UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Verificación de reglas de diseño (DRC) para el desarrollo de un flujo funcional de un circuito integrado con tecnología nanométrica

Trabajo de graduación presentado por Matthias Sibrian Illescas para optar al grado académico de Licenciado en Ingeniería Electrónica

Guatemala,

2021

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Verificación de reglas de diseño (DRC) para el desarrollo de un flujo funcional de un circuito integrado con tecnología nanométrica

Trabajo de graduación presentado por Matthias Sibrian Illescas para optar al grado académico de Licenciado en Ingeniería Electrónica

Guatemala,

2021

Vo.Bo.:

(f) _____

MSc. Carlos Esquit

Tribunal Examinador:

(f) _____

MSc. Carlos Esquit

(f) _____

Ing. Jonathan de los Santos

(f) _____

Ing. Kurt Kellner

Fecha de aprobación: Guatemala, 21 de enero de 2021.

Prefacio

El presente trabajo de graduación se realizó en dos contextos importantes: en medio de la pandemia causada por el COVID-19 y como seguimiento de manera remota de un proyecto intergeneracional de gran relevancia para la universidad y el departamento. Este avance sobre este proyecto de tal importancia no se pudo haber realizado sin el apoyo recibido por la Universidad del Valle de Guatemala por la colocación de accesos remotos hacia el entorno de Synopsys desde agosto del 2020, por MSc Carlos Esquit por su constante asesoría técnica, por Ing Jonathan de los Santos por su constante ayuda con el entorno de Synopsys y finalmente por mis compañeros de proyecto e Ing Kurt Kellner, por el constante seguimiento que terminó siendo un apoyo esencial.

Índice

Lista de figuras

Lista de cuadros

Resumen

El objetivo principal fue proporcionar el apartado de Design Rule Check (DRC) para un flujo de diseño funcional de un circuito integrado con tecnología nanométrica. Para ello, se generaron dos iteraciones de verificación sobre cinco circuitos usando scripts de mejora diseñados tras la investigación y la selección de comandos útiles de las herramientas del entorno. Los cinco sistemas probados fueron compuertas NOT, NOR, NAND, un Full Adder y un 4-bit Counter. En la verificación final del apartado DRC producido, se obtuvo una reducción promedio de 95.53 % en los errores por reglas de diseño.

Abstract

The goal of the present work was to provide the Design Rule Check (DRC) section of a functional design flow for the design of an integrated circuit of nanometric technology at Universidad del Valle de Guatemala. To do so, an initial examination of the Synopsys design suite was completed, followed by a careful selection of available commands to verify violations of design rules. This led to the successful execution of two iterations of scripts tested on five different circuits, namely, on combinational logic gates NOT, NOR, NAND, and circuits Full Adder and 4-bit Counter. Lastly, through the final version of the verification section, an effectiveness of 95.53 % in average reduction of errors caused to design rules was attained on the five systems.

CAPÍTULO 1

Introducción

En un contexto histórico, la electrónica ha venido para potenciar el desarrollo humano. A través de un circuito integrado, se ha podido lograr millones de operaciones lógicas por segundo, las cuales posibilitan el diseño de dispositivos interactivos utilizados extensamente en la industria, la investigación y la educación.

Sin embargo, al haberse consagrado en las últimas decadas ya como una industria madura y sólida, el diseño por Integración a muy gran escala (VLSI, por sus siglas en inglés), no es para nada una tarea sencilla de realizarse. El proceso de diseño de sistemas a escala nanométrica es una tarea con costos elevados, alta necesidad de conocimiento técnico y herramientas especializadas de diseño.

Considerando esto, la Universidad del Valle de Guatemala, en sus esfuerzos por avanzar la ciencia y la tecnología en la región, ha posibilitado a cohortes de estudiantes la utilización de herramientas de VLSI. Es en este contexto en que se realiza este trabajo de graduación, el cual expande el alcance del flujo de diseño de un circuito integrado con tecnología nanométrica, particularmente, en la sección de violaciones por reglas de diseño (DRC).

De manera concreta, se proporciona material perteneciente al apartado de Design Rule Check (DRC) para un flujo de diseño funcional de un circuito integrado con tecnología nanométrica. Asimismo, se presenta un análisis del estado del arte en pruebas por DRC en el contexto de la Universidad y las herramientas proveídas por la empresa taiwanesa TSMC, un script para una corrida sin errores por violaciones a las reglas de diseño y un bagaje de contenido didáctico para futuras generaciones.

CAPÍTULO 2

Antecedentes

En el año 2014, gracias a los esfuerzos de MSc. Carlos Esquit, Director del Departamento de Ingeniería Electrónica y Mecatrónica, la Universidad del Valle de Guatemala logra establecer un convenio con la compañía americana *Synopsys* que permite el acceso a las herramientas de *software* ofrecidas en su programa educativo *University Program*. Esto ocurre un año después de que se empezara a impartir el curso *Introducción al diseño de sistemas VLSI* en la Universidad. De esta manera es como se logra el primer acercamiento al proyecto multigeneracional trabajado en este documento.

Cinco años después, el ahora Departamento de Ingeniería Electrónica, Mecatrónica y Biomédica recibe la oportunidad de fabricar un circuito integrado en tecnología nanométrica de la compañía taiwanesa *Taiwan Semiconductor Manufacturing Company (TSMC*, por sus siglas en inglés). Para este entonces ya se había realizado trabajos alrededor de diseño VLSI con las herramientas adquiridas y ya se había completado un flujo de diseño para tecnologías nanométricas. Los proyectos de graduación resultantes de estos esfuerzos pueden encontrarse en [1] y [2].

CAPÍTULO 3

Justificación

El perfeccionamiento de un flujo de diseño para circuitos integrados de tecnología nanométrica es fundamental para acercarse a los beneficios que conlleva el desarrollo de electrónica con aplicaciones específicas. Al madurar un proceso que permita llevar un diseño de un circuito integrado digital hacia un formato fabricable, se está sentando un precedente vital para el progreso local de la ingeniería electrónica y la tecnología, particularmente en un contexto centroamericano, en donde el enfoque académico y laboral de ingenieros electrónicos no se ha enfocado en diseño a muy gran escala, sino que más en el área de ingeniería de control industrial y de telecomunicaciones. Es en este contexto en el que se desarrolló este proyecto intergeneracional y este trabajo de graduación en particular.

Considerando la oportunidad de aprendizaje y desarrollo que presenta este proyecto para la universidad, los actuales y futuros estudiantes de ingeniería electrónica y la industria centroamericana, se visualiza el éxito de los objetivos en este trabajo como parte de un objetivo más amplio: crear un flujo de diseño funcional para la producción de circuitos sintentizados y libres de errores listos para fabricarse en silicio que pueda ser utilizado por miembros de la comunidad para aprender, desarrollar y producir electrónica a nanoescala. No solo ha sido una oportunidad de aprendizaje para las cohortes involucradas de primera mano con el proyecto, sino que también se espera que se pueda impactar de manera positiva al avance de herramientas que el Departamento de Ing. Electrónica, Mecatrónica y Biomédica tiene a su disposición para avanzar los objetivos científicos, educacionales y profesionales que visualiza en sus estudiantes.

CAPÍTULO 4

Objetivos

4.1. Objetivo general

Proporcionar el apartado de *Design Rule Check (DRC)* para un flujo de diseño funcional de un circuito integrado con tecnología nanométrica.

4.2. Objetivos específicos

- Comprender las secciones correspondientes al *Design Rule Check (DRC)* en el flujo de diseño desarrollado por el grupo de estudiantes que trabajaron anteriormente en este proyecto.
- Determinar comandos existentes en la documentación de *Synopsys* que sean útiles para reducir las violaciones a las reglas de diseño y que no fueron utilizados por la cohorte anterior de estudiantes.
- Elaborar un *script* que lleve a cabo una corrida del *Design Rule Check (DRC)* libre de errores para un circuito integrado de tecnología nanométrica.
- Documentar el uso correcto de las herramientas utilizadas, a través de tutoriales y algoritmos, para su uso futuro por otros estudiantes de la Universidad del Valle de Guatemala.

CAPÍTULO 5

Alcance

El alcance de este trabajo de graduación se limita a la verificación por reglas de diseño (DRC) aplicada en *layouts* de sistemas electrónicos a escala nanométrica. Asimismo, se restringió a utilizar de manera exclusiva herramientas de *software* ofrecidas como parte de la *suite* de *Synopsys* y *runsets* proveídos por *TSMC*. De manera general, se busca la mejora del flujo entero de elaboración de sistemas a escala nanométrica, así como determinar métodos para reducir las violaciones por DRC y utilizarlos para generar un flujo mejorado para generar un *layout* final.

Es importante mencionar también que el alcance del mismo se vio fuertemente afectado por la pandemia originada por el virus COVID-19. Sin embargo, y a pesar de la misma, se hizo un esfuerzo por intentar mantener el camino trazado al inicio del proyecto. Además, por esta razón, se colocó mucho esfuerzo sobre la documentación para generaciones futuras, de tal manera que no se pierda la inercia del proyecto.

Es importante mencionar que existe una gran oportunidad de seguir mejorando las etapas trabajadas por las cohortes de estudiantes pasadas y presentes, de tal manera que se vuelva más robusto y versátil el el flujo de diseño del que puede depender el Departamento. Por esta razón, este trabajo busca de igual manera documentar el uso correcto de las herramientas que se utilizaron, a través de tutoriales y algoritmos concedidos para uso por futuros estudiantes.

CAPÍTULO 6

Marco teórico

6.1. *Very Large Scale Integration (VLSI)* [3]

El proceso de diseño de circuitos por integración a muy gran escala (*VLSI,* por sus siglas en inglés) es un desarrollo tecnológico que debe su avance a esfuerzos que se remontan a 1963. En ese año, Frank Wanlass, ingeniero de la compañía americana *Fairchild* desarrolló la primera compuerta lógica utilizando transistores de efecto de campo metal-óxido-semiconductor (*MOSFET*, por sus siglas en inglés). Para ello, emparejó transistores MOSFET tipo-n y tipo-p, llamados *nMOS* y *pMOS* respectivamente, en un arreglo semiconductor complementario de óxido metálico (*CMOS*, por sus siglas en inglés).

El logro alcanzado por Wanlass permitió reducir la potencia consumida por un transistor al orden de los nanowatts, pero todavía se utilizaba componentes discretos para elaborarlos. Luego, gracias al desarrollo del proceso de fabricación planar con silicio, se convirtió en una opción rentable la fabricación de circuitos con tecnología de efecto de campo metal-óxido-semiconductor (MOS). Una de las empresas pioneras en el proceso de fabricación MOS fue la compañía americana *Intel Corporation.* A continuación, se presenta una vista del famoso procesador de 4 bits, el Intel 4004 [3]:

Utilizar la tecnología CMOS en las compuertas lógicas de un circuito integrado era menos rentable que usar tecnología nMOS, pero proveía mucho menor consumo de potencia. No es hasta en la década de 1980 en que las aplicaciones electrónicas digitales adoptan por completo la tecnología CMOS en sus transistores.

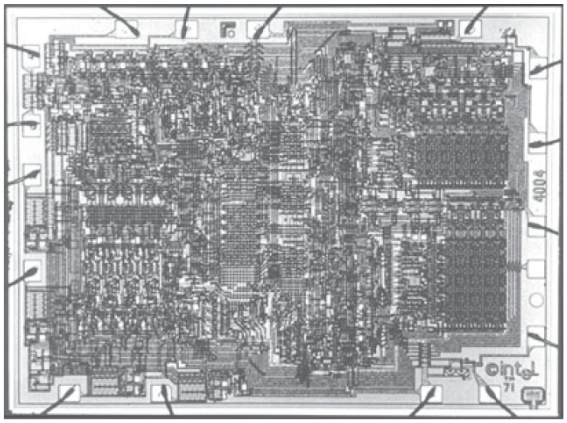


Figura 1: Intel 4004, microprocesador de 4 bits desarrollado por Intel Corporation

En 1965, al observar estos cambios, Gordon Moore conjeturó que, al colocar la cantidad de transistores en función del tiempo bajo una escala semilog, es fácil observar que Intel incrementava la cantidad de transistores dentro de un chip cada 18 meses aproximadamente, empezando desde la invención del Intel 4004. Esta observación se convirtió en el famoso término *Ley de Moore*. A continuación, se presenta evidencia que confirma la predicción hecha por Moore muchos años antes [4]:

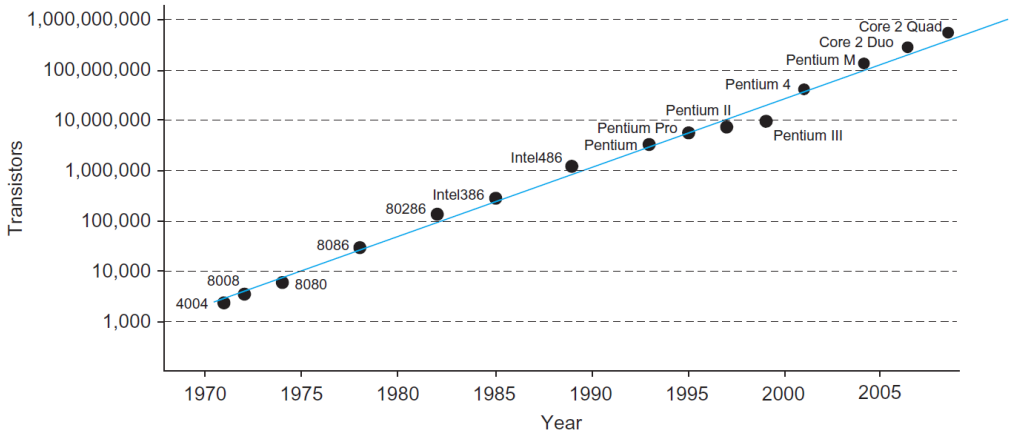


Figura 2: Cantidad de transistores en microprocesadores desarrollados por Intel

Cabe mencionar que Moore, al realizar su observación, tomó en consideración dos hechos importantes: primero, que el tamaño del área efectiva de los microprocesadores de Intel no estaba siendo considerablemente incrementada, y, segundo, que el factor que estaba permitiendo colocar más transistores en cada circuito integrado era la reducción del tamaño en tecnologías CMOS. Es en este contexto en que se empiezan a mencionar los conceptos de *escalamiento* y de *integración* en diseño con electrónica digital.

Es en este contexto en que se formalizó el término *Very Large Scale Integration* (VLSI): al inicio, se estaba colocando un poco más de 10 compuertas lógicas en un circuito integrado, luego se incrementó a 100, luego a 1,000 y, luego, a 10,000, llamando así a estos avances *Small Scale Integration* (SSI), *Medium Scale Integration* (MSI) y *Very Large Scale Integration*

(VLSI) respectivamente. Fue fácil ver que esta tendencia de nombramiento no podía seguir y, por ello, el término VLSI se empezó a utilizar para todos los desarrollos posteriores.

