Técnicas de bajo consumo

7 de octubre de 2024

1. Consumo en circuitos CMOS

Los motivos por los cuáles un circuito CMOS consume corriente son:

- Gate Current I_G, por efecto túnel en el gate.
- Subthreshold Current I_{DL}, con $V_{GS} < V_T$.
- Leakage Current I_{DS}, por junturas en inversa.
- Potencia de Conmutación P_{dt}, por la carga de capacidades.
- Potencia Interna P_I, por los microcortocircuitos entre V_{DD} y GND.

Los tres primeros componententes se engloban en lo que se conoce como potencia estática. La potencia estática está dada por:

$P_S = (I_G + I_{DL} + I_{DS}) \cdot V_{DD}$

La potencia de conmutación o dinámica está mayoritariamente dada por:

$P_{dt} = \sum_{i=1}^{N_g} \alpha_i F_{clk} P_i$

Con $P_i = C_{Li} \cdot V_{DD}^2$, α_i la probabilidad de conmutación, C_{Li} la carga de la compuerta, F_{clk} la frecuencia de operación y V_{DD} la tensión de alimentación.

La potencia interna se debe al paso directo de corriente desde la fuente a través de los transistores cuando están conmutando.

2. Tipos de técnicas de bajo consumo

El problema de reducir la potencia consumida por el circuito puede atacarse desde distintos puntos:

- a nivel de proceso
- a nivel de celdas estándar
- a nivel de netlist
- a nivel de microarquitectura
- a nivel de arquitectura

Las 2 primeras opciones atacan el problema tecnológicamente, haciendo énfasis en el consumo estático. Mientras que las demás apuntan a reducir la cantidad de conmutaciones, disminuir el factor de actividad de los circuitos. Cada nivel de abstracción que se intente optimizar para consumo tiene sus ventajas y desventajas y el diseñador no siempre puede modificarlos a todos.

2.1. Nivel proceso

El factor que más fuertemente afecta el consumo es la tensión de alimentación. Por lo tanto, cuanto menor sea, mejor. Podemos ver en la Figura 1 la tendencia de la tensión de alimentación en función del nodo, según la ITRS (International Technology Roadmap for Semiconductors) y su sucesor IRDS (International Roadmap for Devices and Systems).

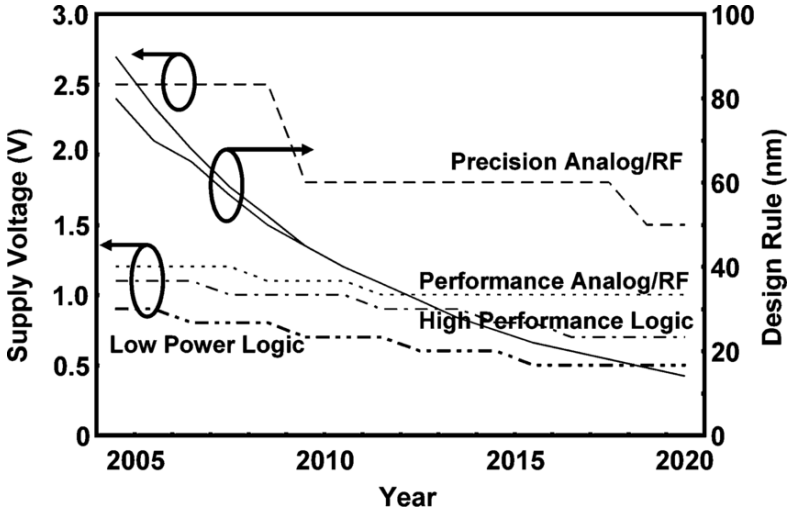


Figura 1: Tendencia de la tensión de alimentación en función del nodo, ITRS

Si reducimos V_{DD} para reducir la potencia, nos vemos obligados a reducir el V_T de los transistores para que no aumente el delay de las mismas. Reducir V_T implica que debemos bajar la corriente de leakage de drain para poder diferenciar entre un 0 y un 1 lógico. Para lograr ambos objetivos se pueden usar varias técnicas:

- Múltiples V_T: distintos tipos de transistores usados en diferentes celdas estándar para usar las mejores para una optimización particular.
- Dieléctricos High-K: disminuye la corriente de leakage, pudiendo bajar V_{DD} y V_T.
- Dispositivos 3D: dispositivos modernos como Fin-FET ó Gate All Around.

2.2. Nivel celdas estándar

En este caso hablamos de modificaciones o variaciones de las celdas estándar para un proceso dado. Esta solución no puede ser implementada directamente por el diseñador físico, ya que este tipo de celdas deben estar disponibles en el PDK utilizado.

2.2.1. Longitud de los transistores

Normalmente, los nodos tecnológicos intentan implementar transistores de L_{min}, por los beneficios a la integración y a la velocidad que ya vimos previamente. Sin embargo, si los transistores que utilizamos tiene $L > L_{min}$, la corriente de leakage disminuye.

2.2.2. Multiples V_T

Si nos fueron provistos transistores de distintos V_T, puedo tener diferentes celdas estándar para diferentes módulos de mi sistema de manera tal de priorizar la velocidad en un caso (menor V_T, más leakage) o el consumo (V_T alto, celda lenta).

2.2.3. Apilamiento de transistores

Si utilizamos un circuito como el de la Figura 2, tenemos la misma funcionalidad que un inversor, pero un comportamiento estático distinto. Logramos que la longitud equivalente del NMOS sea mayor, manteniendo la relación de resistencias equivalentes entre la rama PMOS y la NMOS. Este circuito se comporta como si tuvieramos un transistor cuyo V_T es el doble del que presenta un transistor común.

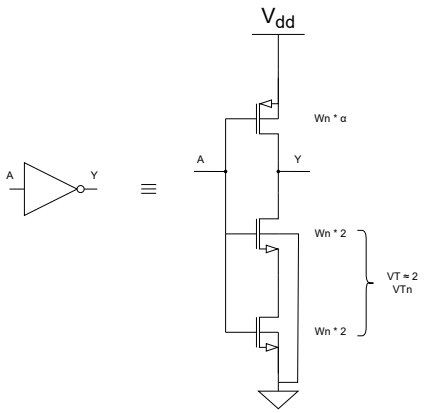


Figura 2: Inversor con NMOS apilados

2.3. Nivel netlist

Hay una serie de acciones que no afectan a la funcionalidad del circuito que pueden alterar su consumo dinámico. Algunas de estas opciones pueden estár implementadas directamente por los algoritmos de síntesis que convierten la descripción RTL en el circuito a nivel de compuertas. Repasaremos algunas de las mismas:

2.3.1. Datapath enable

Podemos realizar operaciónes que nos permitan habilitar o deshabilitar ciertos elementos combinacionales del circuito. Por ejemplo, en la Figura 3 podemos ver como el agregado de algunas compuertas nos permite evitar que ciertos circuitos no utilizados estén conmutando.

