Trabajo Fin de Máster

Máster en Microelectrónica: Diseño y Aplicaciones de Sistemas Micro/Nanométricos

Establecimiento y Medida de Figuras de Seguridad Criptográfica en Función de la Potencia

Autor/Author Dorlin Bonilla Zapata

Tutores/Advisors Antonio Jose Acosta J.

Erica Tena Sanchez

Fecha/Date Noviembre 2021

RESUMEN

Los ataques de canal lateral se utilizan para revelar datos secretos de dispositivos criptográficos mediante la extracción de información física como el tiempo de cómputo, el consumo de energía (potencia) o la radiación electromagnética, entre otros. Los ataques por análisis de potencia diferencial suelen ser efectivos porque existe una correlación entre el consumo de potencia y el dato procesado, razón por la cual en los últimos años se han propuesto varias contramedidas a nivel de arquitectura, algoritmo y *hardware* para prevenir este tipo de ataques, no obstante a veces aparecen vulnerabilidades de canal lateral en implementaciones *hardware* donde el consumo de energía resulta dependiente de la lógica involucrada.

Frente a la necesidad de cifrar la información para que sea ilegible para terceros en el actual escenario mundial donde la ciberseguridad es vital para un sin fin de procesos y sistemas, y donde cada vez hacemos un mayor uso de dispositivos portables que cuentan con recursos limitados, se hace necesario investigar sobre nuevos mecanismos, algoritmos, circuitos y técnicas que permitan desarrollar implementaciones más seguras a la vez que ofrezcan soluciones con mayor eficiencia energética y mejores prestaciones a nivel de *hardware*, a medida que la tecnología interconectada (IOT) continúa impactando y moldeando la forma en que trabajamos, vivimos y nos relacionamos.

En este trabajo se evalúan por simulación eléctrica un conjunto de celdas lógicas digitales implementadas en estilos lógicos *Differential Precharge Logic* - DPL a nivel de transistor (SABL, CRSABL, DyCML, WDDL) en términos de prestaciones de seguridad, potencia y retraso, las cuales son aplicables a circuitos y dispositivos criptográficos seguros.

Palabras claves: DPA, Side channel attacks, dispositivos criptográficos, estilos lógicos DPL, SABL, CRSABL, DyCML, WDDL, seguridad, potencia, retraso

ABSTRACT

Side channel attacks are used to reveal secret data from cryptographic devices by extracting physical information such as computing time, energy consumption (power) or electromagnetic radiation, among others. Differential power analysis attacks are usually effective because there is a correlation between power consumption and the processed data, which is why in recent years several countermeasures have been proposed at the architecture, algorithm and hardware level to prevent this type of attacks, however side channel vulnerabilities sometimes appear in hardware implementations where power consumption is dependent on the logic involved.

Faced with the need to encrypt information so that it is unreadable for third parties in the current world scenario where cybersecurity is vital for an endless number of processes and systems, and where we increasingly use portable devices that have limited resources, It is necessary to investigate new mechanisms, algorithms, circuits and techniques that allow the development of more secure implementations while offering solutions with greater energy efficiency and better performance at the hardware level, as interconnected technology (IOT) continues to impact and shape the way we work, live and interact.

In this work, a set of digital logic cells implemented in Differential Precharge Logic - DPL logic styles at the transistor level (SABL, CRSABL, DyCML, WDDL) are evaluated by electrical simulation in terms of safety, power and delay performance, which are applicable to secure cryptographic circuits and devices.

Keywords: Side Channel Attacks, Dual-rail with Precharge Logic (DPL), Differential Power Analysis DPA, SABL, CRSABL, DyCML, WDDL, safety, delay, cryptographic circuits

AGRADECIMIENTOS

Quiero agradecer a mi amada familia por ser mi soporte y motivación para ser y hacer.

A mis amigos, toda esa gente bonita y cercana al corazón que me ha acompañado en esta caminada.

A Antonio y Erica, por su orientación y acompañamiento.

A las familias Blanco Majuelo y Arbulu Calvo por toda su comprensión y apoyo en esta etapa.

A César e Mauro, pelas dicas e ajuda com o uso das ferramentas analógicas, vocês são demais guris!

Índice de contenido

Índice de ilustraciones

Índice de tablas

1. Introducción

En la era actual el masivo uso de las comunicaciones digitales está desencadenando en un número cada vez más creciente de problemas de seguridad, puesto que las comunicaciones se han convertido en un elemento indispensable en nuestra cotidianidad y donde muchísima información es transmitida y almacenada día a día desde diferentes plataformas y medios. Esta información fácilmente podría ser robada o interceptada por personas no autorizadas para fines fraudulentos e indeseados; por ello se hace necesario implementar mecanismos y técnicas para proteger las transmisiones de documentos y datos, garantizando la confiabilidad de los mensajes transmitidos y autenticando eficientemente a los usuarios de un sistema, procurando que sus comunicaciones se realicen en un entorno seguro y que dicha información llegue solo a quien deba tener acceso a ella y no a receptores no autorizados. De esta necesidad de garantizar la seguridad de la información surgió la criptografía moderna. El uso de estos sistemas es más cercano a nuestras vidas de lo que puede aparecer a simple vista. La ciencia de la criptografía a lo largo de los últimos años está siendo ampliamente empleada y se está convirtiendo en una “piedra angular” de procesos como lo son las operaciones bancarias seguras, las comunicaciones telefónicas seguras, el comercio electrónico, controles de accesos a edificaciones, la identificación y autentificación de usuarios de un sistema o recursos, los sistemas de certificación, Industria 4.0, y todo lo que engloba la ciberseguridad, por mencionar algunos aspectos.

El nuevo escenario mundial digitalizado e hiperconectado demanda implementaciones *hardware* seguras, eficientes energéticamente y con altas prestaciones. Con el aumento de dispositivos portátiles y el uso extendido del *IoT* se precisa que los nuevos dispositivos *hardware* sean pequeños, robustos, seguros y eficientes.

En los últimos años el creciente interés por implementar y desarrollar dispositivos seguros ha dado lugar a muchos trabajos de investigación sobre nuevos ataques a dispositivos criptográficos y también sobre nuevas contramedidas para evitarlos, tanto a nivel de algoritmo como de implementación de hardware.

Este trabajo pretende el diseño y la evaluación por simulación eléctrica con CADENCE de las características de consumo de potencia y seguridad de un conjunto de celdas lógicas digitales tipo DPL, realizadas siguiendo distintas técnicas, aplicables a circuitos microelectrónicos criptográficos seguros ante la necesidad creciente de estudiar y desarrollar tales

implementaciones. En este documento se diseñan y comparan los distintos estilos lógicos en términos de prestaciones de seguridad, rendimiento, potencia y retraso.

En la sección 2 se hará una introducción a los conceptos básicos de la criptografía, hablaremos sobre la seguridad en los dispositivos criptográficos, presentando después técnicas de protección frente ataques no invasivos y contramedidas. Seguidamente, en la sección 3 se abordará el diseño e implementación de las celdas lógicas, seguido en la sección 4 de las métricas para evaluar la seguridad y desempeño. En la sección 5 se presentarán y discutirán los resultados obtenidos, y finalmente se muestran las conclusiones en el apartado 6.

2. Revisión del Estado del Arte

2.1. Introducción a la criptografía

En el mundo actual forman parte de nuestro día a día diversos dispositivos y sistemas integrados de comunicación tales como: Smartphones, ordenadores, tarjetas inteligentes, sistemas RFID y otro tipo de dispositivos portátiles entre otros. Estos dispositivos procesan, almacenan e intercambian información con otros, buena parte de la información que es almacenada o transmitida corresponde a información personal delicada con la que se puede suplantar la identidad para perpetrar hechos delictivos o incluso exponer al peligro la integridad del propietario o su entorno.

En la actualidad el procesamiento de información avanza a pasos agigantados hacia el uso de los dispositivos portátiles integrados, lo cual condiciona a que los canales de transmisión de la información que contienen, adopten sistemas de protección adicionales. De esta manera, se ha hecho necesario garantizar la seguridad de tales dispositivos haciendo uso de la criptografía, llegando esta a convertirse en uno de los ejes del diseño de circuitos integrados y sistemas en la actualidad.

Desde el punto de vista de implementación, la seguridad hoy en día es un elemento crucial para el diseñador de circuitos integrados, ya que una implementación *hardware* débil fácilmente puede ser atacada por terceros y/o personal no autorizado. Dispositivos tales como teléfonos inteligentes, tarjetas bancarias, tarjetas inteligentes, dispositivos RFID, sensores, etc. requieren seguridad adicional cuando albergan información secreta que debe ser protegida.

Ninguno de los algoritmos existentes para cifrado está exento de ser roto o quebrado. Ningún circuito o sistema está exento de ser violado, si el atacante dispone del tiempo, conocimiento y los recursos necesarios para poder llevar a cabo un acceso no autorizado a un sistema o la interceptación y descifrado de cualquier mensaje. Si bien es cierto que en la actualidad los algoritmos criptográficos son matemáticamente seguros, las implementaciones de estos algoritmos a nivel de *hardware* pueden sufrir ataques y el robo de información fruto de la observación del funcionamiento del circuito en modo normal de trabajo. En 1998 Kocher et.al demostró que las tarjetas inteligentes y la información secreta contenida en ellas, podían fácilmente sucumbir ante los ataques de análisis de potencia [8]. Un tipo de ataque que hace uso de este principio son los llamados ataques de canal lateral, *Side Chanel Atack* - SCA.

Básicamente el objetivo que se persigue con este tipo de ataque es procesar e interpretar las medidas de una variable física del dispositivo para conseguir averiguar la clave secreta. A continuación exploraremos este tipo de ataques.

2.2. Tipos de ataques y vulnerabilidades criptográficas

Un ataque en criptografía es cualquier intento que tenga como objetivo romper o quebrar un sistema criptográfico, lo que implica no solamente la obtención de la clave de dicho cifrado, sino que también puede ser la obtención completa o parcial del mensaje original, invertir o conseguir una colisión en un *Hash*, etc.

Existen varios tipos y categorías para los ataques a dispositivos criptográficos, las diferencias entre ellos básicamente radican en términos del coste, tiempo, equipamiento necesario y también en la experiencia del atacante. Es por esta razón que existen varios criterios de clasificación, a grosso modo podemos hablar de ataques activos o pasivos, y dependiendo de la interfaz usada en el ataque existirán ataques invasivos, semi-invasivos y no invasivos [12] y [13]. En la tabla 1 se presenta una clasificación de los diferentes tipos de ataques criptográficos.

Cualquier método conocido como “seguro” tiene un posible ataque que puede aplicarse para romper dicho sistema, los tipos de ataques más conocidos son: ataques por ingeniería inversa, inserción de fallos y Ataques *Side Channel*, para los objetivos del presente trabajo nos centraremos en los ataques *Side Channel*.

El *hardware* de cifrado que se comporta de determinadas maneras dependiendo de los datos que circulen por él y ante diferentes tareas en el cifrado. Este hecho es aprovechado por los ataques de canal lateral o *Side Channel Attack* - SCA, que buscan vulnerabilidades en el sistema en el que están implementados los criptocircuitos o en el medio por el que circulan los datos. Para tal fin estudian comportamentalmente tales sistemas ante la presencia de una operación de cifrado/descifrado procurando un filtrado de información (midiendo cantidades físicas como el tiempo de ejecución, el consumo de energía, la radiación electromagnética, etc.) y de esta manera recuperar parte o la totalidad de la clave secreta almacenada en el dispositivo criptográfico. Este tipo de ataques representa un riesgo alto para los cifrados.

Los ataques de canal lateral pueden ser ataques activos no invasivos (insertando fallas) o también pasivos. Posiblemente a día de hoy los ataques de canal lateral más conocidos sean: Arranque en frío e inicio alternativo, monitorización del hardware, extracción directa, análisis electromagnético, análisis de potencia y de estados, ataques de tiempo, entre otros.

Dentro de los SCA, los ataques de análisis de potencia, representan una gran amenaza para la seguridad de los datos procesados y almacenados en dispositivos criptográficos y a su vez que suelen ser los más efectivos [8, 9], a lo largo de estos últimos años se han propuesto muchas contramedidas frente a ellos [10, 11]. Este tipo de análisis a su vez se encuentra dividido dentro de 3 categorías: *Simple power analysis* - SPA o Análisis de potencia simple, *Differential power analysis* - DPA o análisis de potencia diferencial y *Correlation power analysis* - CPA o análisis de correlación en la potencia.

En los ataques mediante análisis diferencial de potencia (DPA), se acostumbran a utilizar muchas mediciones para filtrar el ruido, explotando la relación entre los datos procesados y el consumo o fuga de energía. Está técnica también se vale del empleo de modelos hipotéticos del consumo del dispositivo bajo ataque, comparando las salidas de modelo con las salidas medidas del dispositivo y de esta manera predecir los valores de la clave. El objetivo central de un ataque DPA es revelar la clave secreta del dispositivo valiéndose de la información proporcionada por las trazas de consumo de potencia de alimentación medidas de los dispositivos mientras cuando el núcleo criptográfico está en proceso de cifrado o descifrado [12].

En cuanto a los ataques, este trabajo se centra exclusivamente en los ataques de análisis de potencia, específicamente a los DPA.

2.3. Protección frente a ataques

Frente a las diferentes técnicas de ataques que sufren los dispositivos criptográficos se hace imprescindible el desarrollo e implementación de contramedidas para proteger la vulnerabilidad de dichos dispositivos. Uno de los frentes de acción radica en el propio diseño de los circuitos. Dependiendo del nivel de abstracción (a nivel de transistor, puerta, arquitectura y algoritmo) o fase de diseño es posible aplicar diferente tipo de contramedidas. El empleo de una sobre otra también va a depender de los requerimientos (área, potencia, frecuencia, coste, etc), prestaciones y criterio de seguridad.

ATAQUE	CONTRAMEDIDA	PROTECCIÓN
FRENTE A INSERCIÓN DE FALLOS	A nivel de hardware	Capa de metal Mecanismos de detección
	Duplicación de la lógica	
FRENTE A ATAQUES DE CANAL LATERAL	Transistor	Masking Hiding
	Puerta	Random masking Random pre-charging Transiciones de estado Distancia Hamming
	Arquitectura	Variar temporización Variar potencia Agregar ruido al consumo de energía del dispositivo Duplicar lógicas con operaciones complementarias Filtrado de corriente Batería en chip Fuente de alimentación extraíble
	Algoritmo	Random masking Variables intermedias

1. Tabla: Contramedidas y mecanismos de protección frente a ataques criptográficos

2.4. Contramedidas DPA a nivel de transistor

Las contramedidas DPA procuran tornar difícil o imposible la adivinación de una clave oculta en un dispositivo criptográfico, para ello se enfocan en desligar el consumo de potencia del circuito del dato durante el proceso de encriptación o desencriptación.

Existen varias técnicas de contramedidas, las cuales pueden ser clasificadas en dos grupos: técnicas de enmascaramiento o *masking* y técnicas de ocultación o *hiding* [4, 9]. Las dos técnicas a sus vez pueden implementar contramedidas a nivel hardware o software, así como también pueden actuar sobre la arquitectura o a nivel de las celdas lógicas.

Las técnicas de *masking* buscan generar valores intermedios aleatorios dentro del dispositivo criptográfico, haciendo uso de máscaras que se mezclan con los datos al procesar, de esta forma se desvincula el consumo de potencia del circuito con los datos procesados y enmascara el dato real procesado. Entre los estilos lógicos que usan esta técnica están: *Masked Dual-rail Precharged Logic* - MDPL e *Improved Masked Dual-rail Precharged Logic* - iMDPL, entre otros [12].

Las técnicas de *hiding* procuran ocultar el consumo de potencia real del circuito fruto de hacerlo independiente del dato procesado, a través del consumo de potencia aleatoria o haciendo que el consumo sea constantemente el mismo frente a cada ciclo de reloj [12].

A nivel de celdas lógicas está técnica se basa en el diseño de celdas con consumo de energía independiente del dato procesado. Numerosos estudios e investigaciones relacionadas han surgido en los últimos años a este respecto, puesto que esta técnica hace posible implementar *hardware* independiente del algoritmo criptográfico, aplicación o técnica de circuito usada [4, 8, 9, 0, 12]. Dentro de este rango de técnicas existen varios enfoques, de hecho los estilos lógicos con consumo de energía independiente de los datos se pueden clasificar en dos categorías: *Dual Rail Precharge Logic* - DPL y *Current Mode Logic* - CML; en este trabajo nos centraremos en el enfoque de las familia lógica DPL ya que es un estilo que maximiza los niveles de seguridad frente a estilos lógicos convencionales.

2.4.1. Dual-Rail Precharge Logic - DPL

Durante los últimos años se han adoptado varios enfoques y contramedidas para proteger los dispositivos electrónicos criptográficos frente a ataques. Uno de estos enfoques adoptados es la llamada *Dual-Rail Precharge Logic* (DPL) o lógica de precarga de doble carril [1], [2], [4], [8], la cual procura ocultar informaciones del circuito criptográfico a los ojos de un posible atacante haciendo que el consumo energético de la lógica involucrada sea el mismo independientemente de las entradas de datos. Esto se logra gracias al uso de lógica diferencial, en DPL Cada valor lógico (“0”/”1”) se presenta mediante dos líneas, es decir, codifica cada valor como un par de señales complementarias.

La lógica DPL al ser un estilo lógico diferencial y dinámico implementa un protocolo de dos fases compuesto por una fase de precarga y una fase de evaluación. Durante la fase de precarga, todas las señales se colocan en un estado inicial o valor fijo para garantizar que durante la fase de evaluación, el número de cálculos sea completamente predecible y constante independientemente de las entradas [2, 4, 7]. En la fase de evaluación las salidas son dependientes de las entradas y de la función lógica que implemente el circuito. Durante la precarga ambas líneas pasan a valor BAJO y sólo una línea pasa a valor ALTO en la fase de evaluación. Esta técnica garantiza que, tras la ejecución, un circuito conmuta siempre el mismo número de nodos durante cada ciclo de reloj.

En la lógica DPL se pueden distinguir 3 tipos de bloques:

- *Bloque controlador de la fase de precarga*: Compuesto por transistores PMOS y NMOS, controlados por una señal de reloj CLK, son los responsables de que el valor de las salidas sea el mismo en la fase de precarga.
- *Bloque controlador de la fase de evaluación:* Como el anterior, consta de transistores NMOS y PMOS, y una señal de control o reloj CLK, su función es conectar en fase de evaluación las ramas diferenciales a la salida.
- *Bloque de función (combinacional):* Implementa la función lógica a través de transistores NMOS o PMOS, en la figura 1 y en la implementación que realizamos, este bloque está representado por el bloque DPDN.

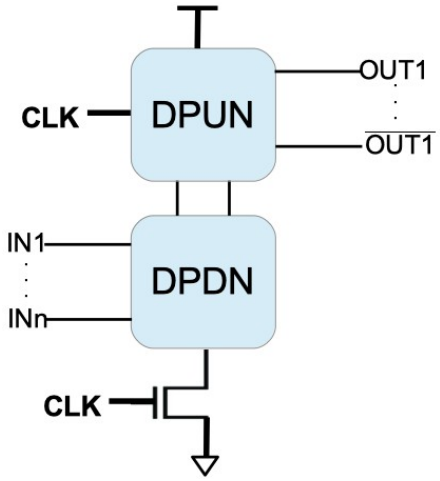


Ilustración 1: Ejemplo de estructura DPL

En el presente trabajo se usa la estructura DPL representada en la figura 1, donde las salidas OUT y su complementaria $\overline{OUT}$ se fijan a V_{DD} en la fase de precarga a través del bloque DPUN, el cual contiene inversores en sus salidas y en la fase de evaluación la función lógica es llevada a cabo por el bloque DPDN, que fija a GND el valor de las salidas durante la precarga.

DPL se considera una contramedida de hardware relevante para proteger la implementación contra ataques de canal lateral (SCA) como por ejemplo el análisis diferencial de potencia (DPA), al hacer que el consumo de energía sea independiente de los datos [2], [4], [7], [10].

Existen varias implementaciones de celdas seguras utilizando la lógica DPL, en este apartado

vamos a abordar tres estilos lógicos que hacen uso del enfoque DPL de la Figura 1:*Sense Amplifier Based Logic*- SABL, *Charge Recycling SABL* – CRSABL, *Dynamic Current Mode Logic* – DyCML. Por otro lado, se analizará también la familia *Wave Dynamic Differential Logic*– WDDL, que siendo un esquema dinámico y diferencial, no opera con la estructura de la Figura 1, sino que basa su esquema de funcionamiento en puertas CMOS lógicas.

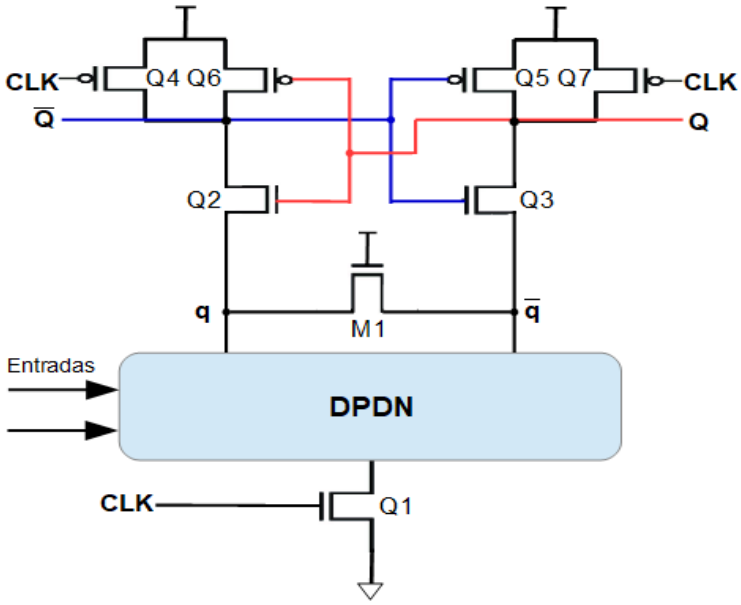
2.4.2. Sense Amplifier Based Logic - SABL

Este estilo lógico usa celdas especiales combinando las lógicas diferencial y dinámica, permitiendo con ello la implementación de puertas lógicas con una cantidad fija de consumo de energía en cada ciclo (un único evento de conmutación), independientemente del valor presente en sus entradas [2]. Es decir, la lógica SABL en cada conmutación del ciclo de reloj descarga y carga la suma de todas las capacitancias internas del nodo a un valor constante (modo dinámico) y se descargan (modo diferencial). En esta lógica las capacidades intrínsecas en las señales diferenciales de entrada y salida son simétricas. Esto hace que el consumo total de energía sea constante en cada ciclo de reloj. Por lo tanto, es independiente de las transiciones de entrada.

Esta lógica funciona en dos fases: precarga y evaluación. En la figura 2 en la fase de precarga (modo dinámico), los nodos de salida n1 y n2 (o q y $\bar{q}$) se cargan durante el nivel bajo del reloj, mientras que la fase de evaluación (modo diferencial), se activa durante el nivel alto del reloj. En fase de evaluación, según las transiciones de entrada la carga se almacena en la capacitancia del nodo de salida y la corriente fluye hacia abajo en el camino de la red de *pull down*. Todos los nodos internos son conectados a los nodos de salida para que la potencia consumida sea la misma en todos los ciclos del reloj.

Esta arquitectura introduce entre los nodos n1 y n2 un transistor NMOS (M1) que siempre está conduciendo entre los nodos n1 y n2 y que conectan la red DPDN con la DPUN, haciendo que n1 y n2 vayan al cero lógico “0” tras la fase de evaluación, y queden flotantes en la fase de precarga. De esta manera, se busca eliminar la información de la conmutación sucedida en la fase de evaluación, dicha información pudiera ser útil para un atacante.

La implementación de este estilo lógico exige que para garantizar un consumo de energía constante, las capacidades de carga de las señales complementarias deben ser las mismas, razón por la cual el circuito debe estar adecuadamente balanceado.

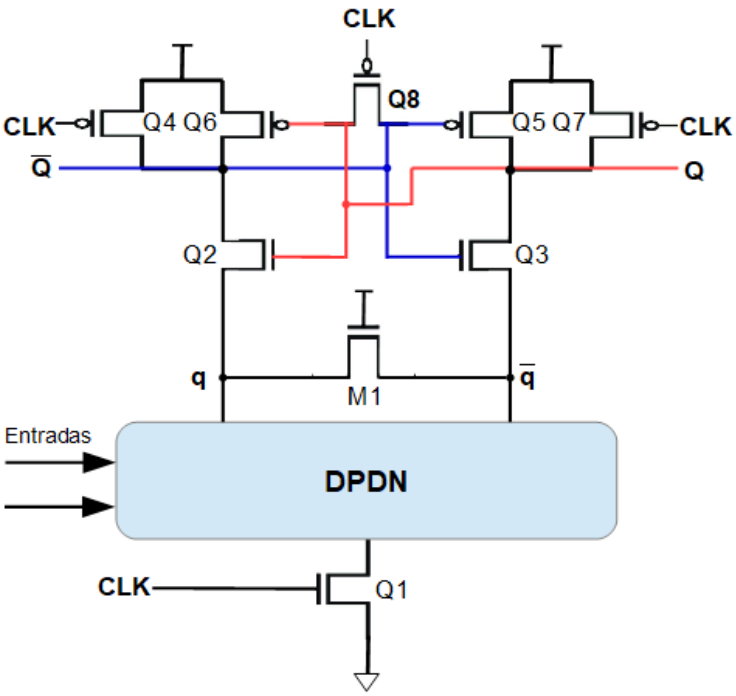


2. Ilustración: Arquitectura genérica de una compuerta SABL

2.4.3. Charge Recycling SABL - CRSABL

Esta lógica se deriva del estilo lógico SABL. Su diferencia comparada con la arquitectura SABL radica en que en la lógica CRSABL los dos transistores PMOS sincronizados con CLK que precargan los nodos de salida del inversor SABL al voltaje VDD, son reemplazados por el transistor PMOS Q8 sincronizado con CLK entre los nodos de salida del inversor CRSABL [3].

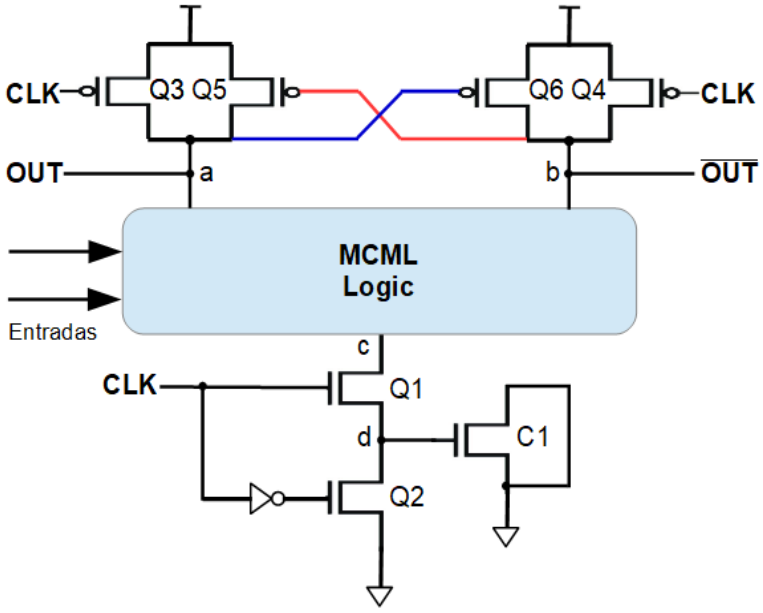
Dado que una salida y todos los nodos internos se descargan en cada ciclo de reloj, cuanto más bajos sean los voltajes de precarga, menor será el consumo de energía. La corriente de suministro de pico es menor porque se requiere menos carga y también porque la corriente de precarga ya no es suministrada por transistores sincronizados con CLK, que están completamente abiertos y proporcionan una corriente pico alta. El esquema general de este estilo lógico lo podemos apreciar en la figura 3.