6.2. Flujo de diseño [5]

El flujo de diseño se utiliza para referirse al proceso por el cual se obtiene el diseño de un circuito integrado funcional con una aplicación específica. Este proceso se puede dividir en seis etapas de diseño, dependiendo del nivel de abstracción requerida: sistema, algoritmo, arquitectura, lógica, física y *signoff*, proceso que se desglosa y explica a continuación:

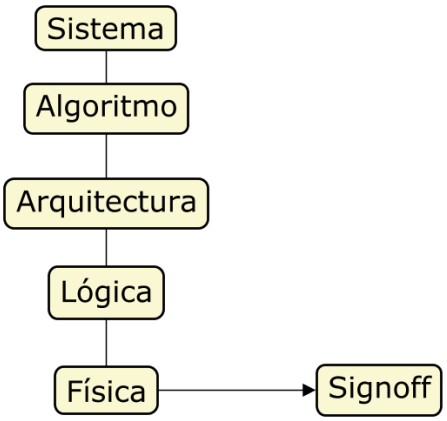


Figura 3: Etapas más importantes en diseño VLSI

- **Diseño a nivel de sistema:** En esta etapa se especifica las características deseadas de funcionalidad, condiciones de operación, costo, forma física, potencia y desempeño que tenga el sistema. Se determina también que protocolos va a necesitar la aplicación electrónica, cuántas y cuáles interfaces va a tener, cómo se va a operar, etc. El reto de esta etapa es adaptar el sistema de la mejor manera a la aplicación que se le quiere dar.
- **Diseño a nivel de algoritmo:** En esta etapa se define el esquema computacional para una fácil implementación hacia hardware. Acá se especifica la complejidad de ls programas, la representación numérica, la cantidad memoria, tamaño y cantidad de registros, operaciones básicas de bajo nivel, etc. Es importante en esta etapa hacer un compromiso entre complejidad, exactitud y capacidad computacional.
- **Diseño a nivel de arquitectura:** En esta etapa, es necesario decidir cuáles recursos de harware se utilizarán para cada una de las tareas, considerando las restricciones de desempeño, costo y potencia, entre otras, que se definieron en el paso anterior. Para ello, se parte de una noción bastante abstracta de la funcionalidad del sistema y se empieza a acercar a representaciones más detalladas. De manera concreta, en esta etapa se realizan tareas como alocar recursos de hardware, definir *datapaths* y memorias RAM para ciertas tareas, decidir protocolos de comunicación entre módulos, establecer tamaños de *buses*, etc.

- **Diseño a nivel lógico:** En esta etapa del diseño se obtiene una lista, llamada *netlist*, que describe la manera en que están conectados los componentes electrónicos que componen el circuito diseñado. Este diseño se obtiene generalmente de manera automatizada, pero el diseñador debe haber decidido factores como la tecnología de fabricación y la colección de piezas elementales, llamada *cell library*, que empleará en el circuito. Sin una *cell library* no se puede completar ningún diseño.
- **Diseño a nivel físico:** Esta etapa consiste en la colocación de todos los subcircuitos que componen al circuito integrado dentro de un área de material semiconductor. En esta etapa sucede el *floor planning*, el *place and route*, el *design rule check* y el *layout versus schematic*. A continuación, se describe cada uno de ellos:
  - *Floor-planning*: El proceso de *floor-planning* consiste en organizar el área rectangular que ocupa el chip con todos los elementos y la menor cantidad de retardos por interconexiones entre módulos.
  - *Place-and-route (P and R)*: El proceso de *place-and-route* consiste en asignarle a cada celda una ubicación y lograr reducir al máximo los retardos que pueda ocasionar la interconexión propuesta.
  - *Design Rule Check (DRC)*: El *Design Rule Check (DRC)* consiste en asegurar que el *layout* final alcanzado durante el *place-and-route* no proponga, a manera de formas geométricas para crear las máscaras de fabricación. un *layout* irrealizable y/o que terminará siendo disfuncional.
  - *Layout-versus-Schematic (LVS)*: A través del análisis *layout-versus-schematic* se hace una comparación entre las *netlists* inicial y final. De esta manera, se logra asegurar que la funcionalidad del circuito fabricado será la deseada.
- ***Sign-off:*** En la etapa de *sign-off* se hace una adaptación a los requerimientos impuestos por el fabricante de circuitos integrados que posibilitará la fabricación. Puesto que no es de interés hacer una fabricación de un prototipo disfuncional, el *sign-off* puede consistir de examinaciones de DRC, LVS, *timing* y demás simulaciones para lograr un resultado exitoso.

6.3. Entorno existente

En el entorno creado con anterioridad para la elaboración del flujo de diseño ya se había descargado librerías tanto de carácter educacional como formal. Las librerías obtenidas para la tecnología de 90nm fue con Synopsys, mientras que, para la tecnología de 180 nm, fue con TSMC. Después de ser descargadas, se colocaron en las carpetas SAED90-EDK y SAED90-EDK90nm-REF ubicadas en las computadoras de la Universidad.

La cohorte anterior de estudiantes han indicado que existen tipos de archivos de vital importancia para completar un flujo de diseño. A continuación, se describen las extensiones de los mismos, junto con una descripción para comprender su importancia y su función:

Cuadro 1: Resumen de archivos y extensiones utilizadas en el flujo de diseño existente

Archivo	Extensión	Información almacenada	Uso en el flujo de diseño
DB	.db	Librería que almacena información de los componentes que están presentes en la síntesis estructural que se hace de un circuito lógico. Al utilizarla, se asegura que las compuertas presentes en el Verilog generado son manejadas por el fabricante o por Synopsys.	Esta extensión es usada comúnmente por librerías link library. Las link libraries se utilizan para que los elementos que están referenciados (posiblemente de manera behavioural) en el Verilog inicial tengan las características específicas de las compuertas fabricables.
SDB	.sdb	Librería que almacena la representación en símbolos de los componentes que se utilizan en la síntesis física.	La herramienta Design Vision sí requiere de symbol libraries, mientras que VCS no.
TF	.tf	Librería que provee con la información referente a la tecnología a utilizarse en la implementación física del diseño, específicamente, en el place and route.	Acá se provee con datos como el ancho de los metales, el espaciamiento, definiciones de vías, etc. Synopsys estos archivos .tf, pero Cadence usa otro tipo.
TLUPLUS	.tlupus	Provee modelos para la extracción exacta de efectos RC causados por características de ancho, espaciado, densidad y temperatura en los coeficientes resistivos presentes.	Se usa en Synopsys, por ejemplo, para hacer extracción de efectos parásitos en el proceso de place and route durante el back end.
MAP	.map	Sirve para decirle al software de síntesis cómo convertir de una capa manejada por el software mismo y una capa descrita en un archivo CIF o GDS..	CIF y GDS son archivos comúnmente utilizados por programas para la síntesis de ICs. CIF viene en formato ASCII, mientras que GDS viene en formato binario.
Reference Library	.db	Contiene información necesaria para el placement y el routing referente a las celdas estándar y de padding.	Define parámetros como el alto de las filas de componentes, la resolución mínima de anchura, dirección preferida en el routing, etc. Synopsys o el fabricante las define.
Milkyway Library	.db	Es un tipo de Reference Library utilizado por Synopsys. Contiene información necesaria para el place and route.	Es soportado por muchas de las herramientas de Synopsys, como Design Compiler, IC Compiler y StarRC.

CAPÍTULO 7

Análisis del flujo anterior

Previo a la generación y obtención de resultados, se recibió el proyecto de la cohorte anterior de estudiantes y se instalaron herramientas de acceso remoto para poder adaptarse a trabajarlo a distancia. Luego, se hizo un análisis de cada una de sus tres partes: del **Runset** utilizado anteriormente, del último **Script** de síntesis física para generar el **Layout** y del proceso de **DRC** utilizado por último.

7.1. Runset utilizado

El último **Runset** utilizado anteriormente para la verificación por **DRC**, el ICVLM18_LM16_LM152_3M_215a_pre041518, proveído por TSMC, cuenta con las siguientes características:

Cuadro 2: Características del runset ICVLM18_LM16_LM152_3M_215a_pre041518

Tipo	Tecnología	C. superior	Voltajes
CMOS	0.152, 0.16 y 0.18 um, 1P3M	40K y 20K	1.5, 1.8, 2.5, 3.3 y 5V

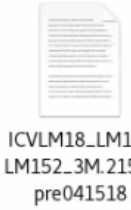


Figura 4: Archivo de **Runset** utilizado anteriormente

7.2. Último script de síntesis física

En los Anexos 1, 2 y 3 se incluye los comandos necesarios utilizados en IC Compiler para la generación del ***Layout*** final a utilizar. En el caso mostrado, se está realizando la síntesis de un circuito Full Adder utilizando celdas proveídas por TSMC. A continuación se especifica, por número de línea, la función que se está llevando a cabo con los comandos: Es importante mencionar que en este **Flujo de diseño** de síntesis, el único comando de

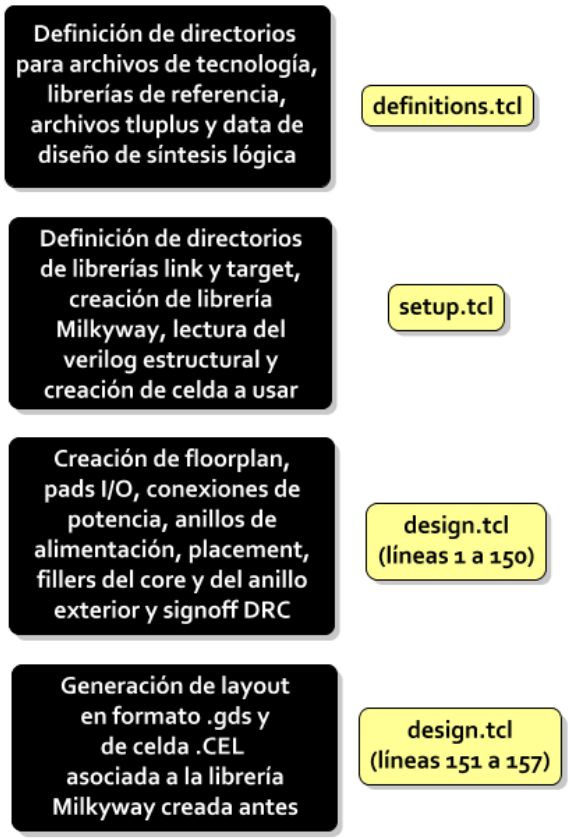


Figura 5: Desglose de funciones efectuadas por el script de síntesis física anterior

relevancia directa con el proceso de **DRC** es el siguiente:

```
144   #Verificacion DRC, Antenna
145   route_search_repair -rerun_drc -loop "10"
```

Figura 6: Ciclo de reparación por errores de **DRC** en síntesis física anterior

Sin embargo, la efectividad de este ciclo de reparación no tiene un efecto sustancioso sobre el ***Layout*** final, dado que este **DRC** no invoca a ICV ni tampoco toma en cuenta el **DRC** de TSMC.

7.3. Problemas con verificación DRC

Durante la transición de la cohorte de estudiantes anterior hacia la actual, se presentaron problemas relacionados a la ejecución del **DRC**. El primero de ellos consistió en la incapacidad de instanciar al ICV VUE desde la interfaz gráfica de IC Compiler. El error que se indicó tenía el código CMD-013, como se muestra a continuación:

Figura 7: Error generado al instanciar a ICV VUE

Para entender la causa de este error, se debe prestar atención al mensaje desplegado en el recuadro [6]. El diagnóstico que se realizó fue que las variables de estado no estaban colocadas adecuadamente y, por tanto, no se logra instanciar a ICV. Para ello, se determinó que la variable PATH, en el archivo bash.rc, estaba apuntando hacia un directorio equivocado, como se muestra a continuación:

Figura 8: Archivo bash.rc previo a la modificación

Luego, debido a que existió la necesidad de actualizar IC Compiler para realizar una corrida del LVS, se pasó de la versión 2018.06-SP5 hacia la versión 2019.12-SP5. Al correr el **DRC** por primera vez, ya con esta diferencia en versiones de software, se experimentó el error mostrado a continuación:

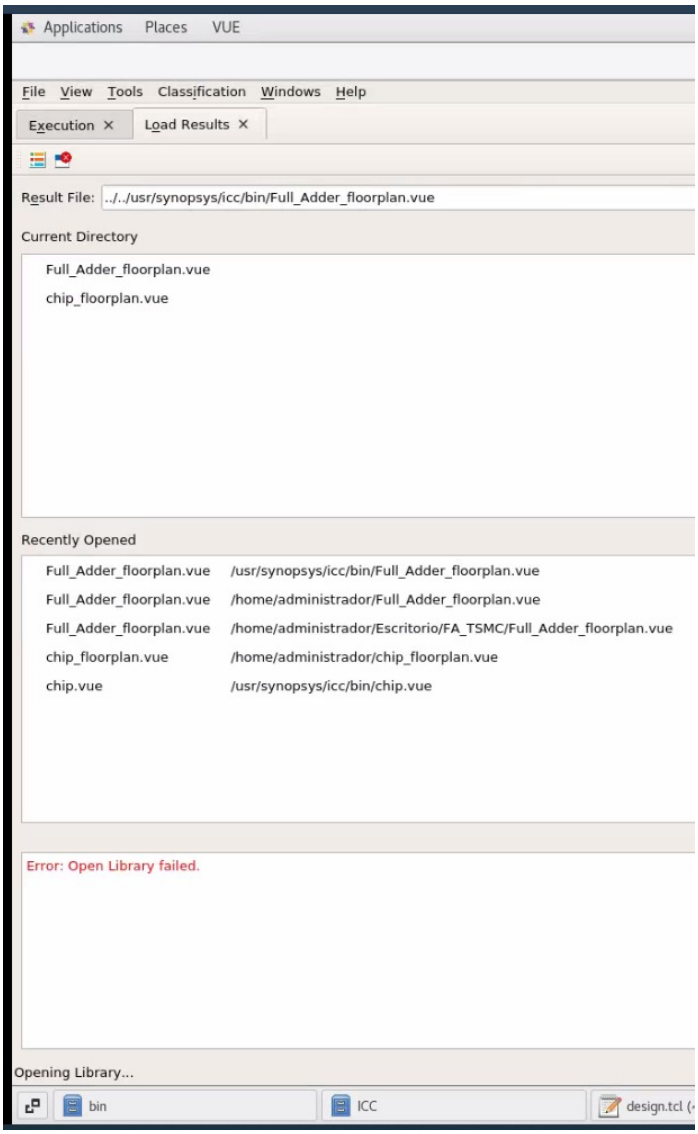


Figura 9: Resultado tras corrida de **DRC** de la cohorte anterior

Este error se estaba generando incluso aunque se estuviese logrando instanciar correctamente a ICV desde la ICV VUE, lo cual puede observarse a continuación:

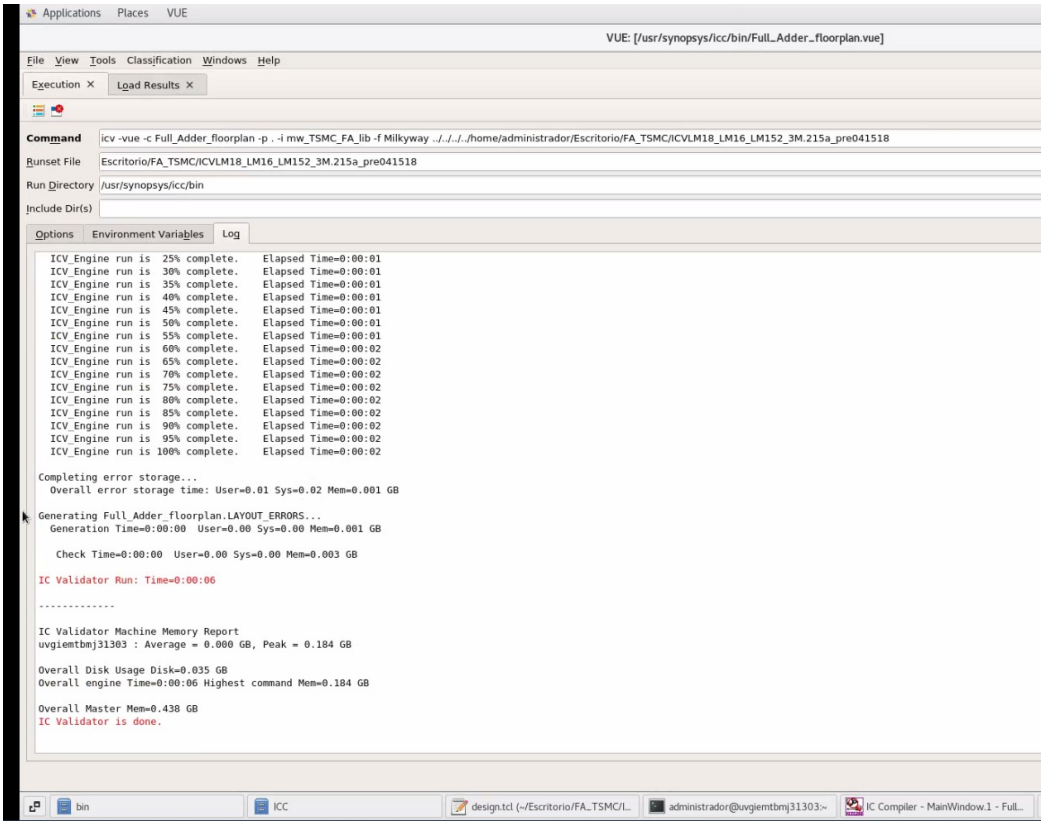


Figura 10: Resultado tras corrida de **DRC** de la cohorte anterior

Anteriormente, la cohorte anterior estaba logrando efectuar una corrida del **DRC** con un entorno en el que las versiones de las herramientas eran distintas. Dado que el entorno está en constante actualización, no se optó por retornar a versiones pasadas de IC Compiler y de ICV.