(a) (b)

Figura 3: Ejemplo de Netlist modificada por consumo

2.3.2. Reducción de timing hazard

Los timing hazard se producen cuando un cambio en una o más entradas de un circuito lógico produce una fluctuación no deseada en la salida, antes de que la salida se estabilice en su valor lógico final correcto. Estos riesgos son transitorios y se deben a las diferencias en los tiempos de propagación de las señales a través de las diferentes rutas dentro del circuito.

Podemos analizar el siguiente circuito de la Figura 4:

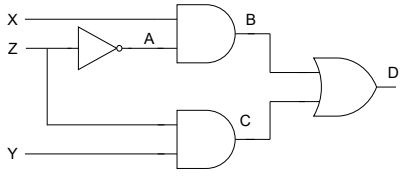


Figura 4: Circuito para analizar Hazards

Las ecuaciones que lo gobiernan con:

$D(X, Y, Z) = X \cdot \overline{Z} + Z \cdot Y$

Podemos ver que para cualquier estado de Z y $X = Y = 1$, la salida es 1:

$D(1, 1, 1) = 0 + 1 = 1$

$D(1, 1, 0) = 1 + 0 = 1$

El problema ocurre cuando hay un cambio en Z, como en el caso de la Figura 5. La sucesión de eventos del circuito puede generar un cambio espúreo en la salida. Esto puede propagarse por todo mi circuito.

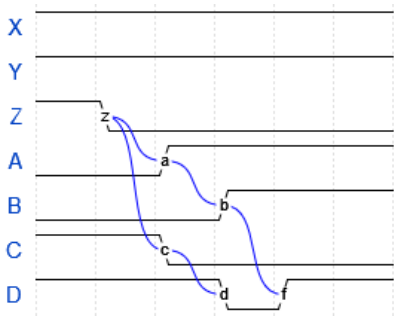


Figura 5: Static Hazard

Static hazard Ocurren cuando la salida debería permanecer constante ante un cambio en la entrada, pero en su lugar se produce una fluctuación momentánea. Los riesgos estáticos se subdividen en dos categorías:

- Static-0: La salida debería permanecer en 0, pero cambia momentáneamente a 1 y luego regresa a 0.
- Static-1: La salida debería permanecer en 1, pero cambia momentáneamente a 0 y luego regresa a 1.

Static-0 hazard Un Static-0 hazard existe si una variable y su complemento se conectan a una misma AND gate, lo cuál no debería ser utilizado.

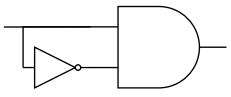


Figura 6: Circuito simple con hazard static-0

Static-1 hazard Un Static-1 hazard puede existir si se conmuta entre 2 minitérminos no solapados. (Minitérmino: producto de todas las entradas o sus negados que da 1 logico. Recordar Mapa de Karnaugh).

Z \ XY	00	01	11	10
0	0	0	1	1
1	0	1	1	0

Figura 7: $D = X \cdot \overline{Z} + Z \cdot Y$

Figura 8: Mapa K con static-1 hazard

Dynamic hazard Ocurren cuando la salida debería cambiar de un valor lógico a otro (por ejemplo, de 0 a 1 o de 1 a 0), pero en su lugar experimenta múltiples transiciones antes de estabilizarse en el valor final. Por ejemplo, si la salida debería cambiar de 1 a 0, un riesgo dinámico podría causar que la salida cambie de 1 a 0, luego a 1 y finalmente a 0, como en la Figura 9.

Este tipo de riesgos nunca se dan en circuitos SOP (Suma de productos: $Z = A \cdot B + C \cdot D$) o POS (Producto de suma: $Z = (A + B) \cdot (C + D)$), sino en circuitos con mas de 2 niveles lógicos, del estilo del que se ve en la Figura 10.

Figura 9: Hazard dinámico

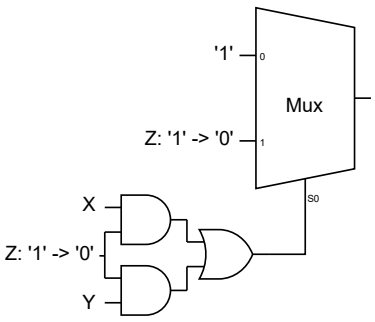
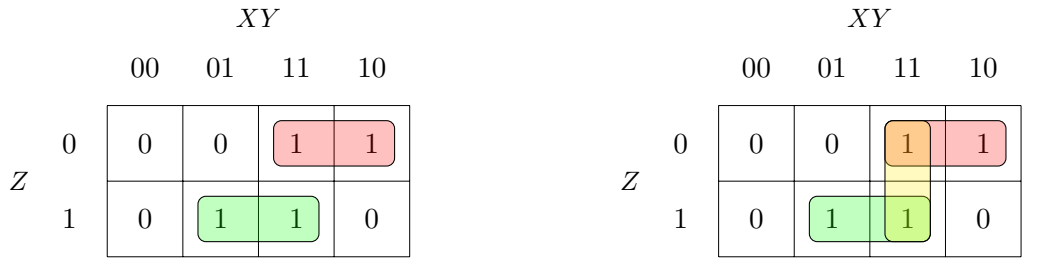


Figura 10: Posible circuito generador de hazard dinámico

Remoción de hazard Una vez que se tiene la síntesis lógica del circuito se hace una restructuración lógica a fin de evitar hazards, generalmente a expensas de incrementar el área. Por ejemplo, un método para eliminar static hazards es generar redundancias en la lógica tomando términos adicionales, si es posible, como en la Figura 11. El costo es una mayor cantidad de compuertas pero no tendremos una conmutación que nos genere consumo.



(a) $D = X \cdot \overline{Z} + Z \cdot Y$ (b) $D = X \cdot \overline{Z} + Z \cdot Y + X \cdot Y$

Figura 11: Término redundante para evitar static-1 hazard

2.3.3. Clock gating

Una de las mayores fuentes de consumo en circuitos digitales son los Flip-Flops. Esto se debe a que cada vez que hay un flanco de clock en su entrada de clock se genera algún tipo de actividad eléctrica. Si bien su operación (D copiado en Q) solo se da con uno de los flancos, el otro flanco también genera actividad en el circuito interno.

Algo que debemos considerar es que no siempre es necesario que el Flip-Flop esté actuando. Por ejemplo, si simplemente tengo que retener un valor hasta que se dé una condición (que podríamos reducirla a una señal de Enable), debo realimentar la salida mediante un multiplexor, como en el circuito de la Figura 12. Tenemos que plantearnos si siempre necesitamos que el FF funcióne.

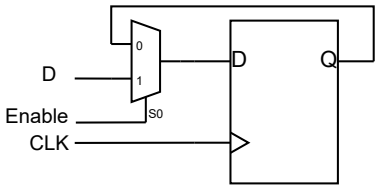


Figura 12: Circuito que retiene el valor, siempre consumiendo

La primer idea que podríamos tener es conectar una compuerta AND en el clock.

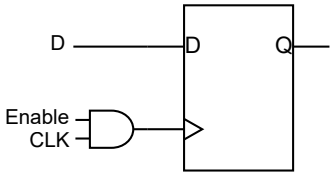


Figura 13: Circuito que deshabilita el FF-D, correctamente?