3. Ilustración: Arquitectura genérica de una compuerta CRSABL

2.4.4. Dynamic Current Mode Logic - DyCML

El estilo lógico DyCML es una familia lógica de modo de corriente dinámica, diseñada para reducir el consumo de energía estática característico de los estilos lógicos CML [8]. Su arquitectura básica se muestra en la figura 4.



4. Ilustración: Arquitectura de una compuerta DyCML

El estilo DyCML está compuesto por 4 bloques:

- *Bloque de funcionalidad:* Encargado de evaluar la función lógica implementada.
- *Bloque de precarga:* Compuesto por los transistores Q3 y Q4.
- *Fuente de corriente dinámica*: Compuesto por Q1, Q2 y C1, este bloque opera como una tierra virtual y como una resistencia con elemento activo (C1) para reducir el consumo de energía estática.
- *Latch:* Compuesto por los transistores Q5 y Q6, utilizado para preservar el valor lógico y acelerar la evaluación.

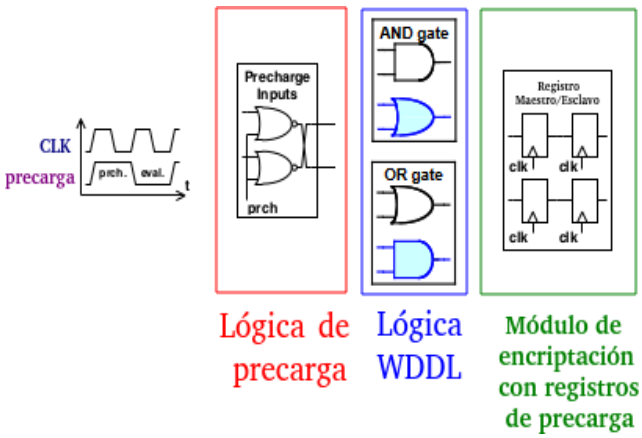
En DyCML durante el valor bajo del reloj, el transistor Q2 es activado para descargar el condensador C1 a GND. A su vez, el transistor Q1 es desactivado, eliminando con ello el flujo de corriente DC en el camino de VDD a GND. Durante la fase alta del reloj Q3 y Q4 son desactivados, Q1 es activado y se crea un camino de corriente desde las 2 nodos de salida precargadas al condensador C1, el cual actúa como una tierra virtual [1].

2.4.5. Wave Dynamic Differential Logic - WDDL

La WDDL o Lógica diferencial dinámica de onda es un estilo lógico DPL basado en bibliotecas de células estándar CMOS [4]. Utiliza una combinación de puertas lógicas complementarias para equilibrar la actividad en el circuito, de modo que la fuga de consumo de energía puede ser minimizada de esta manera. Su idea básica es reducir la dependencia del consumo de energía y el valor intermedio, razón por la cual se implementa utilizando solo compuertas AND y OR CMOS estático en paralelo para construir la lógica diferencial, fruto de la aplicación de la ley de Morgan, en la cual la puerta NAND es equivalente a las entradas invertidas en la puerta OR y la puerta NOR es equivalente a las entradas invertidas en la puerta AND. Esta técnica en lugar de precargar la celda con una señal de reloj como las otras lógicas presentadas, usa la precarga para propagar una onda de ceros a través de las entradas de todas las puertas estáticas. Dado que este estilo lógico usa solo puertas positivas (AND-OR), la onda de precarga fuerza las puertas a un estado en precarga de “0” lógico ya que todas las entradas son 0. Durante la fase de precarga, las entradas al WDDL se cargan a 0 a través de un circuito de precarga o por la “onda” propagada. Esto fuerza a ambas salidas (Y e $\overline{Y}$) a 0, propagando así la onda. Durante la fase de evaluación, las entradas deben ser complementarias empujando a la puerta lógica al estado de evaluación, es decir, el valor de precarga se propaga de las entradas a las salidas, como si fuese una onda y la onda de precarga viaja a la siguiente puerta de la

lógica combinatoria. En lugar de una señal de precarga que restablece la lógica, hay una onda de precarga, donde las puertas se precargan sin distribuir la señal a cada puerta individual.

Existen varias maneras de implementar la onda de precarga, una de ellas es la que muestra la figura 5**,** donde se precargan solo las entradas y se utilizan registros SDDL Maestro-Esclavo, pues estos registros almacenan los ceros precargados durante la fase de evaluación. La otra vía consiste en insertar un operador de precarga al comienzo de cada árbol lógico combinatorio, es decir, las entradas del módulo de cifrado y las salidas de los registros SDDL, este será el método implementado en este trabajo.



5. Ilustración: Arquitectura de una celda WDDL con registro SDDL y lógica de precarga

La WDDL es una lógica fácil de implementar en las bibliotecas de celdas estándar, desarrolla lógica dinámica y diferencial con sólo una pequeña capacidad de carga en la señal de control de precarga y maneja alta márgenes de ruido de CMOS estático, presenta además baja corriente de alimentación.

3. Diseño de Celdas Lógicas e Implementación

En este trabajo el diseño de las celdas lógicas se realizó con las celdas estándar de la tecnología UCM 180 nm, la cual utiliza un VDD de 1.8 V. Se han escogido dimensiones mínimas para los transistores.

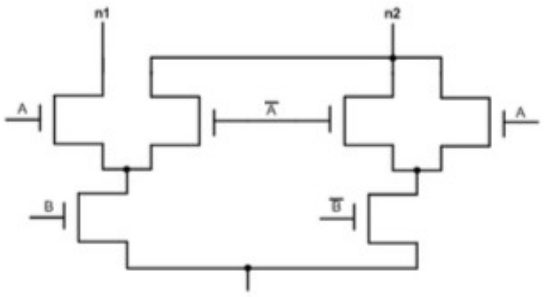
3.1. Diseño del bloque DPDN para las técnicas SABL, CRSABL y DyCML

Como se expresó anteriormente las lógicas SABL, CRSABL y DyCML son lógicas de precarga de doble carril, razón por la cual su estructura consiste de dos bloques: El bloque DPUN (superior) que implementa la lógica de la técnica escogida (SABL, CRSABL o DyCML), y el bloque inferior o DPDN que implementa la función o puerta lógica deseada (AND/NAND, OR/NOR, XOR/XNOR). El esquema general de estos bloques se presenta a continuación.

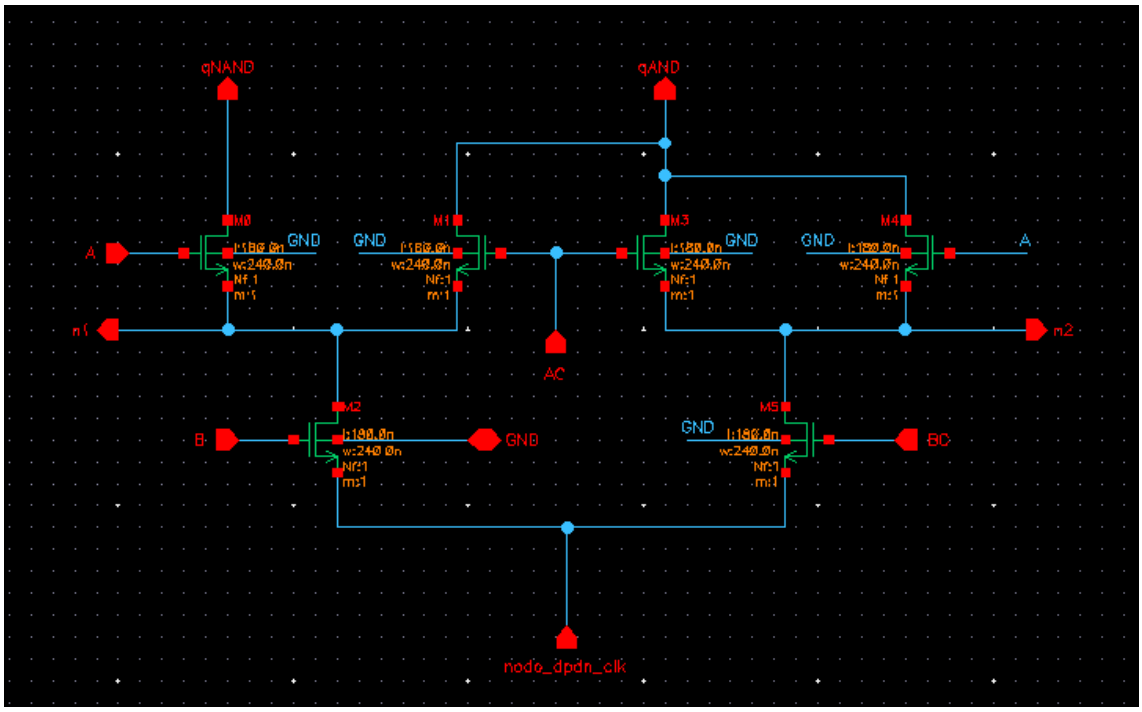
A continuación se presenta el diseño de los bloques DPDN que comparten las tres lógicas anteriormente citadas, teniendo en cuenta que para su diseño se consideró la generación de la salida complementaria correspondiente a lógica de doble rail, como también se trató de tener la máxima simetría e igual número de transistores entre ambas ramas de dicho circuito [7].

3.1.1. Bloque DPND AND/NAND

ENTRADAS		SALIDAS	
A	**B**	**A AND B**	**A NAND B**
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0



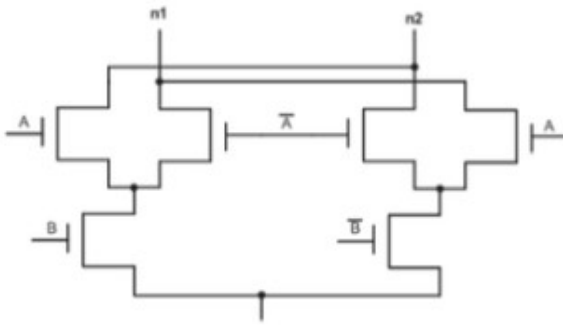
6. Ilustración: Esquema de la celda AND/NAND y su tabla de verdad



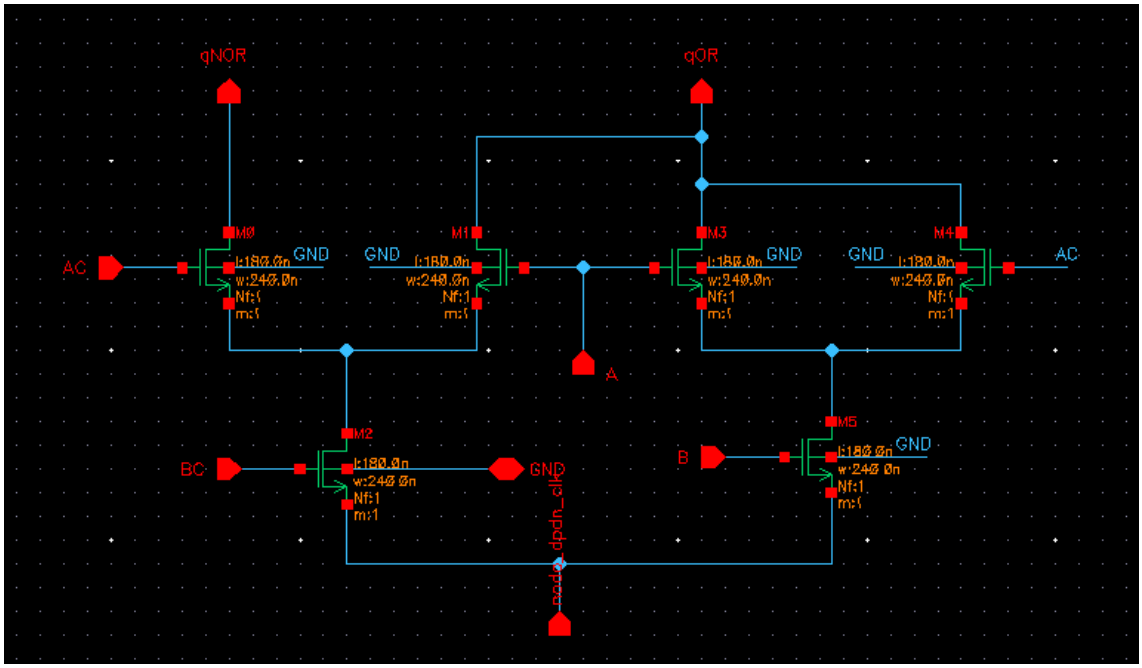
7. Ilustración: Esquemático del DPDN de la celda AND/NAND en Virtuoso (Cadence)

3.1.2. Bloque DPDN OR/NOR

ENTRADAS		SALIDAS	
A	B	A OR B	A NOR B
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0



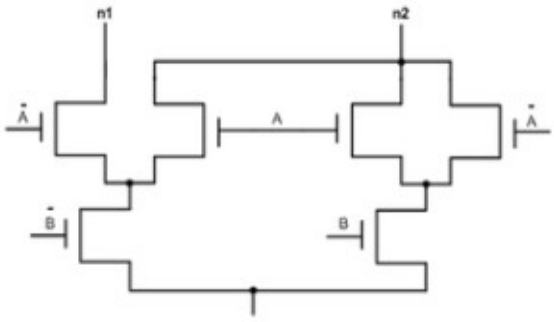
8. Ilustración: Esquema de la celda OR/NOR y su tabla de verdad



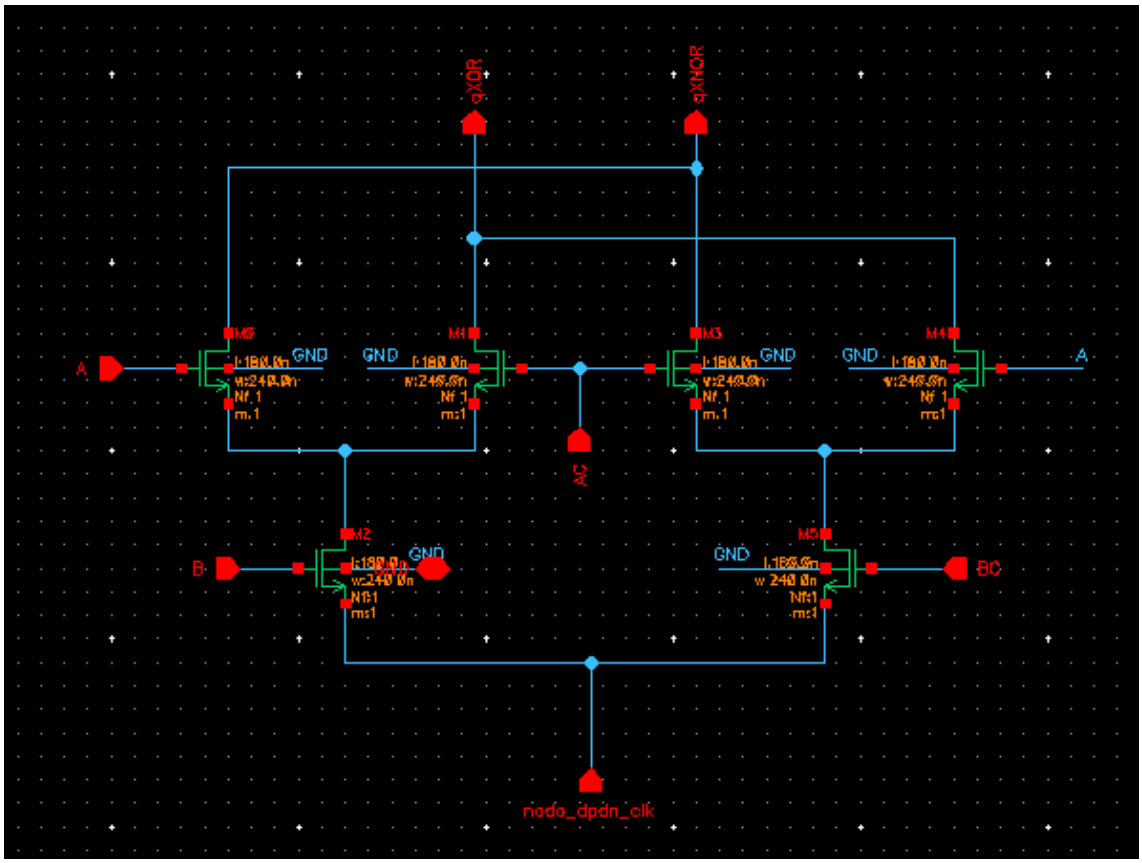
9. Ilustración: Esquemático del DPDN de la celda OR/NOR en Virtuoso (Cadence)

3.1.3. Bloque DPDN XOR/XNOR

ENTRADAS		SALIDAS	
A	**B**	**A XOR B**	**A XNOR B**
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1



10. Ilustración: Esquema de la celda XOR/XNOR y su tabla de verdad



11. Ilustración: Esquemático del DPDN de la celda XOR/XNOR en Virtuoso (Cadence)

3.2. Diseño de la SBOX Piccolo

Las substitution box - S-Box son componentes integrantes de algunos block cipher - cifradores de bloque, que son un componente básico y fundamental en los algoritmos criptográficos de clave simétrica. Operan en grupos de bits de longitud fija, llamados bloques, a los cuales les aplican varias transformaciones, entre las cuales se encuentra la sustitución realizada por la S-Box. En general, una S-Box toma un número *m* de bits de entrada y los transforma en *n* bits de salida. En el proceso de cifrado, el cifrador de bloque toma un bloque de texto plano como entrada y produce un bloque de igual tamaño de texto cifrado, valiéndose de una clave secreta. En el descifrado el proceso es muy parecido: ingresan bloques de texto cifrado y se producen bloques de texto plano.

El objetivo perseguido con el uso de cifradores de bloques es extraviar la relación existente entre texto plano y texto cifrado, razón por la cual se hace necesario que estos dispositivos sean robustos y resistentes a ataques. Los bloques S-Box son los más vulnerables a ataques dentro de los cifradores de bloque [17].

La tendencia actual frente a la necesidad de portabilidad de los dispositivos es hacer uso de *Lightweight block ciphers* que usualmente son optimizados en coste y consumo para implementaciones de hardware más compactas, es este tipo de cifradores los bloques de la S-Box se calculan implementando circuitos que utilizan las puertas lógicas básicas (AND, XOR, OR, NOR, etc.). Ejemplos de estos son el Piccolo [5] o PRIDE [6].

Piccolo es un cifrador ultraligero que utiliza un tamaño de bloque de 64 bits y tiene una clave de 80 o 128 bits. La versión con clave de 80 bits usa 683 puertas lógicas equivalentes - GEs, y la versión de 128 bits usa 758 GEs. Este cifrador requiere otras 60 GEs para realizar el proceso de descifrado. En términos generales este cifrador se adapta bien a las aplicaciones de sensores y RFID [5].

Piccolo usa S-boxes de 4 bits de entrada (x3, x2, x1, x0) y 4 bits de salida (y3, y2, y1, y0), correspondiendo en la notación el 3 con el bit más significativo y el 0 con el bit menos significativo. El algoritmo que usa el Piccolo reemplazaría el texto plano de entrada por un texto cifrado conforme a las siguientes son las ecuaciones lógicas:

y3 = x0 XOR NOT((x3 OR x2))

y2 = x3 XOR NOT((x2 OR x1))

y1 = NOT(x2 XOR NOT((y3 OR x1)))

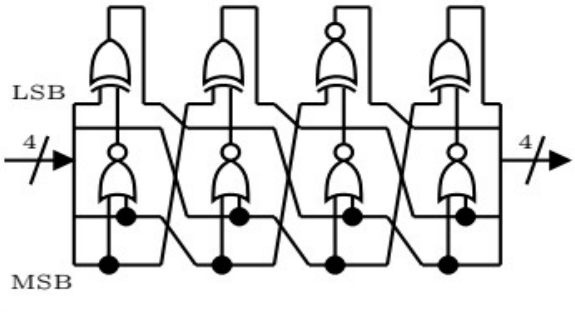
y0 = x1 XOR NOT((y2 OR y3))

La función booleana o tabla de verdad para las ecuaciones lógicas que implementa Piccolo, es la siguiente (valores representados en notación hexadecimal):

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S[x]$	e	4	b	2	3	8	0	9	1	a	7	f	6	c	5	d

Tabla. Tabla de verdad implementada en Piccolo

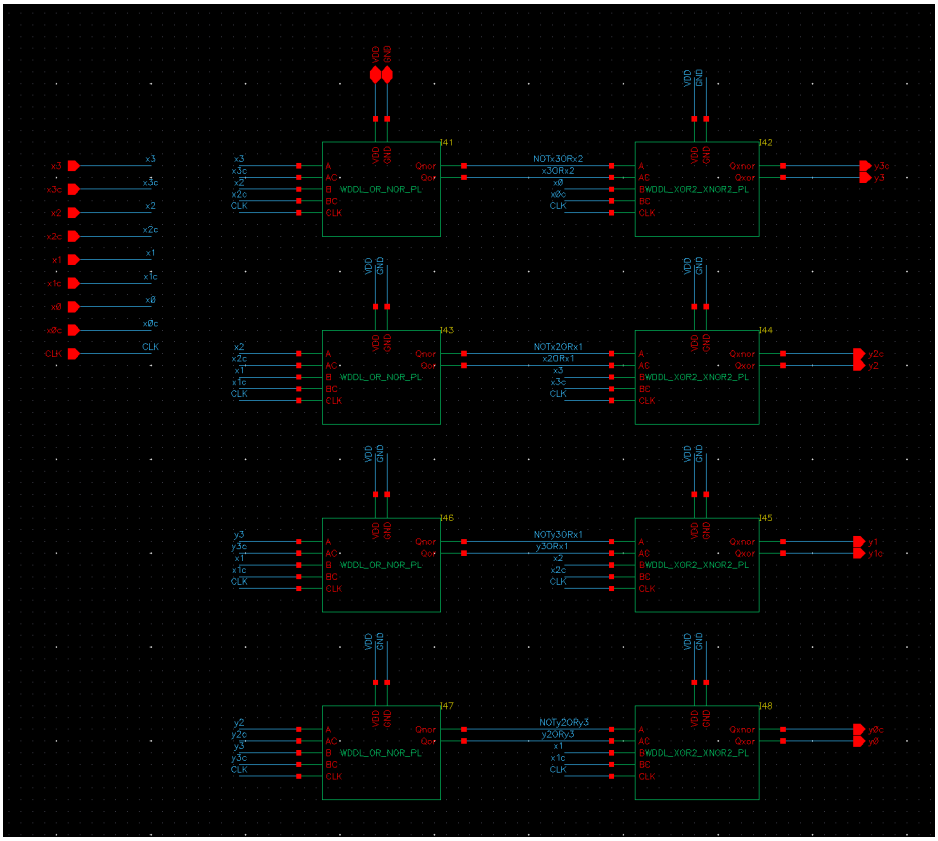
Esta función booleana puede ser fácilmente implementada por hardware mediante el empleo de 4 compuertas XOR/XNOR y 4 compuertas OR/NOR conforme el esquema siguiente:



12. Ilustración: Función lógica implementada en Piccolo

En este trabajo a fin de comparar las propuestas lógicas implementadas (SABL, CRSABL, DyCML y WDDL) se diseñó un demostrador simple Sbox4-Piccolo para cada una de ellas, a fin de obtener de las Sboxes la información del consumo de potencia.

El bloque generado con el framework de Cadence puede ser apreciado en la figura 13, donde tenemos una visión general de sus señales de entrada y salida, igual que las señales complementarias como corresponde a la lógica DPL.



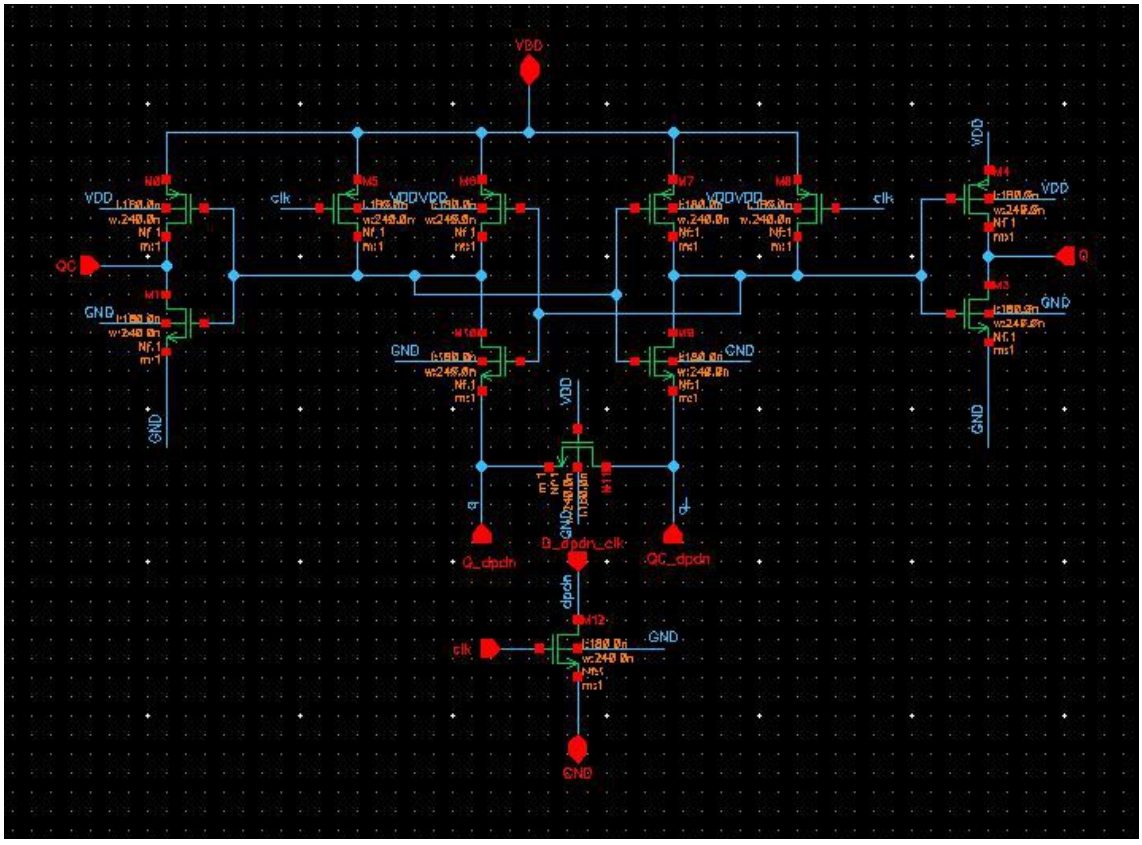
13. Ilustración: Esquemático de la arquitectura de la SBOX implementada en Virtuoso

Descendiendo un poco más de la jerarquía general, podemos apreciar en la figura 13 que la Sbox está compuesta internamente por 8 compuertas, 4 OR/NOR y 4 XOR/XNOR de acuerdo a la función lógica que implementa Piccolo.

3.3. Diseño de Celdas Lógicas con la técnica SABL

3.3.1. DPUN Lógica SABL

El DPUN de la lógica SABL fue implementado conforme la arquitectura de dicha lógica, el transistor en la parte inferior es controlado por el reloj y contiene el nodo (dpdn) que irá conectado a la lógica DPDN encargada de definir la función a implementar (and, or o xor). Al ser la SABL una lógica de doble *rail*, la salida (Q) del circuito posee también la salida complementaria (QC), como se puede apreciar en la figura 14. Donde CLK = 1 corresponde a la fase de evaluación y CLK = 0 corresponde a la fase de precarga.