CAPÍTULO 8

Etapa de mejoramiento

8.1. Actualización de herramientas

Para solucionar el error de instanciamiento del ICV, lo cual era necesario para lograr realizar una corrida de **DRC** completa, se optó como primera medida actualizar la variable de entorno PATH.

Anteriormente, esta incluía una ruta hacia la carpeta LINUX.64. Sin embargo, la nueva versión del ICV no estaba localizada en esta ruta, entonces se debía cerciorar que se estuviese instanciando elementos dentro de la versión correcta de este programa. Por ello, se actualizó la ruta para apuntar hacia la nueva carpeta bin, como se muestra a continuación:

```
# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi

# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=
# User specific aliases and functions
PATH=/usr/synopsys/customcompiler/bin:$PATH
PATH=/usr/synopsys/installer:$PATH
PATH=/usr/synopsys/hspice/hspice/bin:$PATH
PATH=/usr/synopsys/icvalidator/Q-2019.12-SP2-4/bin:$PATH
PATH=/usr/synopsys/starrc/bin:$PATH
#PATH=/usr/synopsys/wv/Q-2020.03/bin:$PATH
PATH=/usr/synopsys/fm/bin:$PATH
PATH=/usr/synopsys/cscope64/ai_bin/:$PATH
PATH=/usr/synopsys/syn/bin:$PATH
PATH=/usr/synopsys/icc/bin:$PATH
PATH=/usr/synopsys/icc2/bin:$PATH
PATH=/usr/synopsys/mw/bin/linux64:$PATH
PATH=/usr/synopsys/pts/bin:$PATH
PATH=/usr/synopsys/pwr/bin:$PATH
PATH=/usr/synopsys/nt/bin:$PATH
PATH=/usr/synopsys/vcs-mx/bin:$PATH
PATH=/usr/synopsys/verdi/bin:$PATH
#source /usr/synopsys/PyCell/quickstart/bashrc

export PATH
#export ICV_HOME_DIR=/usr/synopsys/icvalidator
export ICV_HOME_DIR=/usr/synopsys/icvalidator/Q-2019.12-SP2-4
export VCS_HOME=/usr/synopsys/vcs/P-2019.06-SP2-4
export SNPSLMD_LICENSE_FILE=27020@192.168.6.124
export SAED32_28_PDK=/usr/synopsys/iPDK/SAED32_EDK

PATH="/home/administrador/perl5/bin${PATH:+:${PATH}}"; export PATH;
PERL5LIB="/home/administrador/perl5/lib/perl5${PERL5LIB:+:${PERL5LIB}}"; export PERL5LIB;
PERL_LOCAL_LIB_ROOT="/home/administrador/perl5${PERL_LOCAL_LIB_ROOT:+:${PERL_LOCAL_LIB_ROOT}}"; export PERL_LOCAL_LIB_ROOT;
PERL_MB_OPT="--install_base \"/home/administrador/perl5\""; export PERL_MB_OPT;
PERL_MM_OPT="INSTALL_BASE=/home/administrador/perl5"; export PERL_MM_OPT;
```

Figura 11: Archivo bash.rc posterior a cambios efectuados

Tras este cambio, se logró instanciar, desde la interfaz gráfica de IC Compiler, a ICV VUE, la cual posibilita configurar una corrida de **DRC** y ejecutarla. Con esto, se solventó el error indicado con el código CMD-013, el cual estaba sucediendo por no encontrar variables contenidas dentro del bin correcto tras la actualización.

Esta mejora brindó más robustez al entorno de dos maneras: primero, establece un precedente para la actualización de programas y errores de instanciamiento entre ellos a través de interfaces gráficas. Segundo, permitió actualizar dos de las herramientas más importantes para el **Flujo de diseño** de diseño, IC Compiler y IC Validator.

Para empezar la etapa de mejoramiento, se decidió realizar las primeras pruebas en el **Flujo de diseño** de diseño para una compuerta lógica NOT, diseño el cual consiste solamente de dos transistores. La razón principal para realizar esto de esta manera es porque facilita el descubrimiento y resolución de problemas en el **Flujo de diseño** de diseño, pues es más sencillo entender y solucionar eventos que aparezcan con un diseño más accesible.

8.2. Cambio de librerías

Durante la etapa de mejoramiento se optó por realizar un cambio en las librerías de las **Standard cells**. Inicialmente, la librería que se estaba utilizando para colocar las celdas dentro del core del sistema tenía estas características:

Cuadro 3: Características de la librería para celdas estándar usada con anterioridad

Nombre	Tecnología	Voltaje	Fecha	*Foundry*
TCB018GBWP7T Version 270a	0.18um	N.D.	mayo 2009	TSMC

En el proceso de síntesis física, es necesario referenciar las librerías en las cuales está definido cada uno de los elementos del netlist. Asimismo, es necesario especificar un *typical case*, un *worst case* y un *best case*, en el momento en que el **Flujo de diseño** de diseño aproveche de la capacidad de analizar varios escenarios de funcionamiento.

En el archivo de síntesis, la siguiente sección importa las librerías usadas por la cohorte anterior, y especifica que el caso típico se va a dar por el archivo con el nombre tcb018g3d3tc.db:

```
lappend search_path /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tcb018gbwp7t/LM
set link_library " * tcb018gbwp7ttc.db tcb018gbwp7twc.db tcb018gbwp7tbc.db tpd018nvtc.db"
set target_library "tcb018gbwp7ttc.db"
```

Figura 12: Importe de librerías *link* y *target* utilizado anteriormente

Aunque sí se puede confirmar que la librería proveída por el *foundry* del proyecto, TSMC, estaba diseñada para procesos de 0.18um, no era posible confirmar con absoluta certeza el voltaje de operación que se manejaba en la alimentación de todas las celdas estándar. En la documentación disponible para la librería [7], se indicaba que el voltaje de referencia para la sección de *Absolute Maximum Ratings* y *Recommended Operating Conditions* referenciaba al voltaje del inversor, pero, al examinar el valor indicado, solo se colocaba como Vdd.

En el apartado del inversor, se indicaba que dicha celda utilizaba una alimentación de core de 1.8V. Tras discusión, se decidió utilizar un solo voltaje para todo el sistema, de 3.3V. Se decidió hacer esto porque el hecho que los **Flujo de diseño**s de diseño trabajados en el contexto de este trabajo de graduación no están optimizando para ningún parámetro de potencia, área ni retardo.

Por esta razón, se puede colocar las celdas estándar en el sistema tan distanciadas como sea necesario para soportar este voltaje. Además, en el **Runset** escogido, se ha especificado tras los cambios que se estará utilizando 3.3V como única alimentación. La última razón, y a la vez la más importante, recae en el hecho que se encontró una librería más adecuada para el proceso, la cual cuenta con las siguientes características:

Cuadro 4: Características de la nueva librería para celdas estándar escogida

Nombre	Tecnología	Voltaje	Fecha	*Foundry*
TCB018G3D3 Version 280a	0.18um	3.3V	junio 2010	TSMC

De nuevo, en el archivo de síntesis, la siguiente sección importa las librerías a utilizar, pero esta vez apunta hacia los archivos de *typical case*, *worst case* y *best case* de la nueva librería de 3.3V:

```
lappend search_path /home/administrador/Escritorio/Folder_de_Trabajo/Libs/tcb018g3d3_280a/frame_only/tcb018g3d3/LM/
set link_library " * tcb018g3d3tc.db tcb018g3d3wc.db tcb018g3d3bc.db tpd018nvtc.db"
set target_library "tcb018g3d3tc.db"
```

Figura 13: Importe de nuevas librerías *link* y *target* escogidas

El archivo tpd018nvtc.db se continúa referenciando porque esta librería corresponde a los pines de entradas y salidas, los cuales se decidió seguir utilizando.

8.3. Elección del nuevo runset

El **Runset** que se optó por escoger es distinto al escogido por la cohorte de estudiantes anterior. Considerando que el alcance del trabajo se enfocaba en dejar un **Flujo de diseño** de diseño terminado y, por cuestiones de tiempo, no en optimizar cada etapa del mismo, existía una oportunidad de escoger un **Runset** más de acuerdo al proceso que se escogió manejar en otras etapas también.

Dichos requerimientos establecen un grosor de metal superior de 40K, restringido también como 1P6M, que establece una relación de 1 capa de polisilicio por cada seis capas de metal. Además, la tecnología escogida es de 180 nm, digital CMOS, con TSMC como foundry.

Tomando esto en consideración, se encontró que el **Runset** escogido cumplía con los parámetros mencionados exceptuando la especificación del grosor del metal superior y de la relación 1P6M para los metales y el polisilicio. A continuación, se presenta especificaciones del nuevo **Runset** así como el nombre del archivo colocado por TSMC al mismo:

Cuadro 5: Características del runset ICVLM18_LM16_LM152_6M_215a_pre041518

Tipo	Tecnología	C. superior	Voltajes
CMOS	0.152, 0.16 y 0.18 um, 1P6M	40K y 20K	1.5, 1.8, 2.5, 3.3 y 5V

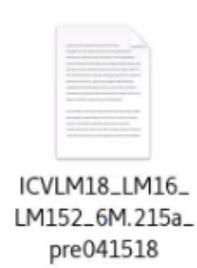


Figura 14: Nuevo runset escogido

Luego, dentro del **Runset** existen instancias bajo el nombre *define* las cuales especifican parámetros que se deben establecer para el proceso en particular que se está ejecutando. En el caso particular de los flujos para este trabajo de graduación, fue necesario definir el grosor de la capa superior de metal en 40K, el voltaje en alto de la lógica en 3.3V y el voltaje en el core, es decir, dentro de las celdas estándar del sistema, en 1.8V, tal y como se muestra a continuación:

```
1  #define THICK_40K
2  #define _3_3V
```

Figura 15: Parámetros redefinidos en el nuevo runset

Luego, de estas redefiniciones, ya se contaba con el archivo del **Runset** listo para el **DRC**.

8.4. Migración hacia Custom Compiler

En cuanto al siguiente problema encontrado para correr el **DRC**, el cual consistía en el mensaje *Error: Open Library failed*, se optó por solucionarlo haciendo un cambio de la herramienta de Synopsys utilizada para continuar el **Flujo de diseño** de diseño.

Las razones por las cuales se decidió migrar la etapa de **DRC** hacia **Custom Compiler** son tres: primero, porque se corroboró que durante la elaboración del archivo .vue, la versión del IC Validator es la misma que la de ICV VUE [8]. Segundo, porque hasta el momento, no existe una versión Q-2019.12-SP5 en ICV de tal manera que se cuente con la misma versión entre ICV y ICC.

Y por último, porque es más factible realizar correcciones a nivel de ***Layout*** utilizando **Custom Compiler** en vez de IC Compiler, dado que el primero está más orientado a realizar diseños propios a nivel difusión, mientras que el segundo, más a realizar un proceso automatizado de diseño.

8.5. Cambios en el script de síntesis física

En el Anexo 4 se incluye el nuevo **Script** de síntesis física a utilizar. Por otra parte, se presenta a continuación un resumen de los cambios efectuados en cada parte del **Flujo de diseño** de síntesis que se lleva a cabo previo al **DRC**:

	Script anterior	Script actual
Celda usada	Una para la celda y otra para el floorplan y el place and route	Una misma desde la apertura del verilog
Celdas estándar	TCB018GBWP7T Version 270a	TCB018G3D3 Version 280a
Fillers/pads	Tamaños 1 a 64	Tamaño 5
Comandos autofix	No incluidos	Incluidos

- **Celdas generadas durante síntesis** En esta etapa, se optó por dejar el **Flujo de diseño** de síntesis dentro de una misma celda en la misma Milkiway Library a través de la cual se genera el **Layout**. Esto simplifica el proceso de diseño al reducir la cantidad de definiciones de elementos, y también demostró no presentar ningún problema al utilizar el mismo archivo .CEL obtenido en el diseño frontend.
- **Celdas estándar del core** Puesto que el circuito no cuenta con requerimientos de optimización hacia una variable del sistema, es factible, como una opción de simplificación, utilizar un solo voltaje para la conducción del core y de las entradas y salidas del circuito.
- **Fillers en el core y en el anillo I/O** Considerando que el floorplan y el place and route se han generado de manera automatizada, existen espacios en el core y en el anillo de entradas y salidas que deben ser llenados con algún elemento definido por el foundry. Para esto, se optó por reducir la cantidad de tipós de celdas que se usaban para rellenar los espacios, a modo de simplificación y de adaptación a los nuevos elementos disponibles tras el cambio de librería.
- **Comandos de autofix** Tras la investigación previa de documentación de **DRC**, se encontraron comandos para el manejo y reducción de errores por reglas de diseño. Considerando que, por la manera en que se escogió realizar la síntesis física, eso es, a través de IC Compiler y de manera automatizada, el **DRC** que se refiere durante estos pasos no es el proveído por el foundry. Por ello, es necesario en cambio realizar el diagnosis de las causas por errores **DRC** hasta el final y con **Custom Compiler**, utilizando la compuerta NOT.