El problema es que las demoras de propagación que podría haber en generar la señal de control podría generarnos que el tiempo en alto de la señal en el pin de clock del FF sea demasiado corta y/o contenga glitches, como en la Figura 14.

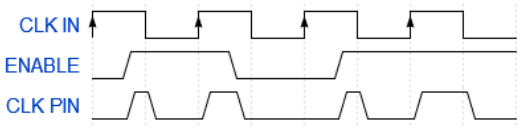


Figura 14: Señal de clock en el pin del FF-D, incorrecta

Por lo tanto, se utilizan celdas especializadas para realizar esta actividad.

ICG: Integrated Clock Cell Un Integrated Clock Cell (ICG) es una celda estándar presente en la mayoría de los PDK modernos. Su funcionalidad es removerle el clock a un FF de a ciclos completos, libre de cualquier tipo de glitch. Están construidos con un latch negativo y una compuerta AND. Se puede ver su símbolo, su circuito, su esquema equivalente en la Figura 15 y su diagrama de tiempos en la Figura 16.

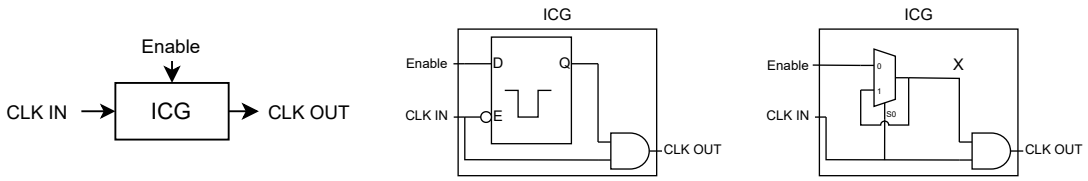


Figura 15: Simbolo, circuito y esquema lógico de un ICG

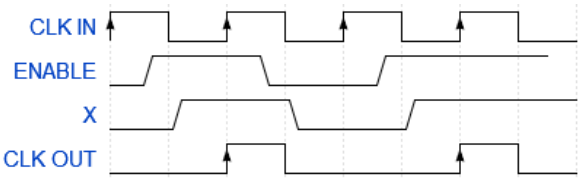


Figura 16: Formas de onda de un ICG

El ICG es tan útil y ominipresente en los circuitos digitales que no es necesario agregarlos manualmente en el RTL. Los sintetizadores tienen algoritmos que detectan la posibilidad de apagar el clock de un FF en base al estilo en el cuál el código fue escrito. Una manera estándar de codificar HDL para inferir ICG es la siguiente:

```
    always @(posedge clk or negedge rst) begin
      if (rst) begin
        q <= 1'b0;
      end
      else begin
        if (enable) begin
          q <= d;
        end
      end
    end

    endmodule
```

Listing 1: HDL para Flip-Flop con Enable detectable ICG

Podríamos tranquilamente interpretar un FF realimentado con un Mux cuya señal de control sea enable, o utilizar esta nueva celda controlada por esta misma señal:

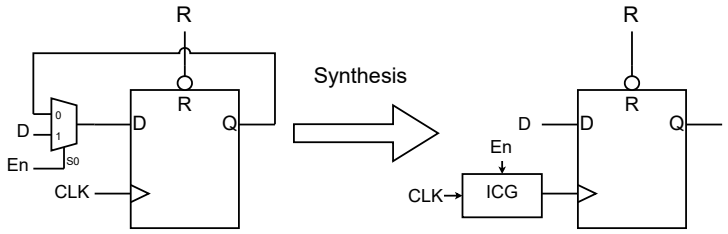


Figura 17: Conversión del sintetizador

Una de las dudas que nos podría surgir es si perdemos área como contraparte de ahorrarnos conmutaciones, debido a que la combinación de un latch con una AND es mayor en tamaño a un multiplexor. Sin embargo, normalmente utilizamos bancos de FF-D en forma de registros que se activan todos con una sola celda de ICG, como en la Figura 18. Por lo tanto, solemos ahorrar área al usar celdas ICG.

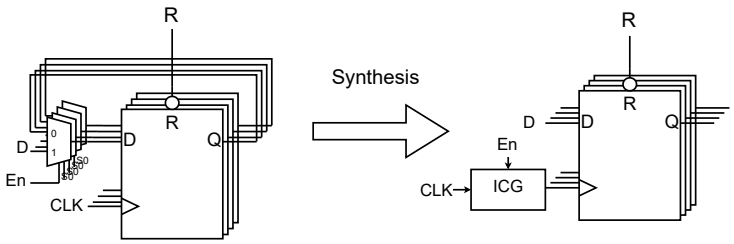


Figura 18: Multiples registros, un solo ICG

En el caso de Genus, el sintetizador de Cadence, alcanza con una configuración muy simple para agregar clock gating a nuestro diseño, manejando el tamaño de nuestro banco de FFs controlados por cada ICG:

```
# Activar clock gating
set_attribute lp_insert_clock_gating True /

# Maxima cantidad de FF-D comandables por un ICG evitando excesivo CL
# elegimos 50
set_attribute lp_clock_gating_max_flops 50 /designs/$toplevel_name
# Minima cantidad de FF-D que justifican un ICG
# elegimos 3
set_attribute lp_clock_gating_min_flops 3 /designs/$toplevel_name
```

2.3.4. XOR Gating

En el análisis previo nos preguntamos si el FF debía funcionar por cuestiones externas, pero no nos hicimos ninguna pregunta sobre si vale la pena a nivel lógico que haga su operación. Dicho en otras palabras, no tiene sentido memorizar un dato nuevo si el dato es el mismo que ya tengo guardado.

Supongamos un ejemplo para clarificar, si estoy muestreando una señal que está en 0 y mi Q está en 0, no podríamos encontrar alguna manera simple de encender el FF si y solo si el dato en D ahora es un 1?

Basados en esta idea, podríamos encontrar que la función de activación del FF-D seria que D y Q sean distintos. Dicho en funciónes lógicas, que no estén los dos en 0 o los dos en 1, tener una cantidad impar de unos. Podemos asociar dicha operación a la compuerta XOR:

D	Q	Operación	XOR
0	0	Retener	0
0	1	Capturar	1
1	0	Capturar	1
1	1	Retener	0

De esta manera, el circuito de control del ICG se vería como el siguiente:

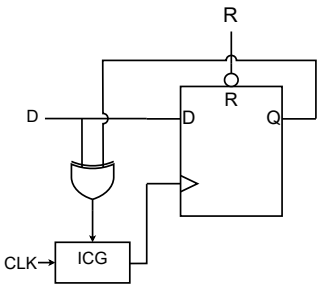


Figura 19: XOR Gating

Una desventaja evidente del esquema es que tengo una compuerta más por cada FF, además del ICG. Se puede utilizar una solución intermedia donde para un banco de registros se utilice un solo ICG y varias XOR conectadas a una compuerta OR. De esta manera actualizo el banco de FF si alguno debe operar, lo cuál no es lo mas eficiente en consumo, pero tampoco es lo más grave en uso de área. Siempre hay un tradeoff.