14. Ilustración: Esquemático del DPUN SABL en Virtuoso (Cadence)

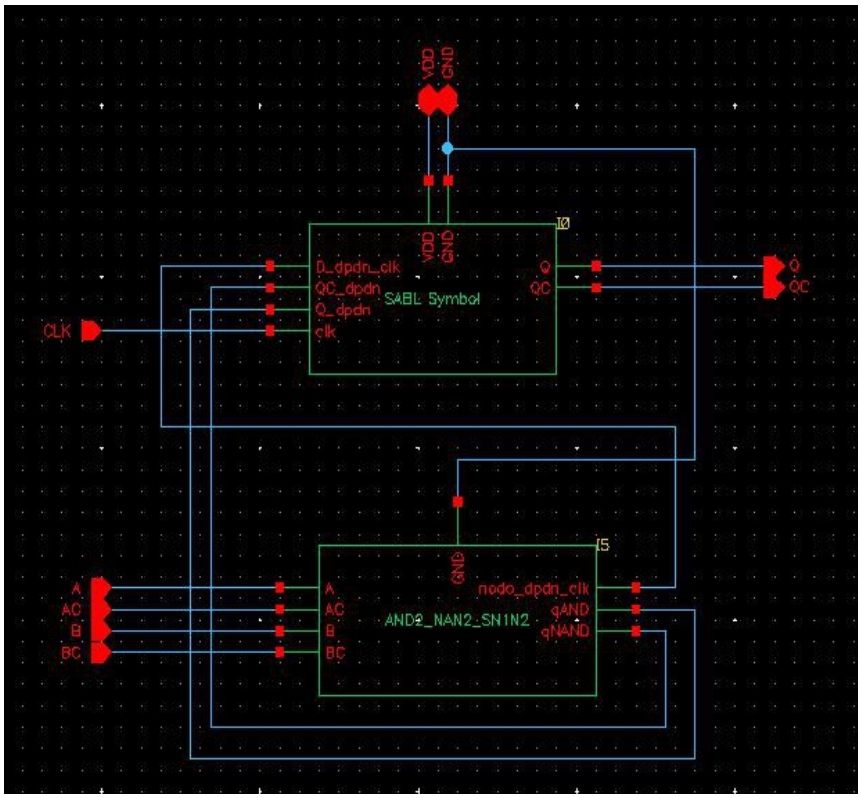
3.3.2. Diseño de la Celda DPDN SABL AND/NAND

Las celdas DPUN fueron diseñadas en tecnología UMC 180 con V_{DD} – 1.8V, conforme lo requería la arquitectura de cada estilo lógico. Cada bloque o celda desarrollado tiene los dos pines de alimentación dedicados (V_{DD} y GND). Las dimensiones de los transistores empleados es la mínima que permite la tecnología (W = 240n y L = 180n). Las celdas DPDN AND/NAND, OR/NOR y XOR/XNOR fueron implementadas siguiendo los esquemas presentados en las figuras 6 – 11 a fin de buscar la máxima simetría posible.

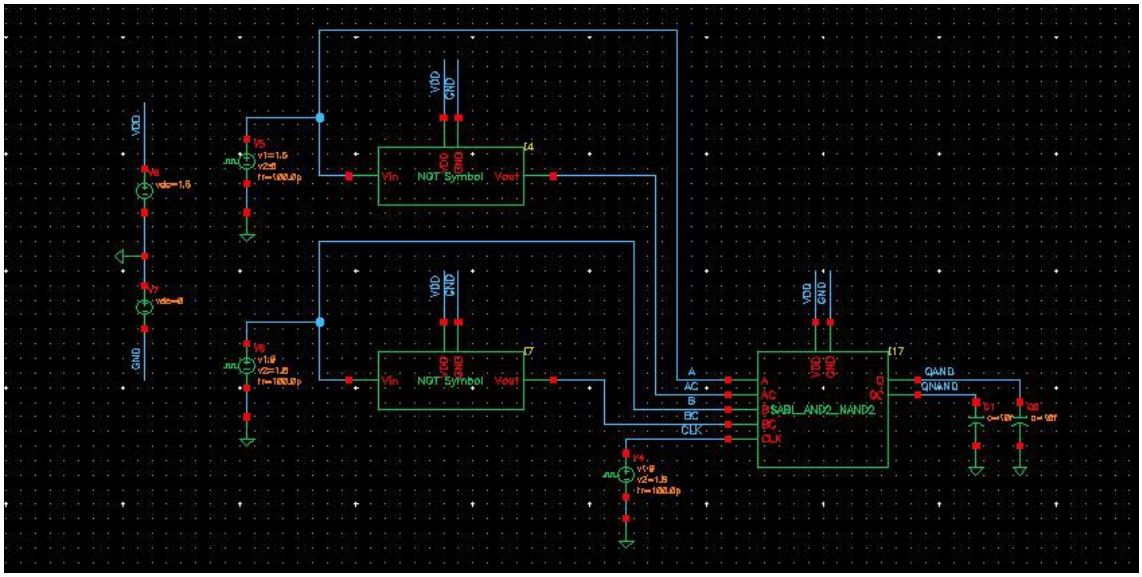
Por practicidad se escogió trabajar los diseños a nivel de jerarquias en la herramienta de captura de esquemáticos Virtuoso de Cadence, creando bloques y símbolos de las celdas diseñadas y no solo vistas a nivel de transistor; de manera que las celdas lógicas básicas por ejemplo, están compuestas de 2 bloques, un bloque que contiene el DPUN en el estilo lógico correspondiente y un bloque DPDN que contiene las compuertas básicas ya que comparten todos los estilos lógicos a excepción del WDDL.

Para comprobar la funcionalidad de cada celda lógica diseñada se creó un esquemático para el test y se realizó inicialmente una simulación transciente usando el Analog Design Environment – ADEL y el simulador Spectre de Cadence, con estímulos dirigidos en las dos entradas y sus complementarias (lógica *doble rail*), asi como también se generó una señal cuadrada de reloj con frecuencia de 50 Mhz, la duración de las simulaciones fue igual a 100 ns. En dichas simulaciones se pudo constatar por inspección visual el funcionamiento de la precarga y evaluación, las formas de onda, la tabla lógica y correcto funcionamiento de cada una de las celdas desarrolladas. Los estímulos, condiciones de simulación y la duración del test fueron iguales para cada compuerta y estilo lógico.

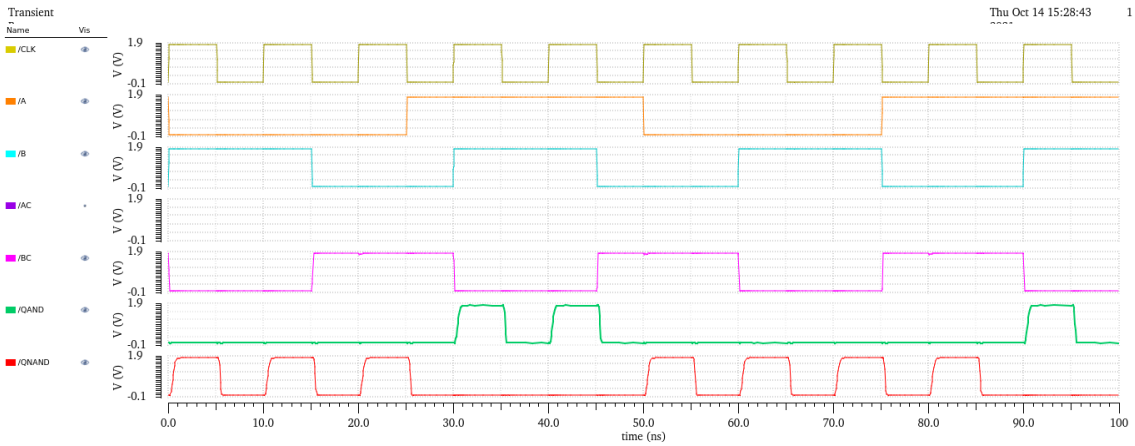
En las figuras siguientes se muestran las capturas de los diseños para cada compuerta y estilo lógico, los test y ondas de salida, así como también la implementación y test de cada una de las Sboxes.



15. Ilustración: Esquemático de la celda AND/NAND en Virtuoso (Cadence)

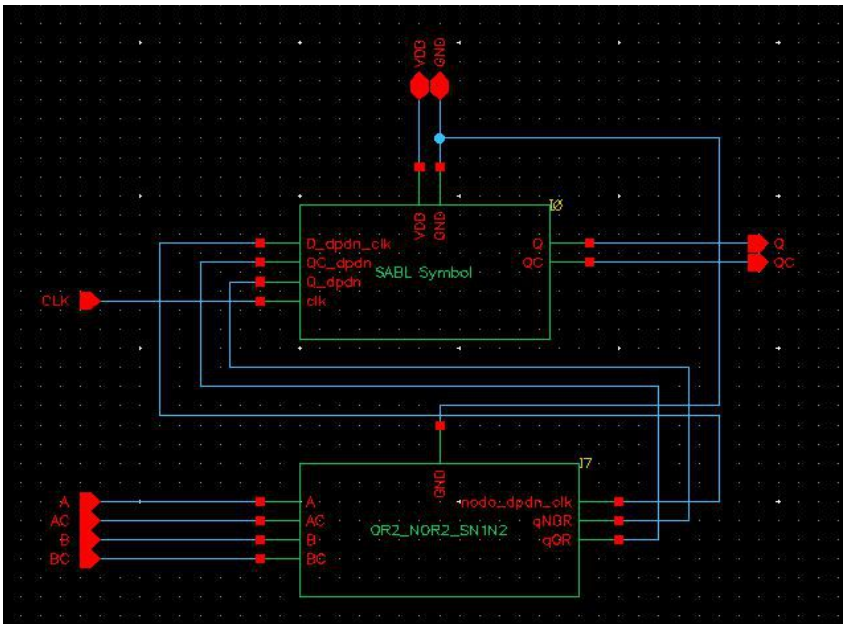


16. Ilustración: Esquemático del test para la celda AND/NAND en Virtuoso (Cadence)

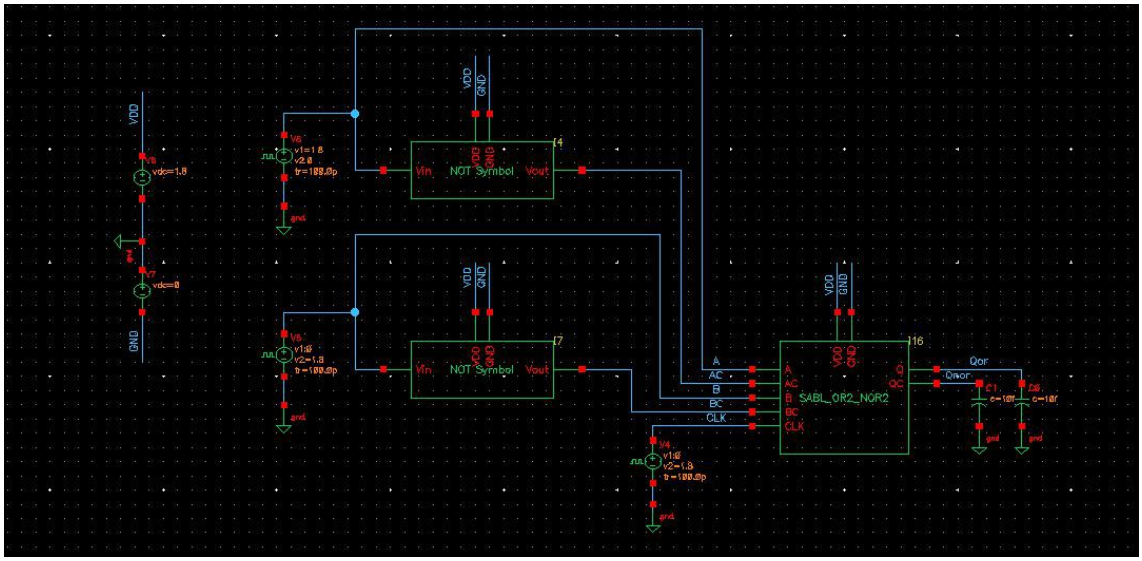


17. Ilustración: Resultado de la simulación de la compuerta AND/NAND en Spectre

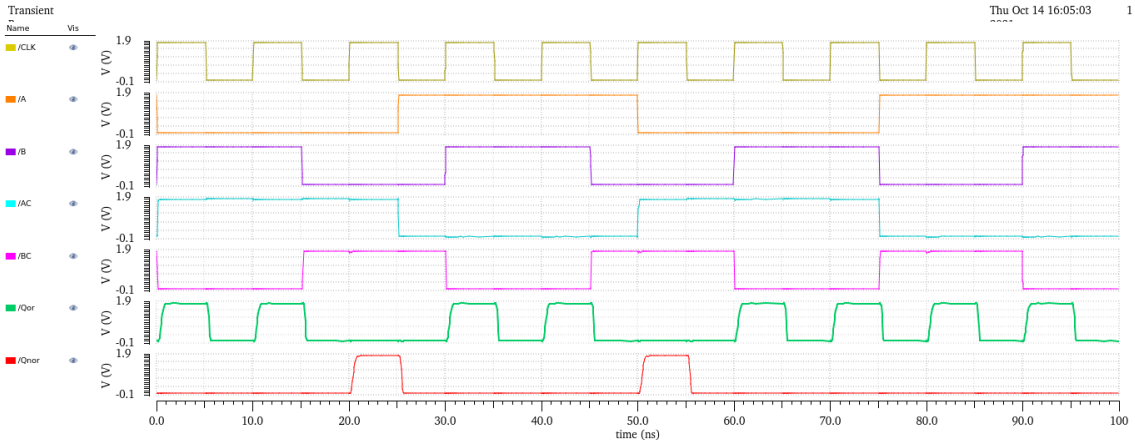
3.3.3. Diseño de la Celda DPDN SABL OR/NOR



18. Ilustración: Esquemático de la celda OR/NOR en Virtuoso (Cadence)

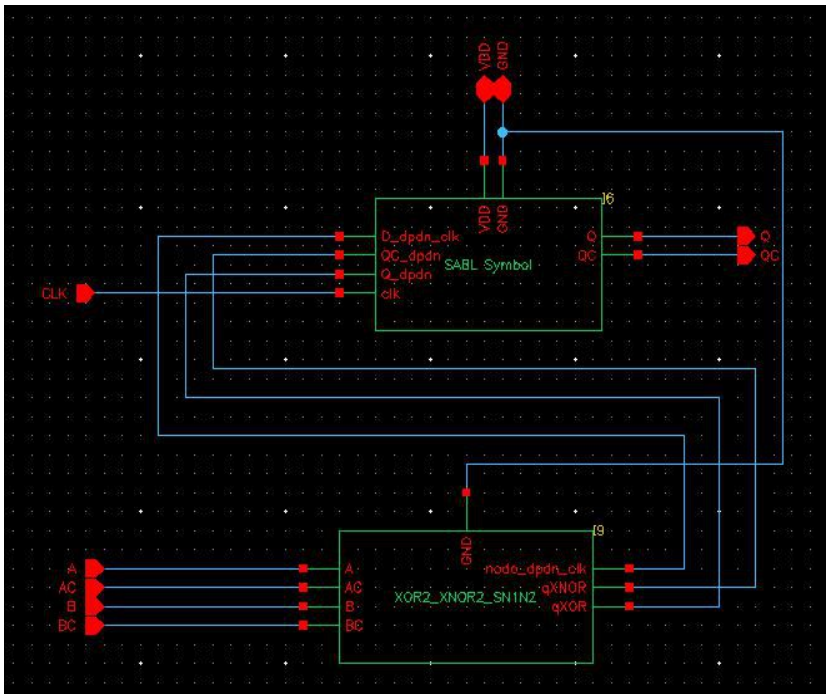


19. Ilustración: Esquemático del test para la celda OR/NOR en Virtuoso (Cadence)

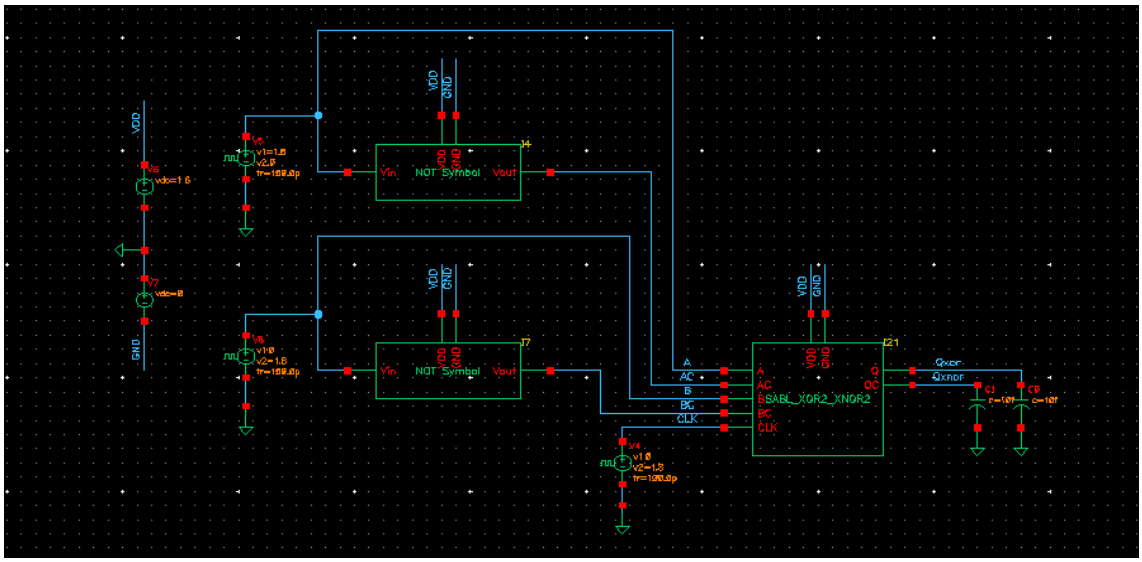


20. Ilustración: Resultado de la simulación de la compuerta OR/NOR en Spectre (Cadence)

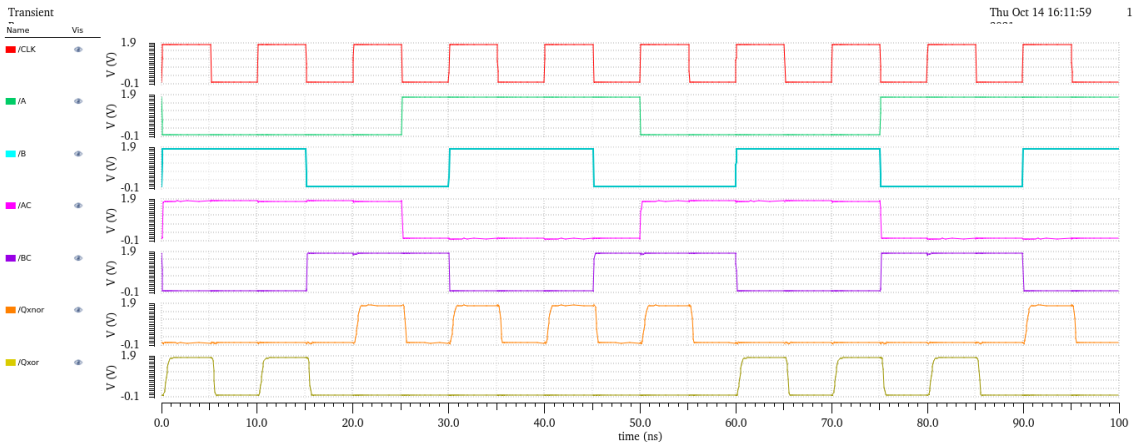
3.3.4. Diseño de la Celda DPDN SABL XOR/XNOR



21. Ilustración: Esquemático de la celda XOR/XNOR en Virtuoso (Cadence)

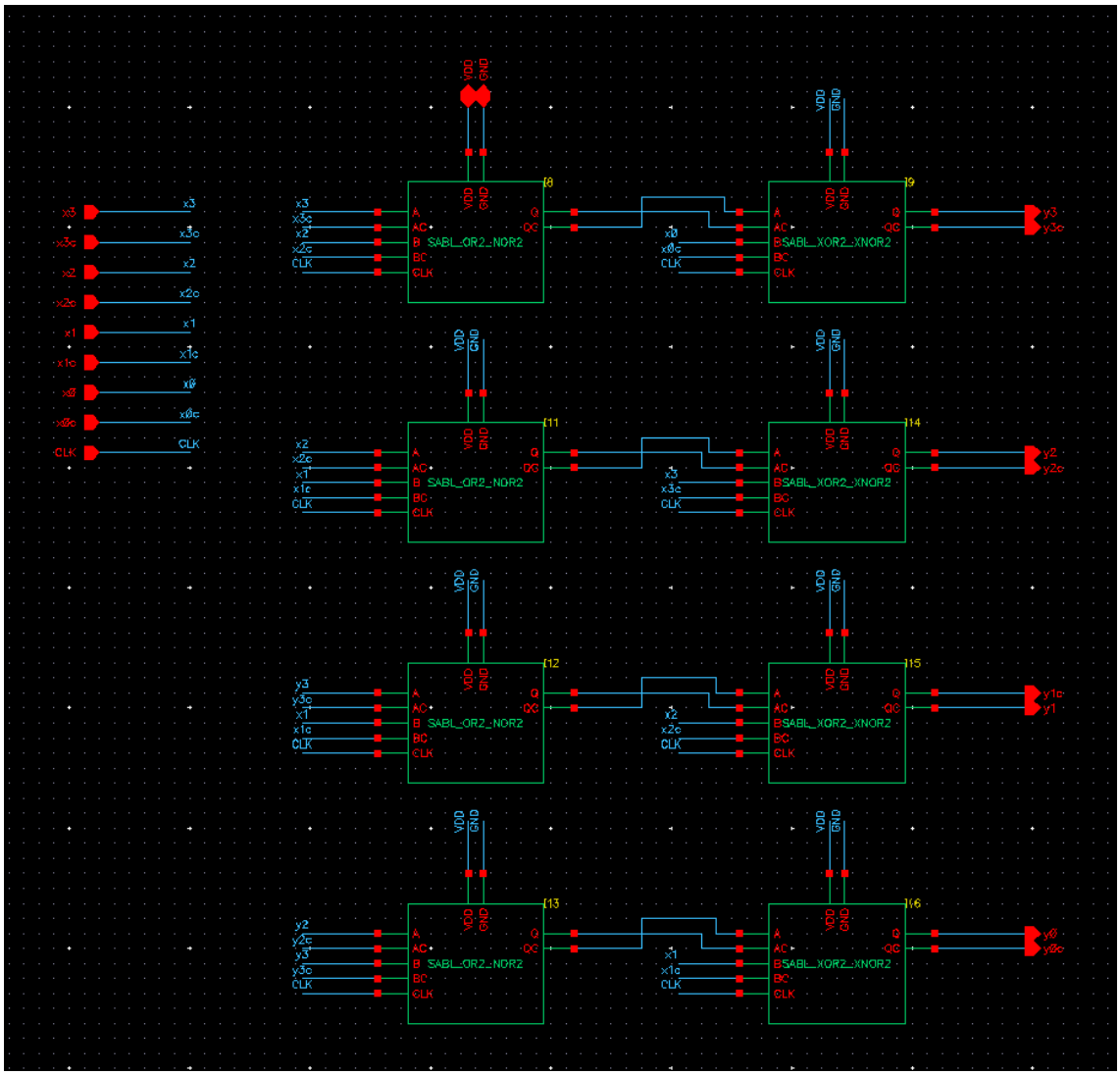


22. Ilustración: Esquemático del test para la celda XOR/XNOR en Virtuoso (Cadence)



23. Ilustración: Resultado de la simulación de la compuerta XOR/XNOR con Spectre

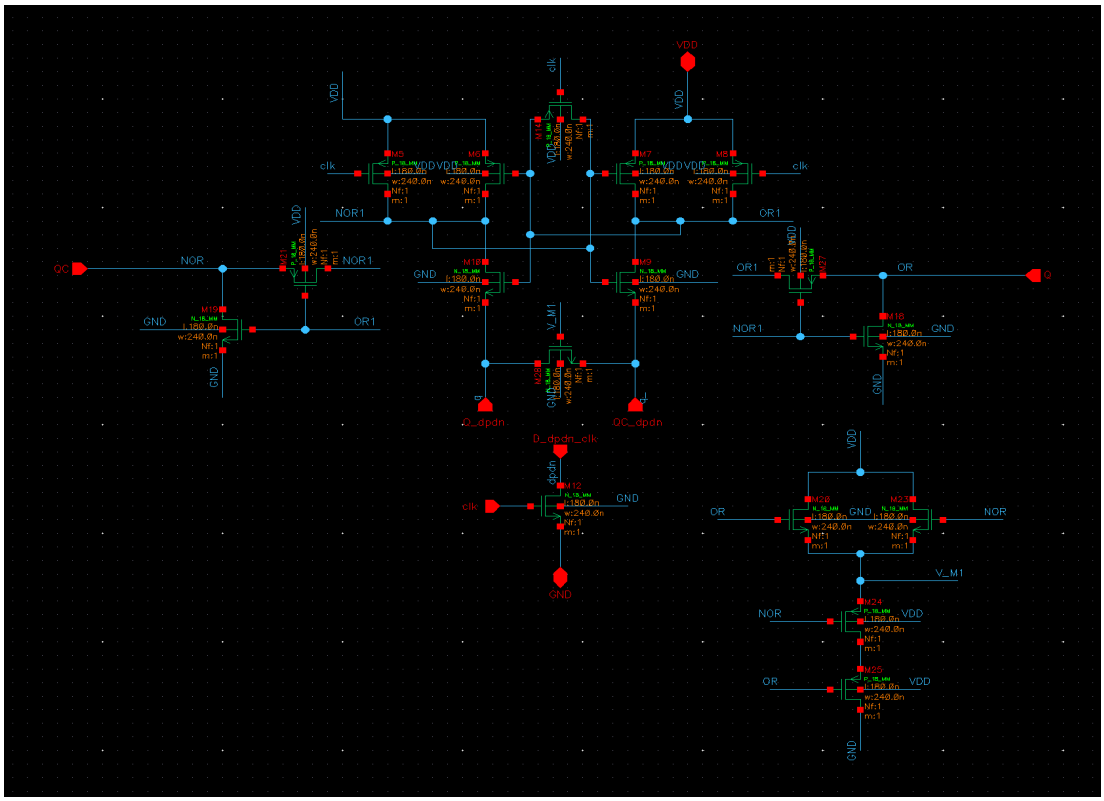
3.3.5. Diseño del demostrador Piccolo SABL



24. Ilustración: Esquemático del demostrador Piccolo en Estilo SABL

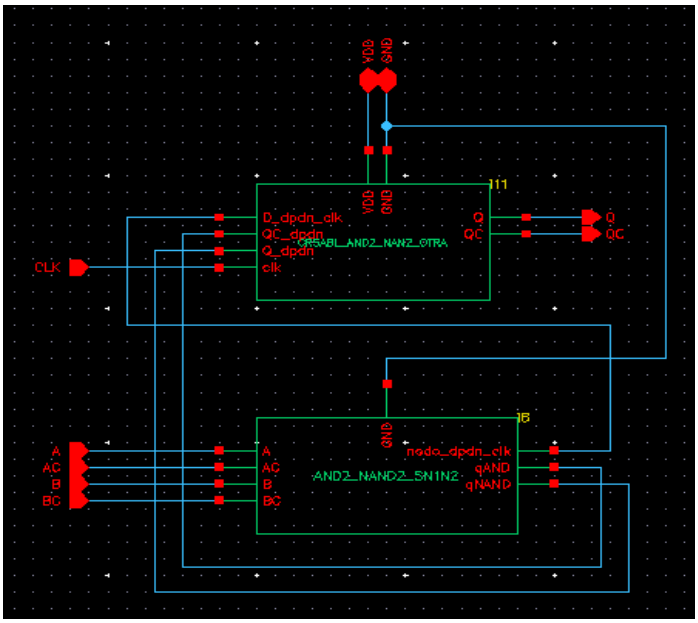
3.4. Diseño de Celdas Lógicas con la técnica CRSABL

