Tema II.

Familias lógicas

2.1. Introducción

A la hora de realizar un layout de un circuito integrado, debemos conocer el esquema a nivel de transistores del circuito para poder disponer las diferentes capas en los lugares correspondientes. Por cuestiones de simetría (tanto geométrica como paramétrica), todas las puertas (o como mínimo la mayoría) que sean necesarias construir se llevarán a cabo siguiendo unos patrones comunes, es decir, utilizando una misma familia lógica.

> Una familia lógica se puede definir como la estructura básica a partir de la cual se pueden construir las diferentes puertas lógicas

La mayoría de familias lógicas utilizadas en el diseño microelectrónico se basan en dispositivos semiconductores, y más concretamente en transistores. Podemos distinguir dos tipos de transistores (de importancia contrastada): transistores bipolares y transistores MOSFET (Metal Oxide Semiconductor Field Effect Transistor), o más comúnmente conocidos como MOS. Los símbolos de estos transistores se muestran en la figura 2.15. En ellos podemos ver que son componentes de tres terminales (aunque en el caso del transistor MOS no es estrictamente cierto, puesto que dispone de cuatro terminales: el sustrato, el cual no suele aparecer porque siempre está conectado al mismo punto).

Figura 2.15. Símbolos de los principales transistores utilizados en Microelectrónica.

Cualitativamente podemos indicar que ambos transistores operan de la misma forma, es decir, como un interruptor controlado por la tensión de base (en el caso de los transistores bipolares) o de puerta (en el caso de los transistores MOS); de tal forma que se permitirá o no el paso de información entre los terminales restantes. No obstante, determinadas características eléctricas han decantado la balanza hacia los transistores MOS ya que presentan un consumo de potencia mucho menor que los transistores bipolares. Los transistores bipolares han quedado para aplicaciones de muy alta velocidad en el campo analógico.

Centrándonos en los transistores MOS, podemos indicar que existen dos tipos: transistores NMOS y PMOS, mostrados en la figura 2.16. En el comportamiento de los dos tipos de transistores, podemos observar que son complementarios, ya que alternan las situaciones de conducción y de corte. También podemos observar una diferencia en la zona de conducción de ambos transistores, de tal forma que uno de los valores lógicos pasa degradado a través del transistor (el valor '1'=V_{DD} en el caso del transistor NMOS, y el '0'=gnd en el caso del PMOS). No obstante, en el modelo de interruptores, esta degradación se suele despreciar (aunque dicha afirmación no es del todo cierta).

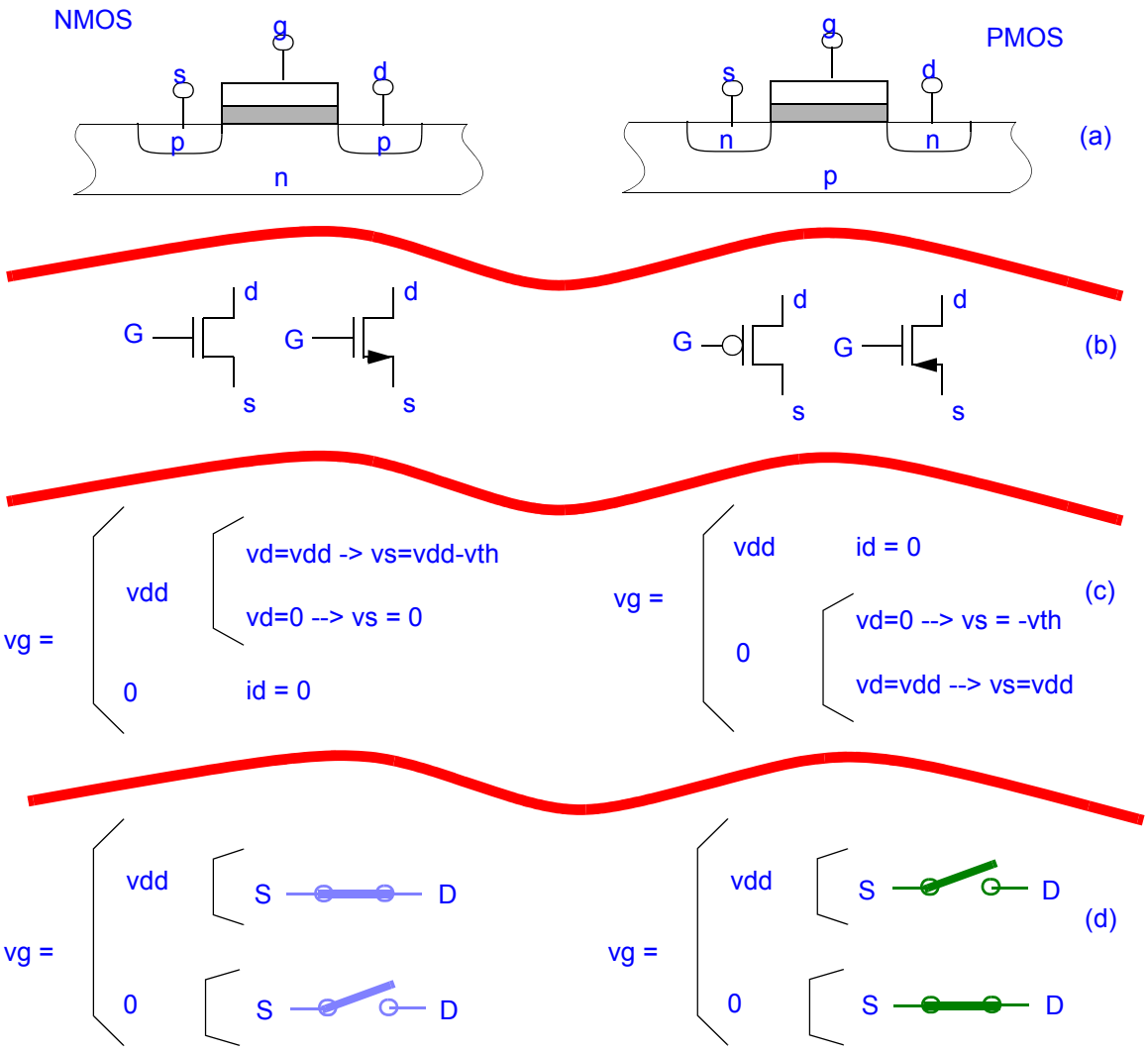


Figura 2.16. Tipos de transistores MOS.
(a) Esquema físico, (b) símbolos, (c) comportamiento, (d) modelo de interruptores.

2.1.1. Análisis de una puerta lógica.

De forma previa a abordar el diseño de puertas lógicas utilizando diferentes familias, vamos a ver cómo podríamos analizar una puerta, es decir, como podríamos obtener el valor lógico de la salida para las diferentes combinaciones de entrada.

Supongamos la puerta mostrada en la figura 2.17, compuesta por dos transistores PMOS, conectados en paralelo, y dos transistores NMOS, conectados en serie. Dicha puerta tiene dos entradas, A y B, conectados a las puertas de dos transistores (NMOS y PMOS) cada una de ellas.

Para cada combinación de señales de entrada, sustituimos cada transistor por su modelo de interruptor, ya sea cortado o conduciendo, y observamos qué nivel lógico es el que está conectado a la salida: ‘1’ (polarización) o ‘0’ (tierra). De esta forma obtenemos cuatro modelos diferentes (por disponer de dos señales de entrada) en los que la salida toma un determinado

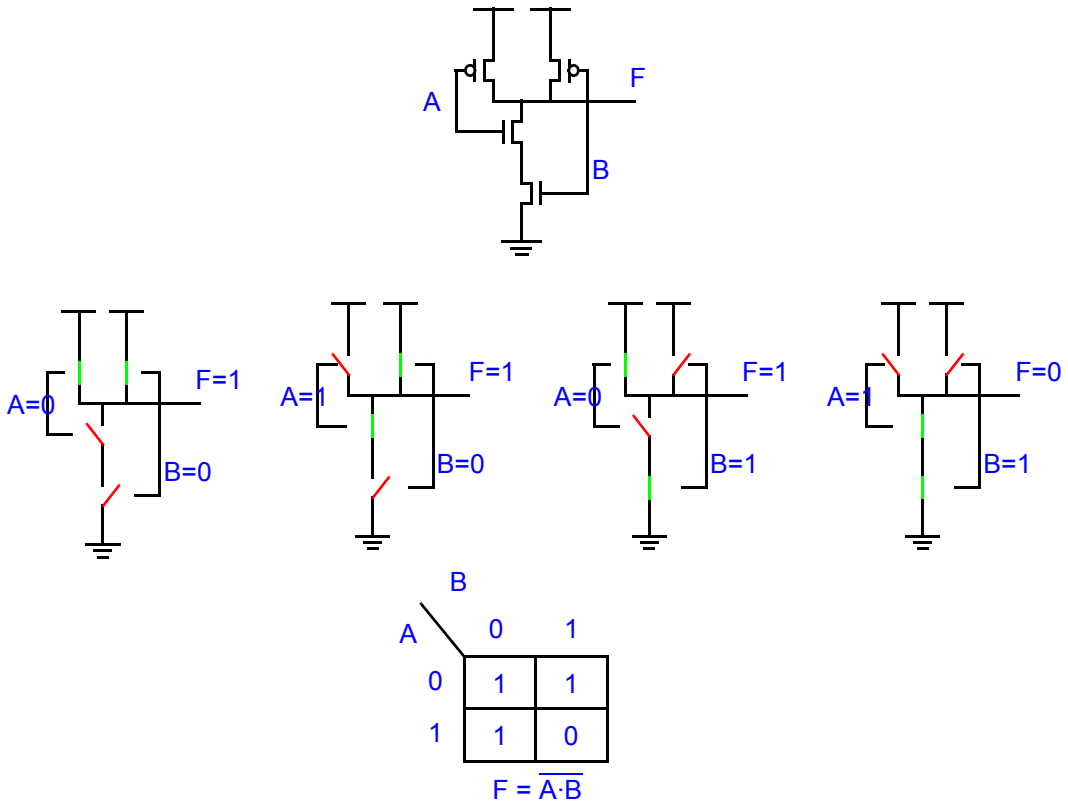


Figura 2.17. Ejemplo de análisis de una puerta lógica.

valor. A partir de cualquier forma de representación (en la figura se ha utilizado un mapa de Karnaugh), podemos apreciar que la puerta lógica estudiada tiene el mismo comportamiento lógico que una puerta NAND de dos entradas.

Veamos otro ejemplo, considerando la puerta mostrada en la figura 2.18(a). La puerta está formada por una conexión serie-paralelo de transistores NMOS y PMOS. En este caso tenemos tres señales de entrada, cada una de ellas conectadas a un par transistor NMOS-PMOS. Siguiendo la técnica del ejemplo anterior, podemos comprobar que la fórmula lógica implementada se corresponde con la siguiente:

$$F = C + A \cdot B$$

Veamos un tercer ejemplo en el que están involucradas más de una puerta lógica. Dicho esquema se muestra en la figura 2.18(b). En ella podemos apreciar varias peculiaridades con respecto a los ejemplos anteriores: existen señales de entrada (en concreto la señal A) que ataca al terminal de fuente de un transistor, la unión de transistores NMOS y PMOS (donde se encontraba hasta ahora el terminal de salida) ataca a otros transistores (ya sea en los terminales de puerta o drenador). El proceso de análisis es el mismo que en los casos anteriores, es decir, sustituimos cada transistor por sus respectivos modelos de interruptores, de tal forma que indiquen cuando se encuentran conduciendo y cuando están cortados. Mostramos dichos modelos para todas las combinaciones de señales de entrada, para obtener los correspondientes valores de la señal de salida. A la vista de los valores de la salida, podemos indicar que la puerta lógica analizada tiene el mismo comportamiento que una puerta XOR, es decir, la fórmula:

$$F = A \cdot \overline{B} + \overline{A} \cdot B = A \oplus B$$

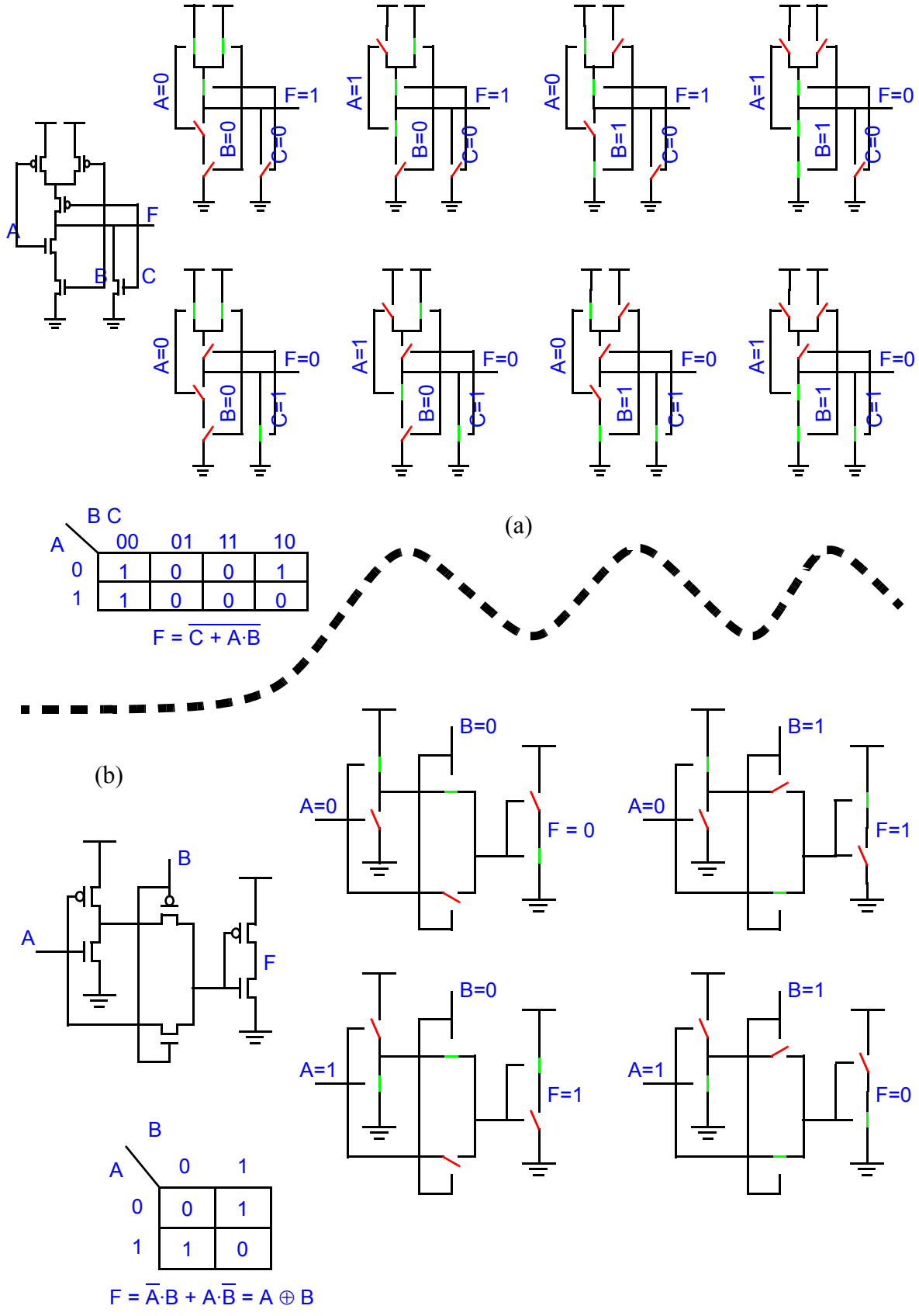


Figura 2.18. Ejemplo de análisis de una puerta lógica.

2.1.2. Familia CMOS estática

Los primeros dos ejemplos pertenecen a una familia lógica denominada familia CMOS (Convencional o Complementaria MOS) estática. Dicha familia se basa en la utilización de un árbol NMOS como pull-down y un árbol PMOS dual como pull-up, como podemos ver en la figura 2.19.

> El pull-down se define como los diferentes bloques que conectan la salida al nodo de tierra.
>
> El pull-up se define como los diferentes bloques que conectan la salida al nodo de polarización.

En función de las fórmulas implementadas podemos indicar que estamos ante una familia negativa en el sentido de que las funciones implementadas son los complementos de alguna función. Por lo tanto, para obtener el comportamiento de una función (en lugar de su complemento) es necesario añadir un inversor a la salida, como sucede en el tercer ejemplo.