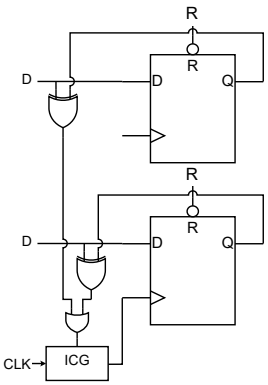


Figura 20: XOR Gating

2.3.5. Power Gating

Una manera mas compleja de ahorrar consumo en circuitos CMOS es apagarlos por completo. Si desconectamos la alimentación de las celdas, las mismas no tendrán ningun tipo de consumo. Esto se puede lograr identificando partes de mi sistema que no tienen porqué estar encendidas en ciertos momentos. El uso de celdas especiales, conocidas como Power Switches, con transistores de paso de muy baja resistencia puede permitirnos hacer esta actividad, que no son tán estándar como los ICG y son considerablemente mas complejas de inferir y ubicar físicamente.

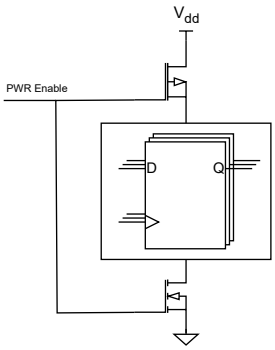


Figura 21: Power Gating

Hay normas estándar para poder realizar este tipo de operaciónes (formatos CPF e IEEE 1804/UPF). Debemos tener en claro la consecuencia de retirarle la alimentación a una parte del sistema: la información almacenada en los FF se perderá. Volveremos sobre esto mas adelante.

2.3.6. Voltage Scaling

Como ya nombramos antes, uno de los componentes principales del consumo está asociado a la tensión de alimentación. La misma puede ser utilizada como una herramienta de diseño del sistema para disminuir la potencia. Esto se puede realizar de forma fija (estáticamente) o variable (dinámicamente).

Static Voltage Scaling: SVS Cuando diseñamos nuestro sistema, elegimos que el mismo opere a una tensión menor que la tensión nominal. Esto se puede hacer si y solo si tenemos seguridad que la peor condición posible de corner PVT (SS, baja tensión, alta temperatura) no es alcanzable de ninguna manera.

Por ejemplo, si nuestro sistema tiene un V_{DD} nominal de 1.8V, lo ponemos a trabajar a 1.6V, solamente porque está diseñado para operar hasta 1.5V y tenemos total seguridad que nunca se alcanzará esa condición.

Muy probablemente este tipo de operaciónes solo podrán ser logradas mediante un regulador integrado del cuál tengamos mucha seguridad sobre su operación.

Dynamic Voltage Scaling: DVS En lugar de plantear que trabajamos a una tensión menor de forma fija, podemos complejizar la situación y modificar la tensión del regulador dinámicamente en función del estado del sistema. Sabemos que los circuitos CMOS consumen más con mayor tensión, pero tambien que funcionan más rápido. Si tenemos momentos en los cúales la velocidad es crítica y otros donde debemos intentar maximizar la duración de una batería, debemos regular la operación de nuestro regulador.

Este tipo de operaciónes está reservada para circuitos de muy alta complejidad, que seguramente estén corriendo algún tipo de software embebido que detecte las distintas situaciones y tenga la capacidad de afectar el regulador del chip.

2.3.7. Frequency Scaling

Otro de los componentes de mi consumo es la frecuencia del sistema. Si la misma puede ser afectada, es otra variable a utilizar para bajar el consumo.

Static Frequency Scaling: SFS De una manera similar a SVS, hacemos trabajar el circuito a una frecuencia menor a la frecuencia nominal para la cuál fue diseñado haciendo algún tipo de configuración sobre el oscilador. Es una técnica muy poco utilizada, debido a que ajustar el timing de los circuitos sincrónicos suele ser una de las mayores restricciones de diseño y es extraño tener margen para reducir la frecuencia y que el sistema siga funcionando correctamente.

Dynamic Frequency Scaling: DFS Esta técnica es más utilizada que SFS. En este caso, subimos o bajamos dinámicamente la frecuencia de oscilación del sistema para cumplir nuestro objetivo de velocidad y/o consumo dependiendo del escenario actual de operación. Suele utilizarse simultáneamente con DVS, por lo que en general se habla de Dynamic Voltage and Frequency Scaling (DVFS), donde tanto el regulador como el oscilador son modificados en tiempo de ejecución según las necesidades del sistema. Los dispositivos a batería con funciónes de StandBy/Sleep son ejemplos típicos de uso intensivo de estas técnicas.

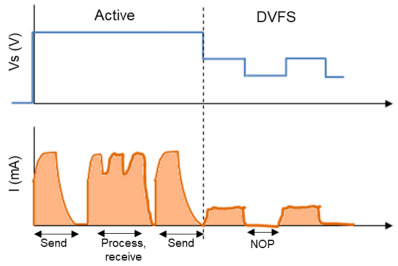


Figura 22: Ejemplo de variación de consumo en sistemas DFVS

2.3.8. Body Bias

Finalmente, la última técnica que tenemos para regular el consumo afectando a la netlist con elementos no funcionales es Body Bias. Como sabemos, el V_T de los transistores depende de la tensión de Body de los mismos. Normalmente en dispositivos CMOS todos los transistores PMOS tienen el body a V_{DD} y los transistores NMOS el body a GND. Sin embargo, esto no es obligatorio. Si tenemos las celdas correspondientes en el PDK, podemos regular la tensión de body para tener distintos V_T utilizando el mismo tipo de transistor y con eso hacer algo similar a correcciones a nivel de proceso.

El caso mas extremo de Body Bias se da en los circuitos de tecnología SOI, donde tenemos la posibilidad de regular la tensión de body de distintos grupos de transistores más sencillamente.

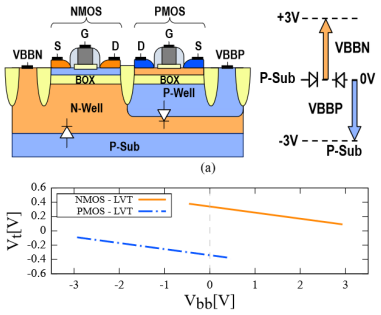


Figura 23: Ejemplo de variación V_T con la tensión de Body

2.4. Nivel microarquitectura

Cuando nos referimos a micro arquitectura, nos referimos a diseñar nuestros circuitos de forma tál que logren la menor cantidad de conmutaciones posibles. Esto dependerá de cada circuito que estemos diseñando y la importancia de la relación área-consumo para nosotros.

2.4.1. Buffer circular

Un ejemplo típico es la utilización de buffers circulares como reemplazo de circuitos tipo FIFO. Supongamos que tenemos un filtro FIR que necesita almacenar las últimas 16 muestras de N bits. Si implementamos una FIFO en cada ciclo de clock tendremos 16*N flops propagando sus datos y consumiendo energía. Si nuestros datos son más lentos que el clock, esto será menor, pero cada vez que tengamos un dato nuevo igualmente moveremos todos los registros.