3.4.1. DPUN de la Celda CRSABL AND/NAND

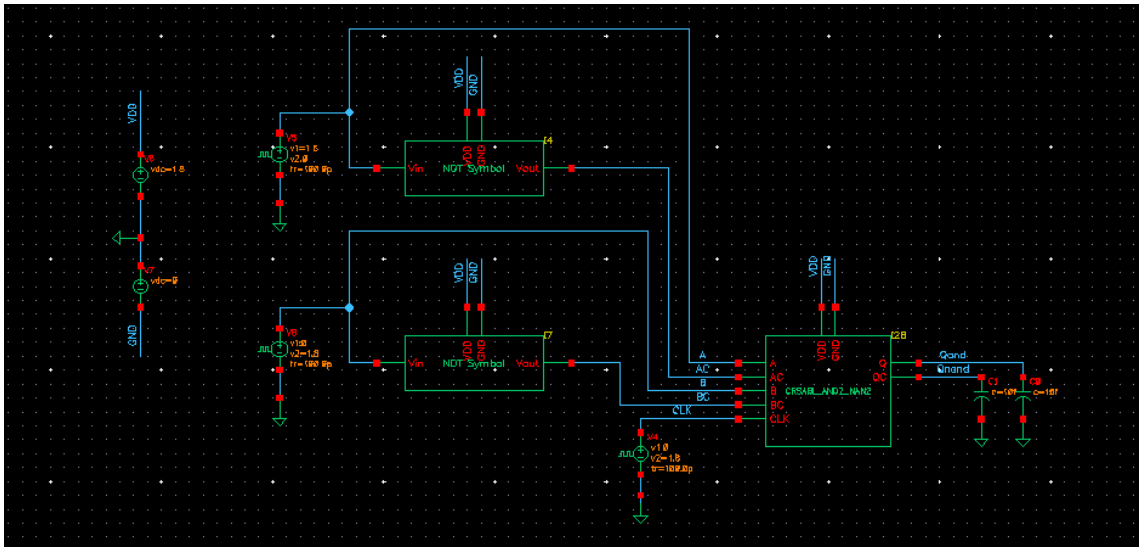


25. Ilustración: DPUN de la Celda CRSABL AND/NAND

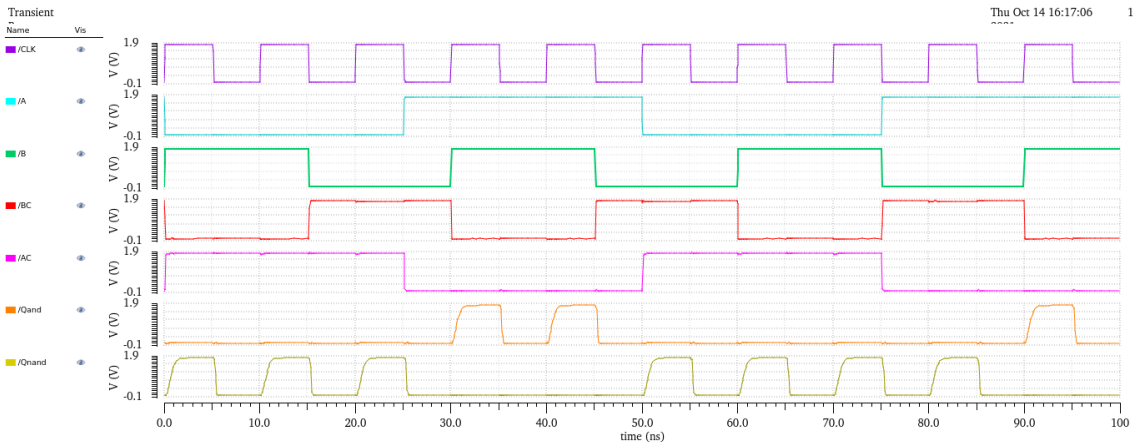
3.4.2. Diseño de la Celda DPDN CRSABL AND/NAND



26. Ilustración: Esquemático del la celda CRSABL AND/NAND en Virtuoso (Cadence)

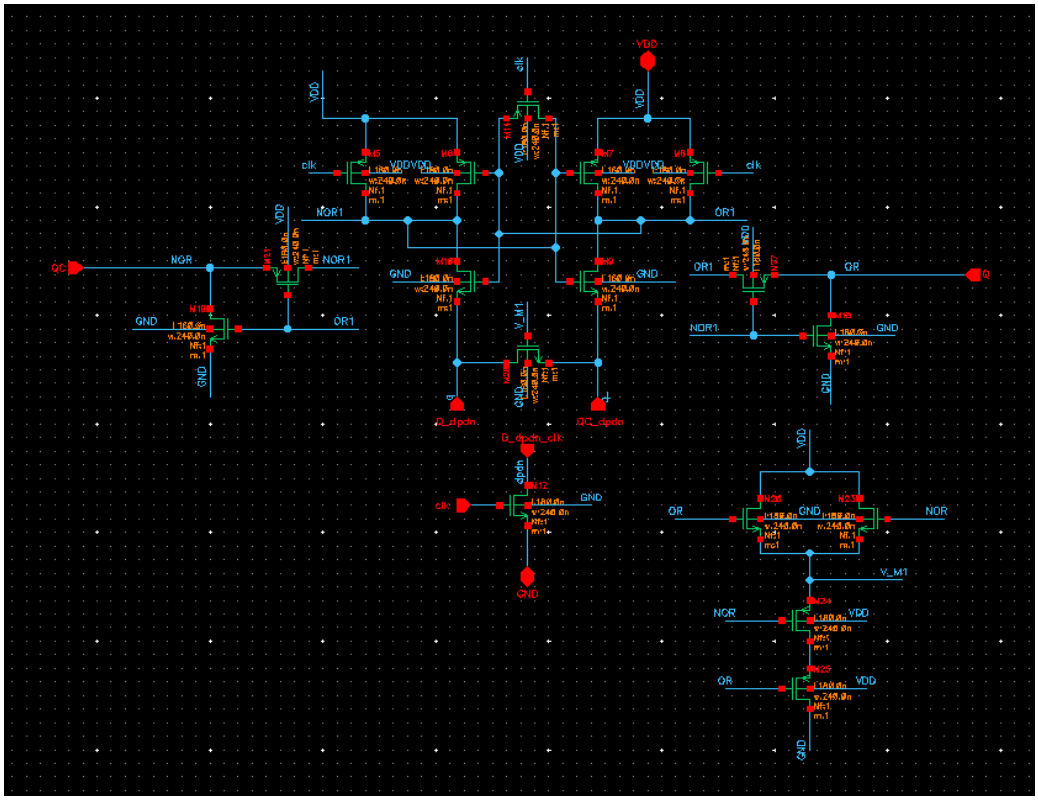


27. Ilustración: Esquemático del test para la celda CRSABL AND/NAND en Virtuoso (Cadence)



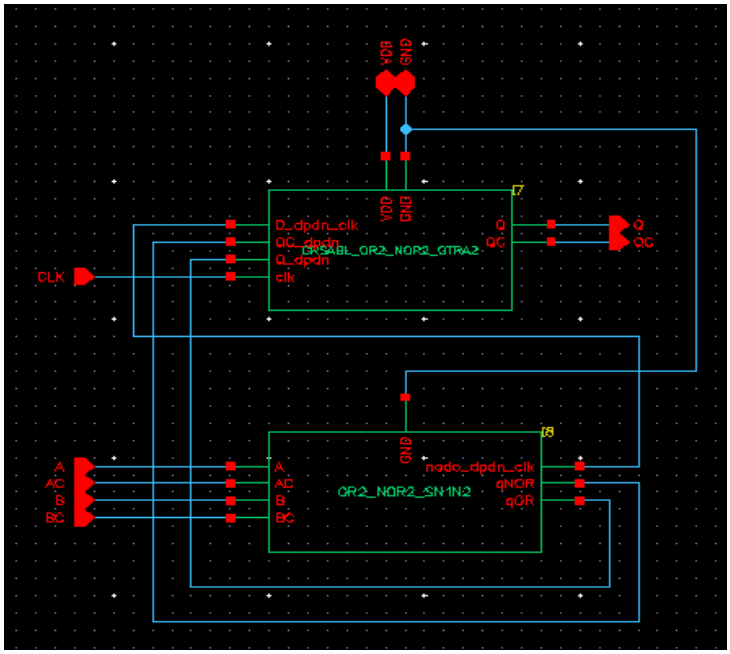
28. Ilustración: Resultado del test para la celda CRSABL AND/NAND en Spectre (Cadence)

3.4.3. DPUN de la Celda CRSABL OR/NOR

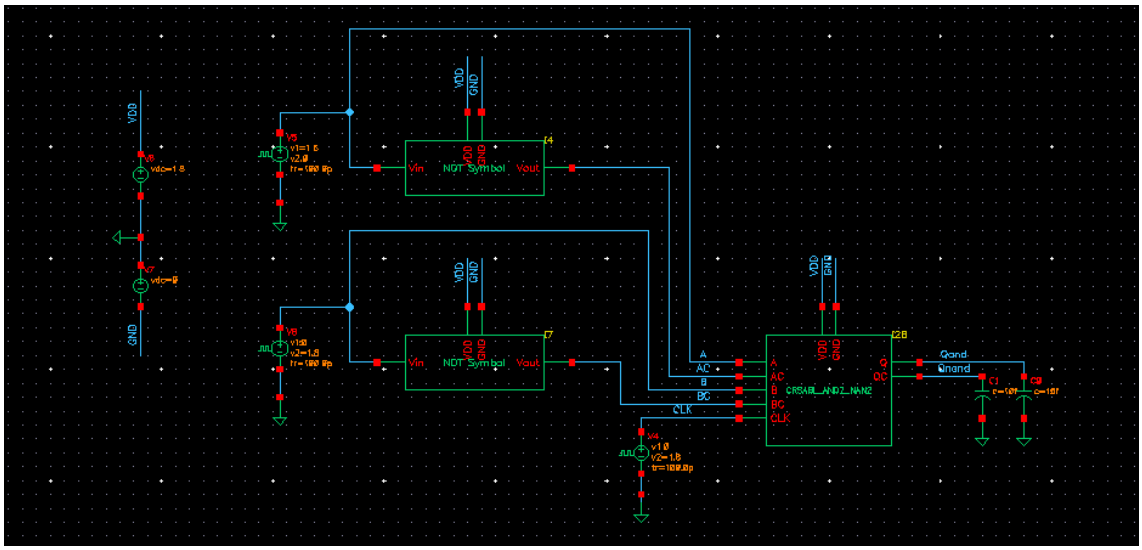


29. Ilustración: DPUN de la Celda CRSABL OR/NOR

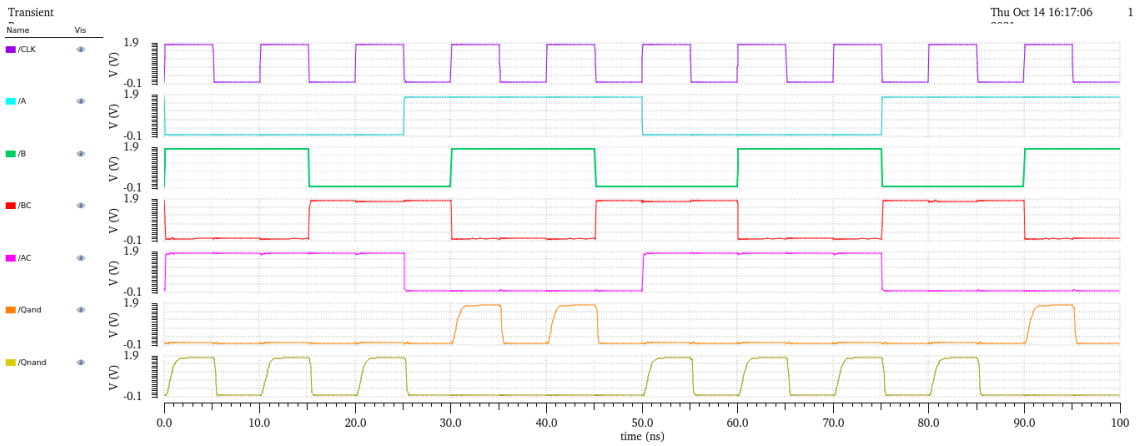
3.4.4. Diseño de la Celda DPDN CRSABL OR/NOR



30. Ilustración: Esquemático de la Celda CRSABL OR/NOR en Virtuoso (Cadence)

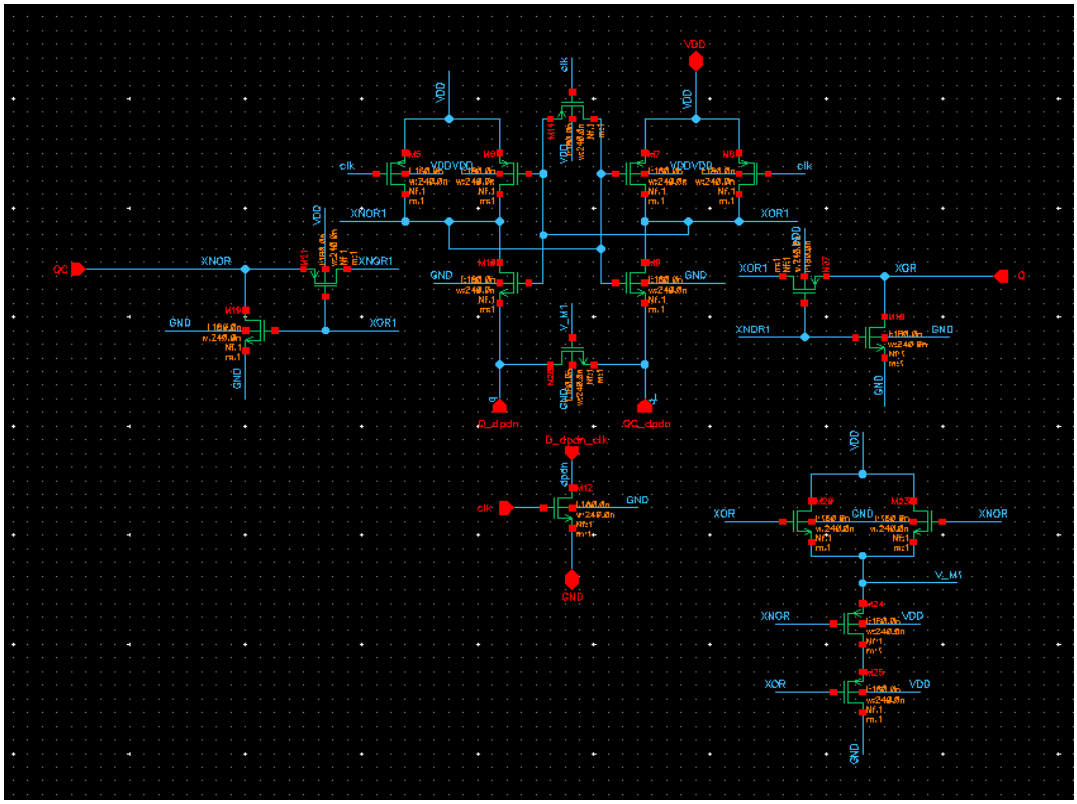


31. Ilustración: Esquemático del test para la celda CRSABL OR/NOR en Virtuoso (Cadence)



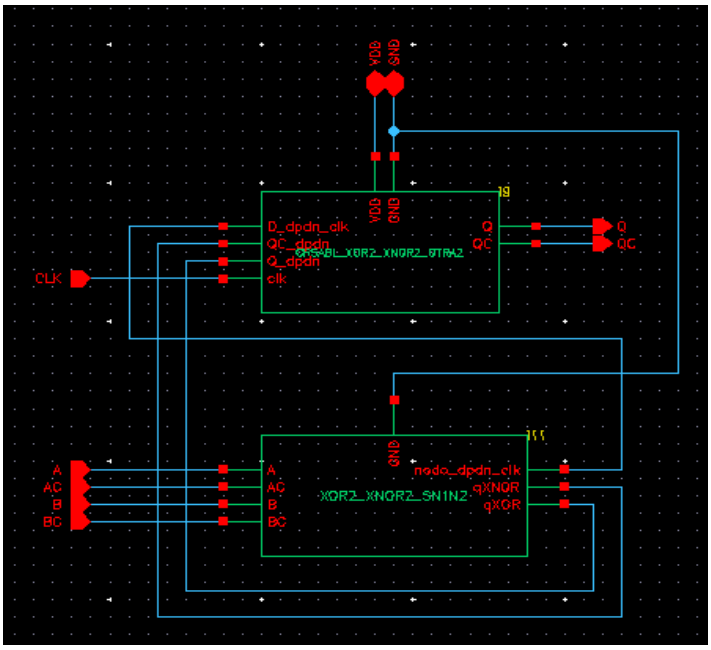
32. Ilustración: Resultado del test para la celda CRSABL OR/NOR en Spectre (Cadence)

3.4.5. DPUN de la Celda Celda XOR/XNOR

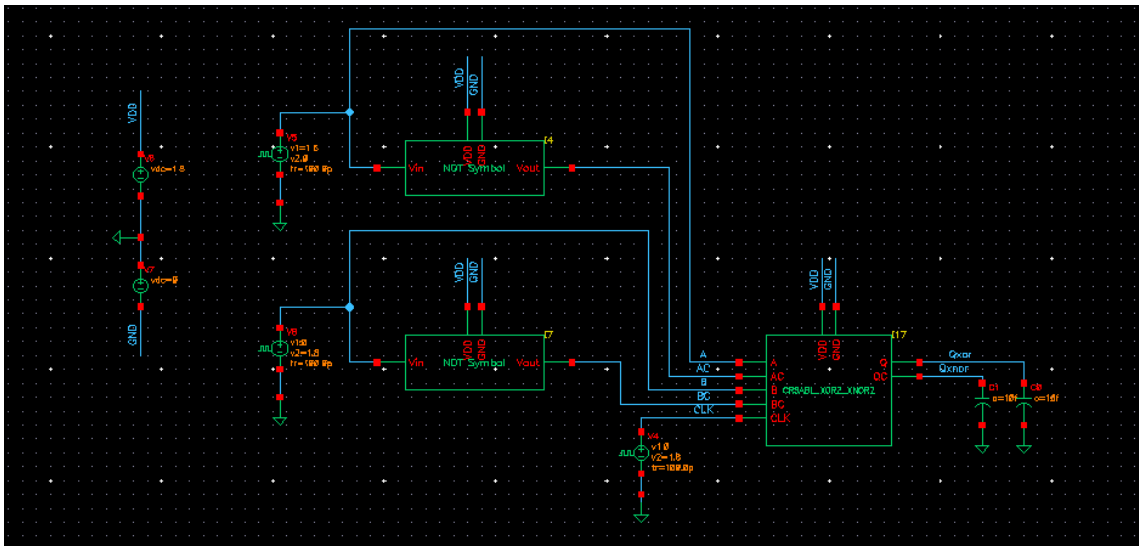


33. Ilustración: DPUN de la Celda Celda XOR/XNOR

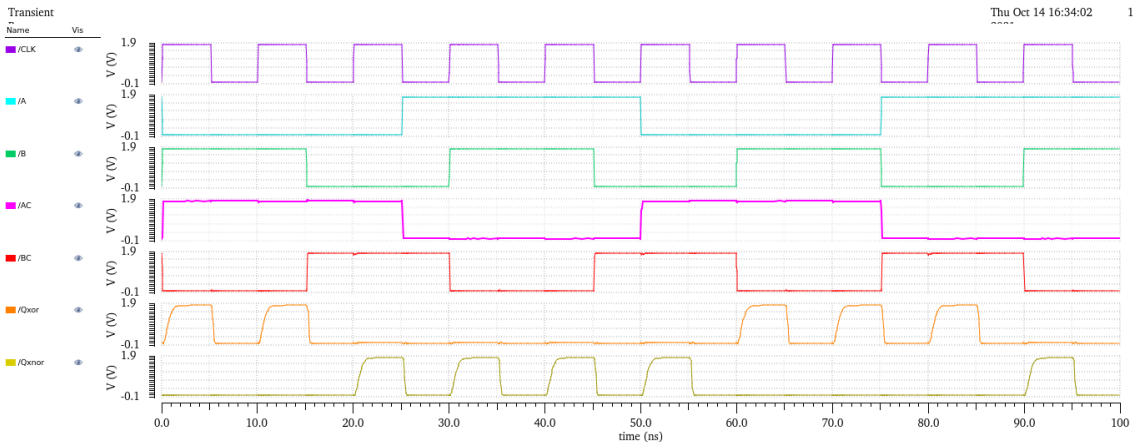
3.4.6. Diseño de la Celda DPDN CRSABL XOR/XNOR



34. Ilustración: Diseño de la Celda CRSABL XOR/XNOR

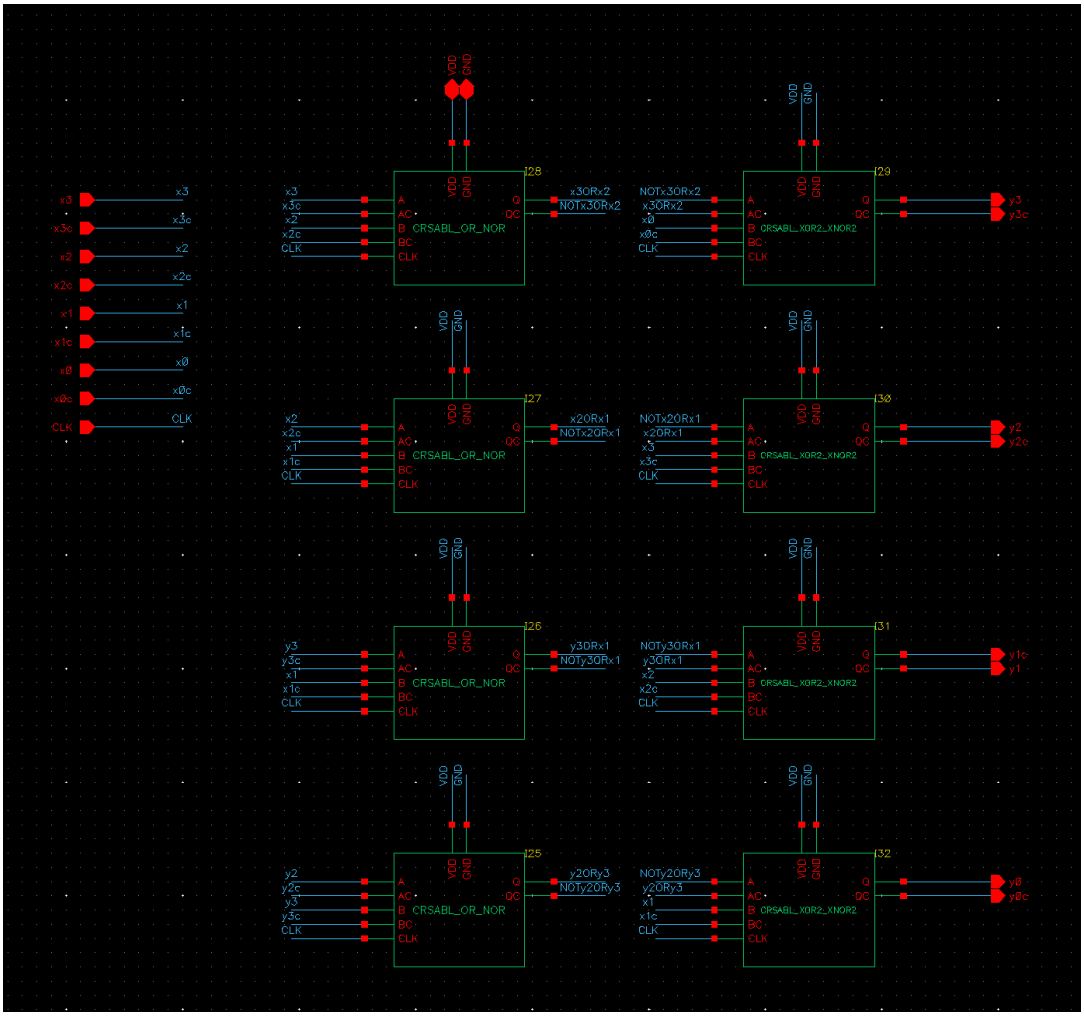


35. Ilustración: Esquemático del test para la celda CRSABL XOR/XNOR en Virtuoso (Cadence)

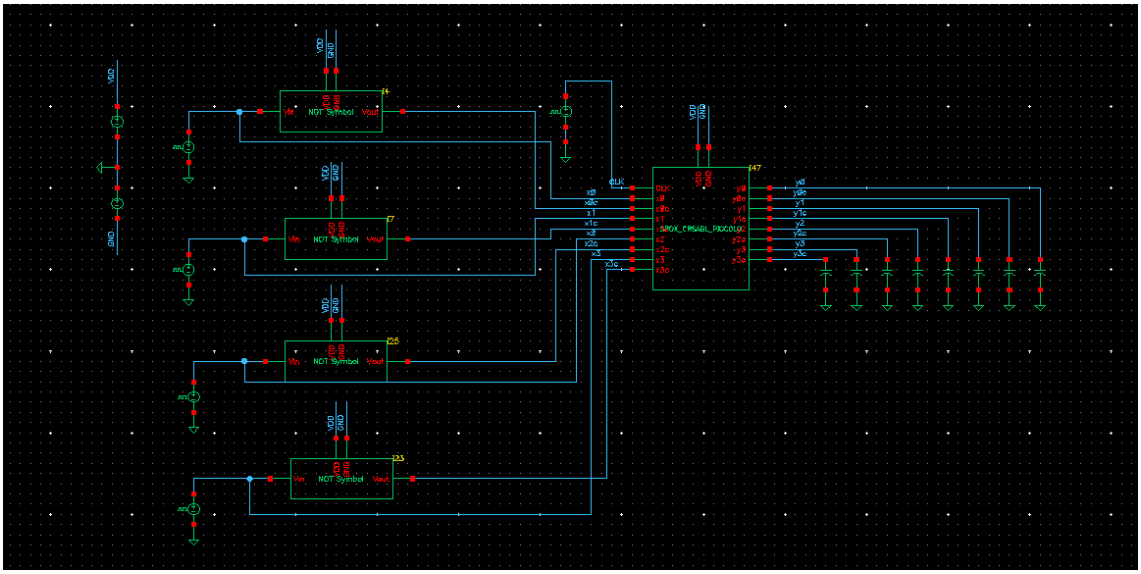


36. Ilustración: Resultado del test para la celda CRSABL XOR/XNOR en Spectre (Cadence)

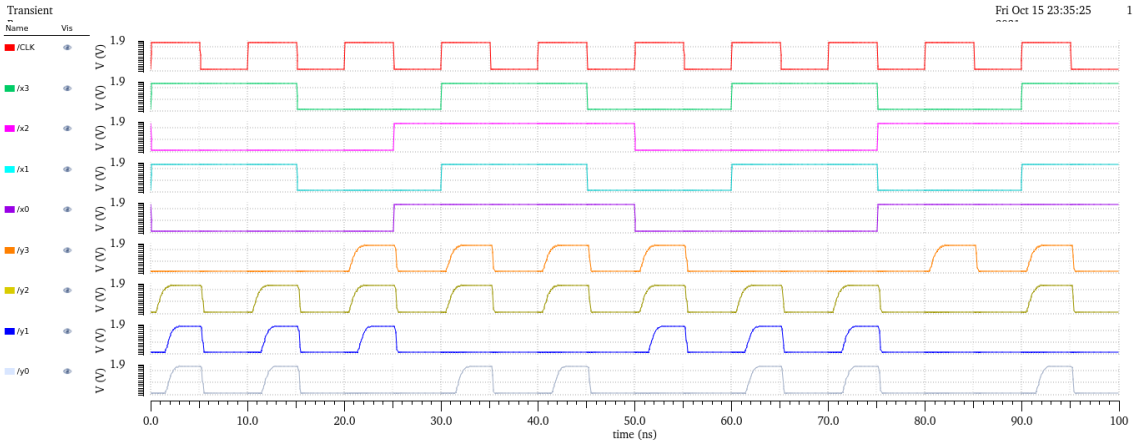
3.4.7. Diseño del demostrador Piccolo CRSABL