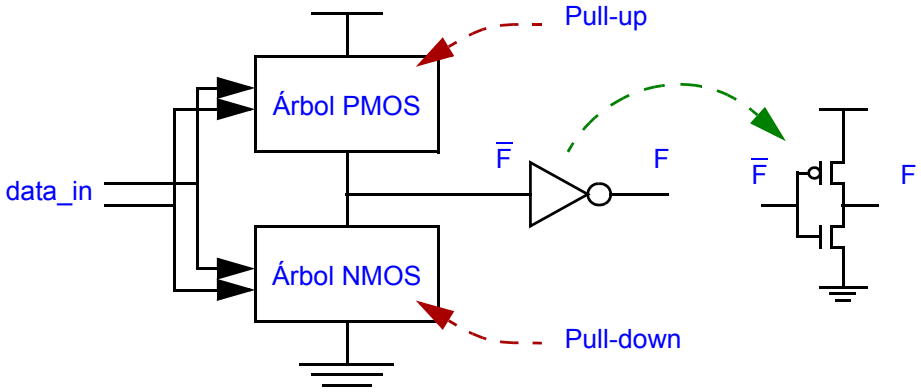


Figura 2.19. Esquema básico de la familia CMOS estática.

En determinadas celdas, se suelen utilizar transistores de paso (como meros interruptores) en unión con las celdas CMOS estáticas como sucede en el tercer ejemplo. Dichas celdas suelen ser multiplexores, puertas XOR y biestables (como veremos en el siguiente tema), ya que la utilización de los transistores de paso aumentan la eficiencia de la implementación. Por lo tanto, debido al uso marginal de este tipo de transistores en la familia CMOS estática, no se hará alusión a ellos en lo que sigue de discusión.

Entre las principales características arquitecturales de la familia CMOS estática podemos destacar las siguientes:

- Los datos de entrada sólo pueden estar conectadas a los terminales de puerta de los transistores (con excepción del posible uso de transistores de paso). De hecho la conexión se debe realizar por parejas, es decir, a un transistor NMOS y a otro PMOS, debiendo tener comportamiento complementarios.
- Los terminales de drenador y fuente sólo pueden estar conectados a nodos internos, a polarización (en el caso de transistores PMOS) o a tierra (en el caso de transistores NMOS) (con excepción del posible uso de transistores de paso).
- El nodo de salida estará conectado a la unión de los árboles NMOS y PMOS.

- Los árboles de transistores NMOS y PMOS no deben conducir simultáneamente, por lo que forzosamente deben ser duales.
- Siempre debe conducir uno (y sólo uno) de los árboles, de tal forma que la señal de salida siempre esté conectada a uno de los raíles de polarización, es decir, a uno de los niveles lógicos. Esta condición estática evita que una variable analógica, como puede ser el tiempo, pueda provocar un cambio en una variable lógica.

Entre las principales ventajas de esta familia podemos destacar las siguientes:

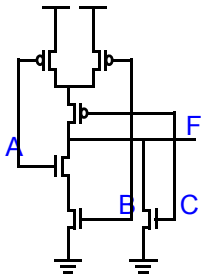
- Consumo de potencia en estática nulo, ya que en ningún estado estacionario existe un camino entre los terminales de polarización y tierra. La principal contribución al consumo de potencia es la contribución dinámica, es decir, situación en la cual las señales de entrada cambian de valor.
- Márgenes de ruido elevados, debido a que la zona de conducción (región que separa los límites lógicos) es muy estrecha. Esta propiedad, que como vimos se podía ver como una medida de la robustez del circuito, hace de esta familia una de las más robustas ante diferentes situaciones: cambios de polarización, cambios de dimensiones de transistores, ruido, etc.

Entre las principales desventajas que existen podemos destacar las siguientes:

- La utilización de un número elevado de transistores PMOS, los cuales son más lentos (tienen una menor movilidad, y por tanto, ganancia) que los transistores NMOS. Esta situación provoca uno tiempos de propagación altos. No obstante, esta situación puede ser mejorada aumentando el dimensionado de dichos transistores (en concreto la razón W/L), lo cual provocará un mayor área ocupada.
- Otro motivo de tiempos elevados es la conexión serie de un número elevado de transistores, el cual disminuirá el fan-in de la puerta en cuestión. Para mejorar la situación se suelen utilizar bufferes/inversores de salida para aislar la conexión serie del resto el sistema, de nuevo a costa de un incremento en área ocupada.

2.1.2.1. Diseño de celdas CMOS estáticas

El diseño de celdas consistirá en el paso de una fórmula lógica a una red de transistores conectados entre sí. Si consideramos el segundo ejemplo, podemos ver las reglas que debemos aplicar en este proceso.



A \ B C	00	01	11	10
0	1	0	0	1
1	1	0	0	0

$$F = \overline{C + A \cdot B} = \overline{C + (A \cdot B)}$$

Figura 2.20. Reglas de diseño de una celda CMOS estática.

En primer lugar debemos eliminar de la función la operación complemento total, ya que está es inherente a esta familia y no se debe implementar específicamente. También vamos a tener cuidado con la jerarquía de las operaciones para poder separarlas de forma correcta. En el caso que nos ocupa tenemos un producto lógico (entre las señales A y B) y una suma lógica (entre el producto y la señal C).

Centrándonos en el árbol NMOS, podemos asociar el producto a una conexión serie (conexión existente entre los transistores controlados por A y B). Del mismo modo, podemos asociar la suma a una conexión paralela (conexión entre el producto, es decir, la conexión serie anterior, y el transistor controlado por la señal C).

También podemos apreciar que el caso del árbol PMOS es completamente complementario, es decir, el producto está asociado a la conexión paralela mientras que la suma está asociada a la conexión serie.

Por lo tanto, el proceso de diseño de las celdas CMOS se basa en la ejecución de los siguientes pasos:

- Obtención de la función lógica (por ejemplo su tabla de verdad)
- Minimización lógica, para obtener una fórmula mínima que se traducirá en un área ocupada mínima (por ejemplo los métodos del mapa de Karnaugh o de McCluskey)
- Aplicar las reglas de conectividad anteriores para el árbol NMOS y el árbol PMOS.

Veamos un ejemplo completo partiendo desde el comportamiento lógico, mostrado en la figura 2.21. Supongamos que deseamos diseñar dos celdas CMOS para las siguientes funciones:

$$F(x_3, x_2, x_1) = \sum m\ (2, 5, 6, 7)$$

$$G(x_3, x_2, x_1) = \sum m\ (4, 5, 6)$$

En primer lugar obtenemos una representación genérica de las funciones lógicas que queremos diseñar. En este caso hemos utilizado las tablas de verdad correspondientes a las funciones F y G.

Una vez que tenemos la representación anterior, pasamos al proceso de minimización. En este caso hemos elegido el método del mapa de Karnaugh, con lo que las fórmulas obtenidas son:

$$F = x2 \cdot \overline{x1} + x3 \cdot x1$$

$$G = x3 \cdot \overline{x1} + x\,3 \cdot \overline{x\,2}$$

Una vez que tenemos las fórmulas mínimas podemos utilizar las reglas de conexión. No obstante, cabe destacar que va a existir un transistor de cada árbol por cada literal (variable o variable negada) que aparezca en la fórmula. Luego, con el fin de reducir al mínimo el número de transistores, se potenciará la utilización del factor común, y así se evitará la repetición de estructuras de transistores. Así, la implementación de G será la correspondiente a la siguiente fórmula:

$$G = x3 \cdot (\overline{x1} + \overline{x2})$$

Es de destacar la existencia de los inversores de salida para obtener la función correcta.

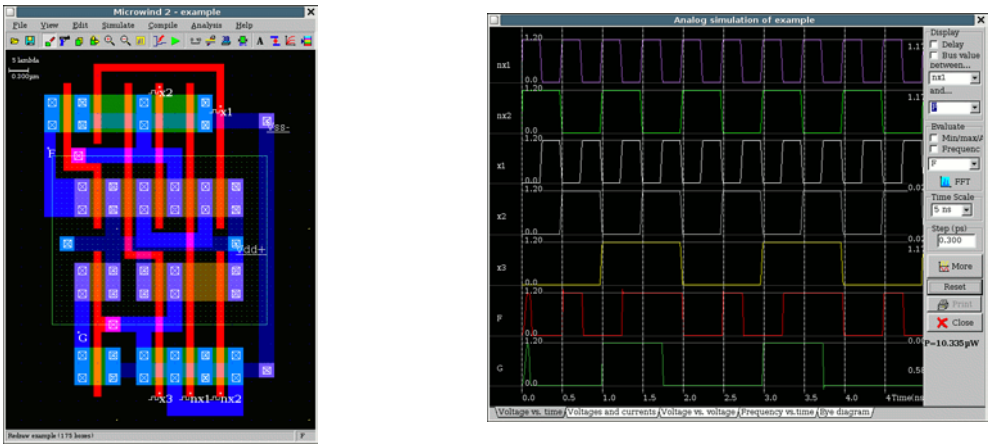


Figura 2.22. Layout y simulación de verificación del circuito mostrado en la figura 2.21c (sin incluir los inversores de entrada).

En la figura 2.22 mostramos un posible layout de la celda CMOS del ejemplo de lafigura 2.21c, en el cual no se han incluido los inversores de entrada. Dicho layout tendría unas dimensiones de 3.60 µm x 5.52 µm, con un área igual a 19.9 μm^2. La verificación del layout se ha realizado a través de una simulación mostrada en la misma figura. Se puede comprobar que obtenemos la misma tabla de comportamiento que la de partida.

En los casos de la figura 2.21b1 y c1, se ha utilizado una estrategia para evitar los inversores de salida. Dicha estrategia consiste en implementar la función complementada, que con el complemento intrínseco de la propia familia, obtendremos la función correcta. Esta estrategia que a primera vista puede ser vista como más óptima debido a la no utilización de los inversores de salida, puede llegar a ser más costosa si el número de inversores de entrada aumenta.

2.2. Familias de transistores de paso

Como ya hemos comentado, los transistores de paso son muy utilizados en todas las puertas cuya implementación con multiplexores es directa, como el caso de los propios multiplexores, puertas XOR o biestables. Todos ellos se pueden modelar con la sentencia VHDL *with ... select*:

```
-- Comportamiento de un transistor de paso
with G select
        D <=  S when ‘1’,
              Z when others;
-- Multiplexor con S como entrada de selección, y A0 y A1 como entrada de datos.
-- Cuando los valores de S sean diferentes de ‘0’ o ‘1’, la salida se pondrá en alta impedancia (Z).
with S select
        F <=  A0    when ‘0’,
              A1    when ‘1’,
```

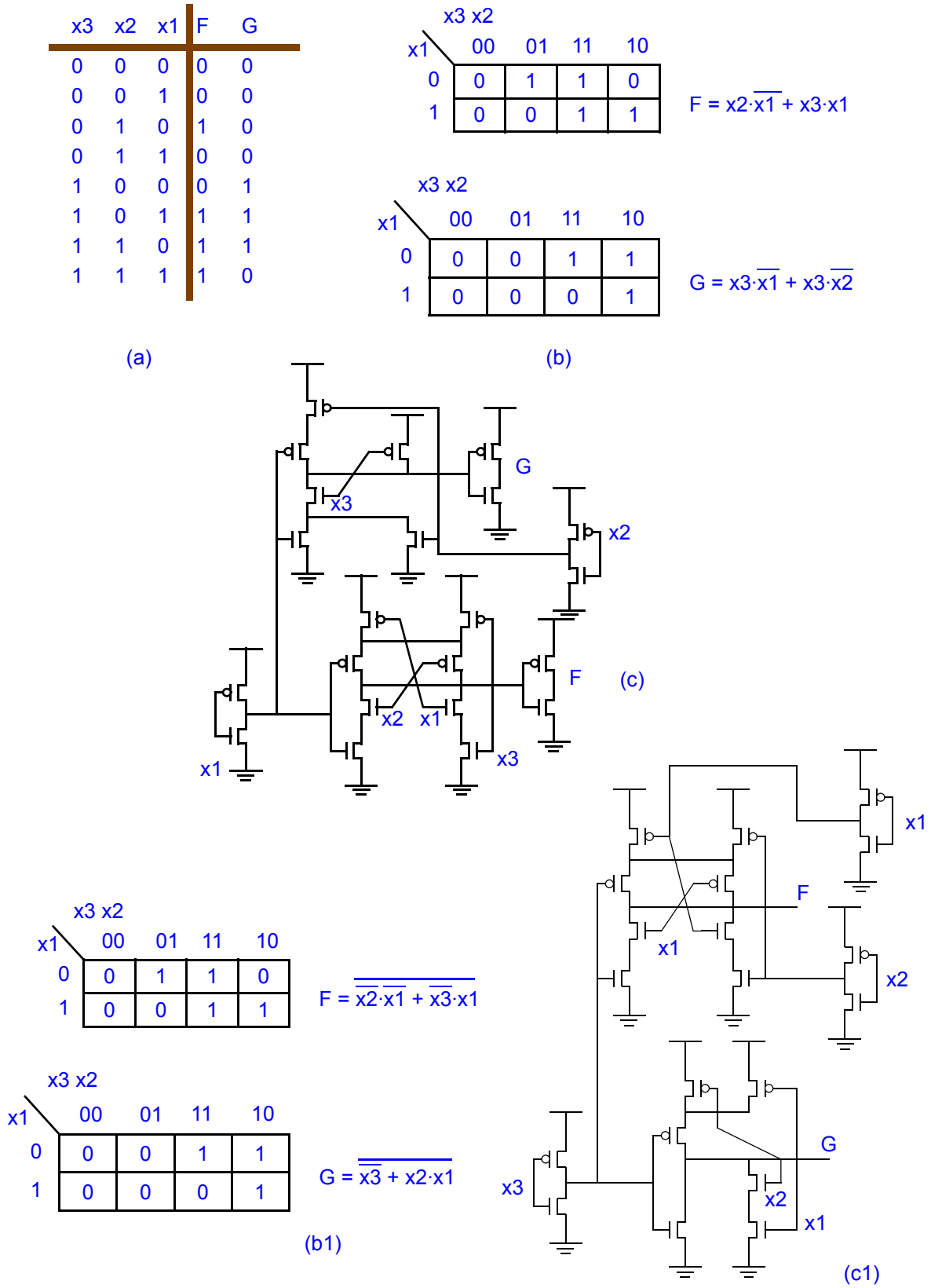


Figura 2.21. Ejemplo de diseño de celdas CMOS estáticas. (a) Función lógica, (b) minimización, (c) aplicación de las reglas de conectividad. (b1) y (c1) para la implementación sin inversores de salida.

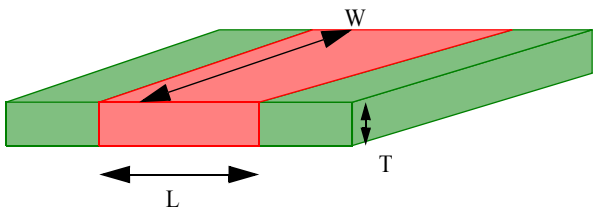
```
        Z       when others;

-- Puerta XOR de dos entradas.
-- Cuando los valores de A sean diferentes de '0' o '1', la salida se pondrá en alta impedancia (Z).
with A select
        F <=  B       when'0',
              not B   when '1',
              Z       when others;
```

Por lo tanto, un multiplexor 2:1 o una puerta XOR de dos entradas estarán formados por dos transistores de paso controlados por señales complementarias.

Si extendemos esta utilización a cualquier función lógica, obtendremos familias lógicas basadas en transistores de paso. En primer lugar, vamos a definir lo que se entiende por un transistor de paso.

> Un transistor de paso es un transistor con una resistencia idealmente nula en su fase de conducción.

La condición de resistencia nula no se puede lograr en dispositivos reales, por lo tanto se tiende a una resistencia tan baja como sea posible (manteniendo una serie de compromisos). En un circuito integrado, todos los parámetros eléctricos están fijados por el proceso de integración, así que la disminución de la resistencia se logrará a través de parámetros geométricos, es decir, aumentando la razón W/L de los transistores de paso. Como podemos ver en la figura 2.23, la resistencia dependerá de la resistividad del material semiconductor (ρ), la longitud del canal (L) y la sección del canal (la anchura del canal, W, por la profundidad de la difusión, T). De todos estos parámetros, los únicos modificables por el diseñador son la longitud y la anchura del canal; de tal forma que se debe aumentar la razón W/L para disminuir la resistencia del canal.



$$R = \rho \cdot l / S = \rho \cdot L / (W \cdot T)$$

Figura 2.23. Resistividad de un transistor.