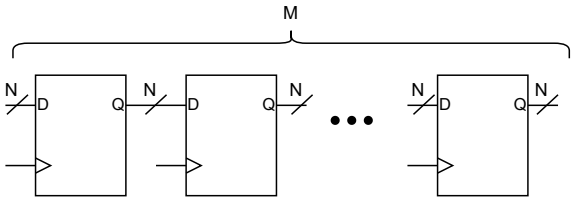


Figura 24: FIFO de M etapas y N bits, muchas conmutaciones

Sin embargo, solo tenemos un dato nuevo y un dato a descartar. Podemos reemplazar nuestra FIFO por un buffer circular, como el de la Figura 25. El mismo necesita mas registros y compuertas debido a la lógica de control y ruteo de señales, pero su cantidad de conmutaciones es mucho menor.

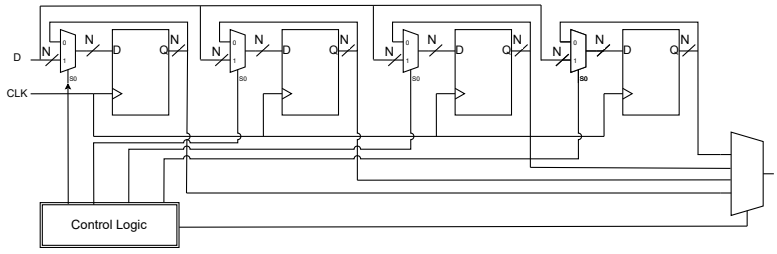


Figura 25: Buffer circular M etapas y N bits, menos conmutaciones

Realizar este tipo de análisis sobre la implementación de cada uno de los bloques puede ser una fuente de ahorro de energía muy importante e independiente del proceso y el sintetizador utilizado.

2.5. Nivel arquitectura

2.5.1. Multiples dominios de reloj

Hasta ahora siempre hemos trabajado con un esquema de un solo clock. Tomando una serie de precauciones, se puede hacer trabajar distintas partes de circuito con distintas velocidades. Podemos hacer trabajar la parte del diseño que no necesita alta velocidad de procesamiento con un reloj mas lento. Este tipo de sistemas se pueden realizar sobre sistemas sincrónicos como asincrónicos.

Sistemas sincrónicos Si analizamos el esquema de la Figura 26a, tenemos un clock mas lento que es función del clock mas rápido, por lo tanto todas las señales estarán naturalmente sincronizadas. Se deben tomar precauciones sobre la lectura de las señales rápidas con el clock lento, ya que si tenemos cambios en flancos del clock rápido pero todavía no tuvimos ningún flanco de clock lento, esos datos no son capturados.

Los divisores de clock son una herramienta perfectamente válida, pero utilizando los conceptos de clock gating que trabajamos antes, podemos poner un contador que genere la habilitación del ICG, como se ve en la Figura 26b.

(a) (b)

Figura 26: Circuitos con múltiples dominios de clock

Sistemas asincrónicos En el caso de sistemas asincrónicos, los osciladores de nuestro clock son independientes. Por lo tanto aparece lo que se conoce como un Clock Domain Crossing. La comunicación entre ambos bloques no es trivial y requiere de lógica circuital adicional, pero no son raros los esquemas como el de la Figura 27.

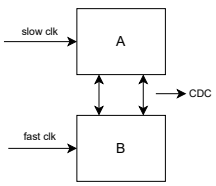


Figura 27: Circuitos asincrónicos entre si, con Clock Domain Crossing

2.5.2. Global clock gating

Si lo sistemas siguen creciendo, podemos pensar en unidades de management que se encargen de decidir que bloques deben recibir clock y cuales no en función de la operación que está realizando el circuito. En este caso seguiré pagando potencia estática, pero no tenemos las complicaciones asociadas a power gating.

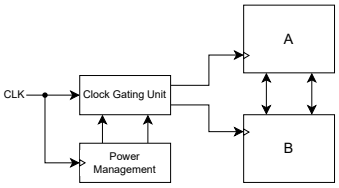


Figura 28: Circuito con unidad global de control de clock

2.5.3. Multiple dominios de potencia

Volviendo al tradeoff velocidad vs. consumo, podemos elegir ciertos bloques que no necesitan alta velocidad y hacerlos trabajar con un V_{DD} mas bajo, mientras que otros bloques trabajan con un V_{DD} mayor. O simplemente podemos tener definidos cuáles se encienden en cada momento dependiendo de lo que estoy haciendo.

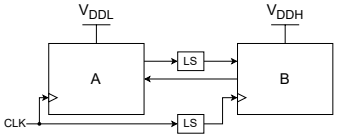


Figura 29: Circuito con múltiples dominios de tensión

Los problemas aquí aparecen en el intercambio de información entre distintos dominios con diferente tensión de alimentación, que pasa con la información en el dominio apagado, que ocurre en las interfaces de un dominio apagado con uno encendido y como recorremos electricamente distancias a través de dominios apagados.

3. Multiple Power Domains

Construir un sistema con múltiples dominios de potencia/tensión no es simple, repasaremos cuales son los elementos necesarios.

3.1. Elementos Circuitales

Level Shifter Si tenemos que enviar un dato desde el dominio de V_{DDH} al de V_{DDL}, no hay problemas. Pero si intentamos activar una compuerta alimentada con V_{DDH} con tensión V_{DDL}, el transistor PMOS no se va a apagar nunca.

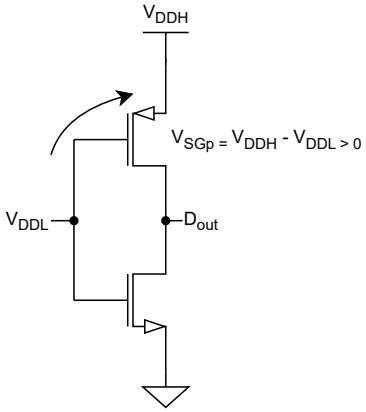


Figura 30: Problemas con múltiples tensiones

La solución para dicha problemática es utilizar celdas Level Shifter. Las mismas utilizan ambas tensiones de alimentación para convertir un dato recibido en una tensión y lo propaga en la otra. Un circuito típico se puede ver en la Figura 31.

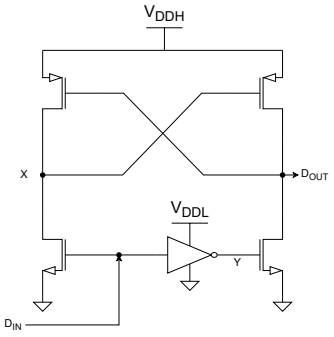


Figura 31: Circuito Level Shifter, cambia la tensión de una señal

- Si $D_i = 0$, $Y = V_{DDL} => D_{out} = 0 => X = 1$
- Si $D_i = 1$, $Y = 0 => X = 0 => D_{out} = V_{DDH}$

Isolation cells Si en nuestro esquema uno de los dominios de tensión se apaga, independientemente de si tenemos la misma tensión o no, debemos analizar que ocurre en las interfaces.