8.6. Prototipo final de apartado de DRC

```
# ##################################################
# ##################################################

# ##############Inicio apartado de DRC #############

# ##################################################
# ##################################################
# ##################################################

#Creado por Matthias Sibrian
#
#Inclusión de runset, cargar acá el archivo proveído por el foundry
set_physical_signoff_options -exec_cmd {icv} -drc_runset {/home/administrador/Escritorio/Matthias/sintesis/NOT/
ICVLM18_LM16_LM152_6M.215a_pre041518} -fill_runset {/home/administrador/Escritorio/Matthias/sintesis/NOT/
ICVLM18_LM16_LM152_6M.215a_pre041518}
#Prevención de errores por DRC al hacer el metal fill
signoff_metal_fill -fix_density_errors
#
#set_preroute_advanced_via_rule -size_by_via_area {1 1}
#Estrategia de floorplanning, adicional, que no intente optimizar DRC considerando las standard cells
set_preroute_drc_strategy -ignore_std_cells
#Definición de costo para resolución de conflictos al optimizar
set_cost_priority -design_rules
#Establecer optimización en prerouting para evitar congestión
set_preroute_focal_opt_strategy
#Setear optimización postrouting en todas las nets con violaciones por DRC
focal_opt -drc_nets all
#Definición de optimización incremental en el diseño solo para design rules
verify_route
#set_dont_touch
psynopt -only_design_rule
#ECO routing -- resintetiza areas con concentración de errores por DRC
route_zrt_eco
```

Figura 16: Comandos más importantes para la definición del apartado de DRC

Tras revisiones exhaustivas de la documentación disponible, proveída tanto por **Synopsys** como por **TSMC**, se pudo elaborar un prototipo final de apartado por DRC para las pruebas, el cual se presenta en la Figura 16. En dicho diseño, los comandos escogidos logran las siguientes mejoras:

- **Inclusión del DRC:** Dentro de las opciones signoff antes de realizar el place and route y en el proceso de metal filling, se optimizó para prevención de errores.
- **Estrategia de floorplanning:** Se definió que no intente optimizar DRC considerando las **Standard cells**, dado que las mismas las proveyó un foundry en formato CEL View.
- **Redefinición de costos para las reglas de diseño:** Se realizó una definición de costo superior a reglas de diseño para el proceso automático de síntesis.
- **Prevención de congestión de buffers durante el placement:** Se logró establecer un proceso de optimización en prerouting para evitar congestión, ya que, durante el placement automático, puede existir inserción de elementos tal que se introduzca más errores por DRC sistemáticamente.
- **Optimización post-routing:** Se fijó optimización postrouting en todas las nets con violaciones por DRC, así como optimización incremental en el diseño solo para design rules exclusivamente.
- **ECO-routing:** Uso de ECO routing, proceso que reduce zonas de resíntesis para reducir el *ping-pong effect*, efecto que ocurre cuando una acción de corrección de errores dispara más errores por resíntesis, y así sucesivamente.

CAPÍTULO 9

Verificación de errores

Para evaluar las mejoras propuestas, se optó por una verificación de resultados con tres partes: en la primera, se ejecutaría el DRC tal y como se realizaba por la cohorte de estudiantes anterior; en la segunda, se incluiría el runset en el DRC, se especifaría estrategias de floorplanning, de redefinición de costos en reglas DRC, de optimización incremental y de ECO Routing; por último, en la tercera, se incluiría una corrección de errores por metal filling, estrategia prerouting para evitar congestión, optimización focal y visualización de errores por ubicación, ya sea en el Core o en el anillo I/O.

Cuadro 6: Desempeño del apartado de DRC sobre los sistemas probados

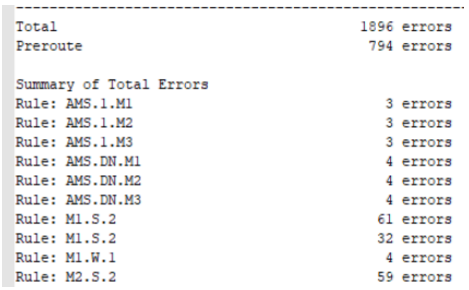
Sistema	Reducción de errores
NOT	99.05 %
NOR 3 entradas	98.73 %
NAND 2 entradas	98.21 %
Full Adder	98.04 %
4-bit counter	83.62 %

En cuanto a los sistemas escogidos, se optó por circuitos relativamente sencillos, de tal manera que se pudiera hallar comandos que solventaran problemas en elementos que, en sistemas más grandes y complejos, funcionan como bloques fundamentales. En el Cuadro 6 se puede ver el desempeño alcanzado por el apartado DRC para cada uno de los cinco sistemas evaluados. Dicho porcentaje indica la reducción de violaciones por DRC usando el apartado final, en forma de script, para la ejecución del runset de TSMC.

9.1. Compuerta NOT

Iteración	Cantidad Errores			
	Preroute	Core	I/O	Total
1	794	1102		1896
2	0	30		30
3	0	18	0	18

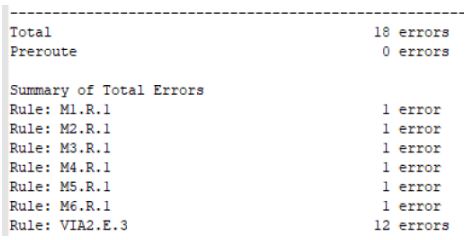
Figura 17: Desempeño final del apartado de DRC propuesto sobre la compuerta NOT



```
------------------------------------------------------------
Total                                    1896 errors
Preroute                                  794 errors

Summary of Total Errors
Rule: AMS.1.M1                              3 errors
Rule: AMS.1.M2                              3 errors
Rule: AMS.1.M3                              3 errors
Rule: AMS.DN.M1                             4 errors
Rule: AMS.DN.M2                             4 errors
Rule: AMS.DN.M3                             4 errors
Rule: M1.S.2                               61 errors
Rule: M1.S.2                               32 errors
Rule: M1.W.1                                4 errors
Rule: M2.S.2                               59 errors
```

Figura 18: Cantidad de errores por DRC para la compuerta NOT previo al uso del apartado



```
------------------------------------------------------------
Total                                      18 errors
Preroute                                    0 errors

Summary of Total Errors
Rule: M1.R.1                                1 error
Rule: M2.R.1                                1 error
Rule: M3.R.1                                1 error
Rule: M4.R.1                                1 error
Rule: M5.R.1                                1 error
Rule: M6.R.1                                1 error
Rule: VIA2.E.3                             12 errors
```

Figura 19: Cantidad de errores por DRC para la compuerta NOT posterior al uso del apartado

La reducción alcanzada en la compuerta NOT fue de 99.05 % (Cuadro 6). Inicialmente, se contaba con 1896 errores por reglas de diseño previo a la utilización de DRC, los cuales se redujeron a 30 luego de utilizar el primer apartado DRC propuesto, y a 18 tras el segundo (Fig 17). Asimismo, al examinar el tipo de errores generados luego de la ejecución con apartado de DRC, se encontró que dos tipos de errores permanecieron, VIAx.E.3 Mx.R.1 (Figs. 18 y 19).

9.2. Compuerta NOR de 3 entradas

Iteración	Cantidad Errores			
	Preroute	Core	I/O	Total
1	1069	1765		2834
2	0	36		36
3	0	18	18	36

Figura 20: Desempeño final del apartado de DRC propuesto sobre la compuerta NOR

```
------------------------------------------------------------
Total                                       2834 errors
Preroute                                    1069 errors

Summary of Total Errors
Rule: AMS.1.M1                                 5 errors
Rule: AMS.1.M2                                 5 errors
Rule: AMS.1.M3                                 5 errors
Rule: AMS.DN.M1                                8 errors
Rule: AMS.DN.M2                                8 errors
Rule: AMS.DN.M3                                8 errors
Rule: M1.S.1                                   2 errors
Rule: M1.S.2                                  68 errors
Rule: M1.S.2                                  38 errors
```

Figura 21: Cantidad de errores por DRC para la compuerta NOR previo al uso del apartado

```
------------------------------------------------------------
Total                                         36 errors
Preroute                                       0 errors

Summary of Total Errors
Rule: M1.R.1                                   1 error
Rule: M2.R.1                                   1 error
Rule: M3.R.1                                   1 error
Rule: M4.R.1                                   1 error
Rule: M5.R.1                                   1 error
Rule: M6.R.1                                   1 error
Rule: VIA2.E.3                                24 errors
Rule: VIA3.E.3                                 6 errors
```

Figura 22: Cantidad de errores por DRC para la compuerta NOR posterior al uso del apartado

Por su parte, con la compuerta NOR, la reducción alcanzada fue de 98.73 % (Cuadro 6). Inicialmente, se contaba con 2834 errores por reglas de diseño previo a la utilización de DRC, los cuales se redujeron a 36 luego de utilizar el primer apartado DRC propuesto, y permanecieron en esta cantidad tras el segundo (Fig 20). De manera similar al caso anterior, al examinar el tipo de errores generados luego de la ejecución con apartado de DRC, se encontró que dos tipos de errores permanecieron, VIAx.E.3 Mx.R.1 (Figs. 21 y 22).

9.3. Compuerta NAND de 2 entradas

Iteración	Cantidad Errores			
	Preroute	Core	I/O	Total
1	773	1240		2013
2	0	36		36
3	0	12	24	36

Figura 23: Desempeño final del apartado de DRC propuesto sobre la compuerta NAND

```
----------------------------------------------------------
Total                                       2013 errors
Preroute                                     773 errors

Summary of Total Errors
Rule: AMS.1.M1                                 4 errors
Rule: AMS.1.M2                                 4 errors
Rule: AMS.1.M3                                 4 errors
Rule: AMS.DN.M1                                6 errors
Rule: AMS.DN.M2                                6 errors
Rule: AMS.DN.M3                                6 errors
Rule: M1.S.2                                  66 errors
Rule: M1.S.2                                  35 errors
Rule: M1.W.1                                   4 errors
Rule: M2.S.2                                  63 errors
```

Figura 24: Cantidad de errores por DRC para la compuerta NAND previo al uso del apartado

```
----------------------------------------------------------
Total                                         36 errors
Preroute                                       0 errors

Summary of Total Errors
Rule: M1.R.1                                   1 error
Rule: M2.R.1                                   1 error
Rule: M3.R.1                                   1 error
Rule: M4.R.1                                   1 error
Rule: M5.R.1                                   1 error
Rule: M6.R.1                                   1 error
Rule: VIA2.E.3                                24 errors
Rule: VIA3.E.3                                 6 errors
```

Figura 25: Cantidad de errores por DRC para la compuerta NAND posterior al uso del apartado

Por otro lado, con la compuerta NAND, la reducción alcanzada fue de 98.21 % (Cuadro 6). Inicialmente, se contaba con 2013 errores por reglas de diseño previo a la utilización de DRC, los cuales se redujeron a 36 luego de utilizar el primer apartado DRC propuesto, y permanecieron en esta cantidad tras el segundo (Fig 23). De manera similar al caso anterior, al examinar el tipo de errores generados luego de la ejecución con apartado de DRC, se encontró que dos tipos de errores permanecieron, VIAx.E.3 Mx.R.1 (Figs. 24 y 25).

9.4. *Full Adder*

Iteración	Cantidad Errores			
	Preroute	Core	I/O	Total
1	1596	2388		3984
2	0	78		78
3	0	36	42	78

Figura 26: Desempeño final del apartado de DRC propuesto sobre el circuito *Full Adder*

```
----------------------------------------------------------
Total                                    3984 errors
Preroute                                 1596 errors

Summary of Total Errors
Rule: AMS.1.M1                             10 errors
Rule: AMS.1.M2                             10 errors
Rule: AMS.1.M3                             10 errors
Rule: AMS.DN.M1                            14 errors
Rule: AMS.DN.M2                            14 errors
Rule: AMS.DN.M3                            14 errors
Rule: M1.S.1                                2 errors
Rule: M1.S.2                              121 errors
Rule: M1.S.2                               43 errors
```

Figura 27: Cantidad de errores por DRC para el circuito *Full Adder* previo al uso del apartado

```
----------------------------------------------------------
Total                                      78 errors
Preroute                                    0 errors

Summary of Total Errors
Rule: M1.R.1                                1 error
Rule: M2.R.1                                1 error
Rule: M3.R.1                                1 error
Rule: M4.R.1                                1 error
Rule: M5.R.1                                1 error
Rule: M6.R.1                                1 error
Rule: VIA2.E.3                             48 errors
Rule: VIA3.E.3                             24 errors
```

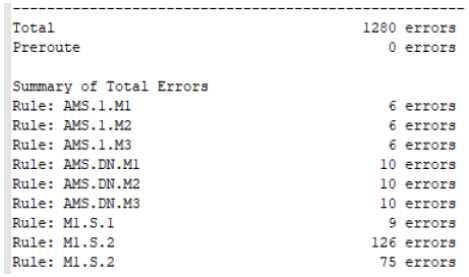
Figura 28: Cantidad de errores por DRC para el circuito *Full Adder* posterior al uso del apartado

Para el sistema *Full Adder*, la reducción alcanzada fue de 98.04 % (Cuadro 6). Inicialmente, se contaba con 3984 errores por reglas de diseño previo a la utilización de DRC, los cuales se redujeron a 78 luego de utilizar el primer apartado DRC propuesto, y permanecieron en esta cantidad tras el segundo (Fig 26). De manera similar a los casos anteriores, al revisar el tipo de errores generados luego de la ejecución con apartado de DRC, se encontró que dos tipos de errores permanecieron, VIAx.E.3 y Mx.R.1 (Figs. 27 y 28).

9.5. *4-bit Counter*

Iteración	Cantidad Errores			
	Preroute	Core	I/O	Total
1	0	1280		1280
2	0	209		209
3	0	144	65	209

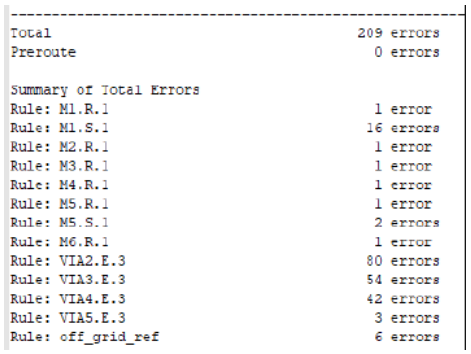
Figura 29: Desempeño final del apartado de DRC propuesto sobre el circuito *4-bit Counter*



```
----------------------------------------------------------
Total                                    1280 errors
Preroute                                    0 errors

Summary of Total Errors
Rule: AMS.1.M1                              6 errors
Rule: AMS.1.M2                              6 errors
Rule: AMS.1.M3                              6 errors
Rule: AMS.DN.M1                            10 errors
Rule: AMS.DN.M2                            10 errors
Rule: AMS.DN.M3                            10 errors
Rule: M1.S.1                                9 errors
Rule: M1.S.2                              126 errors
Rule: M1.S.2                               75 errors
```

Figura 30: Cantidad de errores por DRC para el circuito *4-bit Counter* previo al uso del apartado



```
----------------------------------------------------------
Total                                     209 errors
Preroute                                    0 errors

Summary of Total Errors
Rule: M1.R.1                                1 error
Rule: M1.S.1                               16 errors
Rule: M2.R.1                                1 error
Rule: M3.R.1                                1 error
Rule: M4.R.1                                1 error
Rule: M5.R.1                                1 error
Rule: M5.S.1                                2 errors
Rule: M6.R.1                                1 error
Rule: VIA2.E.3                             80 errors
Rule: VIA3.E.3                             54 errors
Rule: VIA4.E.3                             42 errors
Rule: VIA5.E.3                              3 errors
Rule: off_grid_ref                          6 errors
```

Figura 31: Cantidad de errores por DRC para el circuito *4-bit Counter* posterior al uso del apartado

Por último, para el circuito *4-bit Counter*, la reducción alcanzada fue de 83.62 % (Cuadro 6). Inicialmente, se contaba con 1280 errores por reglas de diseño previo a la utilización de DRC, los cuales se redujeron a 209 luego de utilizar el primer apartado DRC propuesto, y permanecieron en esta cantidad tras el segundo (Fig 29). De nuevo, se encontró que dos tipos de errores permanecieron, VIAx.E.3 Mx.R.1 (Figs. 30 y 31).