Uno de los principales problemas de los transistores de paso es la degradación existente en el paso de los niveles lógicos. El principal motivo de esta degradación es la tensión umbral que cae entre los terminales de fuente y drenador de un transistor al transmitir uno de los niveles lógicos. Esta situación limita el número de entradas (número de transistores conectados en serie) de una puerta. Para evitar que esta situación se extrapole al número de puertas, es necesario regenerar los niveles lógicos en las salidas de las diferentes puertas. Luego la arquitectura general de una familia de transistores de paso estará compuesta por un árbol de transistores de paso seguido de un bloque regenerador, tal como se muestra en la figura 2.24b. Debido a su implementación como multiplexores, la utilización de las variables negadas y sin negar es muy común; por lo tanto, también es muy común el desarrollo de familias de transistores de paso diferenciales, en las cuales se obtienen los dos valores de las señales de salida, como se mues-

tra en la figura 2.24c.

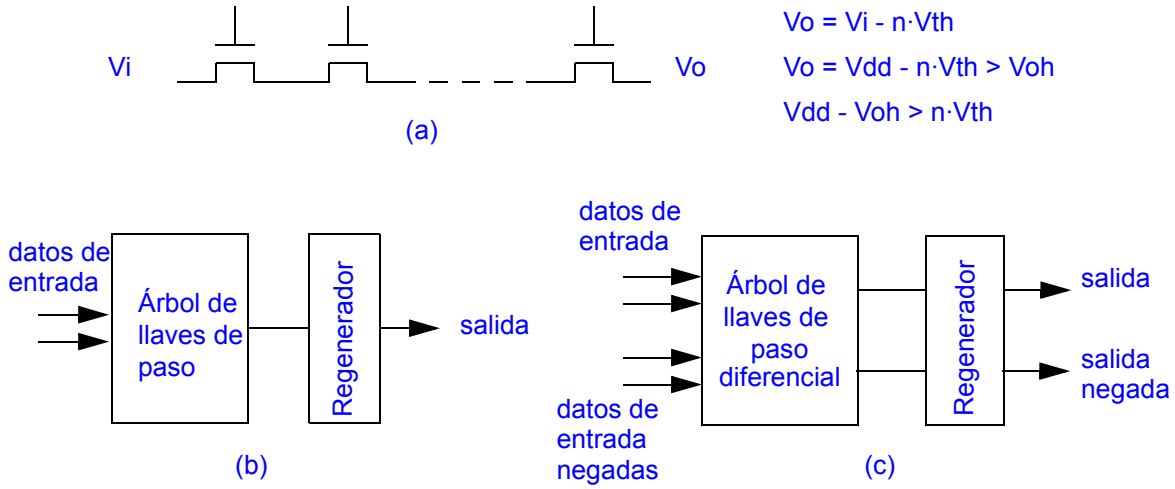


Figura 2.24. (a) Limitación del número de transistores de paso. (b) Esquema genérico de una familia de transistores de paso. (c) Esquema genérico de una familia de transistores de paso diferencial.

Entre las principales características arquitecturales de las familias de transistores de paso podemos destacar las siguientes:

- Las señales de entrada pueden estar conectadas tanto a los terminales de puerta como a los de drenador, obviamente del árbol de transistores de paso.
- Los transistores de paso se distribuirán en niveles, como en un OBDD (Diagrama de Decisión Binaria Ordenado), en los que el primer nivel (terminal de drenador del primer transistor de la rama en cuestión) puede estar conectado a una señal de entrada o a un nivel lógico.
- Los restantes niveles serán el terminal de puerta de todos los transistores de la rama en cuestión.
- Existe un límite (más restrictivo que en la familia CMOS) en el número de transistores conectados en serie, o lo que es equivalente, el número de niveles, es decir, el número de entradas.
- Es necesario un bloque regenerador a la salida de cada puerta para restaurar los niveles lógicos degradados, y así aislar el árbol de transistores de paso del resto del circuito.
- Debido a que en el árbol de transistores de paso pueden coexistir los dos niveles lógicos, es necesario que sólo una de las ramas del árbol conduzca totalmente, con el fin de evitar choques de información.

Entre las principales ventajas de este tipo de familias podemos destacar las siguientes:

- Sólo hace falta una red de transistores (MOS por lo general) para llevar a cabo la función lógica. Luego, el área ocupada será potencialmente más pequeña.
- Debido a que los raíles de polarización no suelen estar conectados al árbol de transistores de paso, el consumo de potencia se ve reducido al consumo de los bloques rege-

neradores, los cuales suelen ser pequeños y consumir poco.

Por contra, ente los principales inconvenientes podemos destacar los siguientes:

- Las familias de transistores de paso son muy sensibles al escalado de la tensión de polarización y al dimensionamiento de los transistores, debido a la necesidad de llegar al límite de resistencia nula. Este hecho implica que este tipo de familias no son robustas.

2.2.1. Diseño de familias de transistores de paso

Como hemos visto en la arquitectura general, la función lógica es generada en el árbol de transistores de paso, por lo que la diferencia básica entre las diferentes familias estará en el bloque regenerador y el tipo de transistores de paso utilizados. No obstante, el método de diseñar el árbol, es decir, de pasar desde la función lógica al árbol de transistores es común para todas las familias.

Una diferencia básica en el diseño con respecto a la familia CMOS estática es que el mismo árbol debe generar los dos valores lógicos, por lo que se deben considerar los '0' y '1' simultáneamente. Luego, ya no existe una relación directa entre una fórmula lógica y el árbol de transistores de paso como sucedía en los árboles NMOS y PMOS de la familia CMOS estática.

Veamos con un ejemplo como podremos interpretar un árbol de transistores de paso. En la figura 2.25 se muestra un árbol de transistores de paso correspondiente al producto lógico de tres variables (x1, x2 y x3), con el siguiente algoritmo de sentencias *if ... then ... else* anidadas:

```
if (x1 = '1') then
        if (x2 = '1') then F <= x3;
        else F <= '0';
        end if;
else F <= '0';
```

Al lado mostramos una posible representación de dicho comportamiento, conocida como BDD (Binary Decision Diagram). Podemos ver que existe una transformación directa entre la representación y el árbol, de tal forma que existirán tantos transistores como arcos haya en el BDD. Las señales que controlan dichos transistores serán los valores sin negar o negadas de las variables de las que parten, según la rama se corresponda con el valor '1' o el valor '0' respectivamente. Por lo tanto, vamos a utilizar esta representación para llevar a cabo el diseño de estos árboles.

Por lo tanto, vamos a utilizar este forma de representación para obtener el árbol de transistores de paso. Una posible definición de diagrama de decisión binaria podría ser la siguiente:

> Un diagrama de decisión binaria (BDD, Binary Decision Diagram) es un grafo dirigido acíclico de tal forma que representa una función lógica como una serie de sentencias *if ... then ... else* anidadas.

En dicho diagrama, los nodos son etiquetados por una variable de la función. De cada nodo

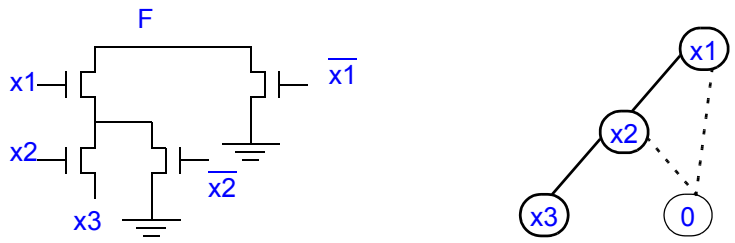


Figura 2.25. Ejemplo de un árbol de transistores de paso y su representación como BDD.

salen dos ramas correspondientes a cada valor que puede tomar cada variable (en el caso de no aparecer el valor; se suele tomar la consigna de que la rama puntuada será la correspondiente al valor '0', y la rama rellena será la correspondiente al valor '1'). A modo de ejemplo mostramos un posible BDD para la función representada por la fórmula en la figura 2.26a

$$F = X \cdot (Y \oplus Z)$$

Podemos comprobar que cada rama completa del árbol equivale a cada una de las combinaciones de la tabla de verdad de la función. En dicha figura se han remarcado las combinaciones 010 y 111. De la misma forma, si analizamos el DD de la figura 2.26b, podemos comprobar que el comportamiento es idéntico al anterior. No obstante, el árbol de transistores de paso obtenido desde este último BSS es más óptimo. Luego debemos obtener el BDD más reducido de forma previa al diseño del árbol de transistores de paso.

Este BDD reducido se conoce como ROBDD (Reduced Ordered Binary Decision Diagram). En primer lugar vamos a dar una posible definición de OBDD.

> Un diagrama de decisión binaria ordenado (OBDD) es aquel en el que cada variable aparecerá siempre en el mismo nivel, y de esta forma aparecerá una sola vez en cada rama del BDD. También existirá un único nodo etiquetado como '0' y un único nodo etiquetado como '1'.

En la figura 2.27a se muestran los OBDDs correspondientes a los BDDs mostrados en la figura 2.26. Si comparamos los diagramas de ambas figuras, podemos observar que el único cambio que se ha realizado ha sido agrupar los nodos etiquetados como '0' y como '1'. Dicha agrupación no implica ninguna alteración en el comportamiento general de los diagramas ya que los subgrafos sucesores (en este caso conjuntos vacíos) coinciden.

La reducción de un OBDD se limita básicamente a la aplicación de dos reglas tantas veces como sea posible. Estas reglas son las siguientes:

- Se reducen a un solo nodo todos aquellos que muestran el mismo comportamiento en los subgrafos sucesores (a partir de dichos nodos).
- Se eliminan todos los nodos que muestran el mismo comportamiento para las ramas '0' y '1'.

La aplicación de la primera regla al OBDD de la figura 2.26a reducirá los dos nodos etiquetados como Z de la rama X=0 a un único nodo. Y la aplicación de la segunda regla eliminará el nodo Z resultante de la regla anterior, así como el nodo Y de la misma rama. Dichas reducciones se muestran en la figura 2.27b.

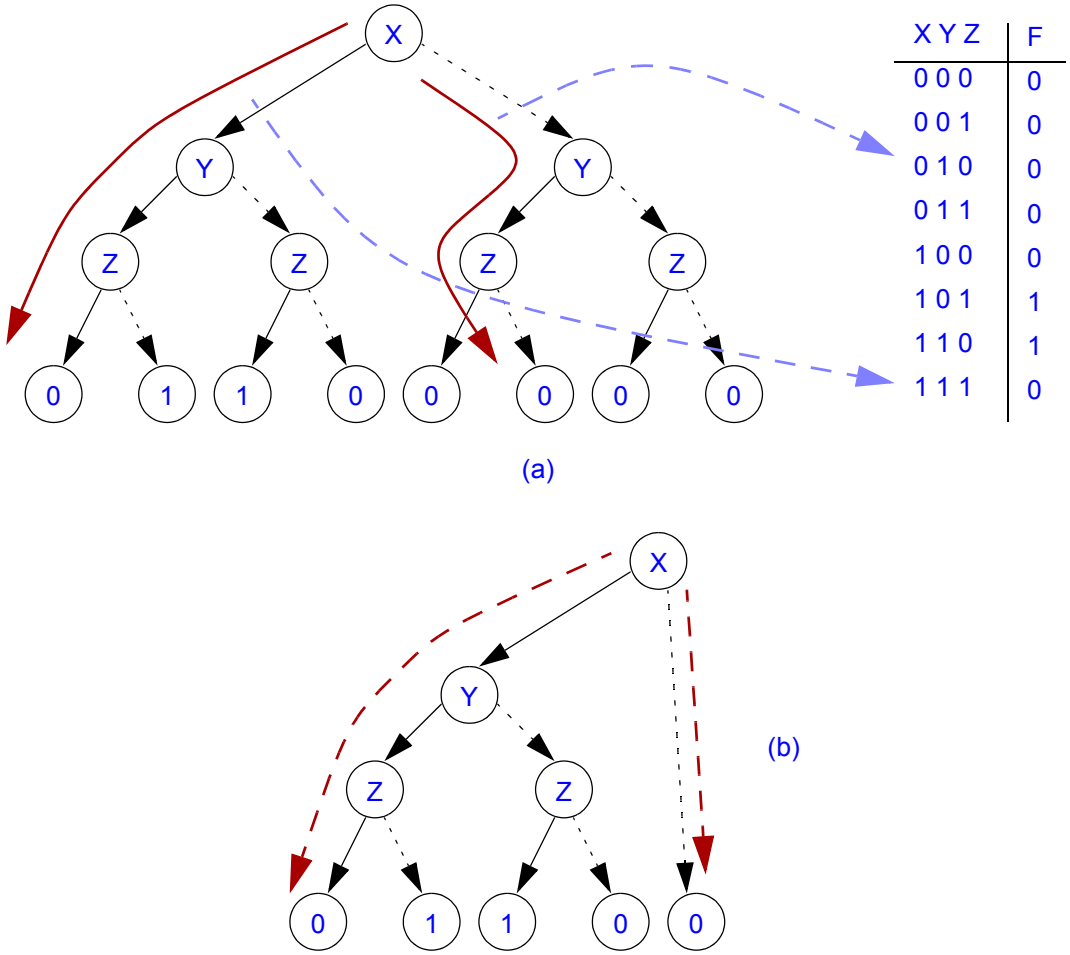


Figura 2.26. BDD y tabla de combinaciones correspondientes a la fórmula X·(Y ⊕ Z)

Se demuestra que un OBDD al que no se le pueden aplicar ninguna de las reglas de reducción anteriores es un OBDD reducido (ROBDD), el cual utilizaremos para construir el árbol de transistores de paso. No obstante, con este OBDD no se aprovecha la situación de que una variable pueda estar conectada a la fuente de un transistor, es decir, ser el último nodo de una rama de un ROBDD. Para ello podemos hacer uso de las representaciones de las funciones unarias (funciones de una sola variable: seguidor e inversor), mostradas en la figura 2.28a. Si utilizamos estas representaciones para reducir aun más el ROBDD de la figura 2.27b, obtendremos el diagrama de la figura 2.28b, desde el cual podemos obtener el árbol de transistores de paso mostrado en la misma figura.

En la figura 2.29 mostramos dos BDDs reducidos para la misma función lógica, es decir, para

$$F = X \cdot (Y \oplus Z)$$

Podemos apreciar que los dos BDDs son diferentes, y por ende, los arboles de transistores de paso también lo son; es más, la diferencia abarca hasta el número de transistores. Luego sería deseable que antes de reducir el OBDD pudiésemos saber cuál sería el ordenamiento de las entradas óptimo, o en su defecto *bueno*.