Supongamos que del dominio apagado sale una señal que leemos en el dominio encendido, osea que ingresa en una compuerta. No veremos un valor de tensión que represente un 0 o un 1, ya que tendremos un nodo de alta impedancia. Por lo tanto, debemos aislar nuestro dominio encendido de los datos corruptos.

Para ello, se usan unas compuertas muy simples, las Isolation Cells. Las mismas tienen una entrada de datos del dominio apagable y una entrada de control para ignorar el dato o no, como se ve en le Figura 32. Otro punto clave es saber que dato estable quiero mientras ignoro la entrada de datos, por lo tanto estas celdas vienen en 2 versiones: aíslo y presento un 0(ISOL), y aíslo y presento un 1(ISOH), como se ve en la Figurea.

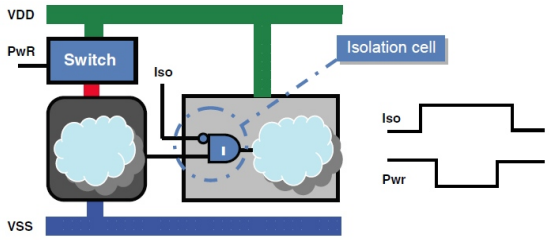
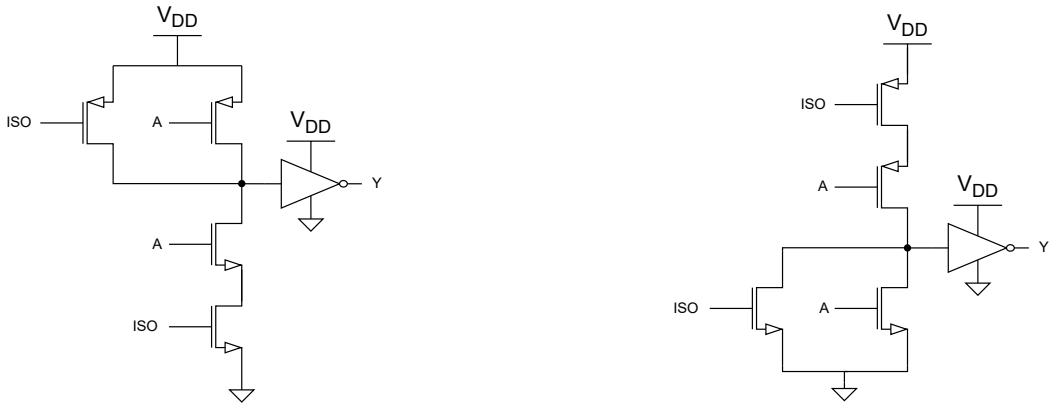


Figura 32: Esquema de aislación de circuitos apagados



(a) ISO L (b) ISO H

Figura 33: Circuitos de Isolation Cells

Retention cell Si en nuestro esquema uno de los dominios de tensión se apaga, quizás necesitamos retener parte de esa información. Entonces, además de los power switches, tenemos que tener FF-D con alimentación secundaria/siempre encendida. Los mismos son conocidos como state-retention cells, su esquema se ve en la Figura 34 y su diagrama de tiempos en la Figura 35.

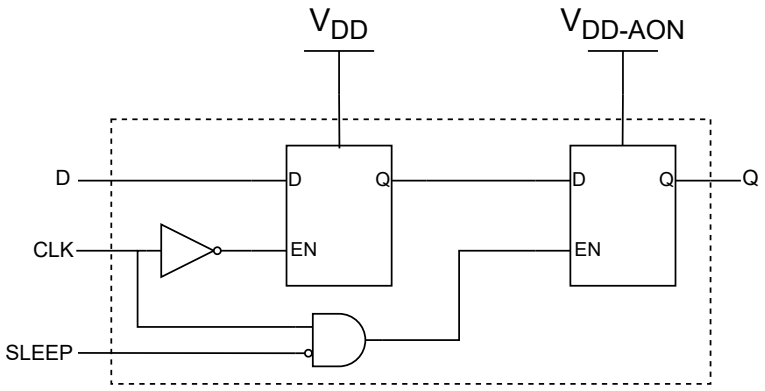


Figura 34: Retention flop

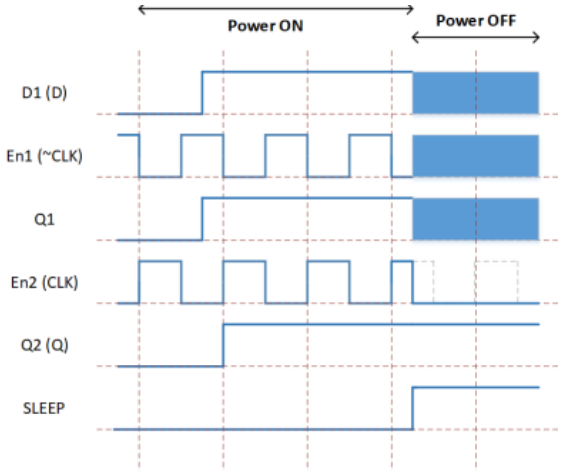


Figura 35: Esquema de señales del retention flop

Always-on buffers Finalmente, existe la posibilidad de tener que llevar señales a lo largo de un dominio de tensión grande a nivel de área, pero que está apagado. Para ello se utilizan buffers que tienen pines de alimentación secundarios, conectados a la tensión siempre encendida.

Para poder implementar este tipo de soluciones debemos tener disponibles en nuestro PDK este tipo de celdas. Debemos considerar que estos circuitos aportan una demo-

ra en la propagación de datos, que debemos considerar al realizar el timing analysis correspondiente.

3.2. Unified Power Format

El Unified Power Format (UPF) es un estándar clave definido en la norma IEEE 1804 para el diseño de baja potencia en circuitos integrados. UPF permite la descripción y gestión de estrategias de ahorro de energía, como la conmutación de potencia y la reducción de voltaje, de forma independiente de las herramientas de diseño. Esto facilita la portabilidad del diseño entre diferentes flujos de trabajo, la abstracción de la implementación de las estrategias y la colaboración entre equipos de diseño.

UPF se utiliza para especificar los modos de potencia y las reglas de control para diferentes bloques del circuito. Permite a los diseñadores definir cómo se comportará el sistema en diferentes estados de energía, lo que es fundamental para optimizar el consumo de energía en dispositivos electrónicos modernos.

