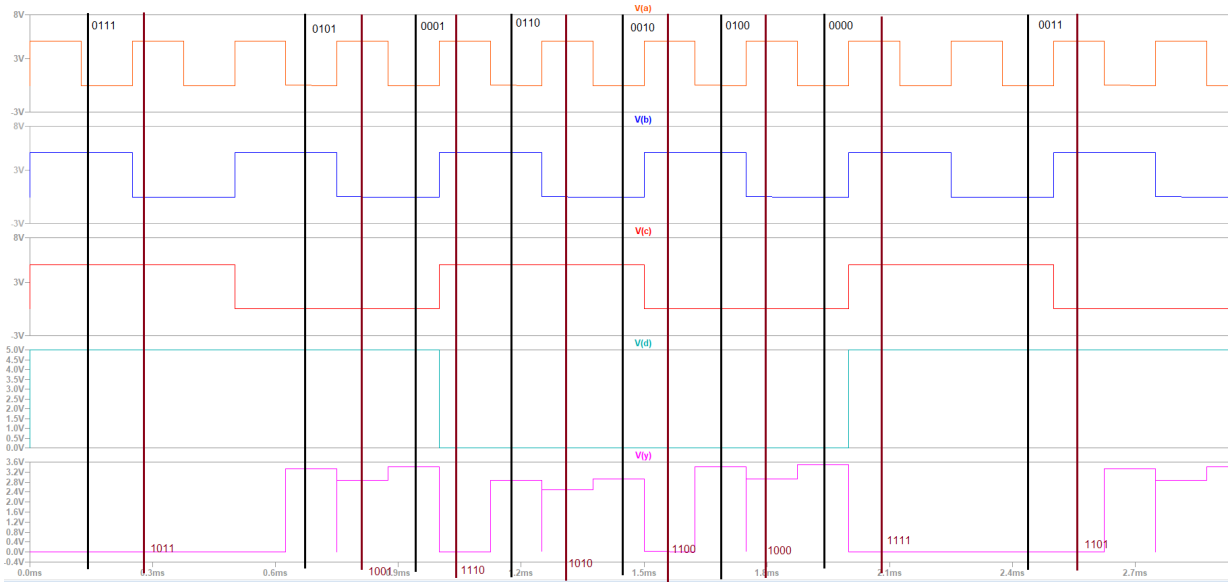

Figura 39: Waveform de la compuerta AOI22 con los 16 posibles casos marcados. Simulado en LTspice

En la figura anterior se puede observar el comportamiento esperado para una compuerta NAND de 3 entradas, donde su salida únicamente se pone en 0 cuando sus 3 entradas tienen un valor de 1.

3. Análisis de resultados

3.1. Parte A

Para esta sección se incluyeron pocas pruebas dado que el único cambio que se realizó al módulo counter de la tarea 1 fue el cambiar dos casos, lo cuál no afecta el funcionamiento anteriormente demostrado. Además, el reducir la duración de la señal rco tampoco afecta el funcionamiento de dicho contador ya que se hace exclusivamente para proveer un funcionamiento óptimo al conectar varios contadores de 4 bits.

3.2. Parte B

Esta fue la sección en la que más se tuvo que trabajar, ya que la conección de los módulos counter de 4 bits no es la que usualmente se utiliza en contadores sin ripple carry out. Una vez conectados dichos módulos se realizaron las pruebas correspondientes con el fin de comprobar el correcto funcionamiento de las señales de salida ante cambios en las señales de entrada como reset, enable, D y mode. Estas pruebas permitieron encontrar un bug en el scoreboard, el cuál no levanta la señal load si enable se encuentra en 0, sin embargo, en el módulo tt_counter esta señal sí funciona correctamente. Además de estas pruebas realizadas, se comprobó que el disminuir el tiempo en que está levantada la señal rco permite a los demás contadores de 4 bits leer el valor de esta señal a tiempo y funcionar de manera correcta debido a que esta señal puede funcionar como un clk para los contadores siguientes.

3.3. Parte C

Para esta parte se eliminó el always @(negedge clk) del módulo counter que reducía el tiempo en que la señal rco se mantenía arriba, esto introdujo problemas al conectar varios módulos counter, ya que los mismos requieren de esta señal para trabajar (esta funciona como clk en ciertos casos).
Al observar el módulo tt_counter, se puede ver que cuando el contador se encuentra en estado de carga (mode == 11), o bien cuando hay un reset o la señal de enable se encuentra en bajo, los contadores de 4 bits que van del bit 7 al 31 funcionan con el mismo clk del primer contador, caso contrario tienen el rco del contador anterior como su clk para poder funcionar correctamente cuando se requiere que saquen datos y al mantener la señal de rco en alto durante todo un ciclo de reloj inevitablemente van a haber lecturas incorrectas en las señales de rco de los contadores siguientes.