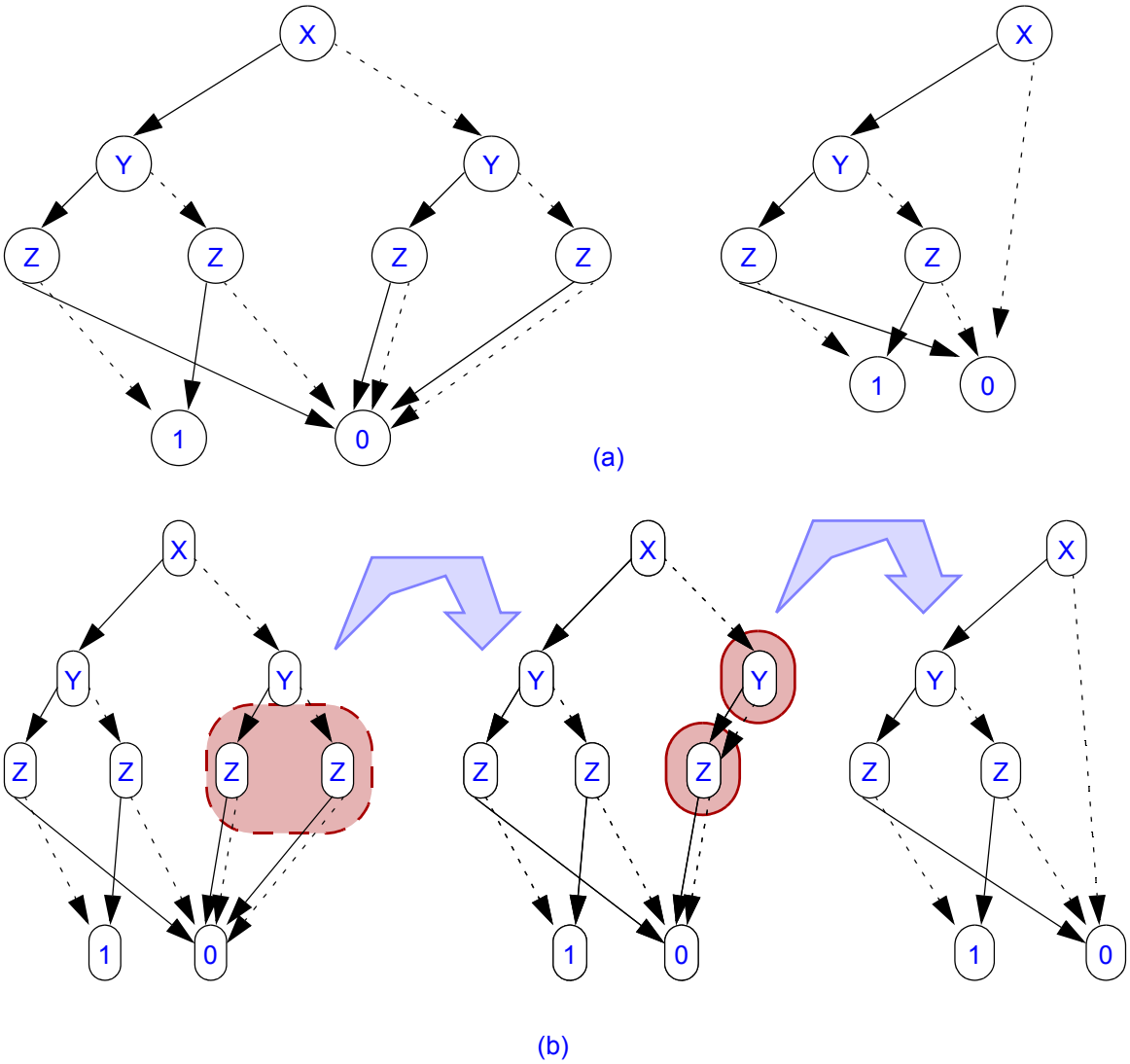


Figura 2.27. (a) OBDD correspondientes a los BDD de la figura 2.26.(b) Aplicación de las reglas de reducción.

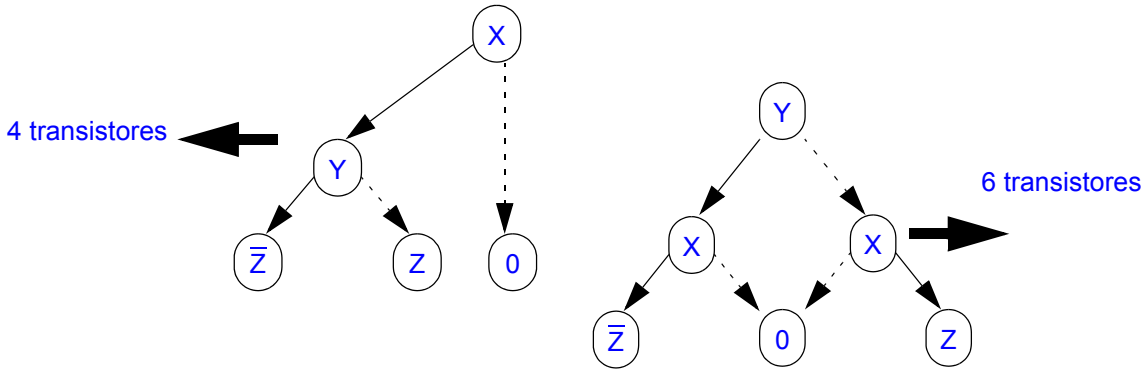


Figura 2.29. Diagramas reducidos para la misma función.

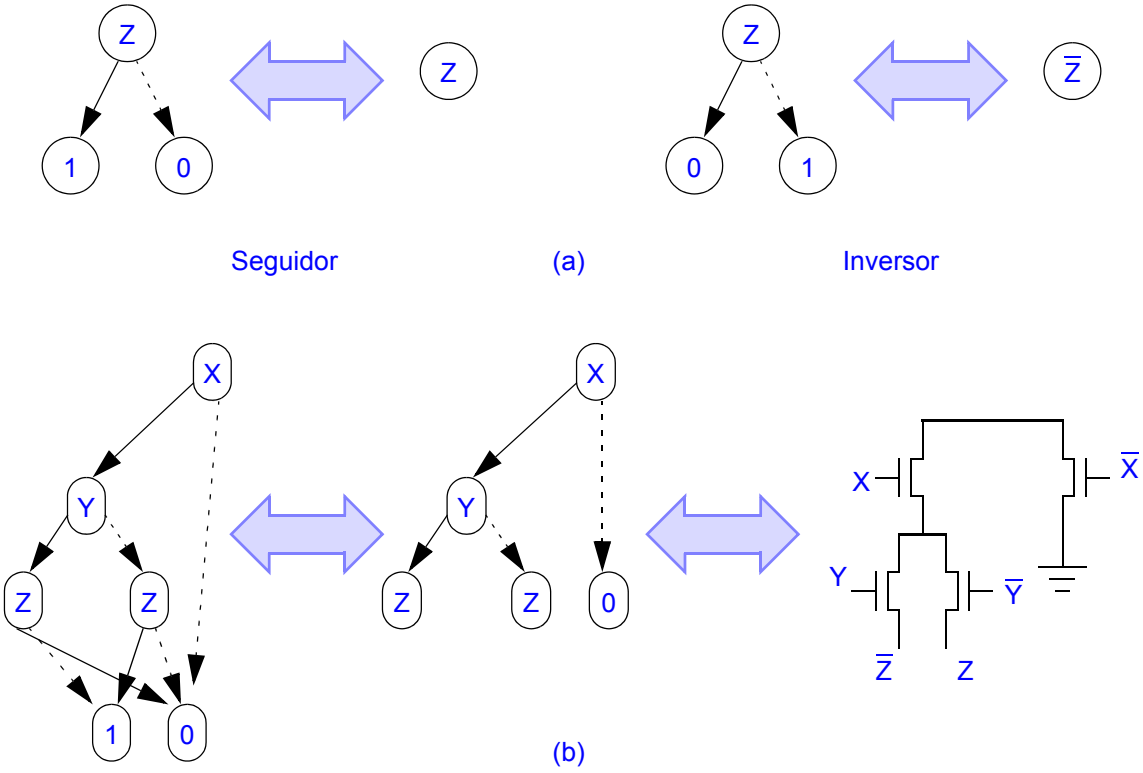


Figura 2.28. (a) BDDs de las funciones unarias. (b) Aplicación de las funciones unarias al ROBDD de la figura 2.27b.

Existen muchos algoritmos de reordenamiento, pero nos vamos a centrar en uno de ellos: el ordenamiento heurístico por pesos. Este ordenamiento se basa en que cuanto mayor sea la influencia de una señal en la función lógica, su posición en el OBDD debe ser lo más alta posible. Esta influencia está relacionada con los pesos calculados de la siguiente forma:

- Si se trata de la señal de salida, el peso de dicha será 1.
- El peso de la salida de una puerta es dividido uniformemente entre sus entradas.
- Si una señal está conectada a más de una puerta, el peso de dicha señal será la acumulación de las contribuciones de todas las puertas conectadas.

En la figura 2.30 mostramos la aplicación del algoritmo de reordenamiento para la función cuya fórmula mínima corresponde a

$$F = X \cdot (Y \oplus Z)$$

En ella podemos ver que el circuito (basado en puertas lógicas) está formado por una puerta AND seguida de una puerta XOR. De dicho circuito, podemos apreciar que el peso de cada nodo se corresponde con la siguiente lista (nodo, peso)

$$\{(F, 1), (X, 1/2), (S1, 1/2), (Y, 1/4), (Z, 1/4)\}$$

donde el nodo S1 será la salida de la puerta XOR. Por lo tanto, la lista de niveles (empezando por el orden superior) será la correspondiente a {F, X, Y, Z}, en la que F no suele aparecer en el BDD, y las señales Y y Z pueden permutarse ya que tienen el mismo peso.

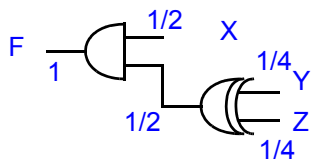


Figura 2.30. Aplicación del ordenamiento heurístico por pesos a la función F = X·(Y ⊕ Z)

En el caso de que tengamos un función multisalida, podemos aprovechar parte de un BDD para obtener el árbol de la salida restante.

Luego un proceso de diseño de celdas basadas en una familia de transistores de paso estaría formado por los siguientes pasos:

- Obtención de la función lógica
- Obtención de un ordenamiento de señales *bueno* (por ejemplo utilizando el algoritmo heurístico de pesos) para la generación del ROBDD* (no sería exactamente un ROBDD por que los nodos finales no tienen porqué ser 1 y/o 0).
- Generación del ROBDD* a partir del OBDD obtenido con el ordenamiento anterior, utilizando las reglas de reducción.
- Generación del árbol de transistores de paso siguiendo la arquitectura dictada por el ROBDD* del punto anterior.

Veamos el proceso de diseño completo para la función mostrada en la figura 2.31. En primer lugar en la figura 2.31a mostramos una primera representación de la función lógica de la cual vamos a obtener los árboles de transistores de paso, es decir, las celdas generadoras de función de cualquier familia de transistores de paso.

Seguidamente, obtenemos un circuito mínimo (utilizando un esquema de puertas lógicas) para lograr un ordenamiento *bueno* para el posterior OBDD. En este caso, se han utilizado cuatro puertas AND (tres de ellas con una entrada negada) y dos puertas OR. Aplicando el algoritmo heurístico de pesos, obtenemos los siguientes pesos para cada una de las señales de entrada.

{(Señal, Peso)} {(x3, 3/4), (x1, 3/4), (x2, 1/2)}

En dicho cálculo se han acumulado las contribuciones de las diferentes puertas.

En el siguiente paso, se genera los OBDD para las dos señales de salida, los cuales serán minimizados utilizando las reglas de reducción vistas en el subapartado anterior. En la figura 2.31c mostramos los OBDD y ROBDD para las salidas F y G. En este caso particular ambos diagramas son totalmente disjuntos, aunque es frecuente el caso de salidas que comparten algunas porciones de diagramas.

Por último, en la figura 2.31d mostramos una celda cuyas salidas son las señales F y G implementada con una familia genérica de transistores de paso (ya que el bloque generador se ha dejado como caja negra). No obstante, como todos los bloques generadores suelen completar la señal de salida, el árbol generado muestra el comportamiento complementario negando el último nivel del ROBDD.

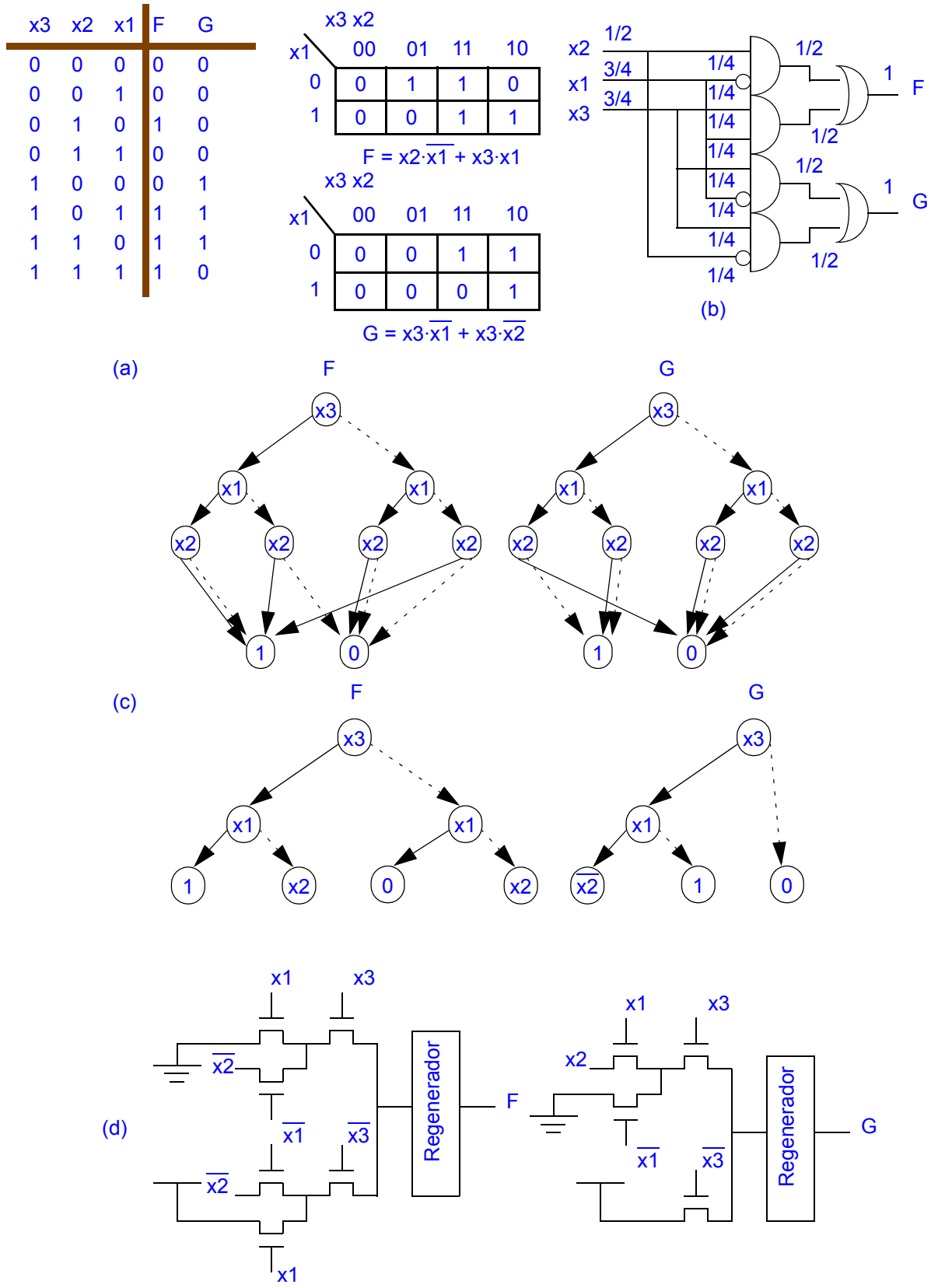


Figura 2.31. Ejemplo de diseño de celdas con familia de transistores de paso. (a) Función lógica, (b) reordenamiento de las señales, (c) generación de los ROBDD y (d) generación de los árboles de transistores de paso.

2.2.2. Tipos de familias de transistores de paso

Siguiendo el esquema de familias de transistores de paso, podemos encontrar un número relativamente elevado de familias. La diferencia básica se encuentra en el bloque regenerador, aunque también existen diferencias en las puertas de paso utilizadas: puertas CMOS, que son parejas de transistores NMOS y PMOS con todos los terminales conectados excepto el terminal de puerta que están conectados a señales complementadas para que la conducción o no de ambos transistores sea simultánea; puertas NMOS, que es la situación más típica en la que se utiliza únicamente transistores NMOS; puertas PMOS, situación menos usual en la se utilizan transistores PMOS. Entre todas las familias reportadas vamos a destacar la familia LEAP (LEAn integration with Pass-transistor), CPL (Complementary Pass-transistor Logic) y DPL (Double Pass-transistor Logic).

La familia LEAP es una familia no diferencial (de raíl simple) formada por un árbol de transistores de paso no diferencial (implementado con transistores NMOS) y un bloque regenerador formado por un inversor y un transistor PMOS a modo de pull-up. El funcionamiento de dicha familia es el siguiente. El árbol de transistores de paso genera la función lógica implementada situando en su salida el valor de salida correspondiente a los valores de entrada. La degradación del '1' debido al árbol es restaurada por el bloque regenerador, ya que dicho '1' pasa a ser un '0' en la salida del inversor, el cual pone en conducción el transistor PMOS del pull-up que sitúa un '1' no degradado en la salida del árbol. Dicho esquema se muestra en la figura 2.32, ya sea a nivel de transistores o layout (con un área ocupada de 22 μm^2). En la misma figura mostramos la verificación del layout a través de la simulación. Podemos apreciar la regeneración de la salida del árbol con el cambio de pendiente en las salidas nF y nG (salidas de los árboles de transistores de paso).

Entre las principales ventajas de la familia LEAP, podemos destacar el hecho de que el árbol de transistores de paso sea no diferencial. Por lo tanto, las capacidades y cableado dentro de la celda es menor. Todo ello implica una implementación con área ocupada baja, un retraso bajo y un consumo de potencia bajo.