9.6. Interpretación de resultados

Como se mencionó anteriormente, existieron dos errores que fueron constantes entre los sistemas y que no se lograron solventar a través de comandos pertinentes a la etapa de DRC. Se debe tomar en cuenta que, para solventar estos errores, es necesario revisar la documentación creada por el foundry del runset utilizado y determinar en qué punto del flujo de diseño debe solventarse.

A continuación, se presenta el primero de los errores:

9.6.1. Error 1: VIAx.E.3

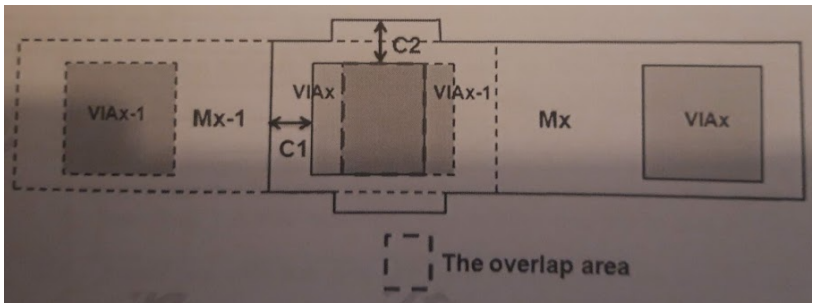


Figura 32: Diagrama ilustrativo para la comprensión del error VIAx.E.3

```
Rule: VIA2.E.3 : Minimum extension of M2 beyond the overlap area that VIA2
and VIA1 are fully or partially touching < 0.06              18 errors
```

Figura 33: Resultado en terminal con error VIAx.E.3 presente en los sistemas probados

Este error indica que es necesario extender el tamaño del área transversal que existe entre la capa 1 y 2 de metal (C2 en Fig. 32). Para el caso particular de los sistemas probados, existía una violación dado que dicha sección medía menos de 0.06 micrómetros (Fig. 33). Cabe mencionar que, dado que este error se introduce en la definición de parámetros dentro de la síntesis física, es necesario corregirlo durante la misma.

9.6.2. Error 2: Mx.R.1

El segundo error recurrente está relacionado a la densidad de la capa de metal en el sistema final, tras la finalización de los pasos de síntesis física. Específicamente, al tomar un ejemplo de las instancias recurrentes de este tipo (Fig. 34), es necesario comprender cómo se calcula la densidad de una capa de metal:

$$\frac{\text{Área total del metal en el layout}}{\text{Área total del chip}}$$

Una vez comprendido esto, es posible comprender que la densidad de las capas de metal en los sistemas generados es demasiado baja, eso es, menor al 30 % (Fig. 34). Esto tiene

```
Details of Total Errors
Rule:            M1.R.1 : Min M1 area coverage < 30%          1 error
Rule:            M2.R.1 : Min M2 area coverage < 30%          1 error
Rule:            M3.R.1 : Min M3 area coverage < 30%          1 error
Rule:            M4.R.1 : Min M4 area coverage < 30%          1 error
Rule:            M5.R.1 : Min M5 area coverage < 30%          1 error
Rule:            M6.R.1 : Min M6 area coverage < 30%          1 error
```

Figura 34: Resultado en terminal con error Mx.R.1 presente en los sistemas probados

sentido en el contexto de las pruebas, ya que no se optimizó un tamaño adecuado de área total, y también porque los circuitos usados para las pruebas consisten de elementos fundamentales. Al ser sistemas básicos, la proporción de elementos/área es muy baja, lo cual explica bastante bien este error, el cual debe corregirse en etapas previas al DRC.

CAPÍTULO 10

Documentación para futuros estudiantes

Considerando que parte de los requerimientos del mismo es lograr una rápida transición de trabajo entre estudiantes, se optó por escribir un manual de usuario y tres videotutoriales como parte del material didáctico para la nueva cohorte de estudiantes. Los temas que se cubrieron en los mismos fueron los procesos de síntesis física, de DRC y de ejecución de DRC en las herramientas disponibles.

10.1. Síntesis física

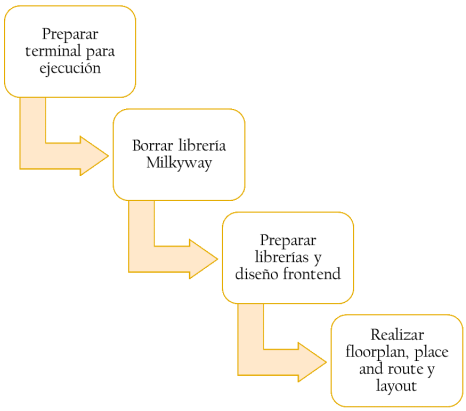


Figura 35: Flujo de trabajo para la síntesis física

La realización de la síntesis física sigue este algoritmo y utiliza IC Compiler para completarse. A continuación, se presentan dos maneras distintas para configurar el entorno de Synopsys en una terminal de CentOS. Personalmente, recomiendo realizar el procedimiento sin la GUI, pues es más rápido y no es necesario ver el proceso.

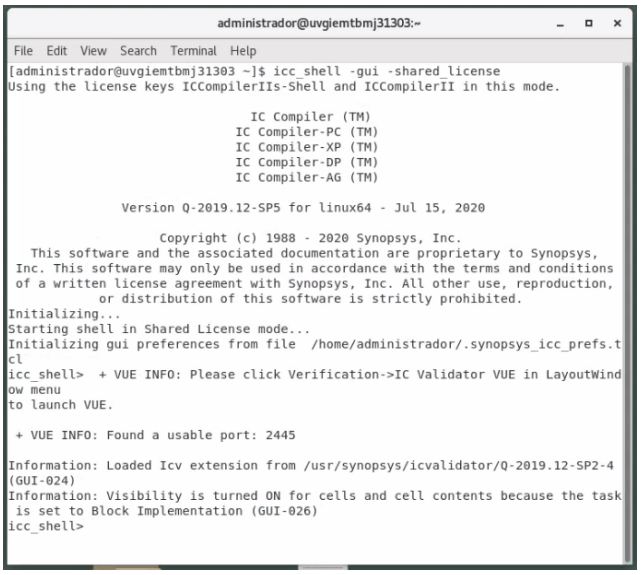


Figura 36: Vista de comandos para inicialización de IC Compiler con GUI

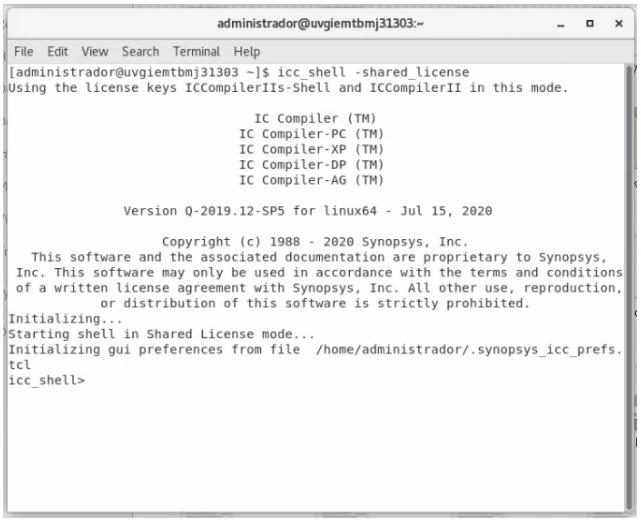


Figura 37: Vista de comandos para inicialización de IC Compiler sin GUI

En la primera manera mostrada, sí se instancia el parámetro de GUI, mientras que en la segunda no. Por la naturaleza de IC Compiler, el cual se utiliza para realizar **Flujo de diseño**s de diseño más automatizados, no causa una diferencia significativa no contar con la interfaz gráfica. Asimismo, dado que las celdas estándar del foundry están selladas, la vista de celdas no es muy relevante para el proceso de diseño.

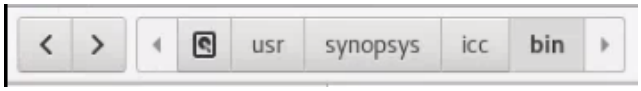


Figura 38: Ruta para llegar a la Milkyway library a crear

Figura 39: Carpeta a borrar

Después del establecimiento de la terminal a utilizar, el siguiente paso para consiste en cerciorarse que no existe una librería Milkyway en la ruta de ejecución del IC Compiler con el mismo nombre a utilizar en la presente síntesis física. La ruta se muestra acá y, si fuese necesario borrar la librería, se muestra un ejemplo de la carpeta a suprimir.

Figura 40: Preparación de librerías

Una vez corroborado lo anterior, se procede a establecer los archivos de librerías de referencia utilizados durante la síntesis lógica. Acá es donde se establece las bases de datos en donde estén definidas las vistas a colocar en el **Layout** de las celdas que se van a emplear. Luego, se crea la Milkyway Library y se lee el Verilog en donde está el sistema a generar. Para realizar esto, se hace un copy paste hacia la terminal abierta de los comandos correspondientes que se prepararon para esta fase.

```
derive_pg_connection -power_net VDD -power_pin vdd -ground_net VSS -ground_pin vss -tie
create_cell {c1 c2 c3 c4} PCORNER
create_cell {VDD} PVDD1CDG
create_cell {VSS} PVSS1CDG
set_pad_physical_constraints -pad_name "c1" -side 1
set_pad_physical_constraints -pad_name "c2" -side 2
set_pad_physical_constraints -pad_name "c3" -side 3
set_pad_physical_constraints -pad_name "c4" -side 4
set_pad_physical_constraints -pad_name "VDD" -side 2 -order 2
set_pad_physical_constraints -pad_name "VSS" -side 4 -order 2
create_floorplan -no_double_back -top_io2core 10 -bottom_io2core 10 -left_io2core 10 -right_io2core 10
adjust_fp_floorplan
set_fp_placement_strategy -adjust_shapes on
create_fp_placement -effort High -optimize_pins -congestion_driven -timing_driven
create_rectangular_rings -nets {VDD VSS} -around core
create_power_straps     \
          -direction vertical   \
          -start_at 155         \
          -num_placement_strap 4        \
          -increment_x_or_y 25  \
          -nets  {VDD}  \
          -layer METAL2 \
          -width 1.8    \
          -do_not_route_over_macros

create_power_straps     \
        -direction vertical     \
        -start_at 150   \
        -num_placement_strap 4  \
        -increment_x_or_y 25    \
        -nets  {VSS}    \
        -layer METAL3   \
        -width 1.8      \
        -do_not_route_over_macros
set_preroute_drc_strategy
set_route_mode_options -zroute false
set_route_opt_strategy
route_opt
insert_pad_filler -cell "PFILLER5"
insert_stdcell_filler -cell_with_metal "FILL1"
signoff_autofix_drc -select_rule {*}
#ANTENNA RULE CHECK
#uplevel #0 source /home/administrador/Escritorio/Folder_de_Trabajo/antennaRule_018_6lm_Mod.tcl
#verifies and reports Antenna Violations
#verify_zrt_route
set_write_stream_options -output_pin {text geometry} -keep_data_type
write_stream -lib_name mw_NOTSOPHIE -format gds "Not_IO.gds"
write_verilog -pg -unconnected_ports Not_IO
save_mw_cel
close_mw_cel
```

Figura 41: Generación del layout

Tras la ejecución de los comandos anteriores, se procede con la generación del floorplan, del place and route y eventualmente, del **Layout** completo del sistema. Para ello, de nuevo, se necesita copiar los comandos hacia la terminal y realizar su ejecución. Al final de esta fase, es necesario guardar y cerrar la celda donde se ha generado el **Layout**, pues se hará un cambio de herramienta de software luego de este procedimiento.

10.2. DRC

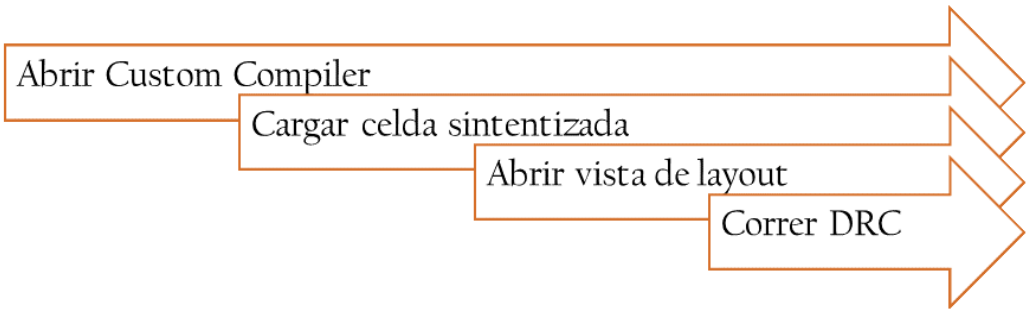


Figura 42: Flujo de trabajo para la ejecución del **DRC**

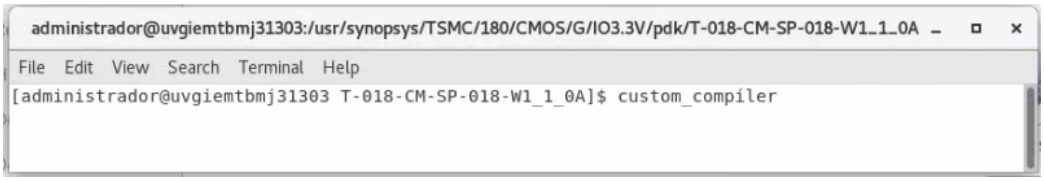


Figura 43: Inicialización de **Custom Compiler**

Como se discutió en la etapa de mejoramiento, la ejecución del **DRC** se optó por realizarse en **Custom Compiler**, el cual es más conveniente para sistemas simplificados. Para inicializar el programa, se necesita abrir una terminal en la ruta mostrada con anterioridad. De esta manera, se estará agregando en pasos posteriores la celda diseñada junto con los demás elementos de TSMC que fueron proveídos. Una vez preparada la terminal de ejecución, se corre el comando para la inicialización del **Custom Compiler**.