37. Ilustración: Esquemático del demostrador Piccolo CRSABL



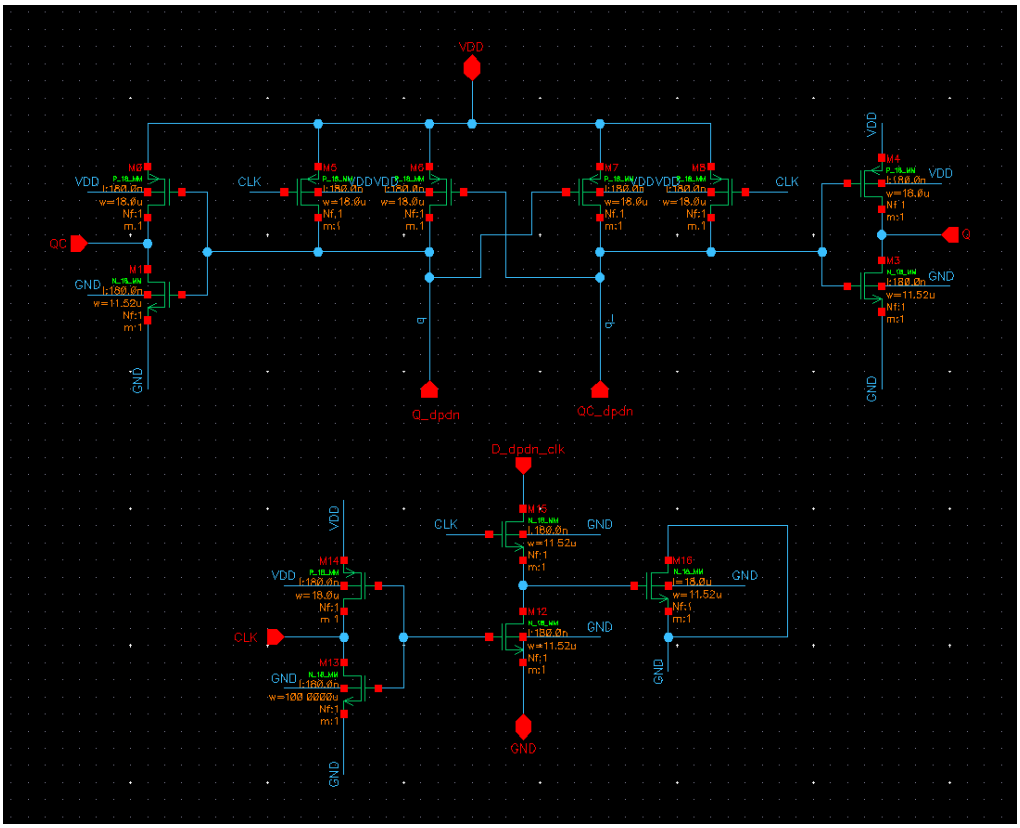
38. Ilustración: Esquemático del test del demostrador Piccolo CRSABL



39. Ilustración: Resultado del test del demostrador Piccolo CRSABL en Spectre

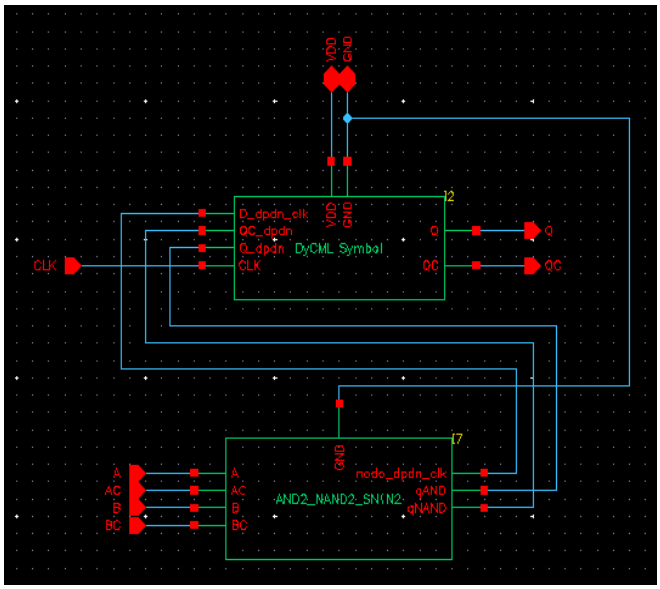
3.5. Diseño de Celdas Lógicas con la técnica DyCML

3.5.1. DPUN Lógica DyCML

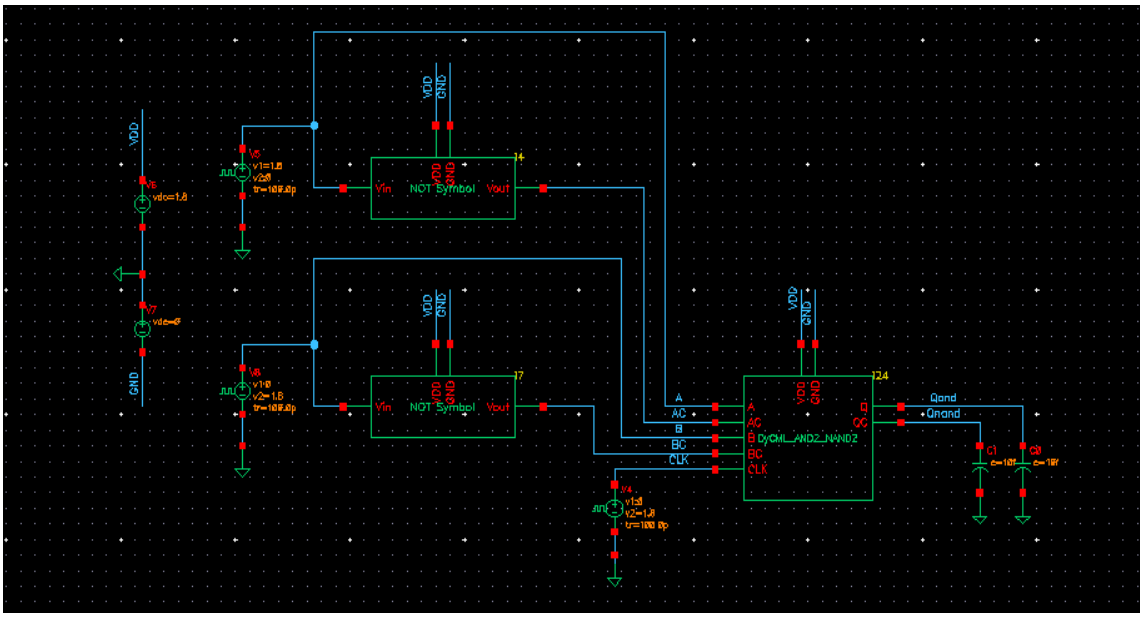


40. Ilustración: DPUN Lógica DyCML

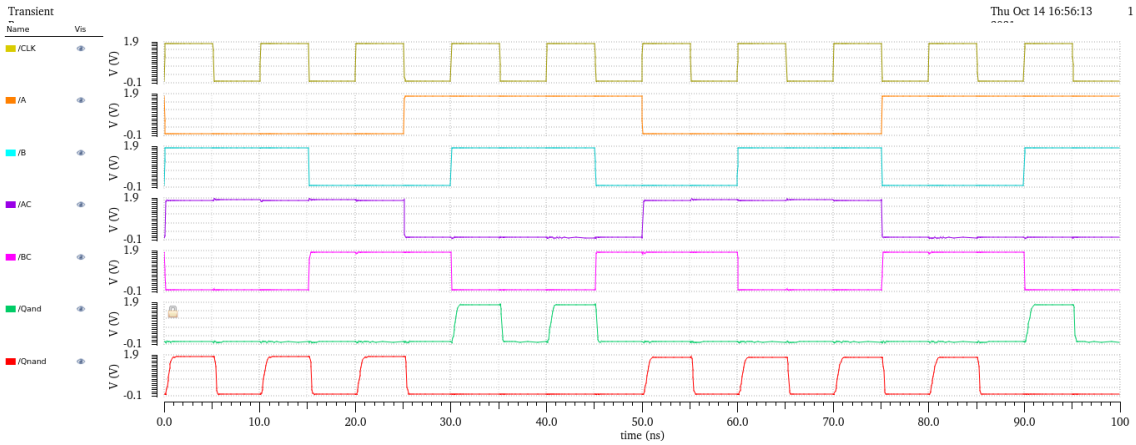
3.5.2. Diseño de la Celda DyCML AND/NAND



41. Ilustración: Esquemático de la Celda DyCML AND/NAND en Virtuoso (Cadence)

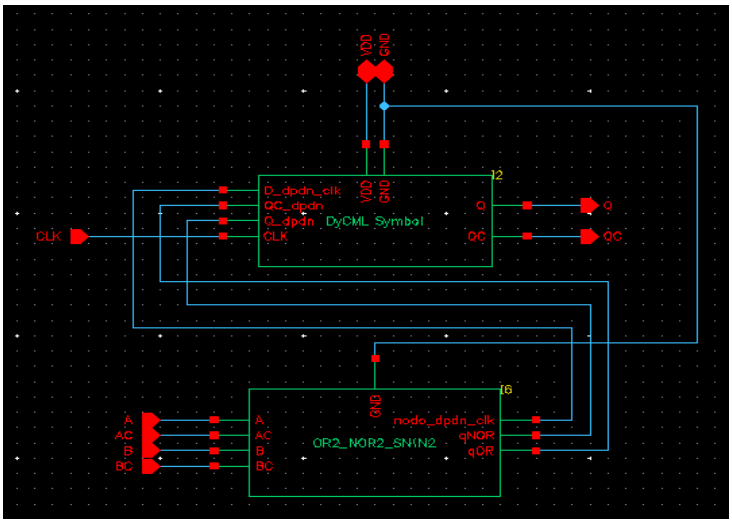


42. Ilustración: Test de la Celda DyCML AND/NAND en Virtuoso (Cadence)

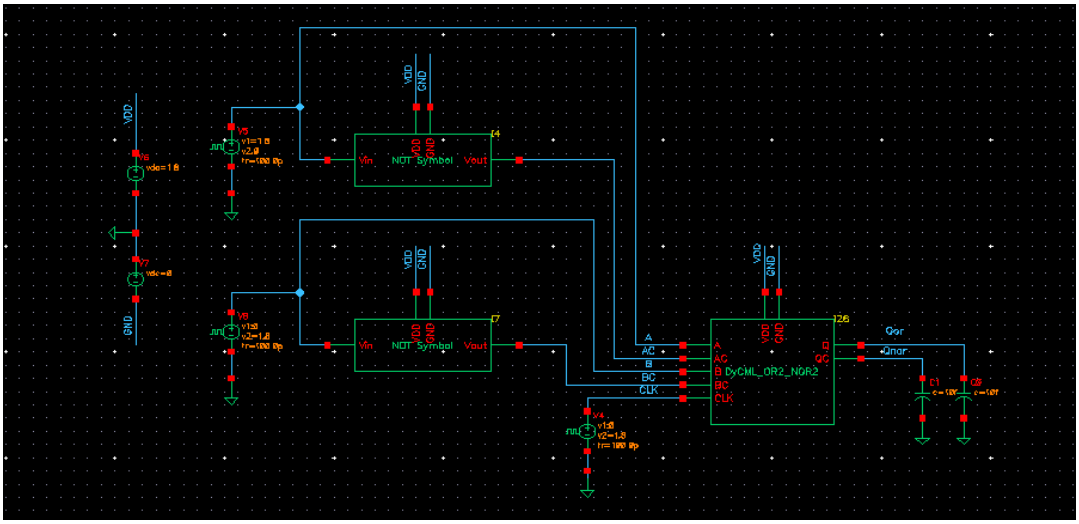


43. Ilustración: Resultados del Test de la Celda DyCML AND/NAND en Spectre (Cadence)

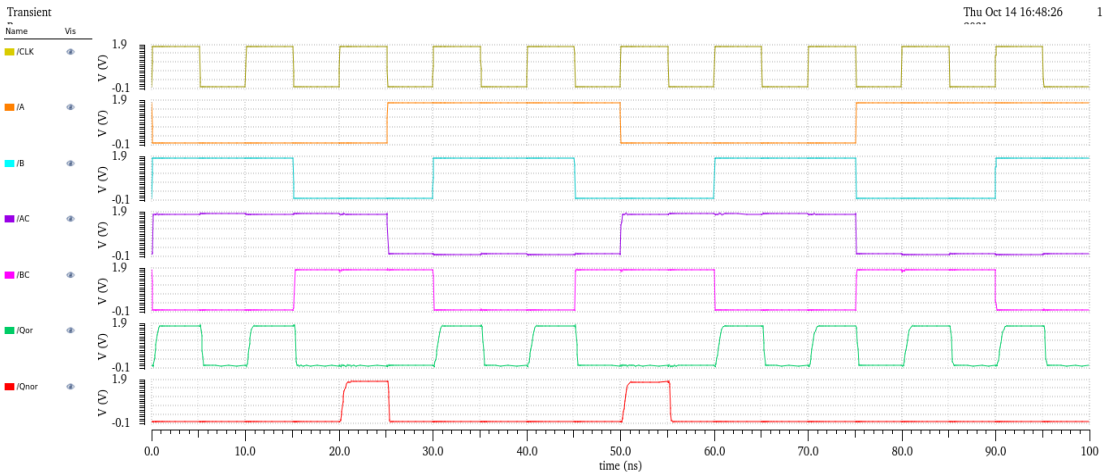
3.5.3. Diseño de la Celda DyCML OR/NOR



44. Ilustración: Esquemático de la Celda DyCML OR/NOR

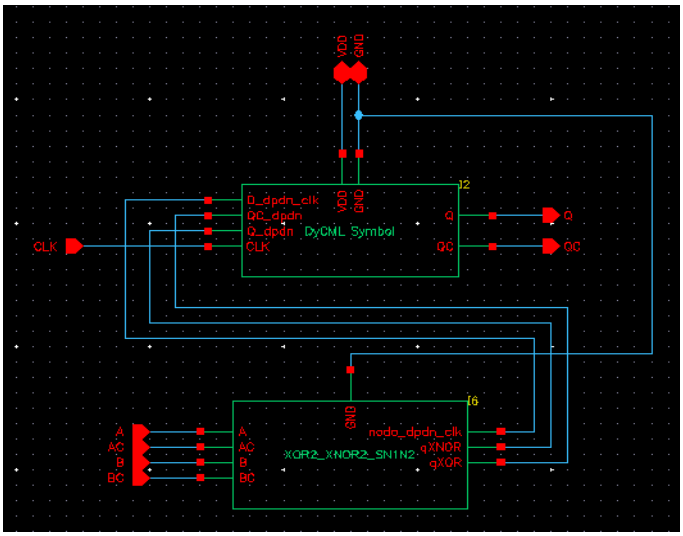


45. Ilustración: Test de la Celda DyCML OR/NOR

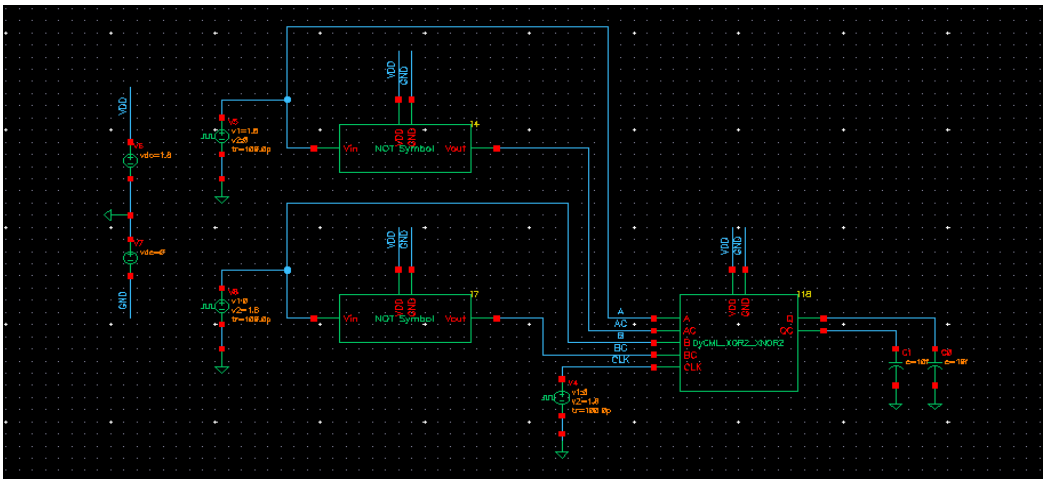


46. Ilustración: Test para la Celda DyCML OR/NOR

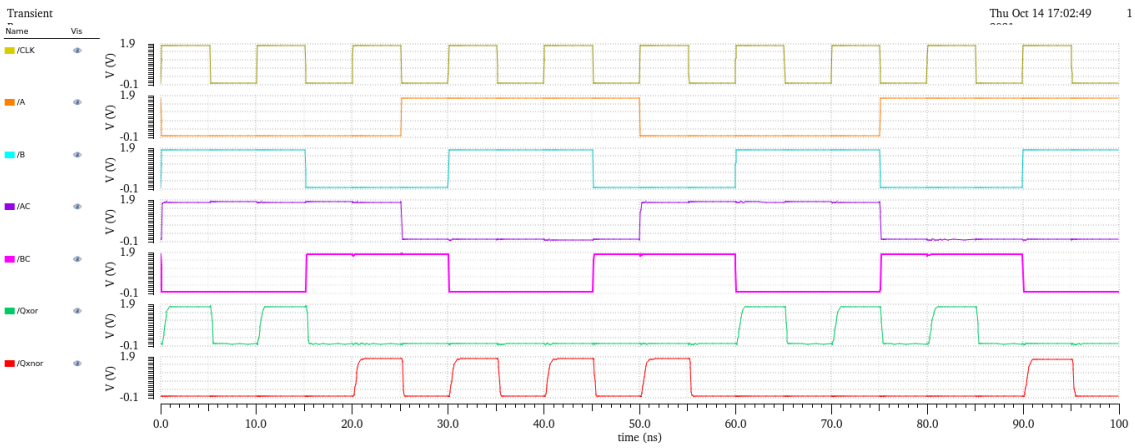
3.5.4. Diseño de la Celda DyCML XOR/XNOR



47. Ilustración: Esquemático de la Celda DyCML XOR/XNOR

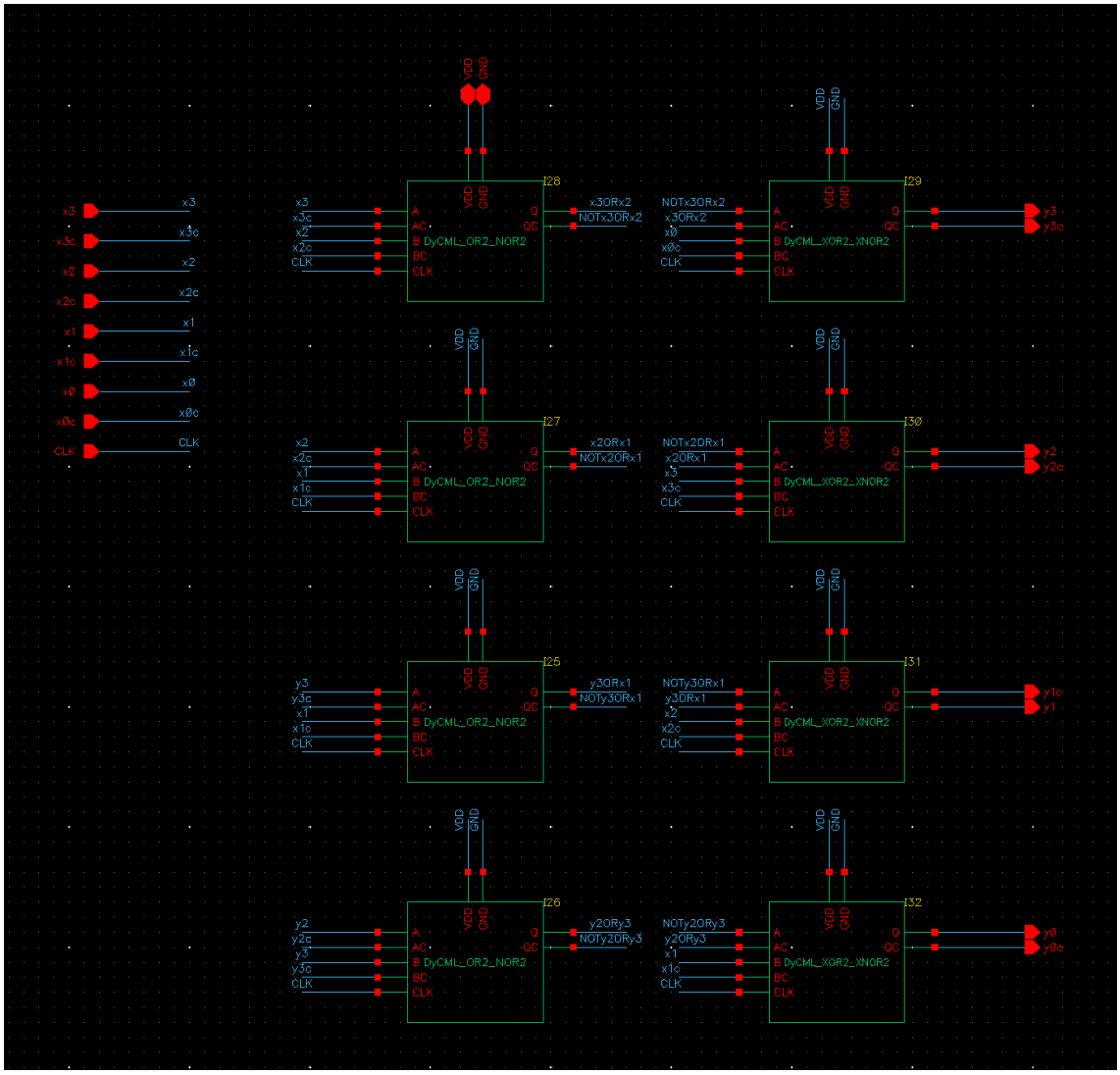


48. Ilustración: Test para la Celda DyCML XOR/XNOR

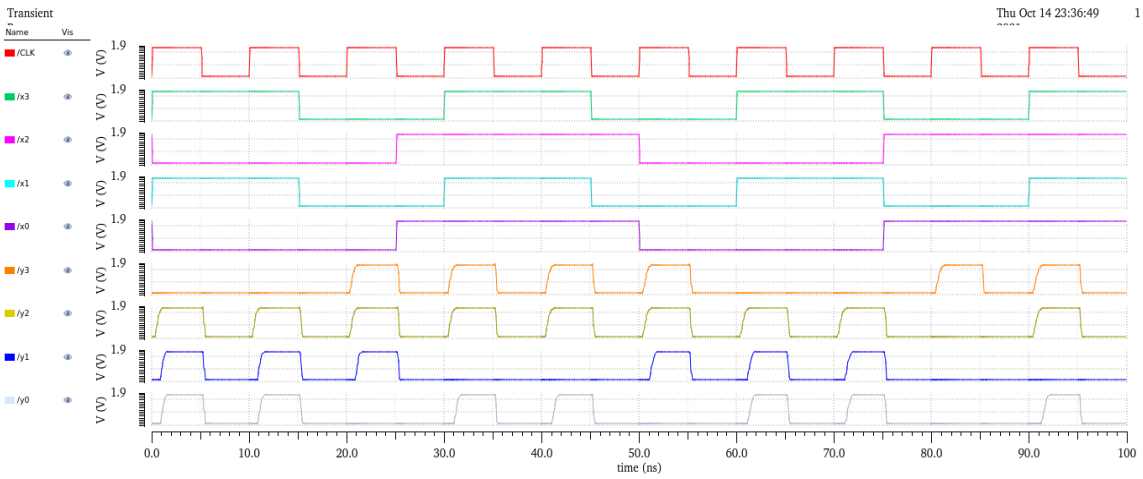


49. Ilustración: Resultado del test para la Celda DyCML XOR/XNOR

3.5.5. Diseño del demostrador Piccolo DyCML



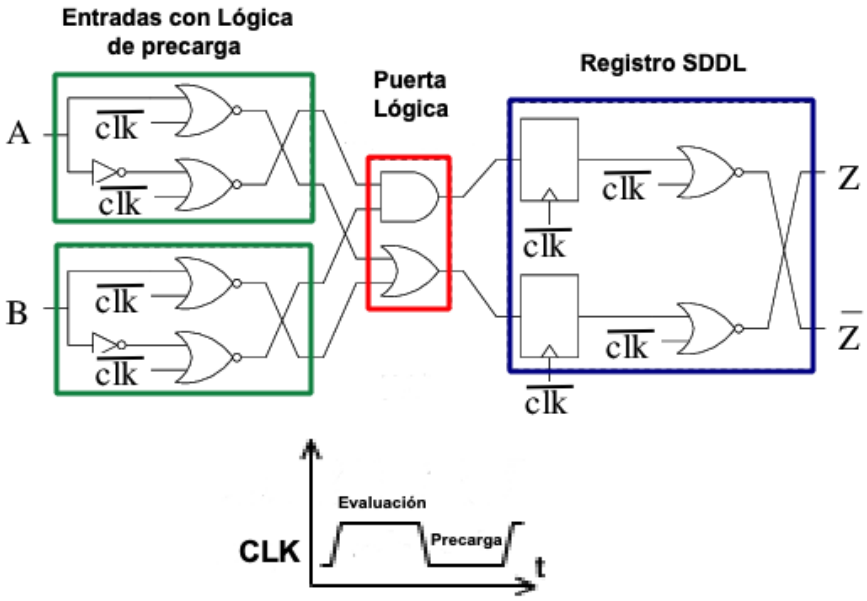
50. Ilustración: Esquemático del demostrador Piccolo DyCML



51. Ilustración: Test para el demostrador Piccolo DyCML

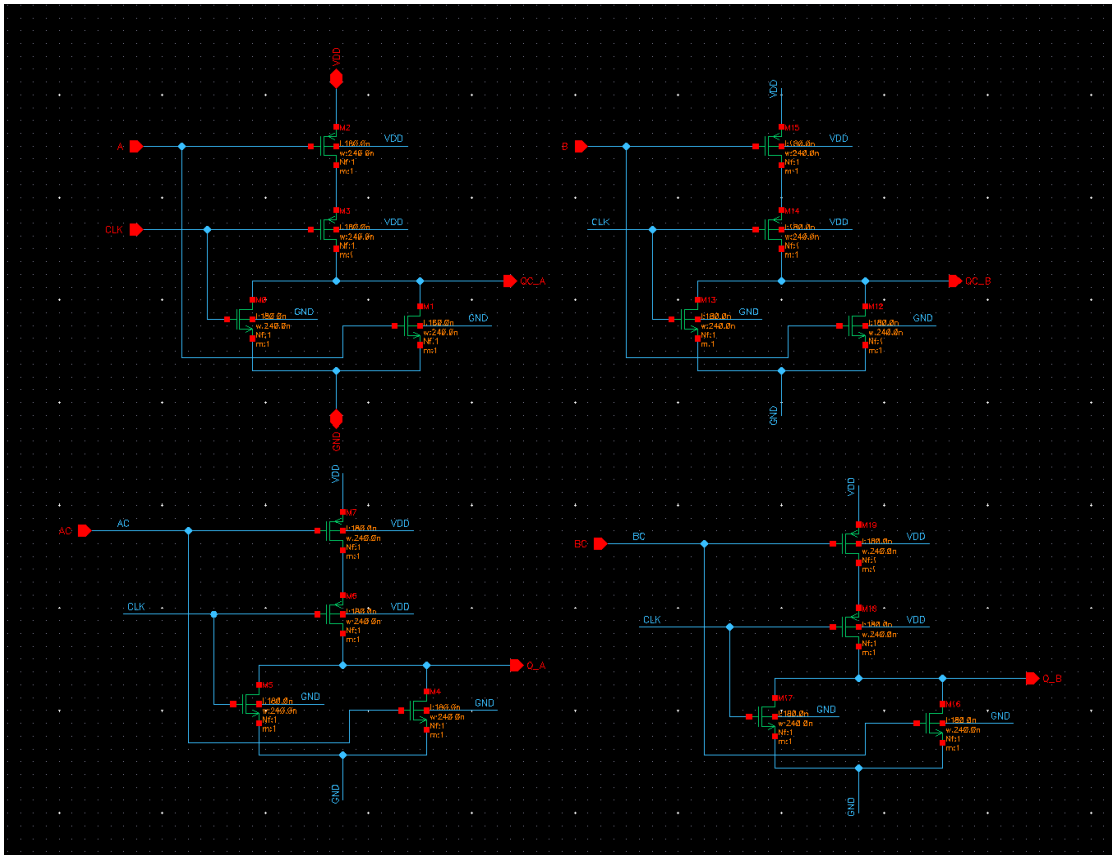
3.6. Diseño de Celdas Lógicas con la técnica WDDL

Para el diseño del estilo lógico WDDL se optó por el circuito presentado en la figura 52 que cuenta con lógica de precarga en las entradas, siendo implementada con compuertas NOR, y cuenta además con un registro SDDL en las salidas. Se desestimó el empleo de los registros SDDL Maestro-Esclavo a fin que que las ondas de salida pudieran ser comparadas con las otras lógicas diseñadas. De manera análoga, también fue colocado un inversor en la entrada de la señales de reloj para sincronizar la evaluación y la precarga de esta lógica en el mismo ciclo de reloj que para los estilo SABL, CRSABL y DyCML, CLK = 1 evaluación y CLK = 0 precarga.