Entre los principales inconvenientes, podemos destacar los siguientes:

- La regeneración del '1' llevado a cabo a través del transistor PMOS realimentado es una regeneración lenta, que es compensada con la disminución de retraso debido al árbol no diferencial.
- La arquitectura del bloque regenerador implica que el comportamiento correcto únicamente se garantizará cuando se cumpla la condición

$$V_{dd} > V_{tn} + |V_{tp}|$$

- De la característica anterior, se deduce que la robustez de la familia LEAP ante pequeñas tensiones de polarización (V_{dd}) sólo se garantiza si los transistores son implementados con la condición de que sus tensiones umbrales (V_{tn} y V_{tp}) sean bajas.

La familia CPL es una familia de transistores de paso diferencial. Está compuesto por un árbol de transistores de paso diferencial, y un bloque regenerador formado por dos inversores y dos transistores PMOS de pull-up controlados por la señal de salida complementaria. En la figura 2.33a mostramos el esquema a nivel de transistores de la familia CPL.

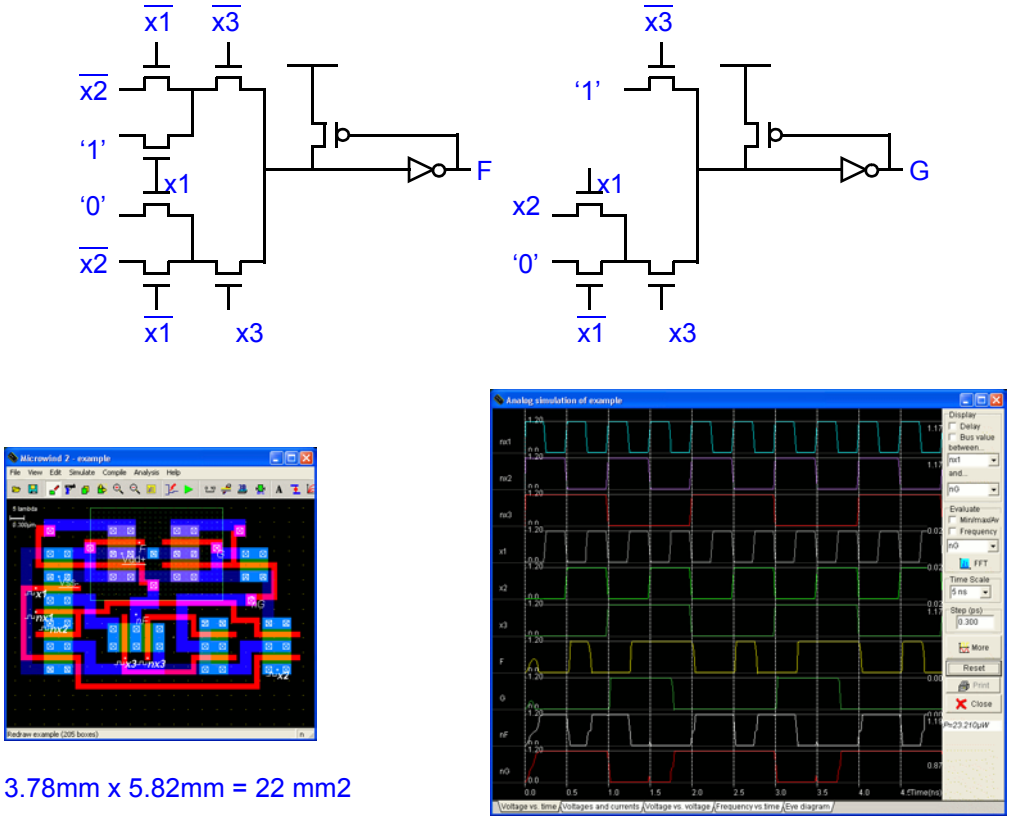


Figura 2.32. Esquema a nivel de transistores de la familia LEAP de la función de la figura 2.31. Layout y simulación de dicho esquema.

Entre las principales ventajas de la familia CPL podemos destacar las siguientes:

- Las capacidades de entrada suelen ser pequeñas, lo cual implica un consumo de potencia y retrasos de propagación bajos.
- La eficiencia de implementaciones de puertas basadas en multiplexores como las puertas XOR que son la base de la mayoría de circuitos aritméticos.
- Los inversores de salida *aceleran* la evaluación del árbol, ya que no hace falta que la salida del árbol llegue al valor lógico nominal para que el inversor complemente la señal. Por lo tanto, el retraso de propagación disminuye.
- De igual modo, los transistores PMOS de pull-up aceleran la evaluación del árbol diferencial, ya que cuando una rama llega a '0', la salida de la rama complementaria tomará el '1' independiente de dicha rama.

Entre los principales inconvenientes de dicha familia podemos destacar las siguientes:

- El árbol diferencial provoca mayores corrientes de cortocircuito, por lo que el consumo de potencia aumenta.
- El árbol diferencial provoca un número sustancial de nodos y de cableado dentro de la celda. Esta situación provoca un aumento en el área ocupada y el consumo de potencia debido al aumento de dispositivos parásitos.

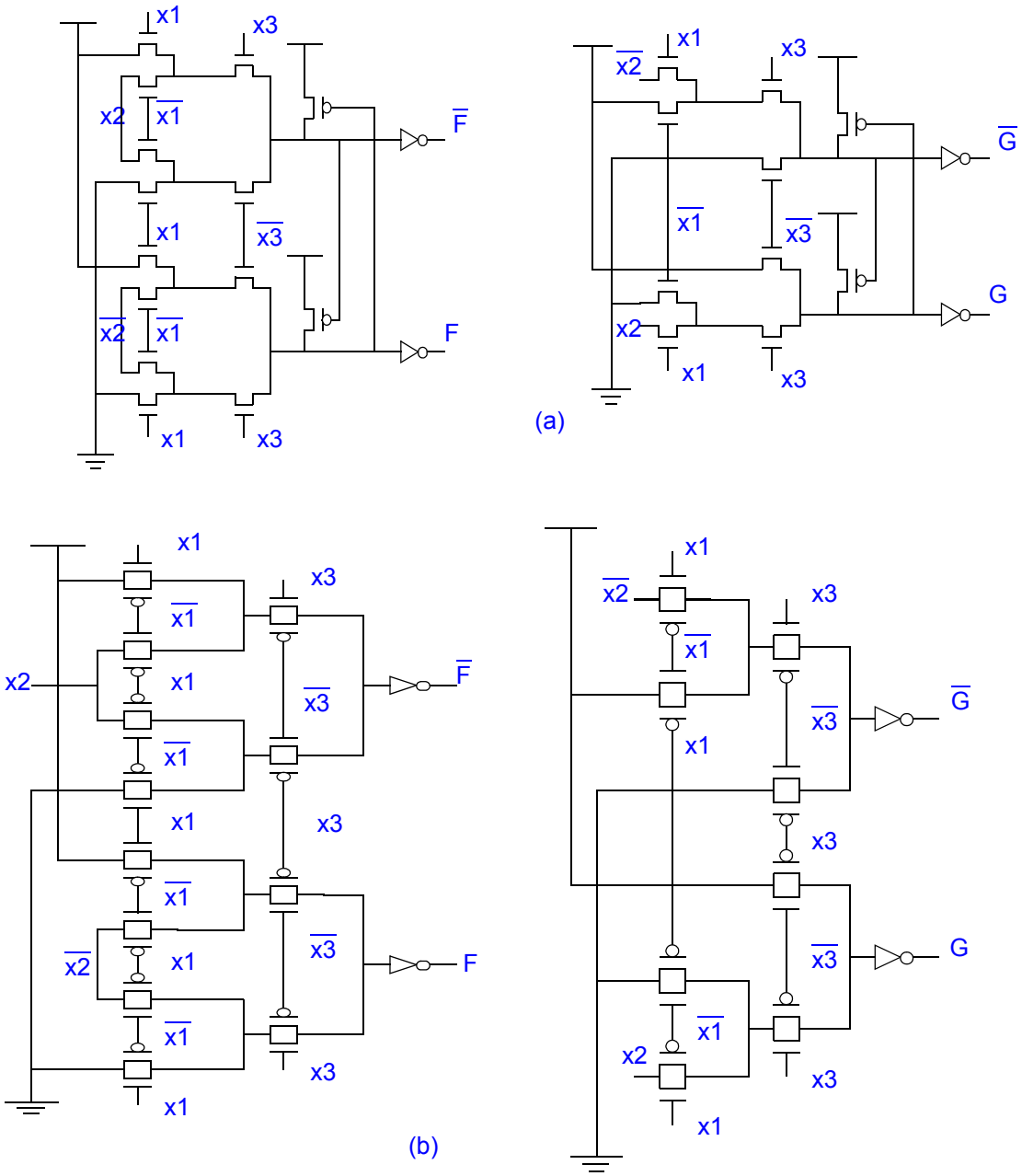


Figura 2.33. Esquema a nivel de transistores de la función de la figura 2.31. (a) Familia CPL, y (b) familia DPL.

La familia DPL se basa en un árbol diferencial implementado con llaves CMOS, es decir, transistores NMOS y PMOS controlados por señales complementarias. Este hecho implica que no existe degradación en ninguno de los niveles lógicos, puesto que el '0' pasará a través del transistor NMOS (sin degradación) y el '1' pasará a través del transistor PMOS (sin degradación). Luego no existe la necesidad de ningún bloque regenerador, no obstante se añade inversores de salida (que en otras familias actuarían como regeneradores de señal) que actuarán únicamente como aisladores del árbol diferencial. En la figura 2.33b se muestra a nivel de transistores la celda DPL correspondiente a la función de la figura 2.31.

Entre las principales ventajas de la familia DPL podemos destacar que el árbol de puertas CMOS de paso implica la no degradación de ningún nivel lógico por lo que no es necesaria la existencia del bloque regenerador.

Entre los principales inconvenientes podemos destacar el elevado número de recursos debido al carácter doble de cada puerta (transistores NMOS y PMOS) y al carácter diferencial. Por lo tanto, el retraso de propagación y el consumo de potencia aumentará debido a los dispositivos parásitos (que serán mayores si el número de recursos aumenta), y el área ocupada será mayor debido al aumento de recursos necesarios.

2.3. Familias Diferenciales

Una estructura diferencial puede considerarse como una situación intermedia entre una familia dinámica (como la mostrada en la figura 2.34) y una familia estática (como la familia CMOS vista en el apartado 2.1.2), de tal forma que trata de aprovechar las ventajas de ambos tipos de familia y evitar, en la mayor medida de lo posible, sus inconvenientes.

De las familias dinámicas tratan de aprovechar su velocidad lograda por la alternancia de fases de precarga y evaluación. La operación se lleva a cabo de la forma siguiente:

- Cuando la señal de control ϕ toma el valor ‘0’, nos encontramos en la fase de precarga. En dicha fase, el árbol está desconectado del nodo de tierra por lo que no realiza ninguna función. n cambio, el transistor PMOS de precarga está conduciendo colocando en el nodo de salida interno un nivel ‘1’.
- Cuando la señal de control ϕ toma el valor ‘1’, nos encontramos en la fase de evaluación. En dicha fase, el árbol está conectado al nodo de tierra por lo que en función de los valores de entrada pondrán dicho nivel, el ‘0’, en el nodo de salida interno. En el caso de que el árbol no conduzca (debido a los valores de las señales de entrada), el nodo de salida interno mantendrá almacenado el valor de precarga (que coincidirá con el valor lógico requerido) de forma dinámica (gracias a los dispositivos parásitos).

Por contra, tratan de evitar el almacenamiento dinámico del nivel ‘1’ en la fase de evaluación (situación a la que hemos aludido antes).

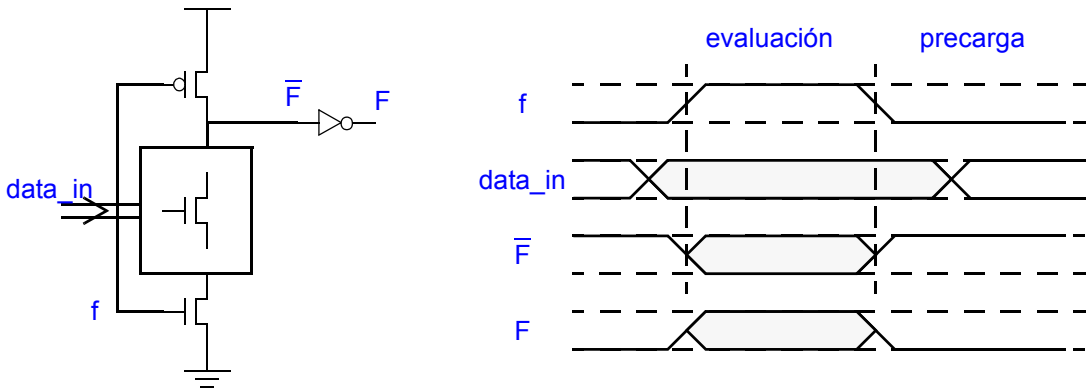


Figura 2.34. Lógica dominó como ejemplo de familia dinámica.

En cambio, de las familias estáticas tratan de aprovechar el comportamiento estático de al menos los nodos de salida (evitando la situación dinámica de las familias dinámicas). Mientras que evita su forma de operación más lenta.

El esquema de una estructura diferencial genérica se muestra en la figura 2.35a. En ella podemos distinguir un árbol diferencial de transistores NMOS y un bloque de carga o pull-up; y adicionalmente se puede encontrar un transistor NMOS de precarga que conecta o desconecta el camino entre el nodo de tierra y el árbol diferencial. Por tratarse de una estructura diferencial siempre vamos a necesitar los dos raíles de las señales de entrada, es decir, las señales sin complementar y complementadas; además de obtener los dos raíles de la señal de salida. Al utilizar los dos raíles de cada señal, cada señal, ya sea de entrada o de salida, va a estar codificada en doble raíl. El esquema de dicha codificación se muestra en la figura 2.35c, en el cual existen dos líneas por cada señal: una entrada de verdad (.v) y una entrada de falso (.f); con tres posibles situaciones: '0' lógico, en la cual únicamente se activa el cable de falso; '1' lógico, en la cual únicamente se activa el cable de verdad; y dato no válido o de precarga, en la que ambos cables tienen el mismo valor (por lo general es siempre el mismo valor, y la cuarta combinación nunca se produce).

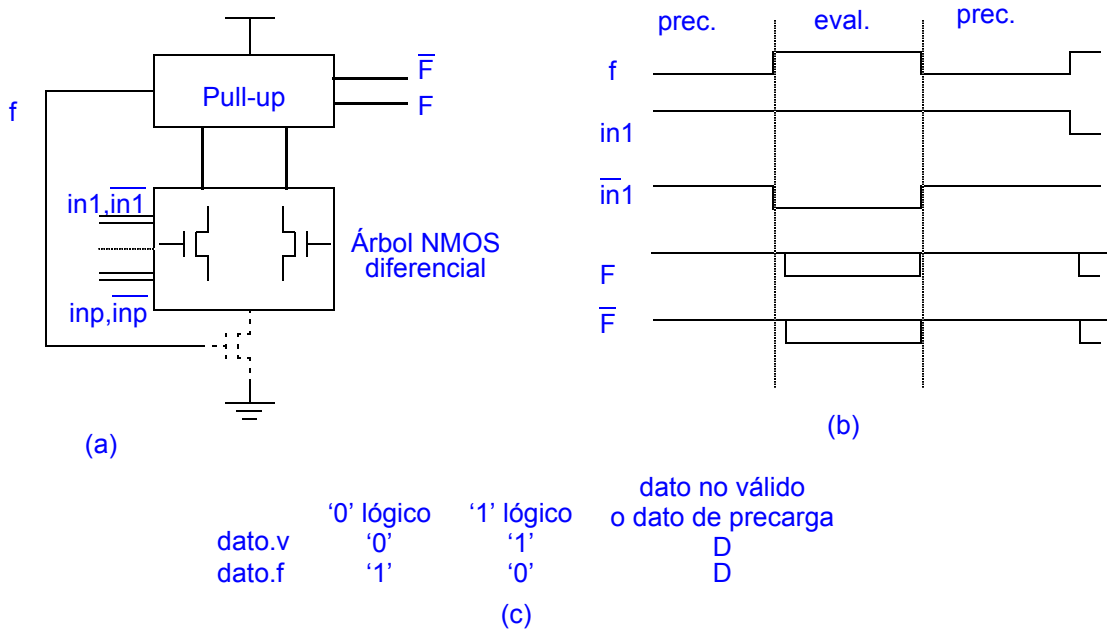


Figura 2.35. (a) Estructura a nivel de bloques de una familia diferencial genérica, (b) su forma de operación., y (c) la codificación en doble raíl.