3.4. Parte D

Para esta parte se utilizó la herramienta Qflow, en dicha herramienta inicialmente se ejecutó el código

```
qflow synthesize place route counter
```

Con el fin de obtener la síntesis, y place route del contador de 4 bits. Luego, la cantidad de componentes que utiliza este contador se lee directamente del archivo counter.rtl.v, ubicado en la carpeta de síntesis ya que la información desplegada en el archivo log no contiene las compuertas finales utilizadas por la herramienta. Después, se ejecuta el comando

```
qflow sta counter
```

Mediante el cuál se obtienen los timings y paths con mayor y menor retardo del circuito.
Finalmente se obtiene el layout mediante la ejecución de dos comandos en el modo magic

```
lef read /usr/share/qflow/tech/osu035/osu035_stdcells.lef

def read ./../layout/counter.def
```

Luego se modifica el archivo qflow_exec.sh para ahora utilizar la librería osu050, se vuelve a repetir el procedimiento anteriormente mencionado para las dos librerías y el módulo tt_counter.
A continuación se muestra una tabla con el resumen de los resultados obtenidos para esta parte

Módulo	**Librería**	**# gates**	**Max freq**	**Medida y**	**Medida x**	**Área**
counter	osu035	49	704.321 MHz	67 um	125 um	8.375 nm
counter	osu050	51	710.839 MHz	72 um	131 um	9.432 nm
tt_counter	osu035	400	722.424 MHz	207 um	306 um	63.34 nm
tt_counter	osu050	415	710.839 MHz	306 um	463 um	141.7 nm

Tabla 1: Resultados de las pruebas de la parte D

De la tabla se observa que los diseños implementados con la librería osu050 consumen una mayor cantidad de compuertas normales así como compuertas compuestas (el contador de 32 bits con osu050 utiliza 5 compuertas OAI21X1 más que el que utiliza osu035).
Por otra parte, para el contador de 4 bits utilizar la tecnología osu050 incrementa su frecuencia máxima de operación, mientras que el contador de 32 bits tiene una mayor frecuencia máxima con la tecnología osu035. En general, ambos contadores tienen frecuencias máximas de operación similares lo cuál indica que la implementación del contador de 32 bits es bastante eficiente en términos de velocidad.

El área de los contadores utilizando la librería osu050 es mayor, esto se debe a que utilizan una mayor cantidad de compuertas y probablemente las dimensiones de estas compuertas sean similares en ambas librerías. Para el contador de 32 bits resulta mucho más eficiente utilizar la librería osu035 ya que, si se utiliza la librería osu050, el área de este contador incrementa más del doble.

3.5. Parte E

Para esta última sección se obtienen los layouts de las compuertas solicitadas a partir de la lógica CMOS de las mismas.

3.5.1. 3 INPUT NOR

En el layout se conectan 3 transistores pmos en serie y 3 transistores nmos en paralelo respetando las dimensiones establecidas. Se implementó un código sencillo que permita mostrar la forma de onda de las entradas y la salida, obteniendo la figura 33, en donde se observa el comportamiento esperado.

3.5.2. 3 INPUT NAND

Esta vez se conectan 3 transistores pmos en paralelo y 3 transistores nmos en serie. Las formas de onda se observan en la figura 35.

3.5.3. AOI22

Para esta compuerta se utiliza la referencia [1], en la cuál se presenta esta compuerta con una salida Y = NOT((A&B) | (C&D)), los transistores se conectan de la misma forma que se muestra en dicha referencia con el fin de utilizar la tabla de verdad de la figura 37. Luego, se procede a elaborar el waveform y se marcan todos los 16 posibles casos utilizando la herramienta Paint. La figura 39 permite observar que dicha compuerta cumple con los 16 posibles casos de la tabla de verdad.

4. Conclusiones y recomendaciones

A partir del análisis de resultados se llega a las siguientes conclusiones

- Se modifica el módulo para el contador de 4 bits exitosamente y a partir de este se implementa un contador de 32 bits funcional
- Se logra obtener la cantidad de compuertas de cada librería que son utilizadas para los diferentes diseños
- Se obtienen características de timings a partir de la herramienta Qflow
- Se comparan las caracteristicas de timing y área de los diseños implementados y las librerías utilizadas.
- Se realiza el layout de tres compuertas y se comprueba el funcionamiento de estas utilizando la herramienta LTspice, también se incluyen las formas de onda.

También se recomienda lo siguiente

- Leer la documentación del software para poder tener un mejor entendimiento de los comandos
- Trabajar de forma ordenada mediante un repositorio de git
- Cerrar el layout después de medir el largo o el ancho, ya que estas medidas se sobreponen y no son legibles si se intenta realizar ambas al mismo tiempo.
- Obtener la lógica CMOS de las compuertas a implementar en la herramienta Electric y constantemente verificar por errores con la tecla F5.
- Utilizar entradas con formas de onda de periodos distintos para poder obtener una mayor cantidad de casos de la tabla de verdad de cada compuerta.

Referencias

1. [1] CMOS AOI22 and-or-invert complex gate. Hades Website. [En linea] Recuperado de: https://tams.informatik.uni-hamburg.de/applets/hades/webdemos/05-switched/40-cmos/aoi22.html/ [Accesado el 7 de Diciembre 2020]