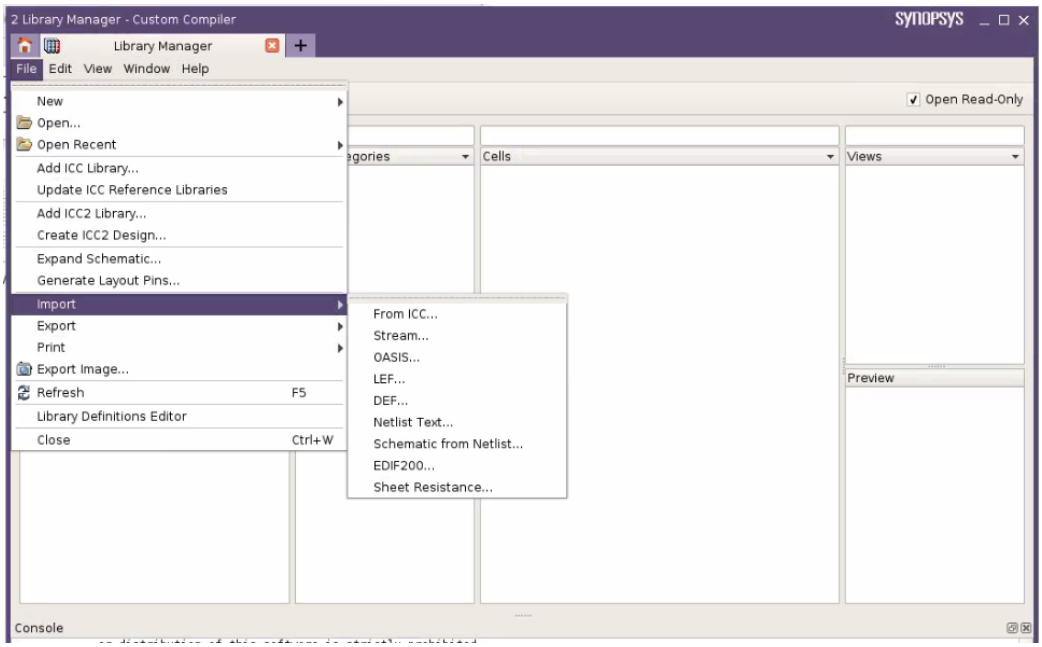


Figura 44: Importación de celda con layout

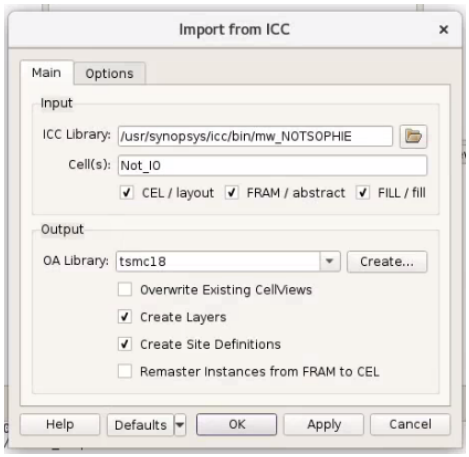


Figura 45: Inserción de librería Milkyway

En **Custom Compiler** es necesario agregar la celda con el circuito actual. Para ello, es necesario también cargar la librería que se generó con IC Compiler. Para ello, se especifica a través de *Import from ICC* la ruta de la librería Milkyway y se agrega como se muestra con anterioridad.

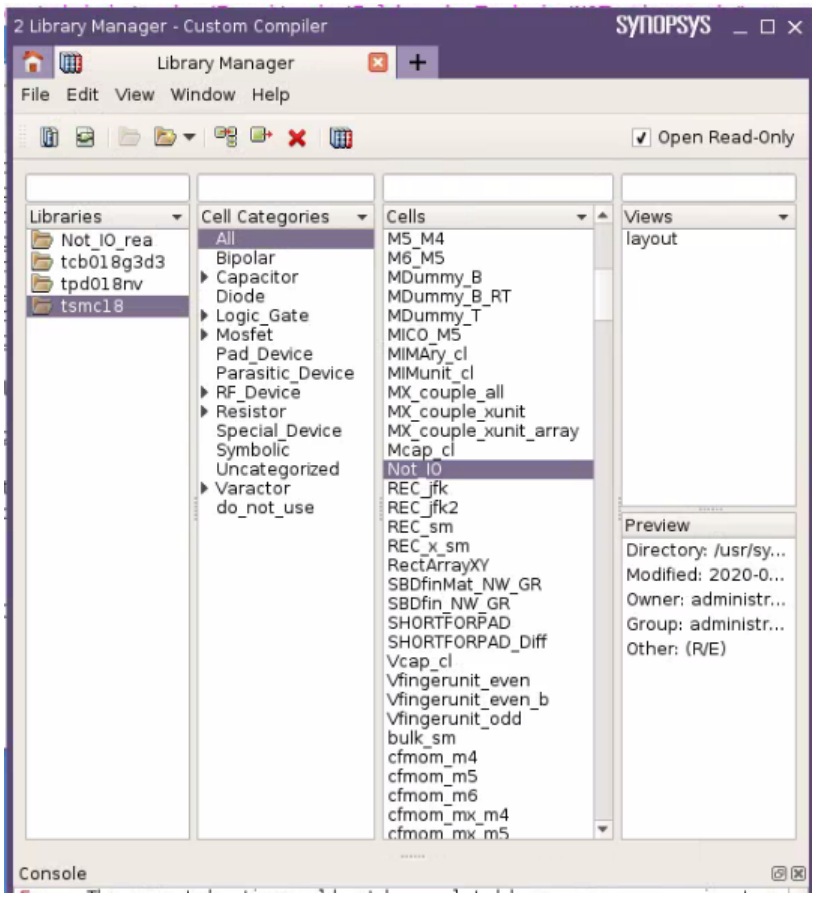


Figura 46: Carga de vista layout para celda Not_IO

Una vez importada la librería generada con IC Compiler, se procede a buscarla dentro de las celdas de TSMC para procesos con tecnología de 180 nm. Luego, se hace doble clic sobre la vista de **Layout** para abrirla.

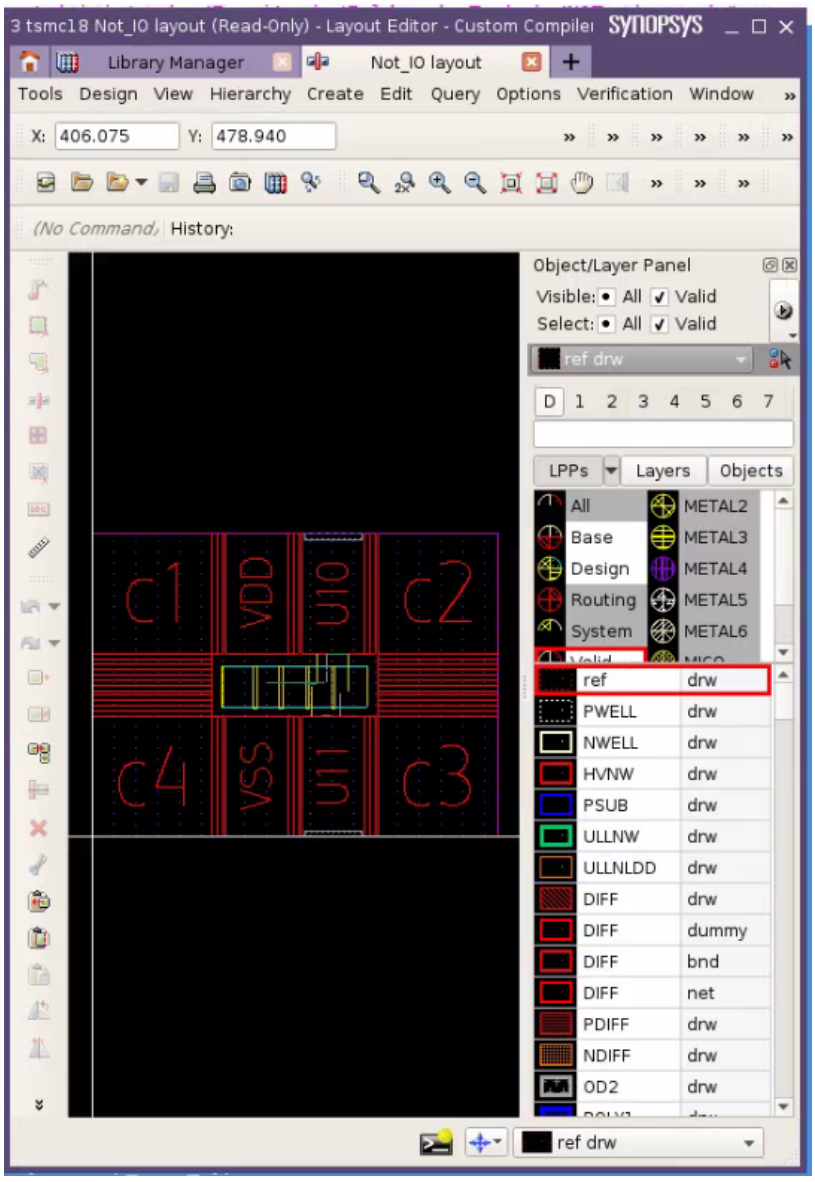


Figura 47: Vista de layout en **Custom Compiler**

Una vez abierta la vista de **Custom Compiler**, es fácil apreciar los pads de esquina en el anillo de I/O, los straps de potencia en el core del circuito, las vistas selladas de las celdas estándar proveídas por el foundry. Es importante notar que, dado que existe confidencialidad en la manera en que se realizan las celdas estándar, no es de mucha ayuda la leyenda para difusiones que provee la herramienta.

10.3. Revisión de errores

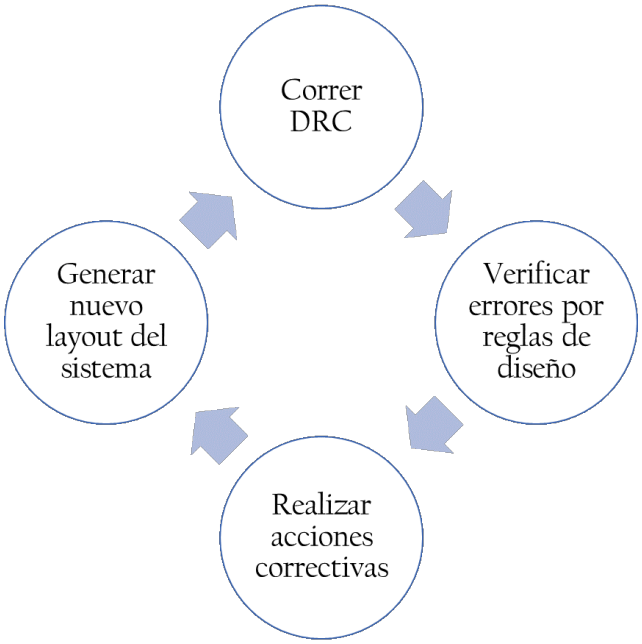


Figura 48: Flujo de trabajo para corrección de errores por **DRC**

De manera general, el proceso de revisión de errores es un ciclo cerrado, a través del cual se necesita hacer correcciones sobre el proceso de generación del **Layout** durante la síntesis física. Una vez en este punto, en donde ya se cuenta con una herramienta capaz de instanciar a IC Validator para realizar el **DRC** y que puede, a su vez, apuntar hacia el segmento en el **Layout** en el cual existe un error, es posible ya la comprensión más completa de los errores por reglas de diseño.

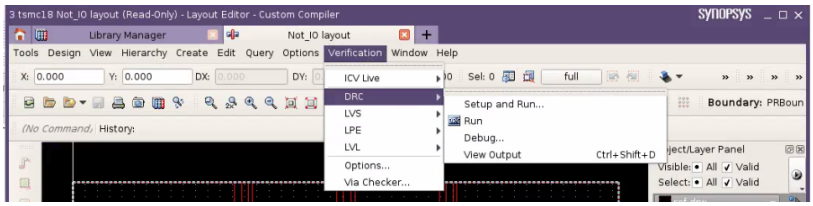


Figura 49: Verificación por **DRC**

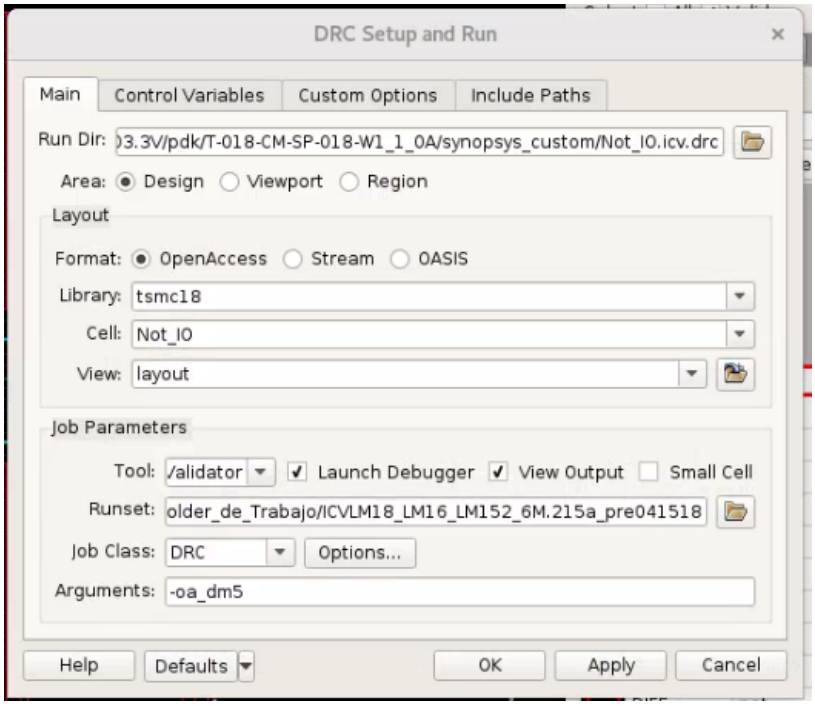


Figura 50: Definición de runset a utilizar

En **Custom Compiler**, se utiliza la interfaz gráfica mostrada anteriormente para instanciar a IC Validator y lograr así la ejecución del runset. Acá, se muestra como se importó exitosamente el nuevo runset que se escogió durante la etapa de mejoramiento.

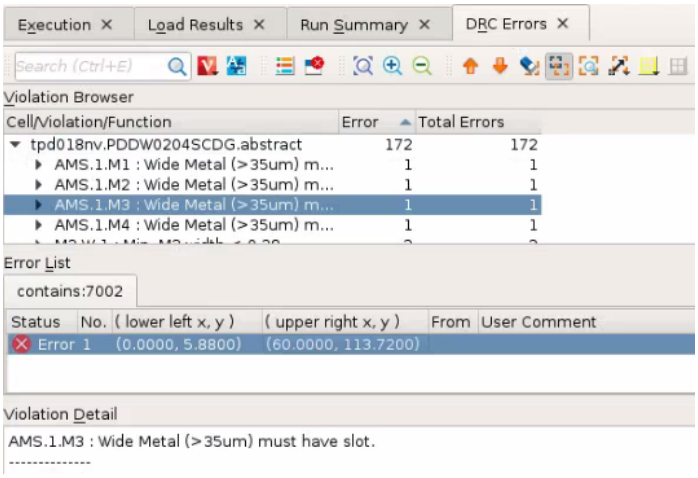


Figura 51: Interfaz para aislamiento de errores

Si existe una ejecución exitosa del IC Validator con el runset especificado, se llegará a esta vista de errores por **DRC**. Acá, la interfaz gráfica permite aislar cada error para examinar a mayor detalle la causa de su aparición. Si se hace doble clic sobre cada error en el *Error List* y se retorna hacia la vista en **Custom Compiler**, se podrá visualizar en dónde es que ocurre cada instancia de los errores. De esta manera se obtuvo un mejor entendimiento de los errores por reglas de diseño y la reducción en los mismos necesaria.