La operación de la estructura diferencial como ya hemos dicho es muy similar a la de una familia dinámica, dividida en fases de evaluación y precarga. Por lo tanto, existe una señal de control de operación, ϕ, que determinará las fases de operación de la estructura. Dicha señal (que estará conectada al bloque de carga y al transistor NMOS de precarga si existiese) habilitará o deshabilitará el camino entre el árbol diferencial y los nodos de salida, y entre el árbol y el nodo de tierra (en caso de existir el transistor NMOS de precarga).

Cuando la señal de control tiene un valor '0' (por ser el indicado en la figura 2.35b), la estructura se encuentra en la fase de precarga. En dicha fase, los caminos entre el árbol y los nodos de salida se encuentran deshabilitados, y el bloque de carga colocará en la señal de salida el dato de precarga o no válido.

Cuando la señal de control toma el valor ‘1’, la estructura se encuentra en la fase de evaluación. En dicha fase los caminos entre el árbol y los nodos de salida están habilitados, pero sólo una rama del árbol conducirá por lo que solamente habrá un camino entre un nodo de salida y el nodo de tierra. En cambio, el nodo complementario de salida se quedará con su valor de precarga, y de este modo los dos cables de la señal de salida tomarán valores diferentes, es decir, tendremos la codificación de un ‘0’ o un ‘1’ lógico (dependiendo de la rama del árbol que conduzca).

Para conseguir el comportamiento anterior, los bloques de las principales estructuras encontradas en la literatura suelen contar con al menos uno de las etapas mostradas en la figura 2.36: transistores PMOS de precarga, con el fin de colocar los datos de precarga; par inversor-transistor PMOS realimentado, para regenerador posibles ‘1’ degradados o forzar un almacenamiento estático; y amplificadores de sensibilidad, con el fin de acelerar la velocidad de operación (ya que no hace falta que la conducción de una rama sea completa para forzar un nivel lógico no degradado).

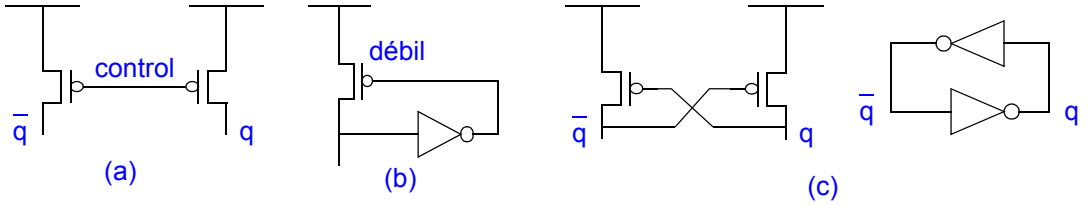


Figura 2.36. Etapas típicas del bloque de carga: (a) transistores PMOS de precarga, (b) par transistor débil PMOS-inversor y (c) amplificador de sensibilidad.

Entre las principales ventajas de las familias diferenciales, podemos destacar las siguientes:

- Alta velocidad de operación, debido a las fases de evaluación y precarga.
- Posible implementación de funciones complejas utilizando una sola estructura diferencial, debido al árbol NMOS diferencial. Las ramas de este árbol no tienen porqué ser independientes, como sucedía con la familia de transistores de paso por lo que el número de transistores no tiene porqué crecer drásticamente con la complejidad de la función. A modo de ejemplo, mostramos el árbol diferencial correspondiente a una puerta XOR de tres entradas. Podemos apreciar que el número de transistores es el mismo en los casos de árbol no diferencial y del diferencial a pesar de que el árbol diferencial tiene una salida adicional, el valor falso. Es más, si utilizásemos una operación XOR de más entradas, el árbol diferencial tendría menos entradas que el no diferencial, ya que el árbol diferencial únicamente necesita añadir cuatro transistores por entrada, mientras que el árbol no diferencial necesita duplicar su tamaño por cada entrada adicional (aunque en realidad es son más transistores que el doble).

Entre los principales inconvenientes de las familias diferenciales, podemos destacar los siguientes:

- La codificación en doble raíl implica generar el doble de cableado por cada señal, situación que puede aumentar considerablemente el área ocupada por el circuito. No obstante, esta situación se puede mejorar generando localmente los valores negados

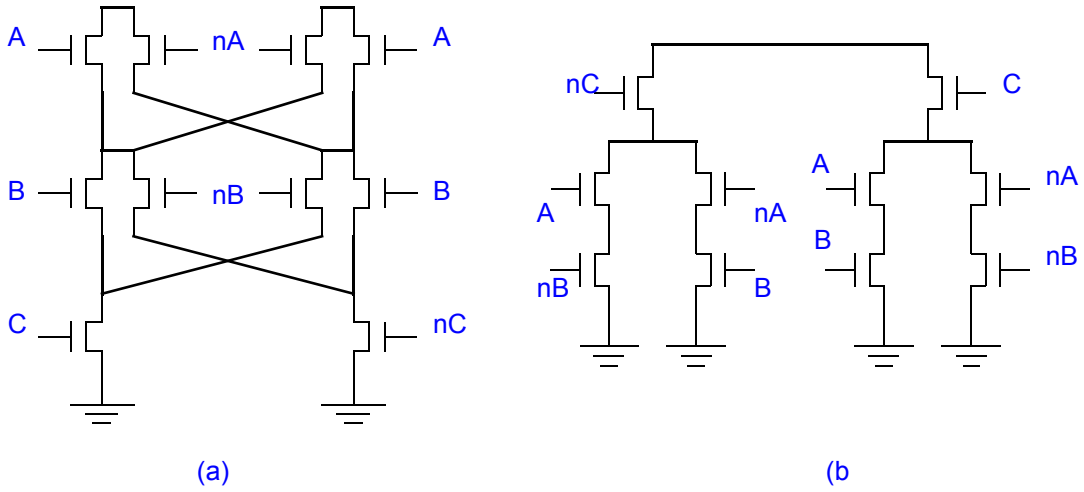


Figura 2.37. (a) Árbol NMOS diferencial y (b) árbol NMOS no diferencial correspondiente a la operación XOR de tres entradas

ya que la codificación no válida en las entradas no es esencial para el correcto funcionamiento de la estructura diferencial.

2.3.1. Diseño de una celda diferencial

Abordar la tarea de diseño de una celda diferencial es la misma que abordar el diseño de un árbol diferencial, ya que dicha parte es común a todas las estructuras (puesto que la diferencia entre las distintas familias diferenciales se encuentra en el bloque de carga y la existencia o no del transistor NMOS de precarga).

Para diseñar estos árboles podemos decantarnos por métodos basados en mapas (como el mapa de Karnaugh) o en árboles (como el ROBDD). Debido a que la manipulación de los BDD ya ha sido introducida para llevar a cabo una implementación a nivel de transistores, vamos a decantarnos por este método. Para la utilización del método del mapa, habría que modificarlo para poder compartir transistores y no operaciones lógicas completas.

En la figura 2.38 mostramos el OBDD, ROBDD y el árbol NMOS diferencial de una puerta XOR de tres entradas. Prácticamente el diseño es el mismo que en el caso del árbol de transistores de paso con leves diferencias:

- Como no se utilizan transistores de paso, las señales de entrada no pueden estar conectados a los terminales de fuente, luego los nodos terminales siempre serán ‘1’ y ‘0’, que en este caso vamos a etiquetarlos como T (true) y F (false).
- La jerarquía del árbol será la contraria, es decir, el nodo que está solo es el último, y estará conectado al terminal de masa. En cambio los nodos T y F estarán conectados a las salidas del árbol diferencial: salida.v y salida.f respectivamente. En el caso de que la estructura sea tal que complemente las salidas, las ramas se intercambiarían, y salida.v estaría conectada al terminal F y viceversa.
- Para que la reducción del árbol sea todavía mayor, en la figura 2.38d mostramos una nueva regla que sería equivalente a la equivalencia booleana.

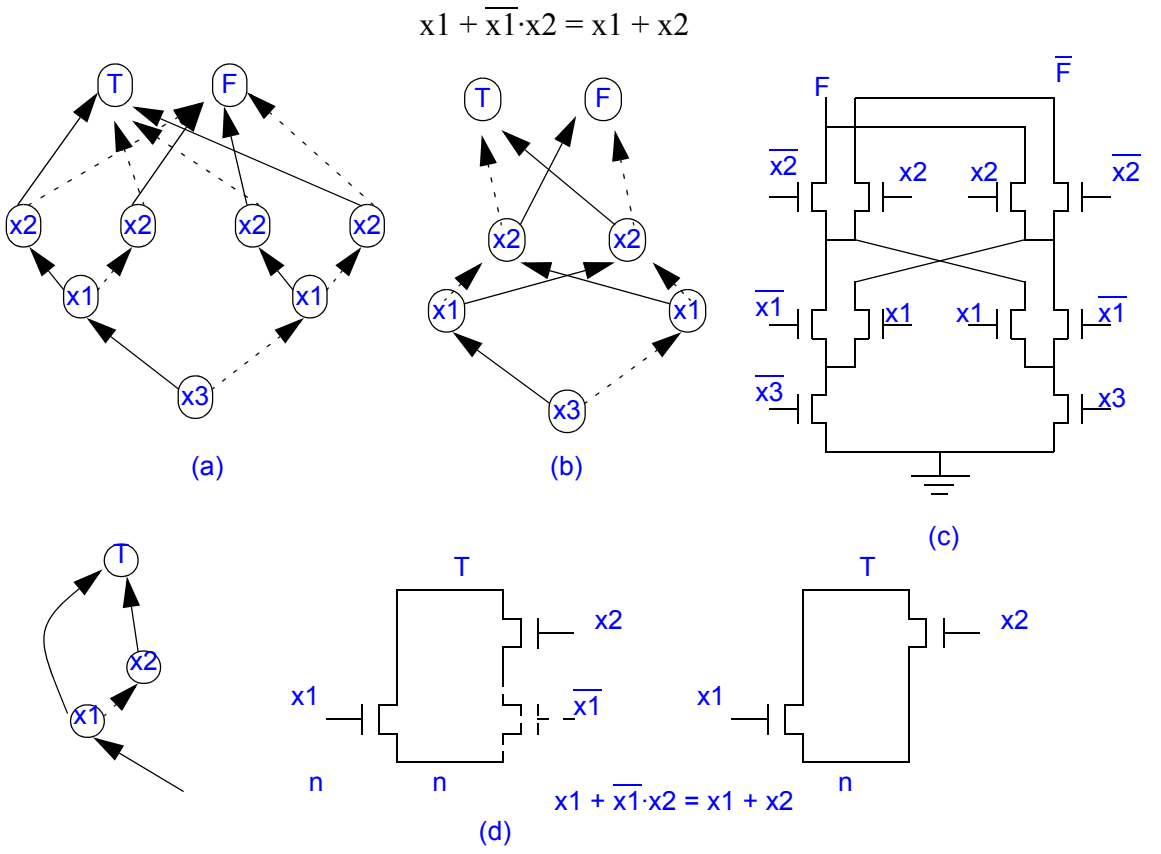


Figura 2.38. (a) OBDD, (b) ROBDD y (c) árbol NMOS diferencial correspondiente a la operación XOR de tres entradas.
(d) Regla de reducción al pasar del ROBDD al árbol diferencial.

Veamos el ejemplo de la función de la figura 2.21. En la figura 2.39 se muestra todo el proceso de diseño. De nuevo empezamos por una representación estándar de la función lógica cuya celda queremos obtener, como puede ser una tabla de combinaciones.

Como el proceso de diseño se va a llevar a cabo mediante diagramas de decisión binaria, debemos obtener un ordenamiento bueno de las señales de entrada, tal que el número de nodos y arcos finales tienda al mínimo. Por ejemplo utilizaremos el algoritmo heurístico por pesos, para lo cual debemos previamente obtener la solución mínima a nivel de puertas para estudiar la influencia de cada una de las entradas en la función. Para dicho propósito utilizaremos el mapa de Karnaugh obteniendo dos ordenamientos diferentes para cada una de las salidas. El tratamiento independiente de cada una de las salidas viene motivado porque cada salida debe estar implementada en una celda diferente, luego los árboles diferenciales deben ser independientes.

Tras la obtención del ordenamiento, generamos el OBDD, el cual será reducido para llegar al ROBDD, utilizando las reglas de reducción vistas previamente. Una vez que tengamos los ROBDDs, sólo debemos aplicar la reglas de cambiar nodo por conexión y arco por transistor controlado por la señal del nodo previo.

Por último, a los árboles diferenciales de transistores NMOS, hay que añadirle el bloque de carga (y el transistor NMOS de precarga, si lo hubiera) para obtener una celda diferencial concreta.

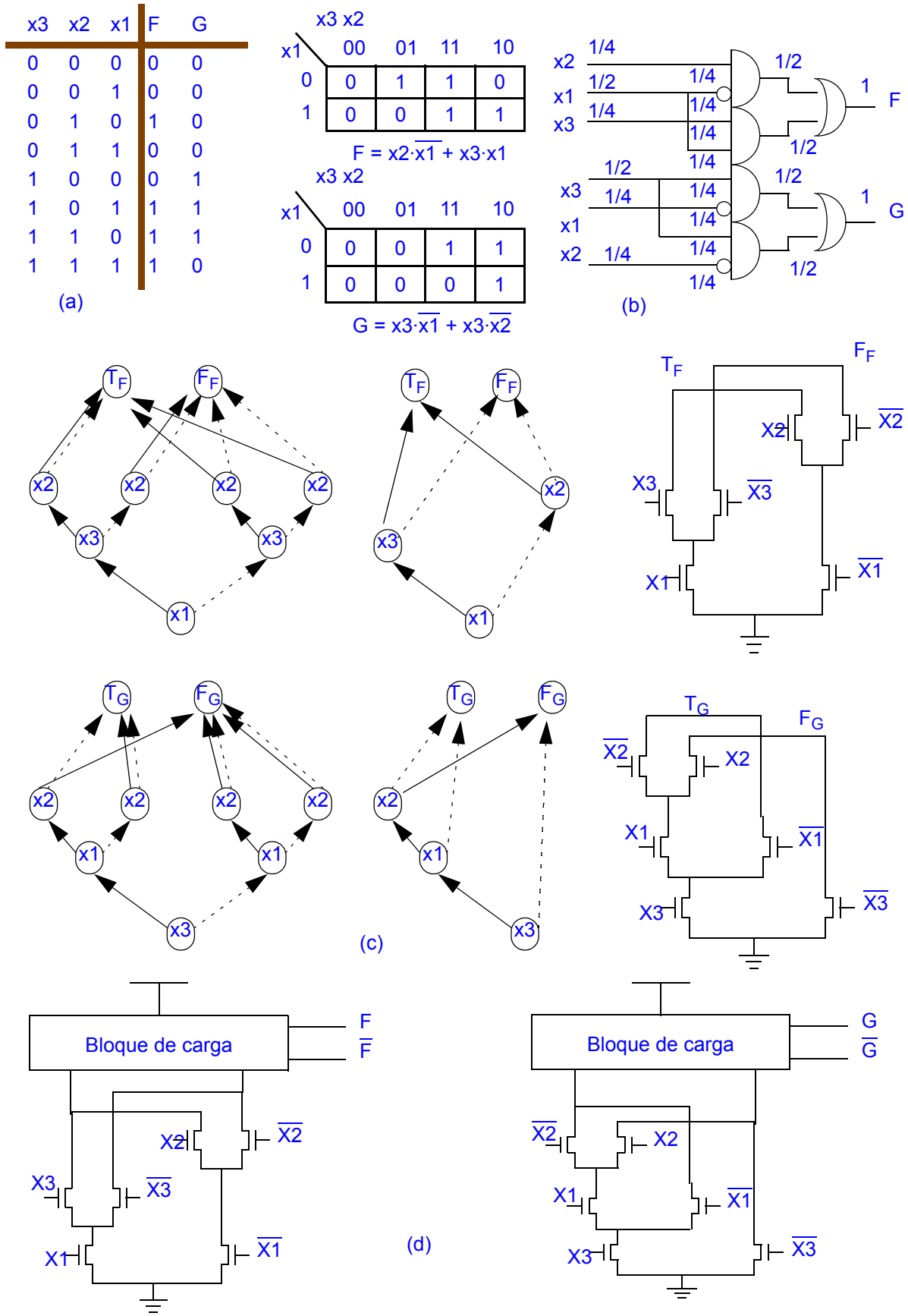
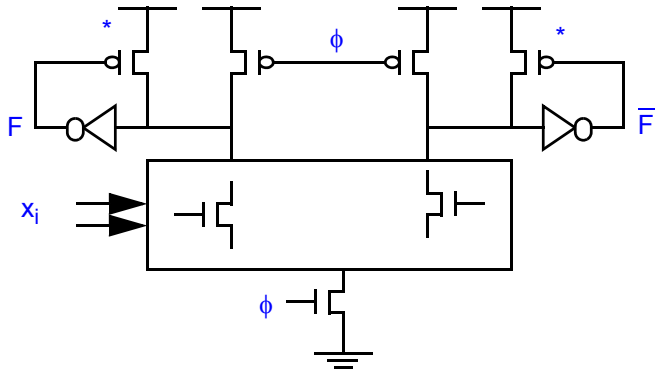


Figura 2.39. (a) Tabla de combinaciones de la función a implementar; (b) obtención de un ordenamiento bueno para la generación del OBDD; (c) obtención del ROBDD y del árbol diferencial NMOS; y (d) obtención de las celdas diferenciales.