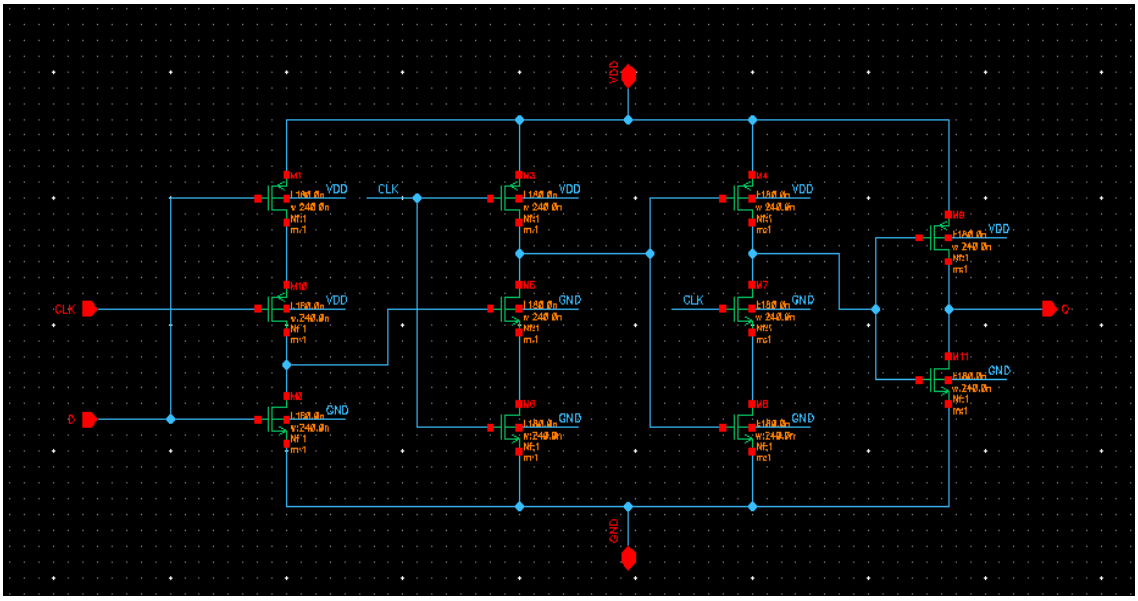
52. Ilustración: Esquema general de la lógica WDDL implementada

3.6.1. Circuito de precarga



53. Ilustración: Esquema del circuito de precarga con NOR

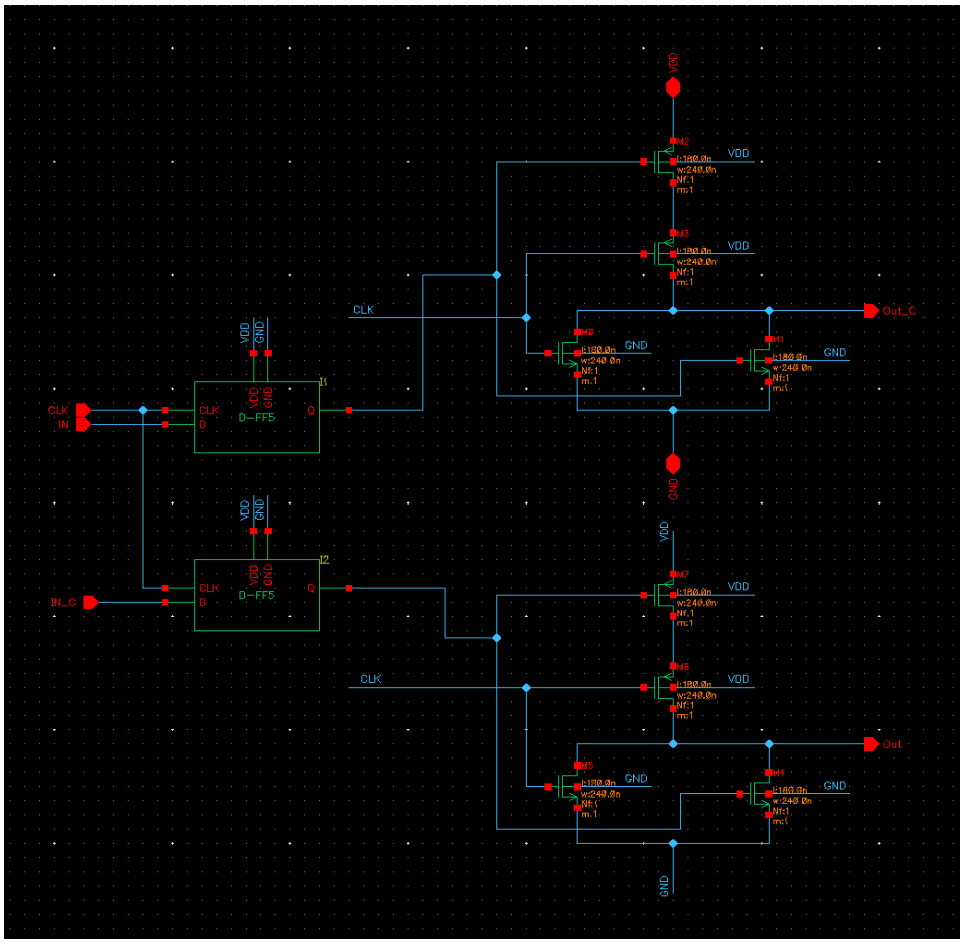
3.6.2. Flip-Flop D



54. Ilustración: Flip-Flop o registro tipo D

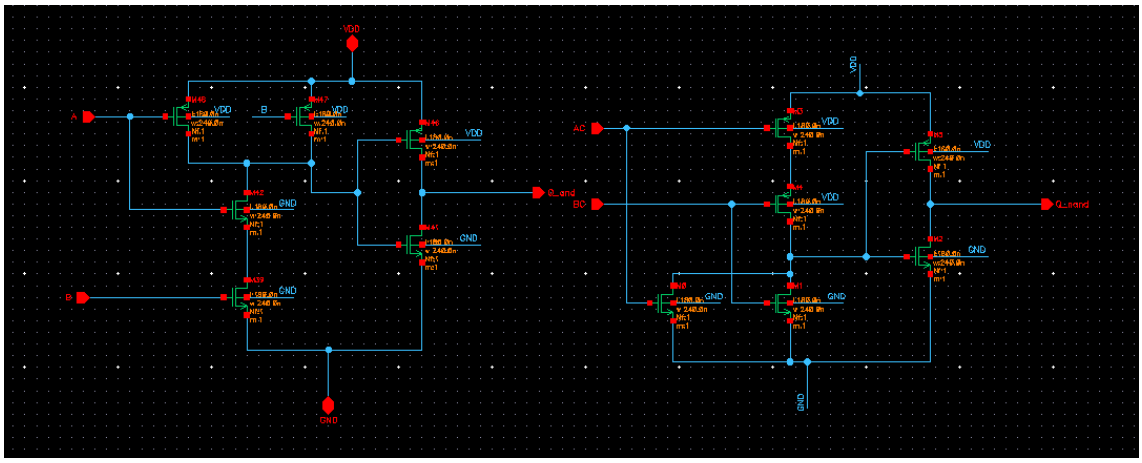
3.6.3. Registro SDDL

Implementado con 2 Flip-Flops tipo D y 2 compuertas NOR.

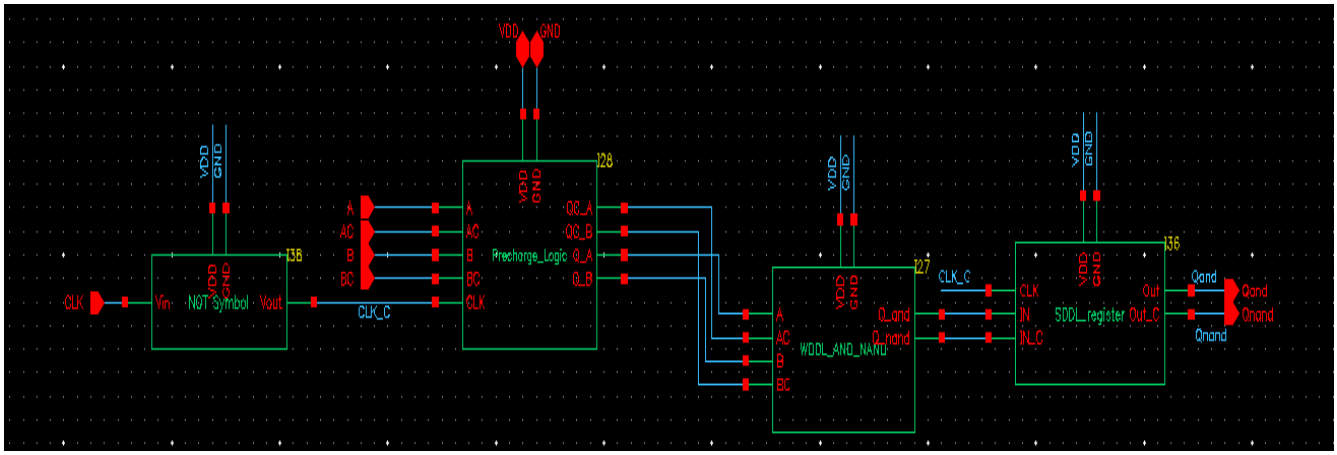


55. Ilustración: Esquemático del Registro SDDL

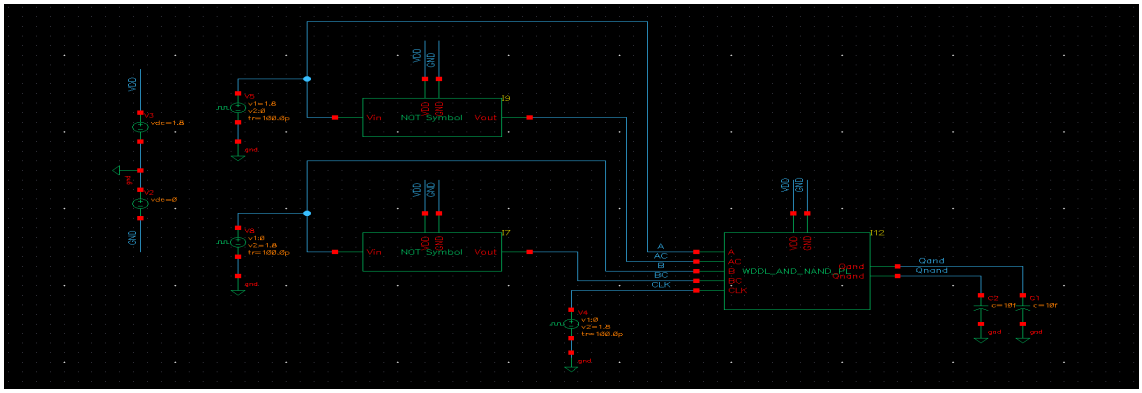
3.6.4. Diseño de la Celda AND/NAND



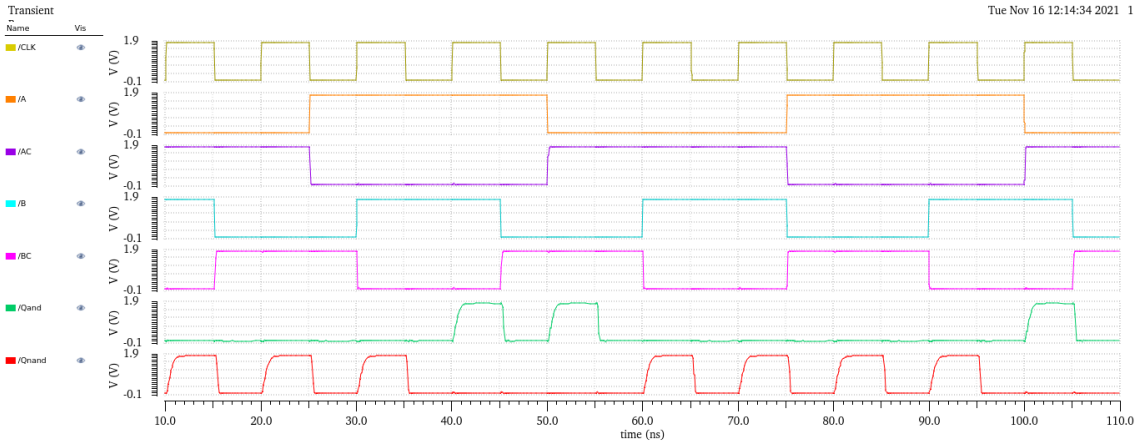
56. Ilustración: Esquemático la celda básica AND/NAND WDDL



57. Ilustración: Esquemático de la AND/NAND WDDL con lógica de precarga

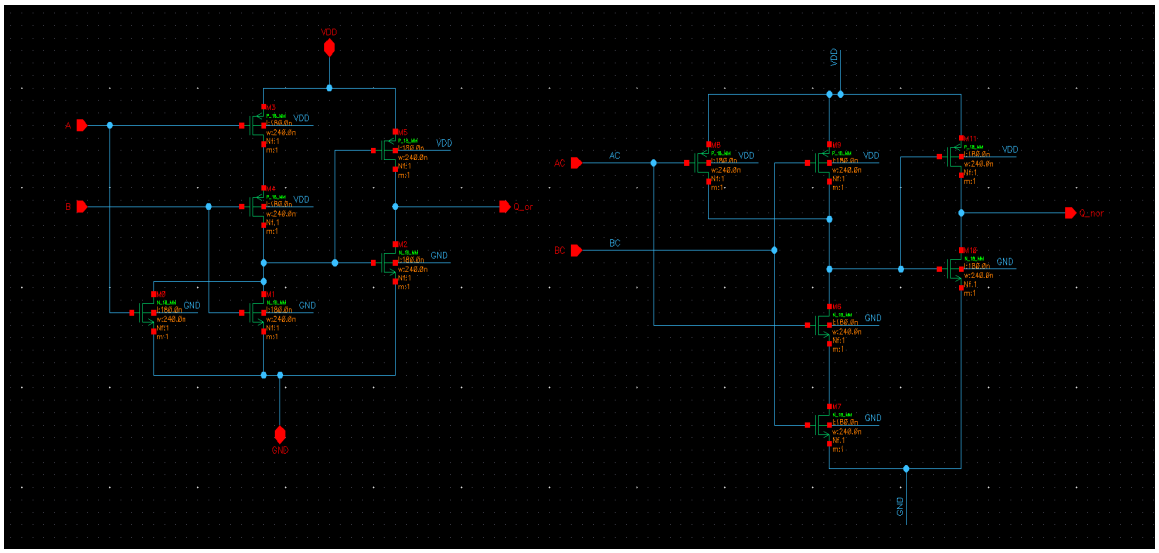


58. Ilustración: Esquema del test para la celda AND/NAND WDDL

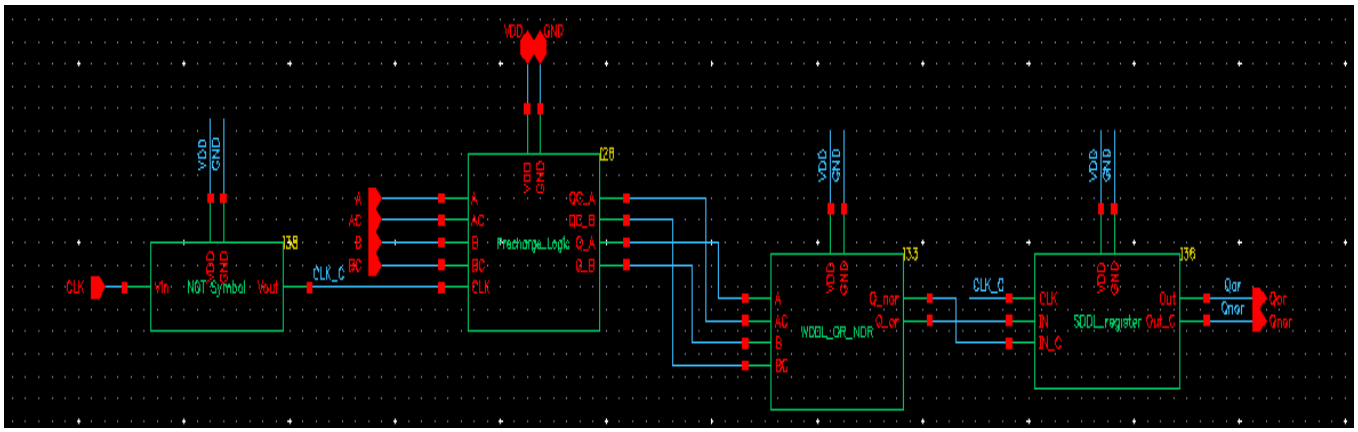


59. Ilustración: Resultado del test para la AND/NAND WDDL

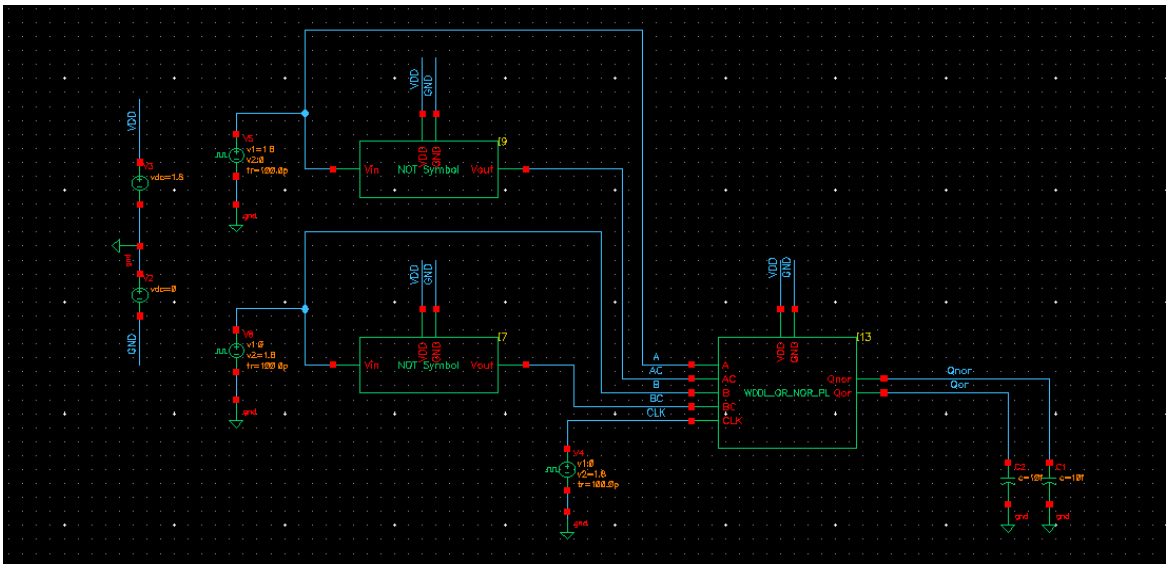
3.6.5. Diseño de la Celda OR/NOR WDDL



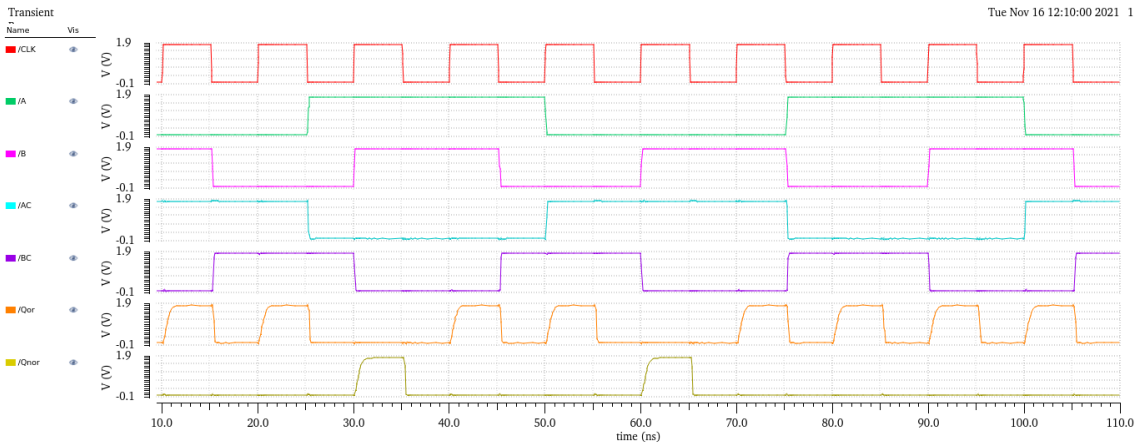
60. Ilustración: Esquemático de la Celda básica OR/NOR WDDL



61. Ilustración: Esquemático de la Celda completa OR/NOR WDDL

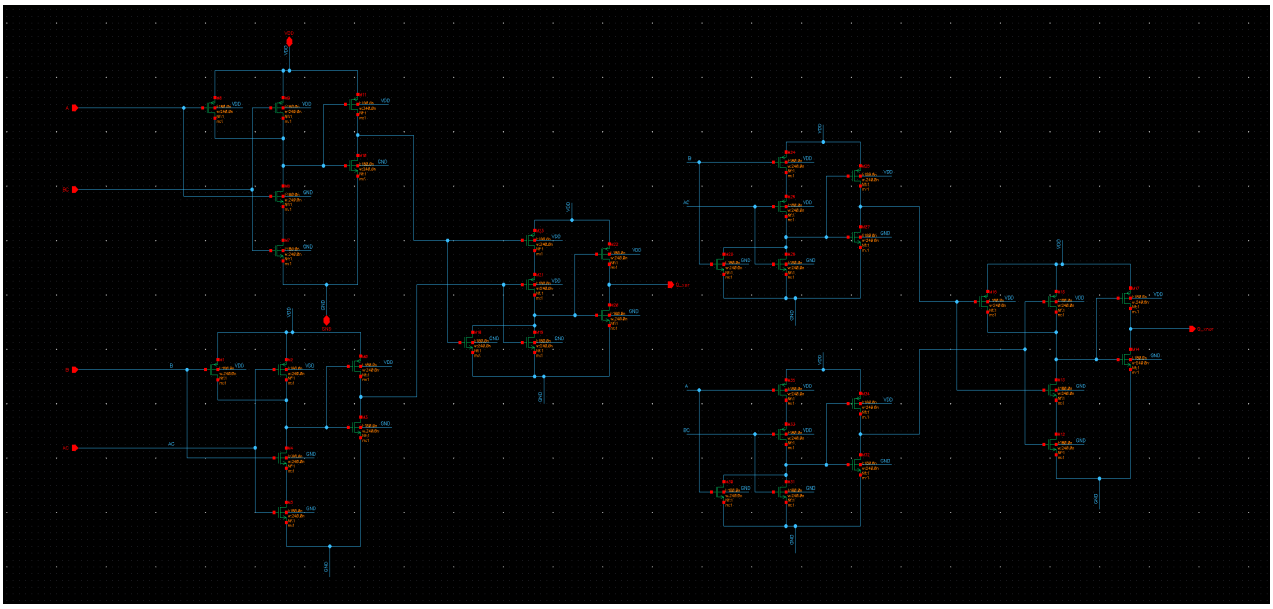


62. Ilustración: Esquemático del Test para la celda OR/NOR WDDL

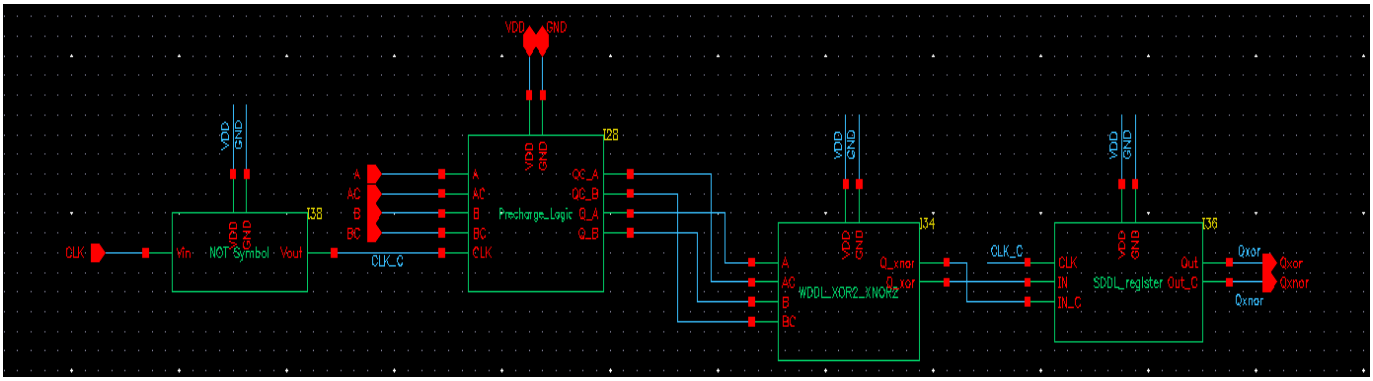


63. Ilustración: Resultados delTest para la celda OR/NOR WDDL

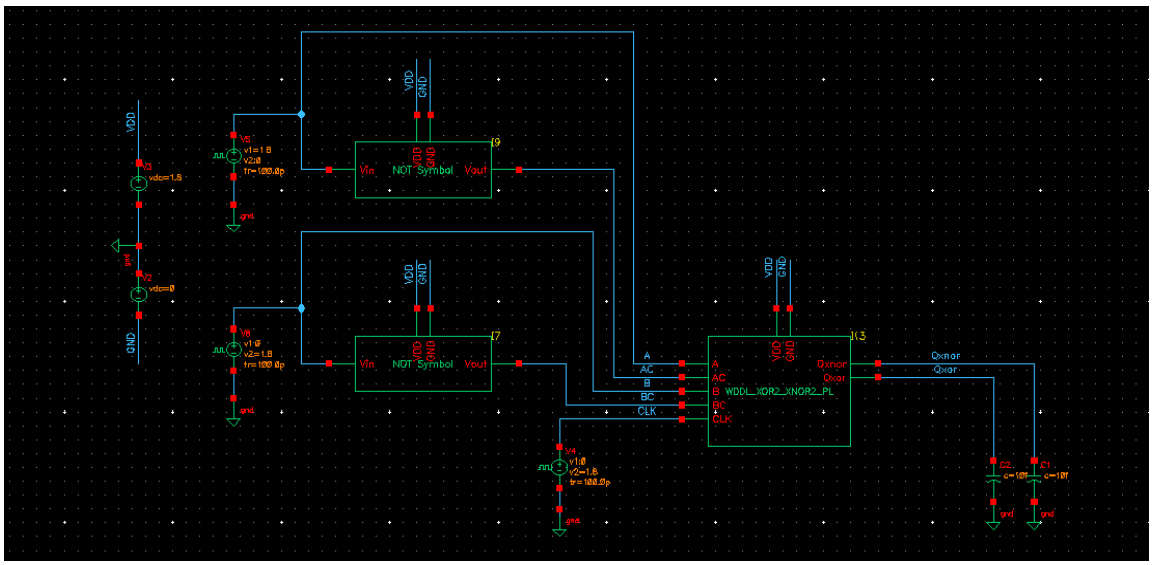
3.6.6. Diseño de la Celda XOR/XNOR WDDL



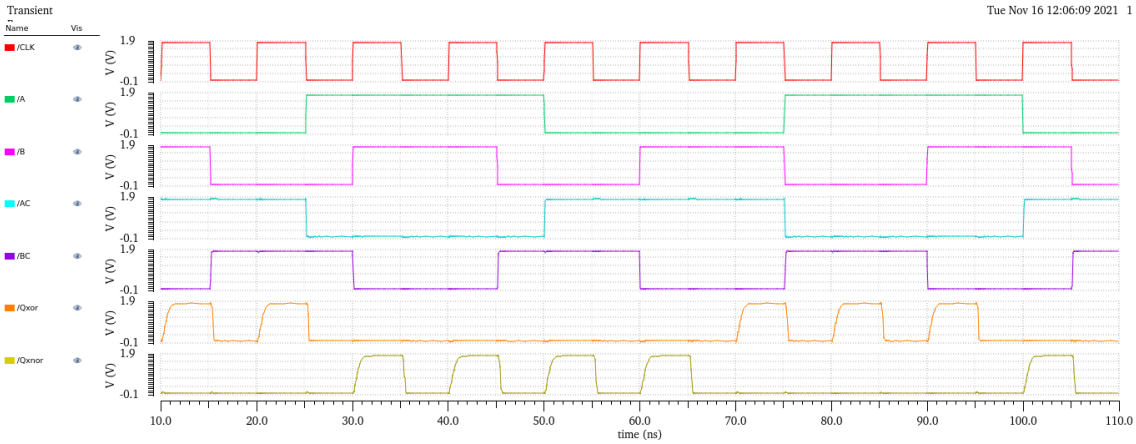
64. Ilustración: Esquema de la celda básica XOR/XNOR WDDL