CAPÍTULO 11

Conclusiones

1. Se creó un apartado de Design Rule Check (DRC) validado a través de dos iteraciones en cinco sistemas fundamentales.
2. Se obtuvo una efectividad promedio de 95.53 % en la reducción de errores a través de las versión final del apartado de DRC.
3. Se mejoró la sección de DRC al lograr incluir un runset de un foundry particular dentro del proceso de optimizaciones previas al signoff.
4. Se creó un apartado de DRC versátil que mostró ser efectivo en cinco sistemas diferentes.
5. Se documentó a través de un manual y tres videotutoriales los pasos previos para la preparación del entorno, la ejecución del DRC y la migración hacia otras herramientas disponibles en el entorno de la universidad.

CAPÍTULO 12

Recomendaciones

1. Solventar los errores del tipo VIAx.E.3 y Mx.R.1 a través de modificaciones a la síntesis física.
2. Instalar Custom Compiler II previo a la nueva iteración de esta sección del proyecto, pues esto libera muchos más comandos para utilizar.
3. Usar los cinco sistemas de este trabajo para validar prototipos. Una vez se cuente con nuevas iteraciones para esta sección, incrementar la complejidad de los sistemas.
4. Explorar la posibilidad de una automatización para el proceso de diseño, ya que un prototyping más rápido permitirá obtener resultados más rápidamente.

CAPÍTULO 13

Bibliografía

[1] L. A. Nájera, **in** *Implementación de circuitos sintetizados a nivel netlist a partir de un diseño en lenguaje descriptivo de hardware como primer paso en el flujo de diseño de un circuito integrado*, Facultad de Ingeniería de la Universidad del Valle de Guatemala, 2019.

[2] S. H. Rubio, **in** *Definición del Flujo de Diseño para Fabricación de un Chip con Tecnologia VLSI CMOS*, Facultad de Ingeniería de la Universidad del Valle de Guatemala, 2019.

[3] N. Weste **and** D. Harris, "CMOS VLSI Design: A Circuits and Systems Perspective," **page** 3, 2011.

[4] ——, "CMOS VLSI Design: A Circuits and Systems Perspective," **page** 4, 2011.

[5] H. Kaeslin, "Digital Integrated Circuit Design: From VLSI Architectures to CMOS Fabrication," **pages** 19–25, 2008.

[6] Solvnet.com, **in** *IC Compiler Error Messages*, Synopsys, 2020.

[7] TSMC, **in** *TCB018GBWP7T TSMC 0.18um Standard Cell Library Databook*, TSMC, 2009.

[8] Solvnet.com, **in** *PYDB Errors When Trying to Load IC Validator VUE Results*, Synopsys, 2019.

[9] I. Synopsys. (). "Custom Compiler," **url**: `synopsys.com/implementation-and-signoff/custom-design-platform/custom-compiler.html`. (accessed: 04.09.2020).

[10] M. McDermott. (). "Lecture 3: CMOS Layout, Floorplanning other implementation styles," **url**: `http://users.ece.utexas.edu/~mcdermot/vlsi1/main/lectures/lecture_3.pdf`. (accessed: 04.09.2020).

[11] I. Synopsys. (). "Wherever Smart Everything Is, You'll Find Synopsys," **url**: `https://www.synopsys.com/company.html`. (accessed: 04.09.2020).

CAPÍTULO 14

Anexos

Anexo 1: Sistema NOT con último apartado de DRC

```
lappend search_path /home/administrador/Escritorio/FA_TSMC_DRC/Libs/
    tcb018gbwp7t_290a_FE/tcb018gbwp7t/LM
set link_library " * tcb018gbwp7ttc.db tcb018gbwp7twc.db tcb018gbwp7tbc.db
    tpd018nvtc.db"
set target_library "tcb018gbwp7ttc.db"
set_tlu_plus_files -max_tluplus /home/administrador/Escritorio/FA_TSMC_DRC/
    Libs/tcb018gbwp7t_290a_FE/tluplus/t018lo_1p6m_typical.tluplus
    -min_tluplus /home/administrador/Escritorio/FA_TSMC_DRC/Libs/
    tcb018gbwp7t_290a_FE/tluplus/t018lo_1p6m_typical.tluplus -tech2itf_map
    /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/
    tluplus/star.map_6M
create_mw_lib  -technology /home/administrador/Escritorio/FA_TSMC_DRC/Libs/
    tcb018gbwp7t_290a_FE/tf/tsmc018_6lm.tf   -mw_reference_library {/home/
    administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/
    tcb018gbwp7t /home/administrador/Escritorio/FA_TSMC_DRC/Libs/iolib/
    tpd018nv} -bus_naming_style {[%d]}  -open  /home/administrador/
    Escritorio/mw_counter_last
read_verilog -dirty_netlist -allow_black_box -verbose {/home/administrador/
    Escritorio/Matthias/verilog/COUNTER/CONT_syn.v}
read_sdc -echo -syntax_only -version Latest "/home/administrador/Escritorio/
    Matthias/verilog/COUNTER/CONT_sdc_p.sdc"
set_app_var mw_logic1_net "VDD"
set_app_var mw_logic0_net "VSS"
derive_pg_connection -power_net VDD -power_pin VDD -ground_net VSS
    -ground_pin VSS
derive_pg_connection -power_net VDD -ground_net VSS -tie
report_cell_physical -connections
create_cell {c1 c2 c3 c4} PCORNER
create_cell {VDD} PVDD1CDG
create_cell {VSS} PVSS1CDG
set_pad_physical_constraints -pad_name "c1" -side 1
set_pad_physical_constraints -pad_name "c2" -side 2
set_pad_physical_constraints -pad_name "c3" -side 3
set_pad_physical_constraints -pad_name "c4" -side 4
```

```
set_pad_physical_constraints -pad_name "VDD" -side 3 -order 2
set_pad_physical_constraints -pad_name "VSS" -side 4 -order 2
create_floorplan -control_type width_and_height -core_utilization 0.7
    -core_width 5 -core_height 50 -no_double_back -top_io2core 10
    -bottom_io2core 10 -left_io2core 5 -right_io2core 5
adjust_fp_floorplan
check_physical_design
set_fp_placement_strategy -adjust_shapes on
report_fp_placement_strategy
create_fp_placement -optimize_pins
legalize_placement
create_rectangular_rings -nets {VDD VSS} -around core
check_physical_design
set_app_var mw_logic1_net "VDD"
set_app_var mw_logic0_net "VSS"
derive_pg_connection -power_net VDD -power_pin VDD -ground_net VSS
    -ground_pin VSS
derive_pg_connection -power_net VDD -ground_net VSS -tie
set_physical_signoff_options -exec_cmd {icv} -drc_runset {/home/
    administrador/Escritorio/Matthias/sintesis/COUNTER/
    ICVLM18_LM16_LM152_6M.215a_pre041518} -fill_runset {/home/administrador/
    Escritorio/Matthias/sintesis/COUNTER/
    ICVLM18_LM16_LM152_6M.215a_pre041518}
signoff_metal_fill -fix_density_errors
set_preroute_advanced_via_rule -size_by_via_area {1 1}
set_preroute_drc_strategy -ignore_std_cells
set_preroute_focal_opt_strategy
focal_opt -drc_nets all
verify_route
set_dont_touch
set_cost_priority -design_rules
psynopt -only_design_rule
remove_net_routing
route_zrt_eco
report_drc_error_type
report_constraint
route_opt
verify_route
save_mw_cel  -design "CONT_IO.CEL;1"
route_search_repair -rerun_drc -loop "20"
insert_pad_filler -cel "PFILLER1 PFILLER5 PFILLER05 PFILLER0005 PFILLER10
    PFILLER20" -overlap_cell "PFILLER0005"
insert_stdcell_filler -cell_without_metal "FILL1BWP7T FILL2BWP7T FILL4BWP7T
    FILL8BWP7T FILL16BWP7T FILL32BWP7T FILL64BWP7T" -connect_to_power "VDD"
    -connect_to_ground "VSS"
save_mw_cel
save_mw_cel {CONT_IO_sdrc.err;1}
signoff_drc -show_stream_error_environment false -run_dir {/home/
    administrador/signoff_drc_run/} -max_errors_per_rule 10000
signoff_autofix_drc -config_file auto -start_repair_loop 1
    -incremental_level off -max_detail_route_iterations 5
    -max_errors_per_rule 1000 -user_defined_options {-V      }
    -read_fram_view_only
signoff_drc -show_stream_error_environment false -run_dir {/home/
    administrador/signoff_drc_run/} -max_errors_per_rule 10000
signoff_drc -show_stream_error_environment false -excluded_cell_types {pad}
    -bounding_boxes {  { 111.800 112.700 } { 356.345 177.625}  } -run_dir {/
    home/administrador/signoff_drc_run/} -max_errors_per_rule 1000
```

Anexo 2: Sistema NOR de 3 entradas con último apartado de DRC

```
lappend search_path /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tcb018gbwp7t/LM
set link_library " * tcb018gbwp7ttc.db tcb018gbwp7twc.db tcb018gbwp7tbc.db tpd018nvtc.db"
set target_library "tcb018gbwp7ttc.db"
set_tlu_plus_files -max_tluplus /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tluplus/t018lo_1p6m_typical.tluplus -min_tluplus /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tluplus/t018lo_1p6m_typical.tluplus -tech2itf_map /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tluplus/star.map_6M
create_mw_lib  -technology /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tf/tsmc018_6lm.tf   -mw_reference_library {/home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tcb018gbwp7t /home/administrador/Escritorio/FA_TSMC_DRC/Libs/iolib/tpd018nv} -bus_naming_style {[%d]}  -open  /home/administrador/Escritorio/mw_nor_last
read_verilog -dirty_netlist -allow_black_box -verbose {/home/administrador/Escritorio/Matthias/verilog/NOR/NOR_syn.v}
read_sdc -echo -syntax_only -version Latest "/home/administrador/Escritorio/Matthias/verilog/NOR/NOR_sdc_p.sdc"
set_app_var mw_logic1_net "VDD"
set_app_var mw_logic0_net "VSS"
derive_pg_connection -power_net VDD -power_pin VDD -ground_net VSS -ground_pin VSS
derive_pg_connection -power_net VDD -ground_net VSS -tie
report_cell_physical -connections
create_cell {c1 c2 c3 c4} PCORNER
create_cell {VDD} PVDD1CDG
create_cell {VSS} PVSS1CDG
set_pad_physical_constraints -pad_name "c1" -side 1
set_pad_physical_constraints -pad_name "c2" -side 2
set_pad_physical_constraints -pad_name "c3" -side 3
set_pad_physical_constraints -pad_name "c4" -side 4
set_pad_physical_constraints -pad_name "VDD" -side 3 -order 2
set_pad_physical_constraints -pad_name "VSS" -side 4 -order 2
create_floorplan -control_type width_and_height -core_utilization 0.7 -core_width 5 -core_height 50 -no_double_back -top_io2core 10 -bottom_io2core 10 -left_io2core 5 -right_io2core 5
adjust_fp_floorplan
check_physical_design
set_fp_placement_strategy -adjust_shapes on
report_fp_placement_strategy
create_fp_placement -optimize_pins
legalize_placement
create_rectangular_rings -nets {VDD VSS} -around core
check_physical_design
set_app_var mw_logic1_net "VDD"
set_app_var mw_logic0_net "VSS"
derive_pg_connection -power_net VDD -power_pin VDD -ground_net VSS -ground_pin VSS
derive_pg_connection -power_net VDD -ground_net VSS -tie
set_physical_signoff_options -exec_cmd {icv} -drc_runset {/home/administrador/Escritorio/Matthias/sintesis/NOR/ICVLM18_LM16_LM152_6M.215a_pre041518} -fill_runset {/home/administrador/
```

```
    Escritorio/Matthias/sintesis/NOR/ICVLM18_LM16_LM152_6M.215a_pre041518}
signoff_metal_fill -fix_density_errors
set_preroute_advanced_via_rule -size_by_via_area {1 1}
set_preroute_drc_strategy -ignore_std_cells
set_preroute_focal_opt_strategy
focal_opt -drc_nets all
verify_route
set_dont_touch
set_cost_priority -design_rules
psynopt -only_design_rule
remove_net_routing
route_zrt_eco
report_drc_error_type
report_constraint
route_opt
verify_route
save_mw_cel  -design "Nor_IO.CEL;1"
route_search_repair -rerun_drc -loop "20"
insert_pad_filler -cel "PFILLER1 PFILLER5 PFILLER05 PFILLER0005 PFILLER10
    PFILLER20" -overlap_cell "PFILLER0005"
insert_stdcell_filler -cell_without_metal "FILL1BWP7T FILL2BWP7T FILL4BWP7T
    FILL8BWP7T FILL16BWP7T FILL32BWP7T FILL64BWP7T" -connect_to_power "VDD"
    -connect_to_ground "VSS"
save_mw_cel
save_mw_cel {Nor_IO_sdrc.err;1}
signoff_drc -show_stream_error_environment false -run_dir {/home/
    administrador/signoff_drc_run/} -max_errors_per_rule 10000
signoff_autofix_drc -config_file auto -start_repair_loop 1
    -incremental_level off -max_detail_route_iterations 5
    -max_errors_per_rule 1000 -user_defined_options {-V      }
    -read_fram_view_only
signoff_drc -show_stream_error_environment false -run_dir {/home/
    administrador/signoff_drc_run/} -max_errors_per_rule 10000
```