2.3.2. Tipos de familias diferenciales

Podemos encontrar bastantes familias que siguen la arquitectura mostrada en la figura 2.35. Todas ellas tienen en común el árbol diferencial, mientras que el bloque de carga determina las características de cada una de ellas. Entre todas ellas, vamos a considerar tres a modo de ejemplo: DCVSL (Differential Cascode Voltage Switch Logic), ECDL (Enable/Disable Differential Logic) y SODS (Swithed Output Differential Structure).

El esquema a nivel de transistores de una puerta DCVSL genérica se muestra en la figura 2.40. Como cualquier estructura diferencial, su operación está dividida en fases de precarga y de evaluación, controladas por la señal ϕ. Cuando la señal ϕ toma el valor '0', la estructura se encuentra en fase de precarga. En dicha fase, el árbol NMOS diferencial se encuentra deshabilitado ya que el camino al nodo de tierra está cortado por el transistor NMOS de precarga. Los transistores PMOS de precarga conducen por lo que colocan un nivel alto en la salida del árbol; que por la acción de los inversores de salida, ambos nodos toman un valor ‘0’ (siendo éste un dato no válido). Cuando la señal ϕ toma el valor ‘1’, la estructura se encuentra en la fase de evaluación. En dicha fase, el transistor NMOS de precarga conduce conectando el nodo de tierra al árbol diferencial. Los transistores PMOS de precarga están cortados, por lo que la rama del árbol que conduce colocará en su salida un ‘0’ (el nodo de tierra) que, con la ación del inversor de salida, pasará a un ‘1’ en el nodo correspondiente de la estructura diferencial. Por contra, la salida del árbol cuya rama no conduce mantiene el nivel alto de la fase de precarga (en principio un almacenamiento dinámico); no obstante, la conexión entre el inversor de salida y el transistor PMOS * fuerza a que el almacenamiento anterior sea estático, es decir, lograr una estructura semiestática. Para garantizar un correcto funcionamiento, se debe forzar a que el transistor PMOS * sea más resistivo que la rama del árbol para que el ‘0’ debido a la rama sea fuerte y no entre en la zona prohibida debido al choque con el ‘1’ de la fase de precarga.



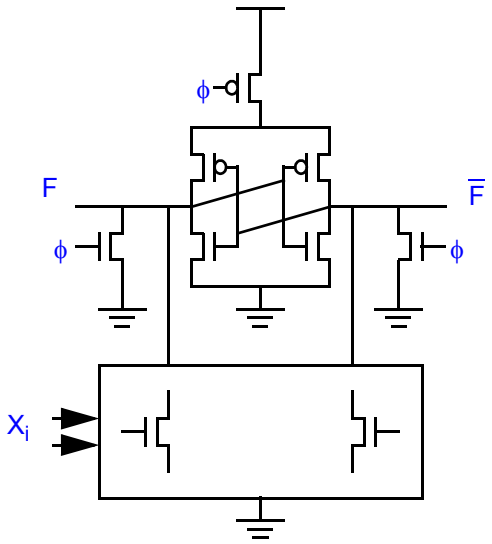
Dato	F $\overline{F}$
No válido	0 0
0	0 1
1	1 0

Figura 2.40. Esquema genérico de una puerta DCVSL, junto con la codificación de sus señales.

Las principales características de la familia DCVSL son las siguientes:

- La existencia de los inversores de salida provocará un aumento en el fan-out de la puerta, que coincidirá con el fan-out de un inversor. Por contra, el retraso y consumo de potencia de dichos inversores habrá que añadírselos al de la estructura diferencial por mantener una conexión serie.
- El transistor NMOS de precarga provocará un aumento en el retraso y consumo de potencia ya que aumentará la capacidad del árbol diferencial.

El esquema a nivel de transistores de una puerta ECDL genérica se muestra en la figura 2.41. Como en el caso anterior, las fases de operación están controladas por la señal ϕ. Cuando dicha señal toma el valor '1', la estructura se encuentra en fase de precarga. En dicha fase, los transistores NMOS de precarga colocan en ambos nodos de salida un nivel bajo. Esta situación no provoca ninguna comunicación entre tierra y polarización, ya que el transistor PMOS de precarga (única conexión con polarización) está cortado. Cuando la señal ϕ toma el valor '0', la estructura se encuentra en fase de evaluación. En dicha fase, los transistores NMOS de precarga están cortados por lo que la conexión al nodo de tierra se debe a una de las ramas del árbol diferencial. El nodo de salida, cuya rama no conduce, tomará el valor '1' debido al par de inversores realimentados (cuya conexión a polarización está habilitada por la conducción del transistor PMOS de precarga).



Dato	F $\overline{F}$
No válido	0 0
0	0 1
1	1 0

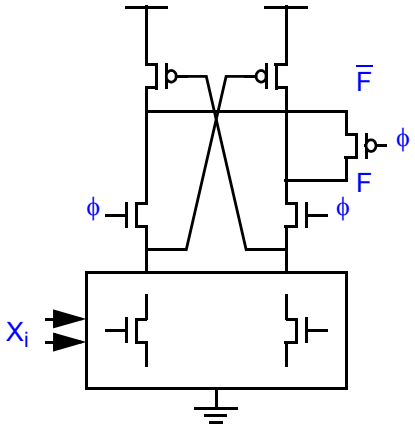
Figura 2.41. Esquema genérico de una puerta ECDL, junto con la codificación de sus señales.

Las principales características de la familia ECDL son las siguientes:

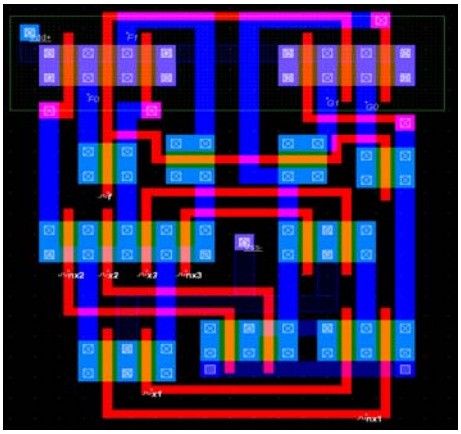
- La utilización de la pareja de inversores realimentados produce un aumento en la velocidad, ya que la carga y descarga de los nodos de salida no se deben únicamente al árbol diferencial. De hecho, el árbol sólo comenzará la operación ya que cuando los niveles estén lo suficientemente definidos, la pareja de inversores acabará la operación más rápidamente que si toda la operación recayera sobre el árbol diferencial. A dichos inversores se les conoce como amplificadores de sensibilidad puesto que amplifican los niveles lógicos.
- Los transistores NMOS de precarga dispuestos en esta configuración no aumentarán la cadena de transistores conectados en serie del árbol diferencial.

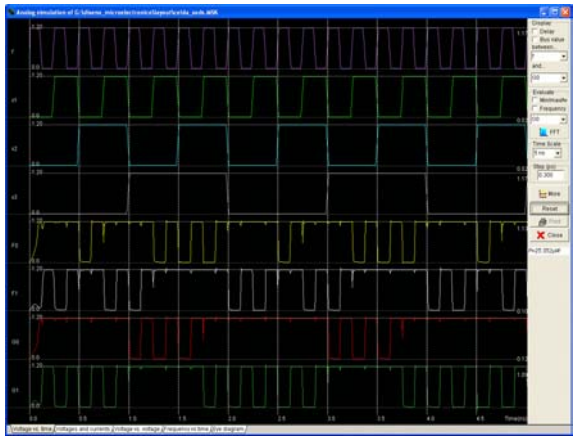
El esquema a nivel de transistores de una puerta SODS genérica se muestra en la figura 2.42. De nuevo la señal ϕ controla la operación de la estructura. Cuando la señal ϕ toma el valor '1', la estructura se encuentra en la fase de evaluación. En dicha fase, el transistor PMOS de precarga está cortado por lo que los nodos de salida no se encuentran cortocircuitados, y los transistores NMOS de precarga conectan las salidas del árbol diferencial con las salidas de la estructura. Con este contexto, una de las ramas del árbol conduce colocando en la salida correspondiente un '0'. Además, el '0' de la salida del árbol colocará un '1' en la salida com-

plementaria debido al transistor PMOS realimentado. cuando la señal ϕ toma el valor '0', la estructura se encuentra en la fase de precarga. En dicha fase, los transistores NMOS de precarga se encuentran cortados deshabilitando la conexión entre las salidas del árbol y las salidas de la estructura diferencial. Como una de las ramas conduce, el nodo de salida complementario tomará un valor '1' debido al transistor PMOS realimentado (como sucedía en la fase de evaluación). Dicho '1' pasará a la salida restante a través del transistor PMOS de precarga, que en esta fase sí conduce.



Dato	F F̄
No válido	1 1
0	0 1
1	1 0





5.34 μm x 5.96 μm = 31.7 μm^2

Figura 2.42. Esquema genérico de una puerta SODS, junto con la codificación de sus señales.

Las principales características de la familia SODS son las siguientes:

- Los transistores PMOS realimentados forman un amplificador de sensibilidad, pero de un solo nivel (en este caso particular, el nivel alto). Como en el caso anterior, este amplificador aumentará la velocidad del sistema.
- La obtención del valor de precarga no involucra directamente a la polarización (como en el caso DCVSL) ni a la tierra (como en el caso ECDL). Esta situación provoca una disminución en el consumo de potencia.
- Los transistores NMOS de precarga no pasan bien el '1' desde la salida de la estructura a la del árbol. Por lo tanto, en la nueva evaluación, la cantidad de tensión que se debe descargar (swing) a través del árbol es menor, aumentando la velocidad de la estructura.

2.4. Familias en modo corriente

Las familias en modo de corriente son aquellas en las que su comportamiento depende explícitamente del valor de la intensidad que circula a través de ella. Un esquema genérico de una familia de corriente se muestra en la figura 2.43. Podemos destacar tres bloques bien diferenciados:

- Una fuente de intensidad, que generará la intensidad que pasará a través de la puerta en todo momento.
- Un árbol NMOS, que generará la función lógica implementada por la puerta en cuestión.
- Y un bloque de descarga, por el cual pasará la corriente cuando el árbol NMOS esté cortado.

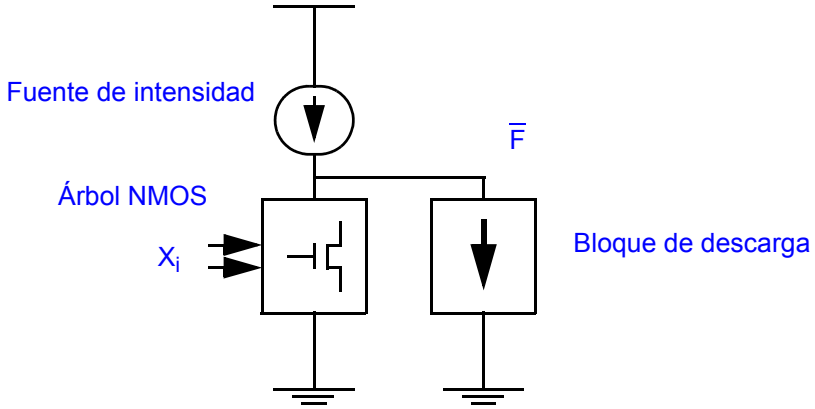


Figura 2.43. Esquema genérico de una familia en modo de corriente no diferencial.

El funcionamiento de dicha familia es el siguiente. Cuando los valores de las señales de entrada hagan conducir el árbol NMOS, la señal de salida (negada) tomará el valor '0' (que coincidirá con el terminal de tierra). En dicho caso, el bloque de descarga estará cortado, por lo que toda la intensidad fluirá por el árbol NMOS. Por contra, cuando los valores de las señales de entrada provoquen el corte del árbol NMOS, toda la intensidad fluirá por el bloque de descarga colocando en la salida un nivel alto. Hay que indicar que el nivel alto no coincide con la tensión de polarización (como sucede en el resto de familias vistas anteriormente), sino que depende directamente de la intensidad suministrada por la fuente, luego los niveles lógicos de esta familia no son compatibles con los de las anteriores, necesitando un conversor de niveles si se quieren conectar puertas de este tipo de familias con puertas de las anteriores. Una restricción del nivel alto es que debe ser lo suficientemente elevado para que conduzca el bloque de descarga y los transistores NMOS de los árboles de las siguientes puertas.

También podemos encontrar familias en modo corriente diferenciales, las cuales siguen el esquema mostrado en la figura 2.44. En ella podemos apreciar que se trata de una familia no diferencial *doblada* (claro está cambiando los dos árboles NMOS por un árbol diferencial). No obstante es de destacar una nueva fuente de intensidad conectada al terminal de tierra del árbol diferencial para garantizar que pasa la misma intensidad por las dos ramas del árbol.

El interés por este tipo de familias se ha renovado debido al creciente interés de los siste-

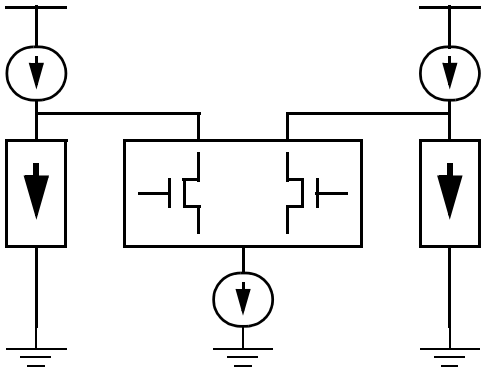


Figura 2.44. Esquema genérico de una familia en modo de corriente diferencial.

mas de señal mixta (analógico-digital) implementados en un mismo sustrato. En dichos sistemas es necesario que la fuente de polarización sea lo más ideal posible, es decir, que no existan picos y la distribución de tensión (e intensidad) sea lo más uniforme posible, aun cuando se produzcan cambios en las señales digitales, como se puede en la figura 2.45. En ella podemos ver que con cada conmutación de las señales digitales, en la fuente de polarización (y en el nodo de tierra) se produce un pico de intensidad debido al cambio de estado de los transistores, lo cual se conoce como ruido de conmutación.

> El ruido de conmutación se puede definir como cualquier desviación de las fuentes de polarización debido a la conmutación de las señales digitales, lo cual se traduce en la presencia de picos de intensidad y tensión en la fuente de polarización y el nodo de tierra.

Estos cambios pueden reducir las prestaciones de las partes más sensibles de un sistema (las cuales son generalmente las zonas analógicas). En las familias en modo corriente, la presencia de la fuente de intensidad produce la uniformidad requerida (que será mayor cuanto más se acerque a una fuente ideal).

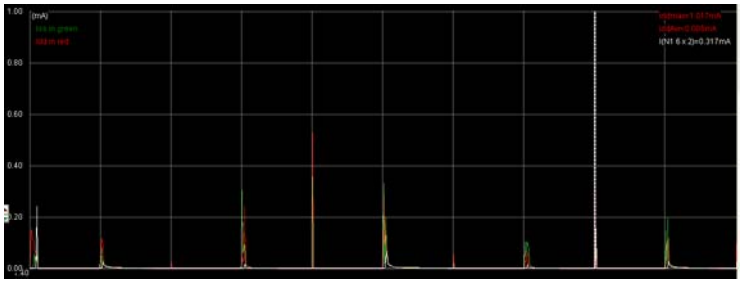


Figura 2.45. Formas de onda de las intensidades de polarización y de tierra para el caso de un sumador completo.