Los elementos a definir son:

- Supply Ports: puertos de energía
- Supply Nets: conexiones de energía
- Supply Sets: conjunto de conexiones de energía
- Power Domains: dominios de potencia, asociados a un Supply Set
- Power State: estado posible de un Power Domain
- Port Attributes: asoción de puertos a un Supply Set
- Isolation Strategies: como aislar Power Domains
- Power Switches: llaves de Supply Nets

A continuación tenemos un pequeño ejemplo de como aislar 2 bloques que tienen cada uno su regulador externo (los power switches no son parte del sistema digital):

```
1    # UPF version
2    upf_version 2.1
3
4    # Design top level
5    set_design_top <block_name>
6
7    # Power domains definition and scope
8    create_power_domain <ONOFF_DOMAIN> -include_scope
9    create_power_domain <AON_DOMAIN>   -elements {<hierarchy elements>}
10
```

```
11    #Define supply ports and nets
12    create_supply_port <ONOFF_VDD_PORT>
13    create_supply_net <ONOFF_VDD_NET>
14    connect_supply_net <ONOFF_VDD_NET> -ports <ONOFF_VDD_PORT>
15
16    create_supply_port <AON_VDD_PORT>
17    create_supply_net <AON_VDD_NET>
18    connect_supply_net <AON_VDD_NET> -ports <AON_VDD_PORT>
19
20    create_supply_port <ONOFF_GND_PORT>
21    create_supply_net <ONOFF_GND_NET>
22    connect_supply_net <ONOFF_GND_NET> -ports <ONOFF_GND_PORT>
23
24    create_supply_port <AON_GND_PORT>
25    create_supply_net <AON_GND_NET>
26    connect_supply_net <AON_GND_NET> -ports <AON_GND_PORT>
27
28    # Connect supply nets to power domain supply sets
29    create_supply_set <ss_ONOFF> -function {power <ONOFF_VDD_NET>} -function {
          ground <ONOFF_GND_NET>}
30    create_supply_set <ss_AON>   -function {power <AON_VDD_NET>} -function {
          ground <AON_GND_NET>}
31
32    # Define power state
33
34    if { !$verification } {
35      add_power_state <ss_AON>     -state on  {-supply_expr {power == `{FULL_ON,
            1.8} && ground == `{FULL_ON,0.0}}}
36      add_power_state <ss_ONOFF>  -state on  {-supply_expr {power == `{FULL_ON,
            1.8} && ground == `{FULL_ON,0.0}}} \
37                                  -state off {-supply_expr {power == `{OFF} &&
                                        ground == `{FULL_ON,0.0}}}
38    } else {
39      add_power_state <ss_ONOFF>  \
40                -state {ON_MODE -simstate NORMAL \
41                  -logic_expr {!<power_down_signal>} \
42                  -legal } \
43                -state {OFF_MODE -simstate CORRUPT \
44                  -logic_expr {<power_down_signal>} \
45                  -legal }
46    }
47
48    # Create Power Domains
49    create_power_domain <ONOFF_DOMAIN> \
50                        -update \
51                        -supply {primary <ss_ONOFF>}
52
53    create_power_domain <AON_DOMAIN> \
54                        -update \
55                        -supply {primary <ss_AON>} \
56
57    set_port_attributes -ports {
58                               <AON_ports_list>
59                               } \
60                        -driver_supply <ss_AON>
61
62    # Isolation strategies
63    set_isolation <RULE_ISO1>  -domain <AON_DOMAIN> \
64                           -elements { <signals_to_isolate>
65                                     }   \
66                           -isolation_signal {<isolation_signal_1>} \
67                           -isolation_sense high \
```

```
68                                  -isolation_supply_set <ss_AON> \
69                                  -clamp_value 0
70
71      set_isolation <RULE_ISO2>   -domain <AON_DOMAIN> \
72                                  -source <ss_ONOFF> \
73                                  -isolation_signal {<isolation_signal_2>} \
74                                  -isolation_sense high \
75                                  -isolation_supply_set <ss_AON> \
76                                  -clamp_value 0
```

Como se puede apreciar, este UPF sirve para definir 2 dominios de tensión e incluir isolation cells entre ellos, así como podria incluir level shifters si las tensiones fueran distintas.

El UPF posse múltiples comandos y configuraciones que pueden explorarse en el estandar.

4. IR Drop

Si bien hasta ahora estuvimos analizando técnicas de bajo consumo solo enfocándonos en consumir menos energía, no es el único motivo por el cuál debemos preocuparnos por la potencia. Las condiciones de operación esperadas pueden verse afectadas por el comportamiento de mi circuito bajo carga.

Daremos algunas definiciones para analizar:

Power Grid Es el conjunto de conexiones (wires, cables, metalizaciones) que transportan la tensión desde la alimentación hasta cada compuerta. En general es un entramado de conexiones que comienzan en los metales más altos (su sección es mayor) que presentan menor resistencia y termina en el metal de mas bajo nivel que está conectado a las celdas estándar.

IR Drop Es la reducción (Drop) en el voltage de alimentación que ocurre a través del power grid, debido a la corriente que circula (I) y la resistencia (R) de los metales.

Ground Bounce Es el incremento en la tensión de GND que ocurre a través del power grid.

4.1. Problema

Con esto en mente, podemos analizar la situación:

- La power grid está hecha de metales, que tienen una resistencia.
- Las compuertas tienen un consumo de corriente, que tiene que atravezar la power grid de camino a la compuerta y de retorno por GND.
- La tensión efectiva que verá el source de un transistor PMOS será menor a V_{DD}.
- La tensión efectiva que verá el source de un transistor NMOS será mayor a GND.
- La tensión total de la compuerta puede ser menor a la de mi corner PVT de caracterización.

Un esquema circuital sería algo como el circuito de la Figura 36:

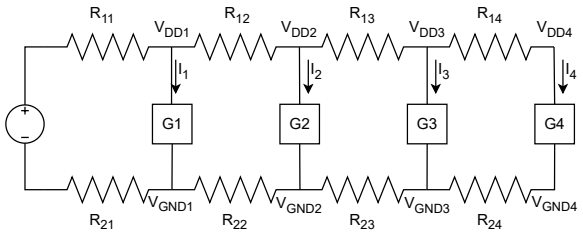


Figura 36: IR Drop

Calculando las tensiones alrededor de la compuerta más lejana:

$V_{DD4} = V_{DD} - [I_1 \cdot R_{11} + I_2 \cdot (R_{11} + R_{12}) + I_3 \cdot (R_{11} + R_{12} + R_{13}) + I_4 \cdot (R_{11} + R_{12} + R_{13} + R_{14})]$

$V_{GND4} = I_1 \cdot R_{21} + I_2 \cdot (R_{21} + R_{22}) + I_3 \cdot (R_{21} + R_{22} + R_{23}) + I_4 \cdot (R_{21} + R_{22} + R_{23} + R_{24})$

Podemos concluir que la compuerta $G4$ está alimentada por una tensión $V_4 = V_{DD4} - V_{GND4}$, lo cuál implica que $V_4 < V_{DD}$.

Esta situación tiene efectos muy indeseados que pueden afectar no ya a la performance del circuito, sino a su funcionamiento.

- Aumento del retardo de propagación: una compuerta que tiene una tensión menor es más lenta.
- Jitter de clock: los buffers que propagan el clock por todo el sistema generan distintas demoras dependiendo de la tensión que reciba cada uno, generando modificaciones en el clock como función de la ubicación del buffer en el diseño.