Anexo 3: Sistema NAND de 2 entradas con último apartado de DRC

```
lappend search_path /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tcb018gbwp7t/LM
set link_library " * tcb018gbwp7ttc.db tcb018gbwp7twc.db tcb018gbwp7tbc.db tpd018nvtc.db"
set target_library "tcb018gbwp7ttc.db"
set_tlu_plus_files -max_tluplus /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tluplus/t018lo_1p6m_typical.tluplus -min_tluplus /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tluplus/t018lo_1p6m_typical.tluplus -tech2itf_map /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tluplus/star.map_6M
create_mw_lib  -technology /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tf/tsmc018_6lm.tf   -mw_reference_library {/home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tcb018gbwp7t /home/administrador/Escritorio/FA_TSMC_DRC/Libs/iolib/tpd018nv} -bus_naming_style {[%d]}  -open  /home/administrador/Escritorio/mw_nand
read_verilog -dirty_netlist -allow_black_box -verbose {/home/administrador/Escritorio/Matthias/verilog/NAND/NAND_syn.v}
read_sdc -echo -syntax_only -version Latest "/home/administrador/Escritorio/Matthias/verilog/NAND/NAND_sdc_p.sdc"
set_app_var mw_logic1_net "VDD"
set_app_var mw_logic0_net "VSS"
derive_pg_connection -power_net VDD -power_pin VDD -ground_net VSS -ground_pin VSS
derive_pg_connection -power_net VDD -ground_net VSS -tie
report_cell_physical -connections
create_cell {c1 c2 c3 c4} PCORNER
create_cell {VDD} PVDD1CDG
create_cell {VSS} PVSS1CDG
set_pad_physical_constraints -pad_name "c1" -side 1
set_pad_physical_constraints -pad_name "c2" -side 2
set_pad_physical_constraints -pad_name "c3" -side 3
set_pad_physical_constraints -pad_name "c4" -side 4
set_pad_physical_constraints -pad_name "VDD" -side 3 -order 2
set_pad_physical_constraints -pad_name "VSS" -side 4 -order 2
create_floorplan -control_type width_and_height -core_utilization 0.7 -core_width 5 -core_height 50 -no_double_back -top_io2core 10 -bottom_io2core 10 -left_io2core 5 -right_io2core 5
adjust_fp_floorplan
check_physical_design
set_fp_placement_strategy -adjust_shapes on
report_fp_placement_strategy
create_fp_placement -optimize_pins
legalize_placement
create_rectangular_rings -nets {VDD VSS} -around core
check_physical_design
set_app_var mw_logic1_net "VDD"
set_app_var mw_logic0_net "VSS"
derive_pg_connection -power_net VDD -power_pin VDD -ground_net VSS -ground_pin VSS
derive_pg_connection -power_net VDD -ground_net VSS -tie
set_physical_signoff_options -exec_cmd {icv} -drc_runset {/home/administrador/Escritorio/Matthias/sintesis/NAND/ICVLM18_LM16_LM152_6M.215a_pre041518} -fill_runset {/home/administrador/
```

```
    Escritorio/Matthias/sintesis/NAND/ICVLM18_LM16_LM152_6M.215a_pre041518}
signoff_metal_fill -fix_density_errors
set_preroute_advanced_via_rule -size_by_via_area {1 1}
set_preroute_drc_strategy -ignore_std_cells
set_preroute_focal_opt_strategy
focal_opt -drc_nets all
verify_route
set_dont_touch
set_cost_priority -design_rules
psynopt -only_design_rule
remove_net_routing
route_zrt_eco
report_drc_error_type
report_constraint
route_opt
verify_route
save_mw_cel  -design "Nand_IO.CEL;1"
route_search_repair -rerun_drc -loop "20"
insert_pad_filler -cel "PFILLER1 PFILLER5 PFILLER05 PFILLER0005 PFILLER10
    PFILLER20" -overlap_cell "PFILLER0005"
insert_stdcell_filler -cell_without_metal "FILL1BWP7T FILL2BWP7T FILL4BWP7T
    FILL8BWP7T FILL16BWP7T FILL32BWP7T FILL64BWP7T" -connect_to_power "VDD"
    -connect_to_ground "VSS"
save_mw_cel
save_mw_cel {Nand_IO_sdrc.err;1}
signoff_drc -show_stream_error_environment false -run_dir {/home/
    administrador/signoff_drc_run/} -max_errors_per_rule 10000
signoff_autofix_drc -config_file auto -start_repair_loop 1
    -incremental_level off -max_detail_route_iterations 5
    -max_errors_per_rule 1000 -user_defined_options {-V      }
    -read_fram_view_only
signoff_drc -show_stream_error_environment false -run_dir {/home/
    administrador/signoff_drc_run/} -max_errors_per_rule 10000
signoff_drc -show_stream_error_environment false -excluded_cell_types {pad}
    -bounding_boxes {  { 117.240 119.825 } { 173.710 223.705}  } -run_dir {/
    home/administrador/signoff_drc_run/} -max_errors_per_rule 1000
```

Anexo 4: Sistema *Full Adder* con último apartado de DRC

```
lappend search_path /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tcb018gbwp7t/LM
set link_library " * tcb018gbwp7ttc.db tcb018gbwp7twc.db tcb018gbwp7tbc.db tpd018nvtc.db"
set target_library "tcb018gbwp7ttc.db"
set_tlu_plus_files -max_tluplus /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tluplus/t018lo_1p6m_typical.tluplus -min_tluplus /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tluplus/t018lo_1p6m_typical.tluplus -tech2itf_map /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tluplus/star.map_6M
create_mw_lib  -technology /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tf/tsmc018_6lm.tf   -mw_reference_library {/home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tcb018gbwp7t /home/administrador/Escritorio/FA_TSMC_DRC/Libs/iolib/tpd018nv} -bus_naming_style {[%d]}  -open  /home/administrador/Escritorio/mw_fa_last
read_verilog -dirty_netlist -allow_black_box -verbose {/home/administrador/Escritorio/Matthias/verilog/FA/FA_SOPHIE.v}
read_sdc -echo -syntax_only -version Latest "/home/administrador/Escritorio/Matthias/verilog/FA/FA_sdc_p.sdc"
set_app_var mw_logic1_net "VDD"
set_app_var mw_logic0_net "VSS"
derive_pg_connection -power_net VDD -power_pin VDD -ground_net VSS -ground_pin VSS
derive_pg_connection -power_net VDD -ground_net VSS -tie
report_cell_physical -connections
create_cell {c1 c2 c3 c4} PCORNER
create_cell {VDD} PVDD1CDG
create_cell {VSS} PVSS1CDG
set_pad_physical_constraints -pad_name "c1" -side 1
set_pad_physical_constraints -pad_name "c2" -side 2
set_pad_physical_constraints -pad_name "c3" -side 3
set_pad_physical_constraints -pad_name "c4" -side 4
set_pad_physical_constraints -pad_name "VDD" -side 3 -order 2
set_pad_physical_constraints -pad_name "VSS" -side 4 -order 2
create_floorplan -control_type width_and_height -core_utilization 0.7 -core_width 5 -core_height 50 -no_double_back -top_io2core 10 -bottom_io2core 10 -left_io2core 5 -right_io2core 5
adjust_fp_floorplan
check_physical_design
set_fp_placement_strategy -adjust_shapes on
report_fp_placement_strategy
create_fp_placement -optimize_pins
legalize_placement
create_rectangular_rings -nets {VDD VSS} -around core
check_physical_design
set_app_var mw_logic1_net "VDD"
set_app_var mw_logic0_net "VSS"
derive_pg_connection -power_net VDD -power_pin VDD -ground_net VSS -ground_pin VSS
derive_pg_connection -power_net VDD -ground_net VSS -tie
set_physical_signoff_options -exec_cmd {icv} -drc_runset {/home/administrador/Escritorio/Matthias/sintesis/FA/ICVLM18_LM16_LM152_6M.215a_pre041518} -fill_runset {/home/administrador/Escritorio/Matthias/sintesis/FA/ICVLM18_LM16_LM152_6M.215a_pre041518}
```

```
signoff_metal_fill -fix_density_errors
set_preroute_advanced_via_rule -size_by_via_area {1 1}
set_preroute_drc_strategy -ignore_std_cells
set_preroute_focal_opt_strategy
focal_opt -drc_nets all
verify_route
set_dont_touch
set_cost_priority -design_rules
psynopt -only_design_rule
remove_net_routing
route_zrt_eco
report_drc_error_type
report_constraint
route_opt
verify_route
save_mw_cel  -design "FA_IO.CEL;1"
route_search_repair -rerun_drc -loop "20"
insert_pad_filler -cel "PFILLER1 PFILLER5 PFILLER05 PFILLER0005 PFILLER10
    PFILLER20" -overlap_cell "PFILLER0005"
insert_stdcell_filler -cell_without_metal "FILL1BWP7T FILL2BWP7T FILL4BWP7T
    FILL8BWP7T FILL16BWP7T FILL32BWP7T FILL64BWP7T" -connect_to_power "VDD"
    -connect_to_ground "VSS"
save_mw_cel
save_mw_cel {FA_IO_sdrc.err;1}
signoff_drc -show_stream_error_environment false -run_dir {/home/
    administrador/signoff_drc_run/} -max_errors_per_rule 10000
signoff_autofix_drc -config_file auto -start_repair_loop 1
    -incremental_level off -max_detail_route_iterations 5
    -max_errors_per_rule 1000 -user_defined_options {-V      }
    -read_fram_view_only
signoff_drc -show_stream_error_environment false -run_dir {/home/
    administrador/signoff_drc_run/} -max_errors_per_rule 10000
signoff_drc -show_stream_error_environment false -excluded_cell_types {pad}
    -bounding_boxes {  { 116.640 120.675 } { 233.780 282.760}  } -run_dir {/
    home/administrador/signoff_drc_run/} -max_errors_per_rule 1000
```

Anexo 5: Sistema *4-bit Counter* con último apartado de DRC

```
lappend search_path /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tcb018gbwp7t/LM
set link_library " * tcb018gbwp7ttc.db tcb018gbwp7twc.db tcb018gbwp7tbc.db tpd018nvtc.db"
set target_library "tcb018gbwp7ttc.db"
set_tlu_plus_files -max_tluplus /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tluplus/t018lo_1p6m_typical.tluplus -min_tluplus /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tluplus/t018lo_1p6m_typical.tluplus -tech2itf_map /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tluplus/star.map_6M
create_mw_lib  -technology /home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tf/tsmc018_6lm.tf   -mw_reference_library {/home/administrador/Escritorio/FA_TSMC_DRC/Libs/tcb018gbwp7t_290a_FE/tcb018gbwp7t /home/administrador/Escritorio/FA_TSMC_DRC/Libs/iolib/tpd018nv} -bus_naming_style {[%d]}  -open  /home/administrador/Escritorio/mw_counter_last
read_verilog -dirty_netlist -allow_black_box -verbose {/home/administrador/Escritorio/Matthias/verilog/COUNTER/CONT_syn.v}
read_sdc -echo -syntax_only -version Latest "/home/administrador/Escritorio/Matthias/verilog/COUNTER/CONT_sdc_p.sdc"
set_app_var mw_logic1_net "VDD"
set_app_var mw_logic0_net "VSS"
derive_pg_connection -power_net VDD -power_pin VDD -ground_net VSS -ground_pin VSS
derive_pg_connection -power_net VDD -ground_net VSS -tie
report_cell_physical -connections
create_cell {c1 c2 c3 c4} PCORNER
create_cell {VDD} PVDD1CDG
create_cell {VSS} PVSS1CDG
set_pad_physical_constraints -pad_name "c1" -side 1
set_pad_physical_constraints -pad_name "c2" -side 2
set_pad_physical_constraints -pad_name "c3" -side 3
set_pad_physical_constraints -pad_name "c4" -side 4
set_pad_physical_constraints -pad_name "VDD" -side 3 -order 2
set_pad_physical_constraints -pad_name "VSS" -side 4 -order 2
create_floorplan -control_type width_and_height -core_utilization 0.7 -core_width 5 -core_height 50 -no_double_back -top_io2core 10 -bottom_io2core 10 -left_io2core 5 -right_io2core 5
adjust_fp_floorplan
check_physical_design
set_fp_placement_strategy -adjust_shapes on
report_fp_placement_strategy
create_fp_placement -optimize_pins
legalize_placement
create_rectangular_rings -nets {VDD VSS} -around core
check_physical_design
set_app_var mw_logic1_net "VDD"
set_app_var mw_logic0_net "VSS"
derive_pg_connection -power_net VDD -power_pin VDD -ground_net VSS -ground_pin VSS
derive_pg_connection -power_net VDD -ground_net VSS -tie
set_physical_signoff_options -exec_cmd {icv} -drc_runset {/home/administrador/Escritorio/Matthias/sintesis/COUNTER/ICVLM18_LM16_LM152_6M.215a_pre041518} -fill_runset {/home/administrador/Escritorio/Matthias/sintesis/COUNTER/
```

```
    ICVLM18_LM16_LM152_6M.215a_pre041518}
signoff_metal_fill -fix_density_errors
set_preroute_advanced_via_rule -size_by_via_area {1 1}
set_preroute_drc_strategy -ignore_std_cells
set_preroute_focal_opt_strategy
focal_opt -drc_nets all
verify_route
set_dont_touch
set_cost_priority -design_rules
psynopt -only_design_rule
remove_net_routing
route_zrt_eco
report_drc_error_type
report_constraint
route_opt
verify_route
save_mw_cel  -design "CONT_IO.CEL;1"
route_search_repair -rerun_drc -loop "20"
insert_pad_filler -cel "PFILLER1 PFILLER5 PFILLER05 PFILLER0005 PFILLER10
    PFILLER20" -overlap_cell "PFILLER0005"
insert_stdcell_filler -cell_without_metal "FILL1BWP7T FILL2BWP7T FILL4BWP7T
    FILL8BWP7T FILL16BWP7T FILL32BWP7T FILL64BWP7T" -connect_to_power "VDD"
    -connect_to_ground "VSS"
save_mw_cel
save_mw_cel {CONT_IO_sdrc.err;1}
signoff_drc -show_stream_error_environment false -run_dir {/home/
    administrador/signoff_drc_run/} -max_errors_per_rule 10000
signoff_autofix_drc -config_file auto -start_repair_loop 1
    -incremental_level off -max_detail_route_iterations 5
    -max_errors_per_rule 1000 -user_defined_options {-V      }
    -read_fram_view_only
signoff_drc -show_stream_error_environment false -run_dir {/home/
    administrador/signoff_drc_run/} -max_errors_per_rule 10000
signoff_drc -show_stream_error_environment false -excluded_cell_types {pad}
    -bounding_boxes {  { 111.800 112.700 } { 356.345 177.625}  } -run_dir {/
    home/administrador/signoff_drc_run/} -max_errors_per_rule 1000
```

CAPÍTULO 15

Glosario

Custom Compiler: Esta es una herramienta dentro de la suite que ofrece Synopsys para el diseño de circuitos integrados digitales, analógicos y mixtos. En este trabajo de graduación, se hizo uso extenso de su compatibilidad con las librerías *Milkyway* para poder trabajar con los *layouts* que se generan a partir de la síntesis física [9]. 27, 28, 43, 44, 46, 48, 49

DRC: Design Rule Check (DRC) es un término utilizado para describir la verificación de violaciones por reglas de diseño que puden ocurrir durante la síntesis física desde un circuito lógico hacia un *layout* fabricable. 17–21, 23, 24, 27, 28, 43, 47–49

Flujo de diseño: Es un término utilizado para describir el proceso que se lleva a cabo para la elaboración completa de un circuito integrado de escala nanométrica, desde su descripción conductal hasta el archivo necesario para la fabricación en silicio.. 18, 24–28, 40

Layout: Este es un término que se utiliza para especificar el archivo descriptivo que detalla la manera en que las capas y las geometrías en el sustrato de silicio implementan las funciones para las que se diseñó el circuito [10]. 17, 18, 27, 28, 41, 42, 45, 47

Runset: : Este término se utiliza para describir un juego de instrucciones que escanea y procesa las formas de un diseño físico con el propósito de detectar violaciones de reglas de diseño para una tecnología de fabricación en particular. 17, 25–27

Script: Es un término en inglés utilizado para describir un archivo de texto a través del cual se da forma a un algoritmo con la sintaxis correcta del lenguaje de programación en el que está escrito. 17, 28

Standard cells: : Es un término utilizado para describir las celdas que se colocan dentro del core del circuito y que no van conectadas directamente a los metales utilizados para conectar el sistema con otros elementos de electrónica discreta. 25, 29

Synopsys: Es una empresa americana que está especializada en el desarrollo de software utilizado en el diseño de circuitos electrónicos. En el contexto de este trabajo de gra-

duación, ha sido el proveedor de la suite de programas necesaria para el diseño y la ejecución del flujo de diseño de circuitos integrados [11]. 29

TSMC: La empresa Taiwan Semiconductor Manufacturing Company Limited (TSMC) es una empresa fabricante de semiconductores de Taiwán. En el presente, es una de las primeras plantas con la capacidad de elaborar ostias de silicio con nodos de 7 nanómetros. En el contexto del proyecto, TSMC es el proveedor de algunos de los *runsets* con los que se elaborarán pruebas de funcionalidad. 29