65. Ilustración: Esquemático de la celda completa XOR/XNOR WDDL

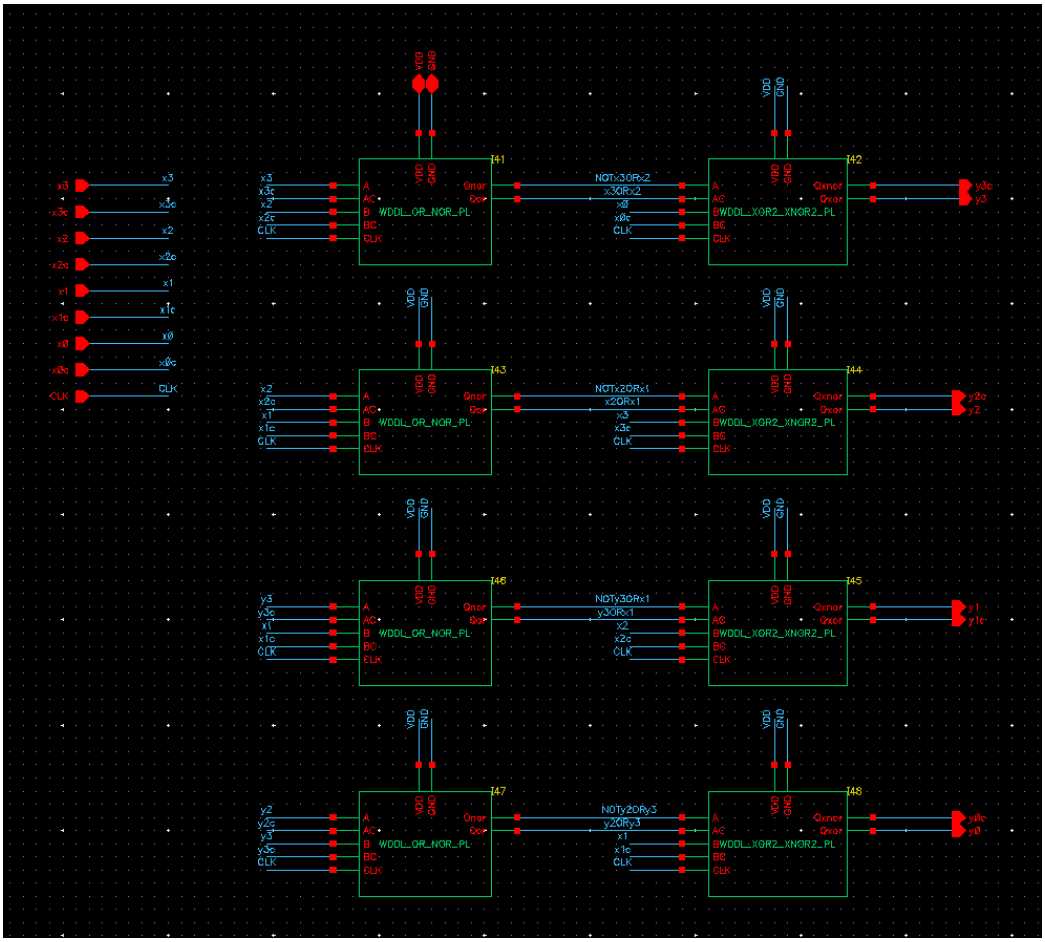


66. Ilustración: Esquemático del test de la celda XOR/XNOR

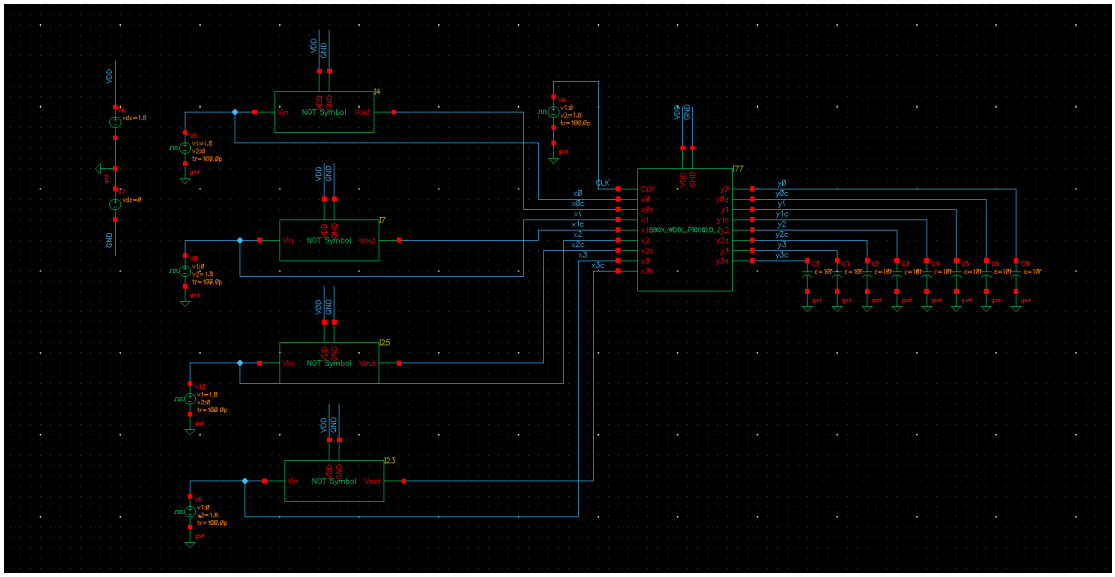


67. Ilustración: Resultados del test para la celda XOR/XNOR

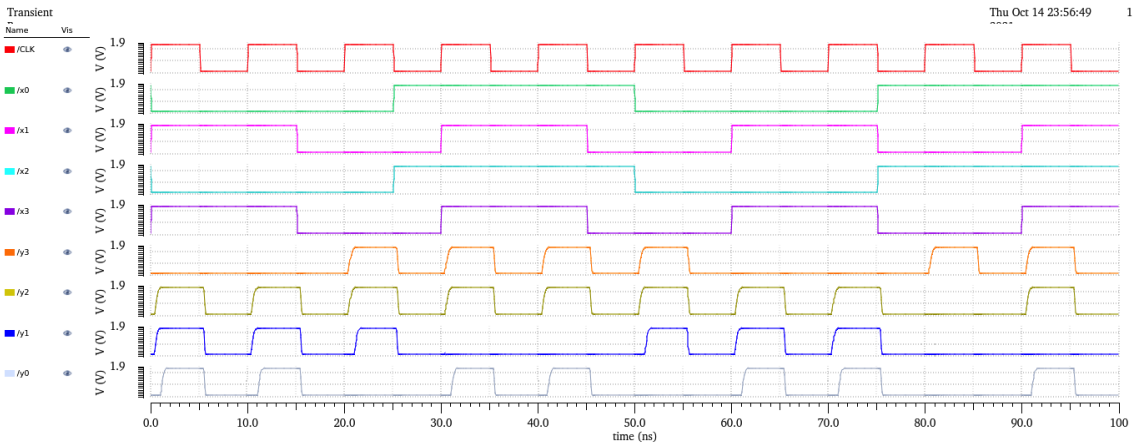
3.6.7. Diseño del demostrador SBOX WDDL



68. Ilustración: Esquemático del demostrador SBOX WDDL



69. Ilustración: Esquemático de test para el demostrador SBOX WDDL



70. Ilustración: Resultados del test para el demostrador SBOX WDDL

4. Métricas para la evaluación de las celdas lógicas DPL

En este apartado vamos a abordar las métricas usadas para evaluar la seguridad y prestaciones de las celdas lógicas. Estas métricas pueden ser divididas dentro de dos clases: Métricas de seguridad y métricas de rendimiento.

En lógica DPL las medidas de la potencia (3) y energía consumida (2) en fase de precarga y evaluación, son una medida importante y necesaria para evaluar las prestaciones o rendimiento de las celdas o circuitos, estas métricas vienen expresadas por las siguientes ecuaciones:

$$\boldsymbol{I_{AVG}} = \frac{1}{T_{CLK}} \int_{\frac{-T_{CLK}}{2}}^{\frac{T_{CLK}}{2}} i_{DD}(t)\,dt \quad (1)$$

$$\boldsymbol{E} = V_{DD} * I_{AVG} * T_{CLK} \quad (2)$$

$$\boldsymbol{P} = V_{DD} * I_{AVG} \quad (3)$$

Siendo:

$I_{avg:}$ El valor de energía promedio

T_{clk}: El periodo del reloj

$V_{DD:}$ El valor de la tensión de alimentación

i_{DD} : La corriente de polarización

P: La potencia consumida

Para evaluar la seguridad de los circuitos implementados, a nivel de celdas lógicas en la literatura encontramos dos medidas indirectas a partir del consumo de energía del circuito que son ampliamente usadas: El NED y el NSD [2,7].

- *Normalized Energy Deviation* - NED

$$\boldsymbol{NED} = \frac{Máx - Mín}{Máx} \quad (4)$$

- *Normalized Standard Deviation* - NSD

$$\boldsymbol{NSD} = \frac{\sigma}{\mu} \quad (5)$$

Siendo:

- Mín: El valor mínimo de energía
- Máx: El valor máximo de energía.
- μ: El valor medio de energía
- σ: La desviación estándar de la energía.

Lo ideal es que los valores de estos últimos dos parámetros (NED y NSD) sean cero o tiendan a ser muy bajos, de manera que cuanto más bajos sean, la seguridad del dispositivo o celda será mayor.

Otra métrica utilizada sobretodo para medir la eficiencia de una familia o puerta lógica es el Power–*delay product*, que como su nombre lo indica es el resultado del producto de la potencia promedio y el retardo promedio o tiempo de propagación desde la entrada a la salida o la duración del evento de conmutación [16]. El PDP básicamente mide la energía consumida por evento de conmutación, de manera que un dispositivo lógico o compuerta energéticamente eficiente idealmente deberán tener un PDP bajo. Está métrica es usada principalmente en dispositivos digitales. La fórmula se detalla a continuación.

$$\boldsymbol{PDP} = P_{avg} * T_{propagación} \quad (6)$$

Los parámetros estadísticos anteriormente presentados son recogidos a partir de la obtención y procesamiento de las trazas de la corriente de polarización del circuito.

5. Resultados obtenidos

Las celdas para cada una de las cuatro técnicas implementadas fueron diseñadas en el entorno Cadence usando tecnología UMC 0.18 (VDD = 1.8 V). Todas las simulaciones fueron realizadas con Spectre bajo condiciones nominales, T = 27°C y transistores con las dimensiones mínimas permitidas por la tecnología. La frecuencia de trabajo del reloj CLK que rige a cada celda se estableció en 50 Mhz, lo cual nos da un periodo T_{CLK} de 20 ns. Este periodo está dividido entre el tiempo de la fase de evaluación, cuando la señal CLK tiene un "1" lógico y el tiempo de la fase de precarga cuando CLK tiene un "0" lógico, repartido en 10 ns para cada una de las dos fases. A continuación se presenta la relación de los costes de cada implementación en términos del número de transistores para cada estilo lógico.

LÓGICA	OR	AND	XOR
SABL	18	18	18
CRSABL	23	19	23
DyCML	19	19	19
WDDL	64	64	88

2. Tabla: Comparación del número de transistores en las compuertas

LÓGICA	SBOX SABL	SBOX CRSABL	SBOX DYCML	SBOX WDDL
# Compuertas Or/Nor	4	4	4	4
# Compuertas Xor/Xnor	4	4	4	4
# transistores totales	144	184	152	584

3. Tabla: Costes en la implementación de las SBOX vs estilos lógicos

Si bien en términos del número de compuertas necesarias, las Sbox comparten idénticos valores (4 OR/NOR y 4 XOR/XNOR), el número de transistores necesarios en cada lógica difiere en algunos valores. En la tabla 3 vemos que la implementación menos costosa fue la Sbox SABL con 144 transistores, y dado que el consumo de potencia depende de la cantidad de lógica involucrada, tendríamos que suponer que la lógica SABL debería ser una de las que menos potencia consuma, por esta misma razón le sigue en costes la Sbox DyCML con un factor de x1,05, la Sbox CRSABL con un factor de x1,27 y la Sbox WDDL que cuadriplica ese valor con un factor de x4,05.

Se realizaron inicialmente simulaciones eléctricas para comprobar la funcionalidad de las celdas con compuertas básicas, como también de las SBOX, inspeccionando visualmente que la salida de las celdas estuviera acorde a su función lógica. Todas las simulaciones eléctricas para las celdas lógicas estudiadas se realizaron bajo las mismas condiciones y fueron presentadas en a sección 3.

Una vez comprobado el correcto funcionamiento de las celdas para cada una de las cuatro lógicas implementadas se creó un bloque de test en lenguaje VerilogA a fin de poder inyectar aleatoriamente y automáticamente los estímulos a nivel de las celdas SBOX y poder obtener los datos de las trazas de corriente, consumo y tiempo de propagación. Este bloque genera e inyecta el reloj y los patrones aleatorios en las 4 entradas de las Sbox y sus complementarias, se incluyó una semilla *seed* para garantizar que todas las simulaciones realizadas fueran reproducibles. El número de patrones aleatorios inyectados para cada una de las entradas a las SBOX y sus complementarias fue de 1000, que nos da un tiempo de simulación de 20 us. Puesto que tenemos 4 entradas de datos y 16 valores posibles, existen 256 combinaciones de patrones, así que se escogió la cantidad de 1000 patrones para intentar garantizar que todas las combinaciones posibles de entradas sean generadas y cubiertas con este test.

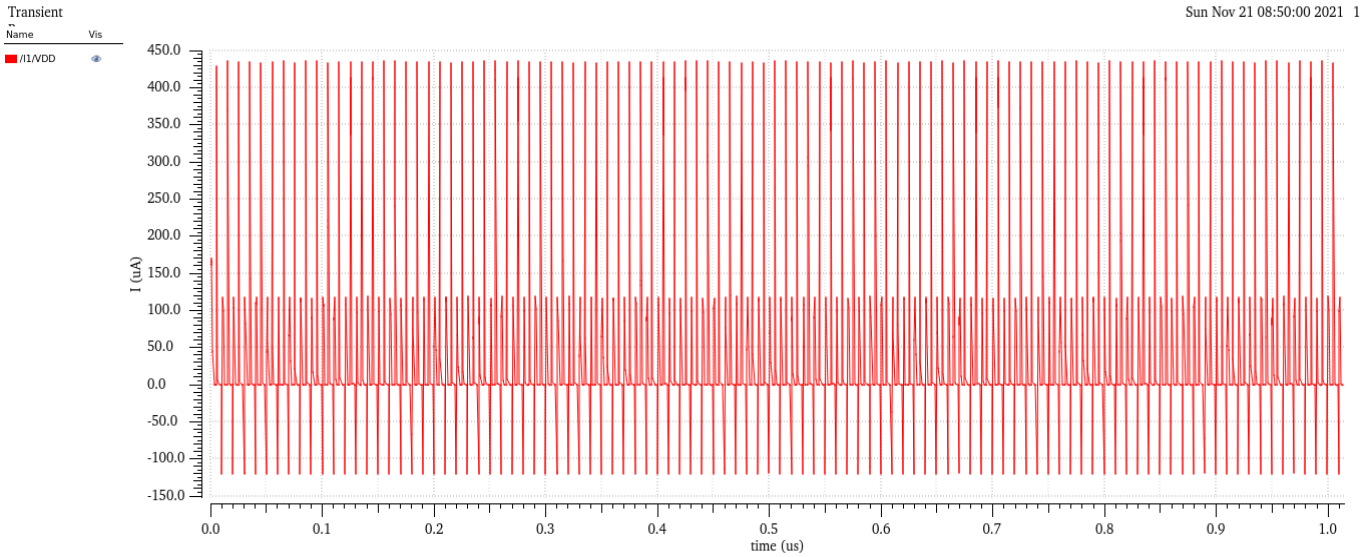
Para garantizar que las simulaciones con Spectre fueran reproducibles y se inyectarán los mismos patrones en cada una de las Sboxes se aplica el mismo test con idéntica semilla en cada uno de los bloques Sbox (SABL, CRASABL, DyCML y WDDL) , de manera tal que el test es el mismo cada que se corre una simulación.

De cada simulación eléctrica aplicada en cada una de las Sboxes, se obtuvieron las trazas de corriente de polarización y se midieron los parámetros estadísticos de las métricas de evaluación de la seguridad y el rendimiento expuestos en el capítulo anterior. Para rodar las simulaciones y obtener dicha información de corriente, salidas y delays se creó un script en *Ocean Script* la herramienta del *Framework* de Cadence que lanza automáticamente la simulación y almacena en ficheros la información de la corriente de polarización I_{VDD}, así como también los valores de salida de las SBOX.

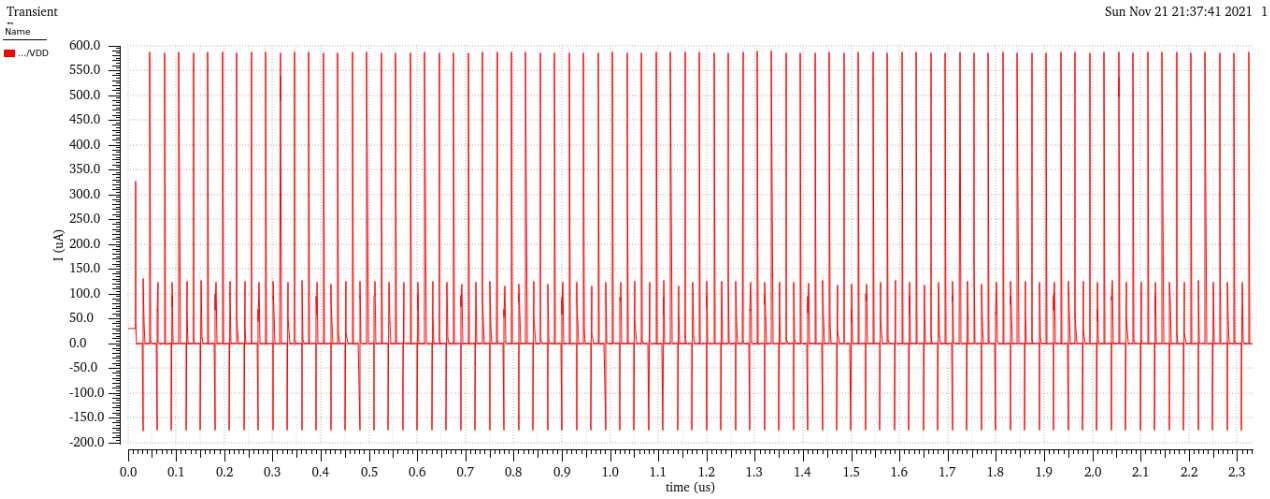
Para procesar estadísticamente la información de las simulaciones contenida en los ficheros de texto generados con *Ocean script* se creó otro script en MatLab [15] cuya función es cargar los ficheros arrojados por Ocean y calcular automáticamente el NED, NSD, la corriente promedio, potencia, energía, como también los valores mínimos y máximos del tiempo de propagación y

delays, dichos valores se encuentran discriminados para la fase de precarga y evaluación. Al analizar los datos de las trazas de corriente para obtener las medidas de consumo y potencia de las celdas, se separaron los datos de evaluación y precarga, así los datos contenidos en la señal alta del CLK son para la evaluación y los datos obtenidos durante la señal baja de CLK para precarga. Con la anterior metodología se obtuvieron los siguientes resultados de simulación:

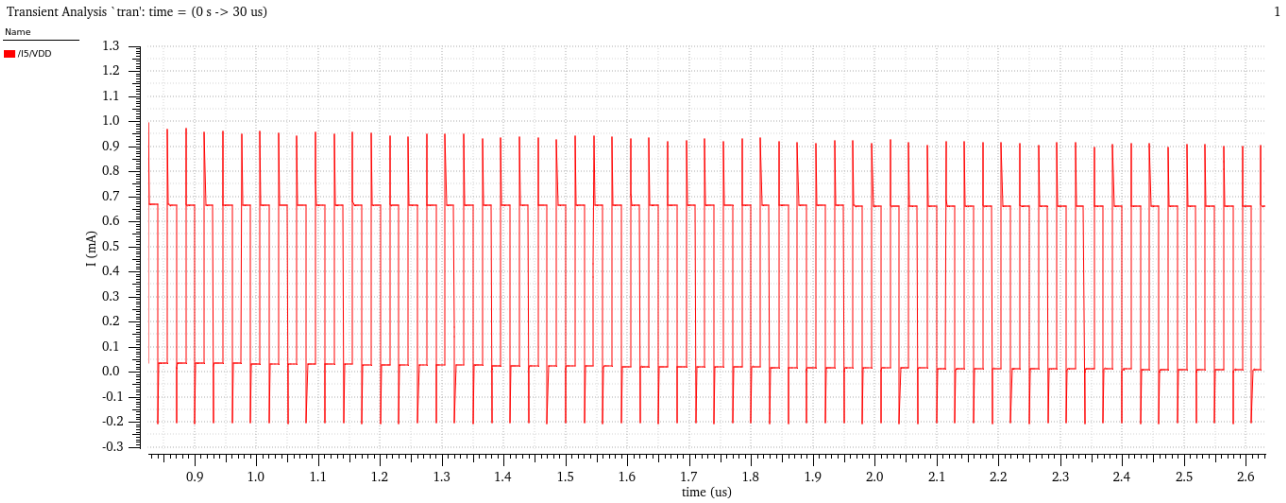
Las trazas de corriente de polarización para cada una de las lógicas en cuestión se muestran a continuación.



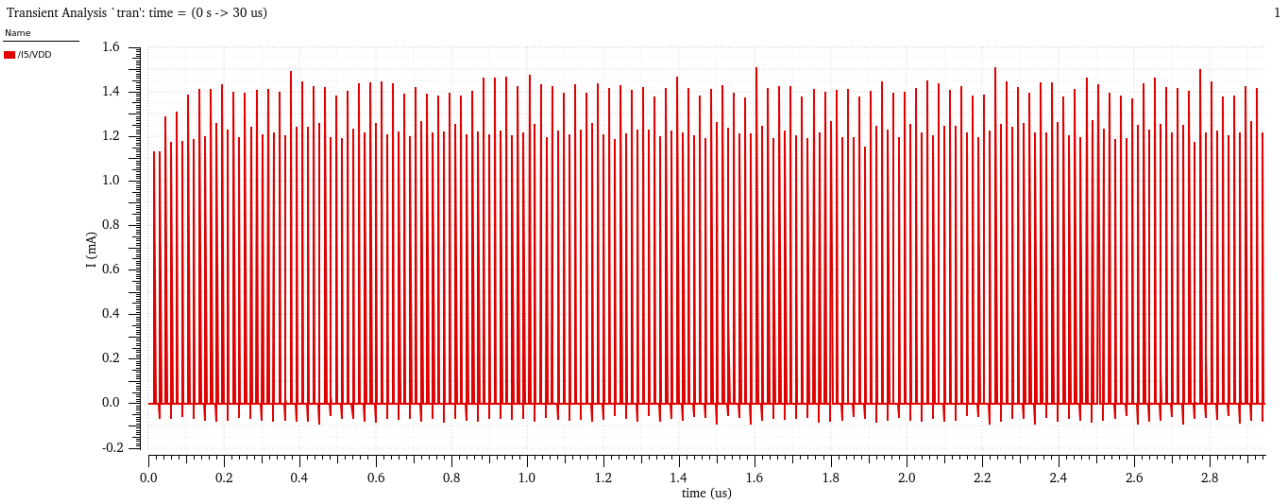
71. Ilustración: Traza de corriente IDD para la celda SBOX SABL



72. Ilustración: Traza de corriente IDD para la celda SBOX CRSABL



73. Ilustración: Traza de corriente IDD para la celda SBOX DyCML



74. Ilustración: Traza de corriente IDD para la celda SBOX WDDL

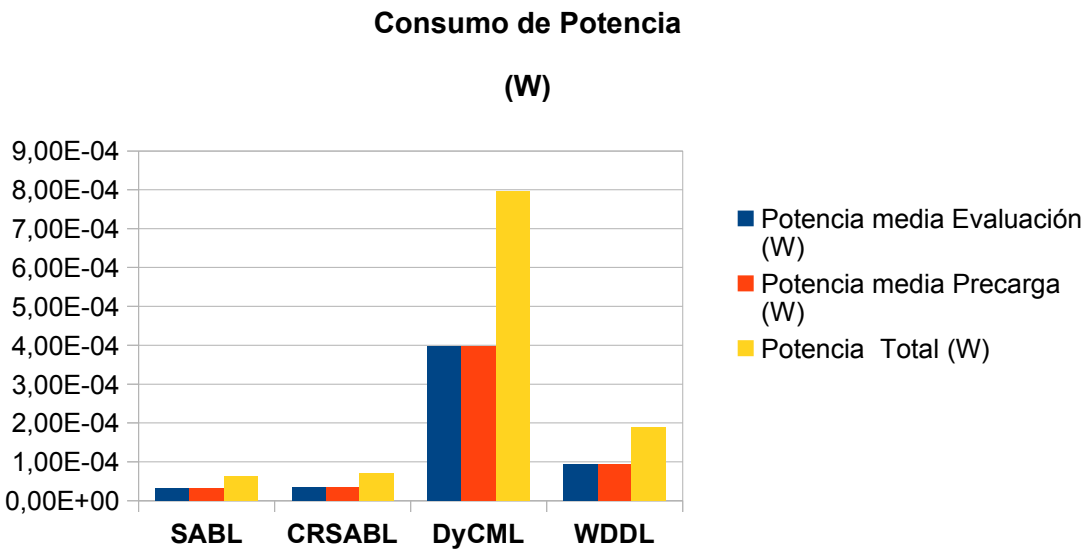
A partir de las métricas presentadas en la sección 4 compararemos las características generales de las celdas para cada una de las cuatro lógicas abordadas y determinaremos su nivel de robustez y seguridad. Los resultados del consumo de potencia promedio, potencia total y energía de las celdas son presentados en la tablas 4, 5 y en la gráfica de barras de la figura 62.