4.2. Soluciones

No hay solución perfecta para este problema, pero se pueden mitigar sus efectos y asegurarnos que los mismos no comprometan el funcionamiento del sistema.

Power Grid robusta Al momento de diseñar las metalizacions que llegan a todas las compuertas, se sobredimensionan los anchos de los metales, se agrega redundancias de vias, se intenta que la grid tenga multiples caminos en paralelo que disminuyan la resistencia, entre otras técnicas.

Caracterización Robusta Al momento de realizar los cálculos de timing, utilizo caracterizaciones de las celdas que tengan un margen de seguridad en el nivel de tensión.

Una manera típica es realizar mis cálculos asumiendo que la tensión puede ser +-10 % respecto de la tensión nominal de mi regulador. Si estoy seguro que mi regulador bajo carga no va a caer y que mi power grid nunca me va a generar tanta caída como para llegar a ese 10 % para todos los escenarios, puedo tener seguridad de que no tendré violaciones de timing.

Capacitores de Desacople La caída de tensión se debe a que cuando transicionan las celdas tiene que circular corriente desde el regulador. Una manera de mitigar el efecto es proveer la corriente desde un capacitor que esté físicamente cerca de las celdas. Para esto los PDKs cuentan con una celda estándar que utiliza transistores NMOS y PMOS conectados de forma tál de agregar un par de capacitores en parelelo entre V_{DD} y GND. El circuito lo podemos ver en la Figura 37. Estas celdas se las conoce por su nombre en inglés de Decoupling Capacitor, o DCAP. Vienen en distintos tamaños, que ocupan distintas áreas y aportan distinta cantidad de capacidad.

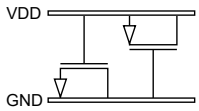


Figura 37: Celda DCAP, arreglo de MOS como capacitores

5. Power analysis

Se puede realizar simulaciones para verificar el consumo de potencia, tanto promedio como instantáneo. Dichas simulaciones podrían ser a nivel de SPICE, pero tenemos modelos en nuestra caracterización que nos permiten acelerar el proceso y reducir su costo computacional.

En general, se realizan 2 tipos de análisis, dependiendo de si tenemos total conocimiento sobre el comportamiento del circuito en operación o si tenemos un estimado.

Otra variable a analizar es el estado actual del diseño. No tiene sentido hacer un análisis profundo de potencia e IRDrop si todavía no tengo el diseño placeado y solo tengo la síntesis lógica, por ejemplo.

5.1. Vectorless Power Analysis

En este caso no sabemos cuantas veces en nuestro período de análisis las celdas van a conmutar. Esto se puede deber a que el diseño todavía no fue simulado logicamente por completo o que estamos en una etapa temprana del diseño.

Para calcular el consumo seguimos los siguientes pasos:

1. Se asume que cada celda tiene una probabilidad de switching α_i (tipicamente $\alpha_i = 20\,\%$)
2. De la caracterización de la celda, conocemos la potencia de switching, de donde se despeja I_{DD} para cada celda
3. Sumamos la I_{DD} leakage, tambien dato
4. Se calcula $\alpha_i \cdot I_{DD}$ y la resistencia de la power grid $R_{powergrid}$ para cada celda, considerando que $R_{powergrid}$ es función de la ubicación de la celda
5. Se calcula la caída de tensión para cada celda
6. Verificamos que todas las celdas tengan un valor de alimentación mayor a la tensión de alimentación nominal menos un $10\,\%$

5.2. Vector Based Power Analysis

En este caso, no trabajaremos con suposiciones sobre la actividad de las celdas. En base a simulaciones lógicas realizadas sobre el circuito, podemos conocer exactamente la probabilidad de conmutación de cada net del circuito. Para realizar dichas simulaciones

se usan unos estímulos externos que varían en el tiempo, que están en formato de vectores, por lo tanto se habla de consumo basado en vectores.

Para calcular el consumo basados en vectores:

1. Realizamos una simulación y anotamos todas las conmutaciones.
2. Calculamos el α_i para cada net del sistema
3. De la caracterización de la celda, conocemos la potencia de switching, de donde se despeja I_{DD} para cada celda
4. Sumamos la la I_{DD} leakage, tambien dato
5. Se calcula $\alpha_i \cdot I_{DD}$ y la resistencia de la power grid $R_{powergrid}$ para cada celda, considerando que $R_{powergrid}$ es función de la ubicación de la celda
6. Se calcula la caída de tensión para cada celda
7. Verificamos que todas las celdas tengan un valor de alimentación mayor a la tensión de alimentación nominal menos un 10 %

Como se vé, el proceso es virtualmente el mismo, pero el resultado es mucho mas exacto, mientras que un α mal estimado en un análsis vectorless puede ser excesivamente pesimista.

5.3. Software

Para realizar los cálculos, se utilizan software de EDA especializados. La anotación de actividad la realiza el simulador, en un formato estándar que cuenta la cantidad de conmutaciones, conocido por su extensión .TCF (Toggle Count Format).

Otro software lee el diseño, tanto a nivel de netlist como a nivel de layout físico, hace coincidir la información del TCF con las nets del circuito, lee la caracterización de las celdas, los datos del proceso y realiza los cálculos.

En el caso de Cadence, el software se llama Voltus (tanto para power analysis como IR Drop) y en Synopsys se llama PrimeTime (solo power analysis).

Para realizar simulaciones más precisas donde no me alcanza la probabilidad de conmutación sino la cantidad exacta de conmutaciones en un instante de tiempo o actividad del clock, se puede realizar otro tipo de simulación dinámica. En este caso los software son los mismos, pero necesitan del simulador los eventos en función del tiempo. El formato estándar de la industria es el archivo .VCD (Value Change Dump) y hay versiones cerradas, como el .SHM de Cadence.

6. Resumen

La siguiente table resume correctamente muchos de los conceptos:

Técnica	Leakage	Consumo Dinámico	Timing	Penalidad Área	Impacto Metodológico	Cambio Metodológico
Optimización de árquitectura	10 %	10 %	0 %	10 %	No	No
Múltiples VT	6X	0 %	0 %	0 %	Bajo	Librerías Multi-Vt disponibles
Clock Gating	0 %	20 %	0 %	<2 %	Bajo	Clock gating cells disponibles
Múltiples Dominios de Tensión	2X	40-50 %	0 %	<10 %	Medio	Micro arquitecturay metodología apropiadas. Reguladores integrados, Level Shifters, etc. Desafíos de verificación
Power Gating	10-50X	0 %	4-8 %	5-15 %	Medio-Alto	Inserción de power switches. Retention flops. Tiempo de encendido y apagado. Verificación de tiempos
Dynamic Voltage Frequency Scaling (DVFS)	2-3X	40-70 %	0 %	<10 %	Alto	Schedule dinámico de tareas. Análisis y optimización multimodo complejo. Sincronización
Body Bias	10X	0 %	10 %	<10 %	Alto	Wells separados. Múltiples rieles de power. Timing análisis complejo