Las principales características arquitecturales de las familias en modo de corriente son las siguientes:

- La presencia de la fuente de intensidad potencia la idealidad de la fuente de polarización, de tal forma que reduce el ruido de conmutación. Existen muchas técnicas para crear fuentes de intensidad, y hay que llegar a un compromiso entre la complejidad arquitectural y el carácter más ideal de la fuente (el hecho de que su entorno no afecte

al comportamiento de la fuente). En la figura 2.46 mostramos varias arquitecturas simples de fuentes de intensidad basadas en un transistor MOS (figuras a y b). Para un aislamiento mejor de la fuente del resto del sistema se suelen utilizar estructuras de espejos de corriente como el mostrado en la figura 2.46c.

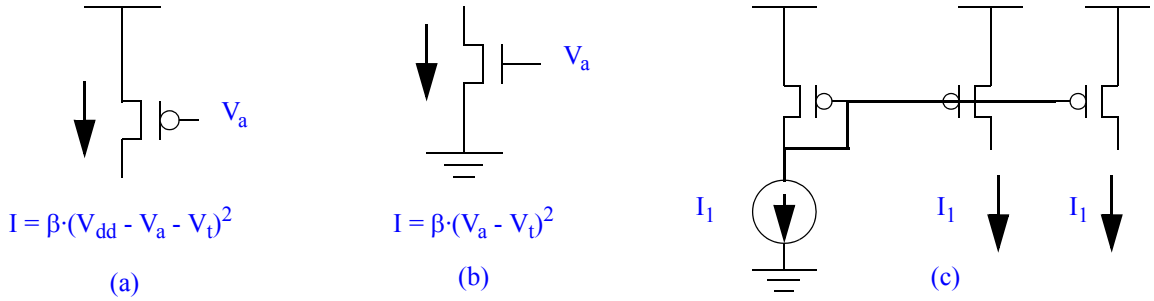


Figura 2.46. Fuente de intensidad generada con (a) un transistor PMOS y (b) con un transistor NMOS. (c) Espejo de corriente.

- El árbol NMOS será el mismo que para la familia CMOS estática.
- El bloque de descarga puede ser un diodo (un transistor conectado como diodo), o una fuente controlada por tensión.

Entre las principales ventajas de las familias en modo de corriente podemos destacar las siguientes:

- Las principales características de la puerta, tanto de tensión como de tiempo, muestran una insensibilidad relativamente alta con respecto a los parámetros tecnológicos y a la fuente de polarización. De hecho es debido a que la influencia con respecto a la fuente de intensidad es muy superior y anula al resto de dependencias.
- Es una familia de muy bajo ruido, ya que la fuente de intensidad absorve todas las desviaciones del carácter uniforme de la fuente de polarización.

En cuanto a los principales inconvenientes de este tipo de familias, podemos destacar los siguientes:

- La presencia de la fuente de intensidad provoca la existencia de un consumo de potencia en estática, equivalente a V_{dd}·I, por lo que estamos ante unas familias que no se deberían utilizar para aplicaciones de bajo consumo. Esta situación ha motivado que su uso se limite a las zonas de mayor sensibilidad ante el ruido, es decir, las zonas adyacentes a los circuitos analógicos.
- Por lo general, los niveles lógicos no son compatibles con los de otras familias, por lo cual es necesario añadir lógica extra para la conversión entre los diferentes niveles lógicos para la conexión con puertas de otras familias, como se desprende del punto anterior.

2.4.1. Diseño de celdas de familia en modo de corriente.

El diseño de celdas de esta familia consistirá en la obtención del árbol NMOS de la familia CMOS estática (para familias no diferenciales) o del árbol NMOS diferencial (para familias

diferenciales). Si nos centramos en las familias no diferenciales, el proceso de diseño es muy similar al de la familia CMOS estática.

Por lo tanto, el proceso de diseño de las celdas en modo corriente se basa en la ejecución de los siguientes pasos:

- Obtención de la función lógica (por ejemplo su tabla de verdad)
- Minimización lógica, para obtener una fórmula mínima que se traducirá en un área ocupada mínima (por ejemplo los métodos del mapa de Karnaugh o de McCluskey)
- Aplicar las reglas de conectividad de la familia CMOS para el árbol NMOS.

Veamos un ejemplo completo partiendo desde el comportamiento lógico, mostrado en la figura 2.47. Supongamos que deseamos diseñar dos celdas en modo corriente para las siguientes funciones:

$$F(x_3, x_2, x_1) = \sum m\ (2, 5, 6, 7)$$

$$G(x_3, x_2, x_1) = \sum m\ (4, 5, 6)$$

En primer lugar obtenemos una representación genérica de las funciones lógicas que queremos diseñar. En este caso hemos utilizado las tablas de verdad de las funciones F y G.

Una vez que tenemos la representación anterior, pasamos al proceso de minimización. En este caso hemos elegido el método del mapa de Karnaugh, con lo que las fórmulas obtenidas son:

$$F = x2\cdot\overline{x1} + x3\cdot x1$$

$$G = x3\cdot\overline{x1} + x3\cdot\overline{x2}$$

Una vez que tenemos las fórmulas mínimas podemos utilizar las reglas de conexión. No obstante, cabe destacar que va a existir un transistor de cada árbol por cada literal (variable o variable negada) que aparezca en la fórmula. Luego, con el fin de reducir al mínimo el número de transistores, se potenciará la utilización del factor común, y así se evitará la repetición de estructuras de transistores. Así, la implementación de G será la correspondiente a la siguiente fórmula:

$$G = x3\cdot(\overline{x1} + \overline{x2})$$

Es de destacar la existencia de los inversores de salida para obtener la función correcta.

2.4.2. Tipos de familias en modo de corriente.

Debido a la relativamente reciente aparición de las familias en modo de corriente y a su aplicación tan específica, el número de este tipo de familias es relativamente reducido. En este caso vamos a presentar las más significativas, como son las familias CSL (Current Steering Logic), CBL (Current Balanced Logic) y FSCL (Folded Source Coupled Logic), las dos primeras familias no diferenciales mientras que la tercera es una familia diferencial.

La familia CSL sigue la estructura mostrada en la figura 2.48. En dicho esquema podemos identificar la fuente de intensidad, que ha sido implementada con un transistor PMOS

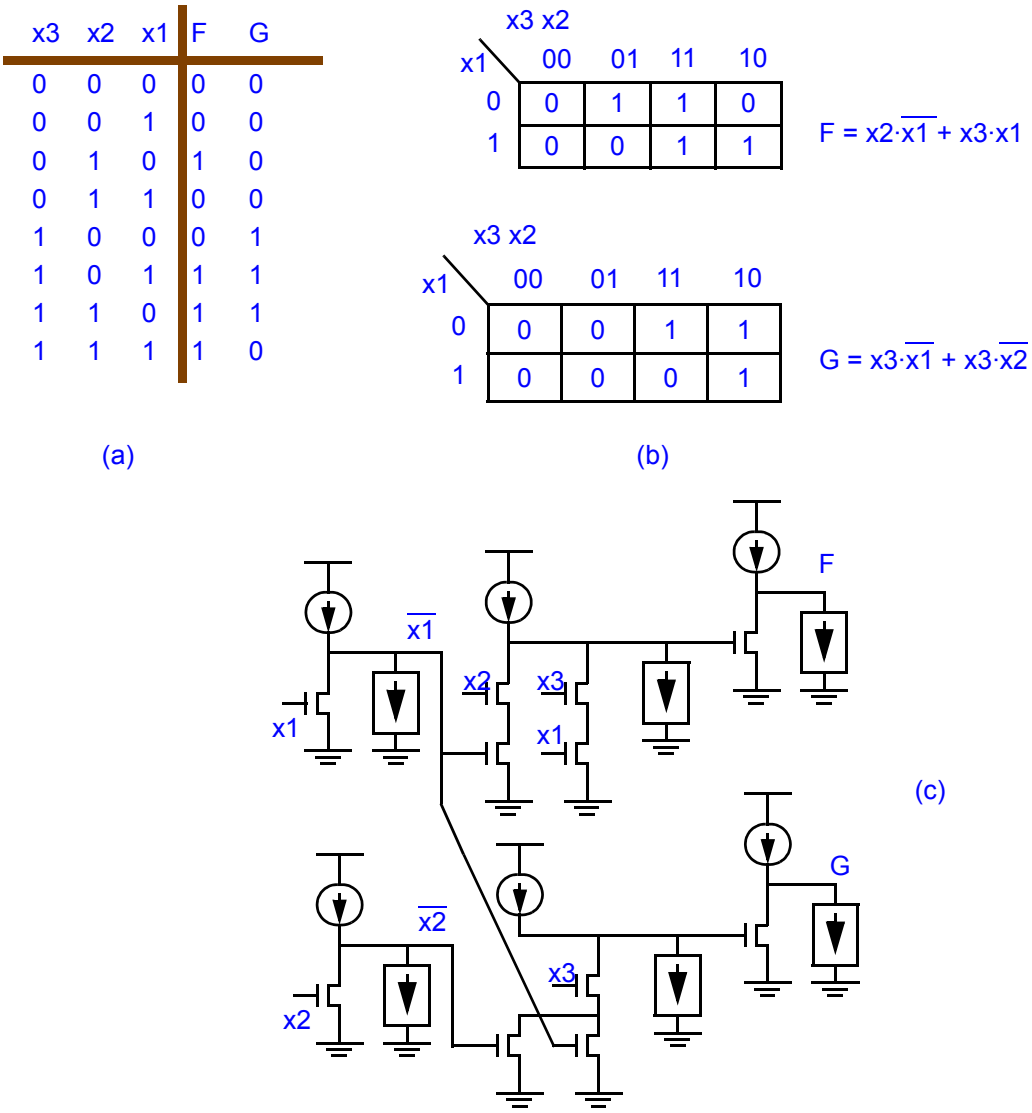


Figura 2.47. Ejemplo de diseño de celdas en modo corriente. (a) Función lógica, (b) minimización, (c) aplicación de las reglas de conectividad.

polarizado en zona de saturación. Así mismo, el bloque de descarga está compuesto por un transistor NMOS conectado como diodo (cortocircuitando los terminales de puerta y drenador), de tal forma que cuando en la salida hay un nivel alto, el diodo conduce y pasa la intensidad por él; mientras que si hay un nivel bajo, el diodo está cortado y no pasa intensidad por él.

La familia FSCL sigue la estructura mostrada en la figura 2.49. Podemos apreciar que dicho esquema es un *duplicado* de la familia CSL (uno por cada rama del árbol diferencial). Debemos destacar que para garantizar que por todo el árbol diferencial circula la misma intensidad, independientemente de los valores de las señales de entrada, se ha colocado una nueva fuente de intensidad e el terminal de tierra del árbol, cuyo valor será diferente (pero forzosamente menor) que las fuentes de transistores PMOS. Por lo tanto, por los bloques de descarga siempre circulará intensidad, la única diferencia será la cantidad que circule, que será diferente si su rama del árbol conduce o no conduce. En este caso, como en el anterior, la cantidad de intensidad suministradas por las fuentes depende del terminal de puerta y de las dimensiones de los transistores.

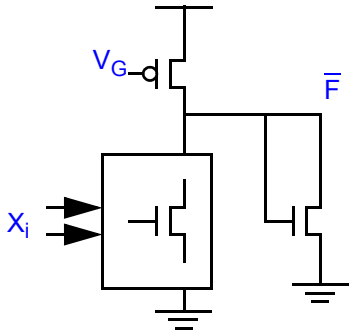


Figura 2.48. Esquema genérico de la familia CSL.

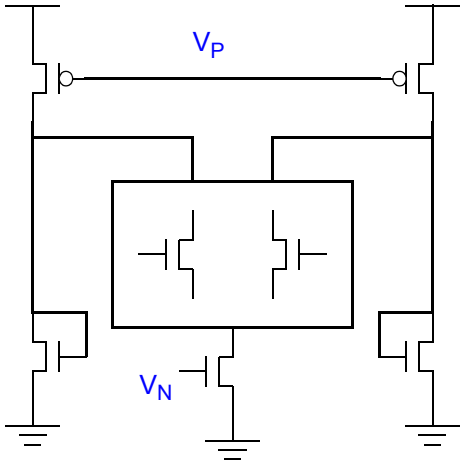


Figura 2.49. Esquema genérico de la familia FSCL.

La familia CBL sigue la estructura mostrada en la figura 2.50. De dicho esquema podemos destacar dos caminos de intensidad bien diferenciados, de los cuales únicamente conducirá uno de ellos en función del valor lógico de la señal de salida. En el camino de descarga principal, tenemos una fuente de intensidad compuesta por un transistor PMOS en la zona de conducción y el árbol NMOS; mientras que el segundo camino de descarga está compuesto por un transistor NMOS (conectando polarización y tierra), el cual es controlado por la salida (negada) de la puerta lógica. En este caso hay que igualar concienzudamente la fuente de intensidad y el camino de descarga para que siempre pase la misma intensidad (ya que los caminos por los que pasa la intensidad desde la polarización son diferentes). Como en este caso no existe ninguna tensión que controle la intensidad, únicamente hay que encontrar unas dimensiones idóneas para cada uno de los transistores. En la figura 2.50 mostramos un posible layout de la celda en modo de corriente del ejemplo de la figura 2.47c, en el cual no se han incluido los inversores de entrada. Dicho layout tendría unas dimensiones de 3.96 μm x 4.38 μm, con un área igual a 17.4 μm^2. La verificación del layout se ha realizado a través de una simulación mostrada en la misma figura. Se puede comprobar que obtenemos la misma tabla de comportamiento que la de partida.

En este caso podemos destacar que los niveles lógicos coinciden con los raíles de polarización, por lo que son compatibles con el resto de familias lógicas vistas en el tema.

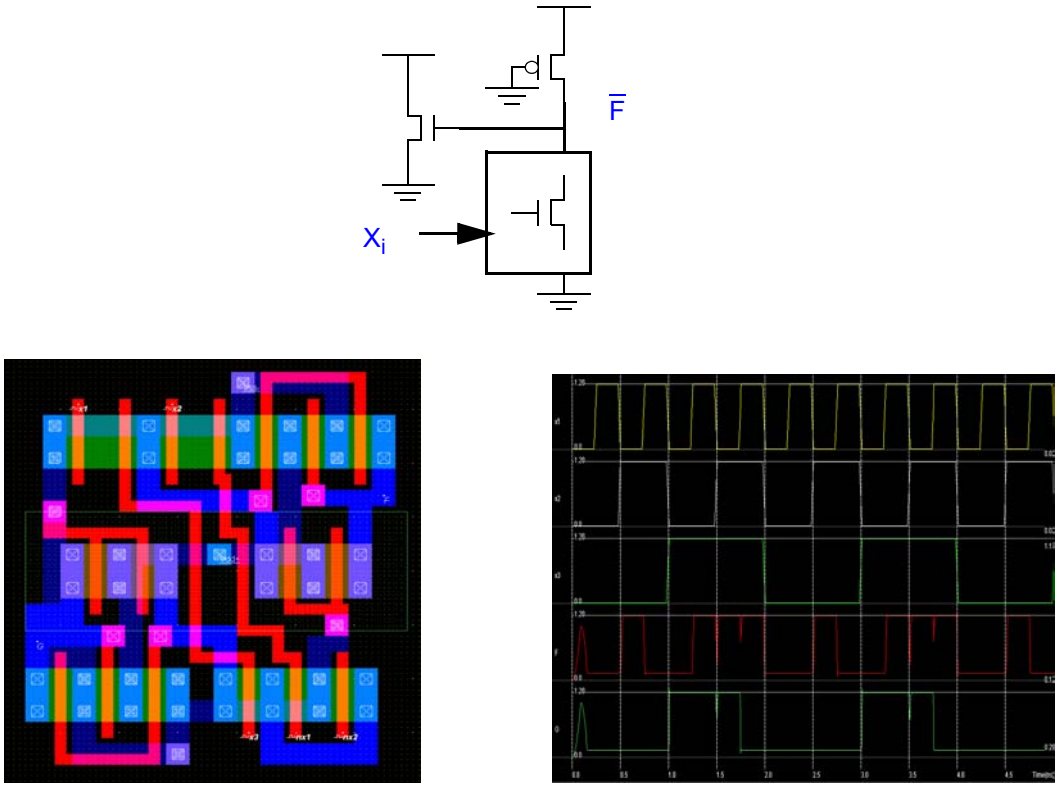


Figura 2.50. Esquema genérico de la familia CBL. Layout y simulación de verificación del circuito mostrado en la figura 2.47c (sin incluir los inversores de entrada).