SBOX PICCOLO	Energia media Evaluación (fJ)	Energia media Precarga (fJ)	Energia media Total (fJ)
SABL	3,13E-014	3,20E-014	6,33E-014
CRSABL	3,52E-014	3,52E-014	7,05E-014
DyCML	3,99E-013	3,99E-013	7,97E-013
WDDL	9,40E-014	9,42E-014	1,88E-013

4. Tabla: Comparativa del consumo de energía

SBOX PICCOLO	Potencia media Evaluación (uW)	Potencia media Precarga (uW)	Potencia Total (uW)
SABL	31,3071	32,0386	63,3456
CRSABL	35,2134	35,2396	70,4530
DyCML	398,5640	398,5470	797,1110
WDDL	94,0381	94,1729	188,2110

5. Tabla: Comparativa del consumo de potencia



75. Ilustración: Datos comparativos del consumo de potencia de las SBOX

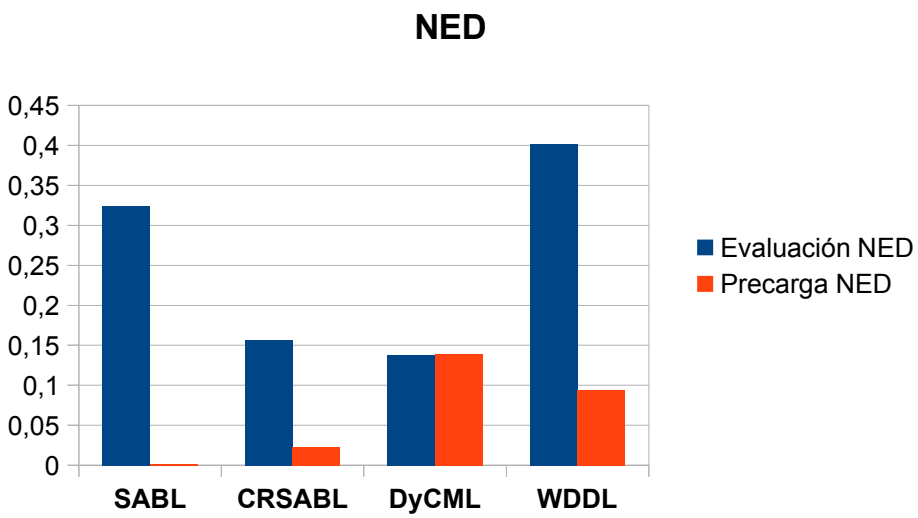
En la tabla 5 podemos apreciar el consumo de potencia de cada lógica, discriminado para la fase de evaluación, precarga y el consumo total. Como vemos en la tabla, la implementación que exhibió menor consumo de potencia fue la SABL, seguida por la CRSABL en un factor de x1,124. El consumo de la WDDL fue superior al de la SABL en un factor de x3, mientras que el consumo de la lógica DyCML fue un x12,73 superior a esta. Comparando este resultado con lo condensado en las tablas 2-3 podemos corroborar que el costo en hardware de la celda es directamente proporcional al consumo de potencia y probablemente también en área, pues la técnica WDDL fue la que mas recursos consumió, seguida por la CRSABL. En la literatura se plantea que el estilo DyCML ofrece un consumo inferior al del estilo SABL, no obstante estos resultados no avalan ese planteamiento, las razones podrían encontrarse en las dimensiones del transistor que actúa como condensador y en la tensión de los nodos intermedios que pueda provocar conducción en alguno de los transistores, ya que en este trabajo utilizamos la geometría mínima permitida por a tecnología. No se aprecia una diferencia significativa en el consumo de potencia entre las fase de evaluación y la de precarga, en los 4 casos la relación es

en un factor de aproximadamente x1, los datos son casi semejantes. Se sabe que el consumo de energía en la fase de precarga no es dependiente del los datos procesados, por tanto, podríamos decir que en este sentido el propósito de la lógica DPL se cumple y que es eficiente para “enmascarar” el consumo de potencia de los circuitos internos en la fase de evaluación.

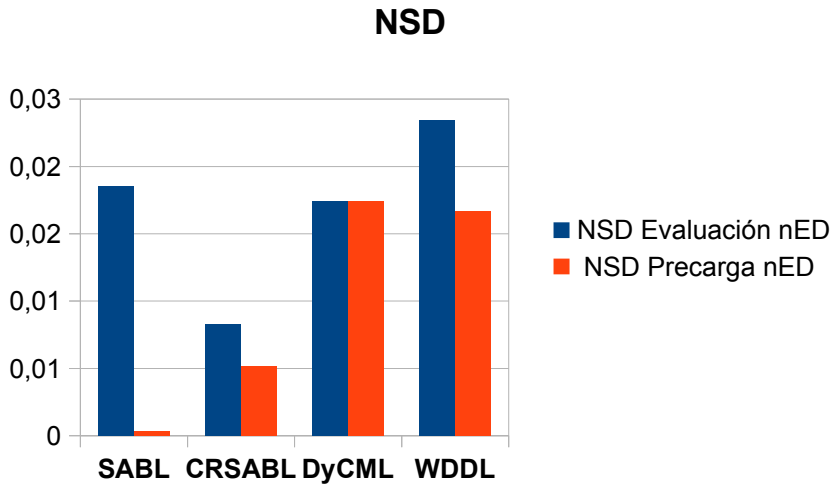
Ahora, vamos a analizar el NED y NSD de cada celda lógica para determinar su nivel de seguridad, nos centraremos en los datos de la fase de evaluación pues es allí donde los datos procesados podrían variar con base a a la variación de las entradas. La tabla 6 y las figura 76-77 nos ofrecen una comparativa de estas métricas.

SBOX PICCOLO	Evaluación	
	NED	NSD
SABL	0,3240	0,0185
CRSABL	0,1569	0,0083
DyCML	0,1377	0,0174
WDDL	0,4019	0,0234

6. Tabla: Comparativa de las métricas NED y NSD



76. Ilustración: Gráfico de barras con el NED de las SBOX



77. Ilustración: Gráfico de barras con el NSD de las SBOX

De las cuatro lógicas estudiadas la DyCML demostró tener los menores valores tanto de NED, le sigue la CRSABL en un factor de x1,14 , la SABL en un factor de x2,35 y la WDDL en factor de X2,92. Con respecto al NSD los resultados de simulación muestran que la CRSABL ofrece menor valor con 0,0083, mientras que le siguen la DyCML con factor de x2,1, la SABL con un factor de x2,23 y la WDDL con factor de x2,83. De las métricas anteriores podemos extraer que el estilo CRSABL demuestra un nivel de seguridad superior al SABL. El estilo DyCML aunque demostró en este trabajo tener superioridad en el consumo energético frente a otras lógicas, es el que muestra resultados mas parejos en cuanto a NED y NSD, mostrando consumo más constante en cada transición y mayor independencia del dato procesado frente a las otras, y por tanto es el que tiene mejor resistencia DPA y por lo tanto, el estilo lógico más seguro de los analizados. Usando el mismo criterio anterior la técnica WDDL demuestra ser la que menor resistencia DPA ofrece y por lo tanto la menos segura de las cuatro.

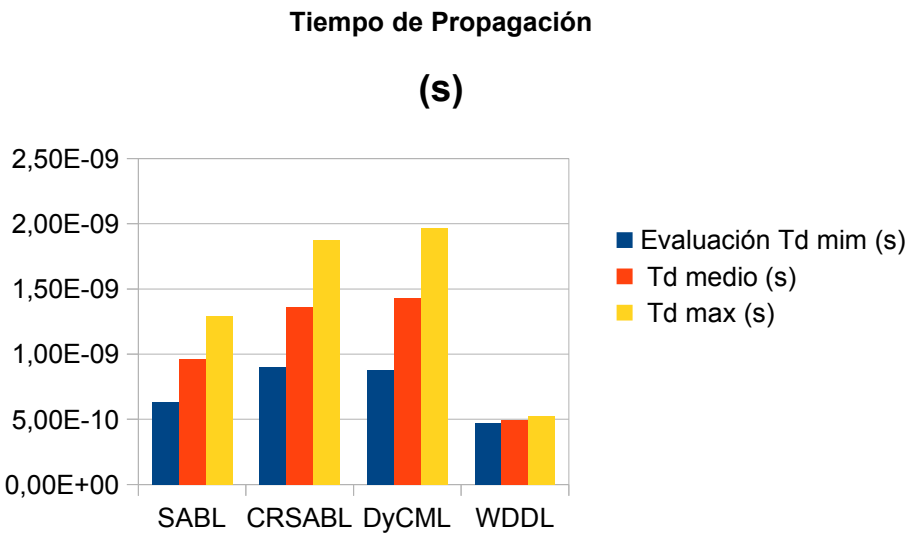
En la tabla siguiente y la figura 78 se presentan los resultados relativos al tiempo de propagación.

SBOX PICCOLO	Evaluación		
	Td mim (ns)	Td medio (ns)	Td max (ns)
SABL	0,6364	0,9591	1,2937
CRSABL	0,8979	1,3633	1,8743
DyCML	0,8809	1,4307	1,9633
WDDL	0,4719	0,4913	0,5236

7. Tabla: Tiempo de propagación en las SBOX

Los resultados muestran que con respecto al tiempo de propagación promedio, la lógica más rápida es la WDDL, pese a tener un consumo de recursos hardware substancialmente superior (mayor numero de transistores), esto tal vez podría asociarse con la pequeña capacidad de

carga que maneja en la señal de control de precarga en relación a las otras técnicas. Esta lógica WDDL evidenció para las compuertas básicas en algunas de las simulaciones realizadas a distintas frecuencias, efectos y errores debido de la evaluación temprana y precarga anticipada, es decir, fruto de la diferencia en los tiempos de llegada de las entradas. A la técnica WDDL Le sigue en rapidez la SABL con un factor de x1,35, luego está la DyCML con un factor de X1,86 y finalmente la más lenta de las 4, la CRSABL con un factor de x1,90.



78. Ilustración: Gráfico de barras comparativo del Tiempo de propagación en las SBOX

A nivel de resumen la tabla 8 nos muestra una comparación de las lógicas en términos de potencia, retraso y el producto Potencia y retraso (Power-Delay product).

SBOX PICCOLO	Potencia media Evaluación (uW)	Tiempo de Propagación – Td (ns)	Potencia*Td PDP (uW*s)
SABL	31,3071	0,9591	30,0281
CRSABL	35,2134	1,3633	48,0071
DyCML	398,5640	1,4307	570,2056
WDDL	94,0381	0,4913	46,2042

8. Tabla: Comparativa de desempeño en las SBOX

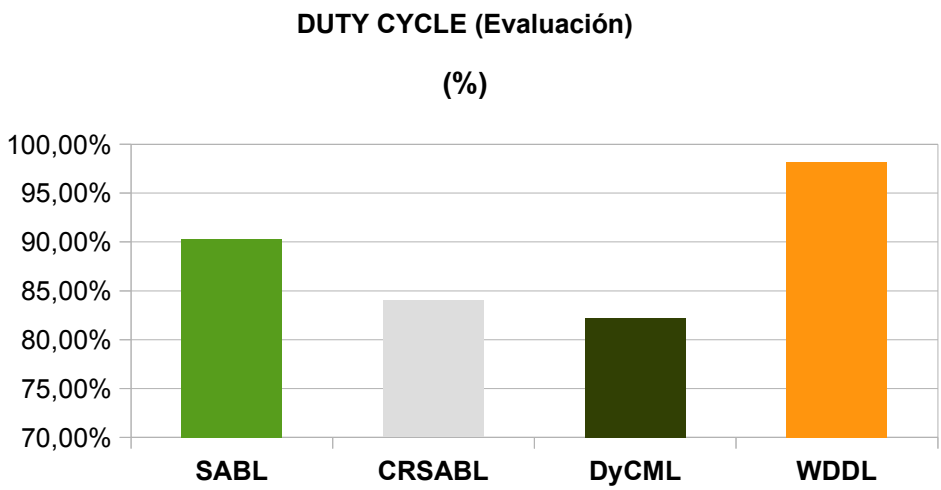
Del PDP en la tabla 8 podemos extraer que la lógica que consume menor potencia por evento de conmutación es la SABL, mostrando con ello ser la mas eficiente energéticamente, le siguen la WDDL que tiene un factor superior de x1,54, la CRSABL con un factor de x1,60 y por ultimo la WDDL con un factor de aproximadamente x19, esta última aunque tiene mejor tiempo de propagación, también es la segunda en consumo de potencia, atrás de la DyCML. Las lógica

SABL, seguida de la CRSABL demuestran ser las más eficientes energéticamente hablando, resultando la CRSABL más segura que la SABL.

Para finalizar analizaremos también el *duty cycle* de la fase de evaluación para cada uno de los estilos lógicos, obtenido de las simulaciones. Los valores extraídos se encuentran condensados en la tabla 9 y la figura 79.

SBOX PICCOLO	DUTY CYCLE
SABL	90,31%
CRSABL	84,00%
DyCML	82,19%
WDDL	98,20%

9. Tabla: Duty Cycle de las Sbox



79. Ilustración: Gráfico de barras con la comparativa del ancho de pulso de las Sbox

En los datos presentados en la tabla y figura anteriores, podemos apreciar que la lógica con el ancho de pulso más estrecho es la DyCML, la cual decrementó el duty cycle en un 17.81%, le sigue la CRSABL cuyo decremento fue del 16%, la SABL en un 9,69% y la WDDL en un 1,8%; siendo está ultima la que exhibe un rango más amplio para la duración del pulso alto. Si bien en este caso el duty cycle no afecta la funcionalidad en ninguna de las lógicas analizadas, es un factor que permite que los circuitos evolucionen durante más tiempo, un detalle que puede ser importante o necesario para algunas aplicaciones.

6. Conclusiones

En este trabajo se han comparado las prestaciones de seguridad, potencia y retraso de cuatro estilos lógicos tipo DLP (SABL, CRSABL, DYCML Y WDDL). La metodología para realizarlo contempló el diseño de una librería de compuertas básicas de 2 entradas (AND/NAND, OR/NOR y XOR/XNOR) para la red DPDN en tecnología UMC 180nm (VDD = 1.8 V) conforme las técnicas SABL, CRSABL, DyCML y WDDL. Para cada una de las técnicas propuestas se implementó un demostrador simple Sbox4-Piccolo y se realizaron simulaciones eléctricas, obteniendo los datos de las trazas de corriente de polarización para procesarlos estadísticamente y evaluar el consumo, retraso y seguridad, esta ultima fue evaluada usando las métricas NED y NSD; la eficiencia energética se midió con la figura PPD.

La resistencia DPA de la técnica DyCML es superior a la de las otras técnicas, de los cuatro estilos lógicos este demuestra ser el más seguro y robusto. Aunque exhibió el peor consumo de potencia, a futuro seria interesante optimizar las dimensiones de los transistores para mejorar las prestaciones de desempeño y eficiencia energética.

La lógica WDDL demuestra ser la más rápida de todas, aunque su superior coste en hardware se ve penalizado en el consumo de potencia. También demuestra ser la mas vulnerable de todas en términos de seguridad. Esta lógica evidencia efectos de evaluación temprana y precarga anticipada, razón por la cual se hace necesario la sincronización de datos a fin de evitar fallos y los efectos de esta propagación.

Las métricas NSD y NED resultan apropiadas en una primera aproximación al diseño de celdas criptográficas seguras, pero se hace necesario incorporar otras métricas, técnicas y simular ataques para determinar con mayor propiedad el grado de seguridad ofrecido por una celda lógica.

Se ha verificado que en los estilos lógicos estudiados del tipo DLP y que utilizan una fase de precarga y señalización complementaria con salidas duales, el consumo de energía no depende de los datos procesados, siempre que las salidas y sus complementarias tengan las mismas cargas capacitivas.

La comparación entre cada técnica analizada, demanda también analizar los compromisos entre las restricciones de área, potencia, velocidad y aplicación.

Bibliografía

[1] L. H. L. Style, M. W. Allam, S. Member, and M. I. Elmasry, “Dynamic Current Mode Logic (DyCML): A New Low-Power High-Performance Logic Style,” J. Solid-State Circuits, vol. 36, no. 3, pp. 550–558, 2001.

[2] K. Tiri, M. Akmal, and I. Verbauwhede, “A Dynamic and Differential CMOS Logic With Signal Independent Power Consumption to Withstand Differential Power Analysis on Smart Cards,” in In Proc. of the 28th European Solid-State Circuits Conference (ESSCIRC’02), 2002, pp. 403–406.

[3] K. Tiri and I. Verbauwhede, “Charge Recycling Sense Amplifier Based Logic: Securing Low Power Security IC’s against DPA,” in In Proc. of the 30th European Conference on Solid-State Circuits (ESSCIR’04), 2004, pp. 179–182.

[4] K. Tiri, I. Verbauwhede, B. Hall, P. O. Box, and L. Angeles, “A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation,” in In Proc. of the conference on Design, Automation and Test in Europe (DATE’04), 2004, pp. 246–251.

[5] K. Shibutani, T. Isobe, H. Hiwatari, A. Mitsuda, T. Akishita, and T. Shirai, " Piccolo: An Ultra-Lightweight Blockcipher", in Cryptographic Hardware and Embedded Systems - CHES 2011. Springer, Berlin, Heidelberg, 2011, pp. 342 357.

[6] M. Albretch, B. Driessen, E. B. Kavun, G. Leander, and T. Yalcin, Block Ciphers: Focus on the linear layer (Feat. PRIDE), 2014, pp. 57 76.

[7] E. Tena-sánchez, J. Castro, and A. J. Acosta, “A Methodology for Optimized Design of Secure Differential Logic Gates for DPA Resistant Circuits,” IEEE J. Emerg. Sel. Top. CIRCUITS Syst., vol. 4, no. 2, pp. 203–215, 2014.

[8] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun, Differential Power Analysis, in Advances in Cryptology - CRYPTO 99. Heidelberg, Germany: Springer-Verlag, 1999, vol. 1666, Lecture Notes in Computer Science, pp. 398–412.

[9] S. Mangard, E. Oswald, and T. Popp, Power analysis attacks: Revealing the secrets of smart cards. Springer Science & Business Media, 2008, vol. 31.

[10] T. Popp, S. Mangard, Implementation aspects of the DPA resistant logic style MDPL, in Circuits and Systems, IEEE International Symposium on, 2910-2916, 2006.

[11] N. Avirneni, A. Somani, Countering Power Analysis Attacks using Reliable and Aggressive Designs, IEEE Transactions on Computers, 1-10, 2013.

[12] Tena Sánchez, E. Diseño y caracterización de criptocircuitos seguros y resistentes a ataques físicos. Tesis Doctoral. Universidad de Sevilla, Sevilla, 2019.

[13] A. J. Acosta, T. Addabbo, E. Tena-Sánchez, “Embedded electronic circuits for cryptography, hardware security and true random number generation: an overview”, International Journal of Circuit Theory and Applications, vol. 45, no. 2, pp. 145-169, 2017

[14] M. W. Allam and M. Elmasry. Dynamic Current Mode Logic (DyCML):A New Low-Power High-Performance Logic Style. In IEEE Journal of Solid-State Circuits, volume 36, pages 550–558. IEEE, 2001

[15] Delgado Lozano, I.M. Diseño microelectrónico de circuitos criptográficos de altas prestaciones y evaluación de su seguridad. (Trabajo Fin de Grado Inédito). Universidad de Sevilla, Sevilla. 2017

[16] Gaudet, Vincent C. (2014-04-01) [2013-09-25]. "Chapter 4.1. Low-Power Design Techniques for State-of-the-Art CMOS Technologies". In Steinbach, Bernd (ed.). Recent Progress in the Boolean Domain (1 ed.). Newcastle upon Tyne, UK: Cambridge Scholars Publishing. pp. 187–212. 2014.

[17] Kim, Hyunjun & Sim, Minjoo & Jang, Kyoungbae & Kwon, Hyeokdong & Uhm, Siwoo & Seo, Hwajeong. (2021). Masked Implementation of Format Preserving Encryption on Low-End AVR Microcontrollers and High-End ARM Processors. Mathematics. 9. 1294. 10.3390/math9111294

Anexos

Código en VerilogA para la generación de estímulos automáticos

```
// VerilogA for TFM, AUTOM_TEST, veriloga
// ARCHIVO: AUTOM_TEST.
// Test automatico para la generacion de entradas de las SBOX
// AUTOR: Dorlin Bonilla Zapata
// FECHA: Octubre 18 de 2021

`include "constants.vams"
`include "disciplines.vams"

module AUTOM_TEST(clk, x0, x0c,x1,x1c, x2, x2c, x3, x3c);
//parameter real freq = 1000 from [0:inf] ;
parameter ciclo = 20; // periodo de clock en (ns) -- 20 ns = 50 MHz
parameter real periodo = 1 ;
parameter integer inicio_rango = 0 ;
parameter integer fin_rango = 7 ;
parameter real  VoltageRail = 1.8 ;
output clk, x0, x0c,x1,x1c, x2, x2c, x3, x3c;
electrical clk, x0, x0c,x1,x1c, x2, x2c, x3, x3c;
integer seed = 0 ;
integer clock_0 ;
integer num_x0, num_x1, num_x2, num_x3;

   analog begin
    @(initial_step)  clock_0 = 0 ;
   //Valores aleatorios de entrada a las Sbox
           num_x0 = abs($random(seed) % 2);  // division por modulo 2 para generar 1 & 0
   if (num_x0 < 0.5)
     begin
       V(x0) <+  0.0 ;
       V(x0c) <+ VoltageRail ;
     end
   else
     begin
       V(x0) <+ VoltageRail ;
       V(x0c) <+ 0.0 ;
     end

  num_x1 = abs($random(seed) % 2);  // division por modulo 2 para generar 1 & 0
  if (num_x1 < 0.5)
     begin
       V(x1) <+ 0.0 ;
       V(x1c) <+ VoltageRail ;
     end
   else
     begin
       V(x1) <+ VoltageRail ;
       V(x1c) <+ 0.0 ;
     end

           num_x2 = abs($random(seed) % 2);  // division por modulo 2 para generar 1 & 0
   if (num_x2 < 0.5)
     begin
```

```
      V(x2) <+ 0.0 ;
      V(x2c) <+ VoltageRail ;
    end
  else
    begin
      V(x2) <+ VoltageRail ;
      V(x2c) <+ 0.0 ;
    end

  num_x3 = abs($random(seed) % 2);  // division por modulo 2 para generar 1 & 0
 if (num_x3 < 0.5)
    begin
      v_x3 = VoltageLow ;
      v_x3c = VoltageHigh ;
    end
  else
    begin
      v_x3 = VoltageHigh ;
      v_x3c = VoltageLow ;
    end
end
end //end for loop

V(clk) <+ transition(clock_0 ? VoltageHigh : VoltageLow, t_rise, t_fall) ;
V(x0) <+ transition(v_x0,  t_rise , t_fall ) ;
V(x0c) <+ transition(v_x0c, t_rise, t_fall ) ;
V(x1) <+ transition(v_x1,  t_rise, t_fall ) ;
V(x1c) <+ transition(v_x1c, t_rise, t_fall ) ;
V(x2) <+ transition(v_x2,  t_rise, t_fall ) ;
V(x2c) <+ transition(v_x2c, t_rise, t_fall ) ;
V(x3) <+ transition(v_x3,  t_rise, t_fall ) ;
V(x3c) <+ transition(v_x3c, t_rise, t_fall ) ;
  end
endmodule
```

Scripts de OCEAN

Scripts de MATLAB

```
%Codigo para realizar el calculo automatico de los parametros estadisticos y consumo electrico
de la SBOX
clear all;
close all;
clc;
key = 1 ;  %15 ;
file = strcat ('ivdd_', num2str(key));
ivdd0 = importdata(file) ;
ivdd =ivdd0(:,2);
n = length (ivdd0);
tiempo =ivdd0((1:length (ivdd0)),1);

 nombre_test = 'TEST SBOX SABL' ;
 figure
 plot(tiempo,ivdd)
 xlabel('Tiempo (ns)')
 ylabel('Corriente (A)')
 title(nombre_test)
 grid on; box off; axis on

%n va a ser el numero de patrones menos uno, porque sino al haber quitado
%las primeras componentes el numero de puntos que va a indexar va a ser
%mayor que el numero de componentes del vector.
%saco a partir de la componente 13 en adelante que es cuando comienza la
%evaluacion y a partir de ahi corto de 7500 en 7500 puntos (voy a tener una
%componente de evaluacion, la siguiente de precarga y asi
%sucesivamente). Calculo la media de intensidad en cada corte, es decir, en
%cada evaluacion y cada precarga y las almaceno en un vector.
for j = 1:n
      i = (j-1)*7500+1:(j-1)*7500+7499;
      avg (1,j) = mean (ivdd(i)) ;
end

%ahora separo las componentes pares de impares, a fin de separar las
%evaluaciones de las precargas. Y calculo todo lo demas con las formulas que tengo.
```

```
avg_evaluation = avg (1:2:length (avg)) ;
avg_precharge = avg (2:2:length (avg)) ;
Eavg_evaluation = 1.2*1e-9*mean (avg_evaluation) ;
Eavg_precharge = 1.2*1e-9*mean (avg_precharge) ;
Pavg_evaluation = 1.2*mean (avg_evaluation) ;
Pavg_precharge = 1.2*mean (avg_precharge) ;

NED_evaluation = (max (avg_evaluation) - min (avg_evaluation))/max(avg_evaluation) ;
NED_precharge = (max (avg_precharge) - min (avg_precharge))/max(avg_precharge) ;
NSD_evaluation = std (avg_evaluation)/mean (avg_evaluation) ;
NSD_precharge = std (avg_precharge)/mean (avg_precharge) ;
Eavg_Total = Eavg_evaluation + Eavg_precharge ;
Pavg_Total = Pavg_evaluation + Pavg_precharge ;

%salida de datos
% avg_out =  [avg_evaluation, avg_precharge, Pavg_Total];
Eavg_out = [Eavg_evaluation, Eavg_precharge, Eavg_Total];
Pavg_out = [Pavg_evaluation, Pavg_precharge,Pavg_Total];
ned_nsd_out = [NED_evaluation, NED_precharge, NSD_evaluation, NSD_precharge];

fid1 = fopen ('data_out_sim_sabl.txt','w');
fprintf (fid1, 'Datos de simulacion\n\n') ;
fprintf (fid1, 'Test de SBOX SABL PICCOLO\n\n');
fprintf (fid1, 'Eavg_evaluation Eavg_precharge Eavg_Total\n');
fprintf (fid1, '%3.5E  %3.5E  %3.5E\n',Eavg_out) ;
fprintf (fid1, '\n\n') ;
fprintf (fid1, 'Pavg_evaluation Pavg_precharge Pavg_Total\n');
fprintf (fid1, '%3.5E  %3.5E  %3.5E\n',Pavg_out) ;
fprintf (fid1, '\n\n') ;
fprintf (fid1, 'NED_evaluation NED_precharge NSD_evaluation NSD_precharge\n');
fprintf (fid1, '%3.5E  %3.5E  %3.5E  %3.5E\n',ned_nsd_out) ;
fprintf (fid1, '\n\n') ;
fclose (fid1);
```