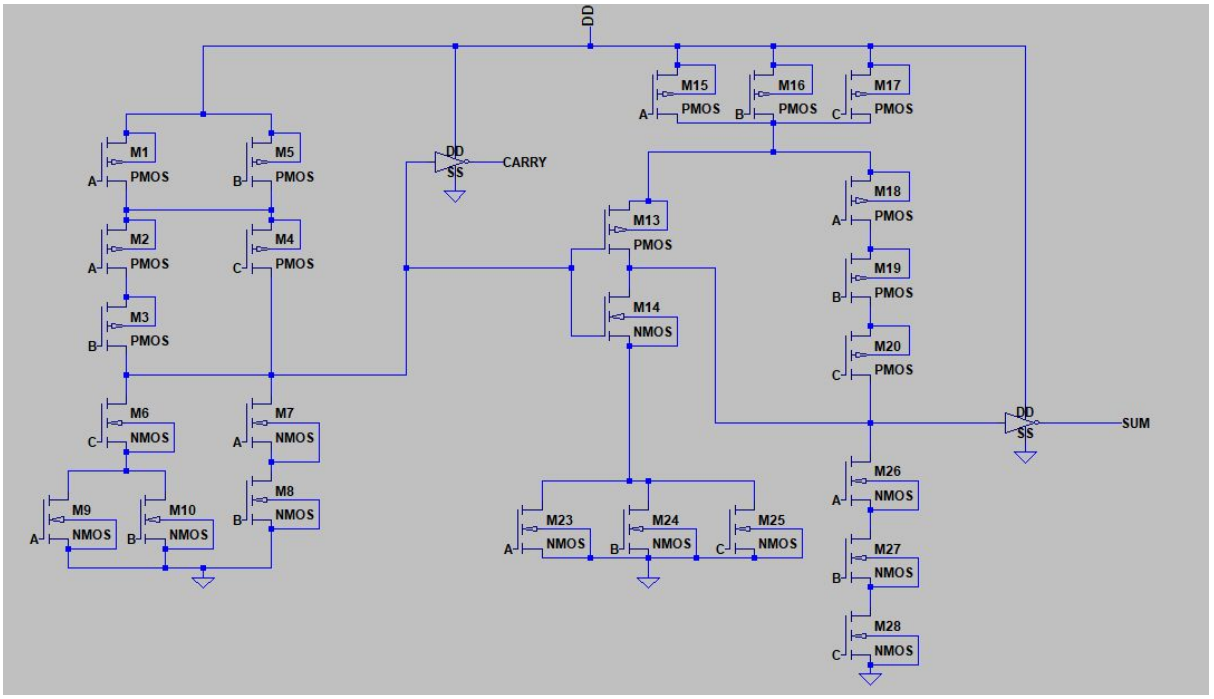
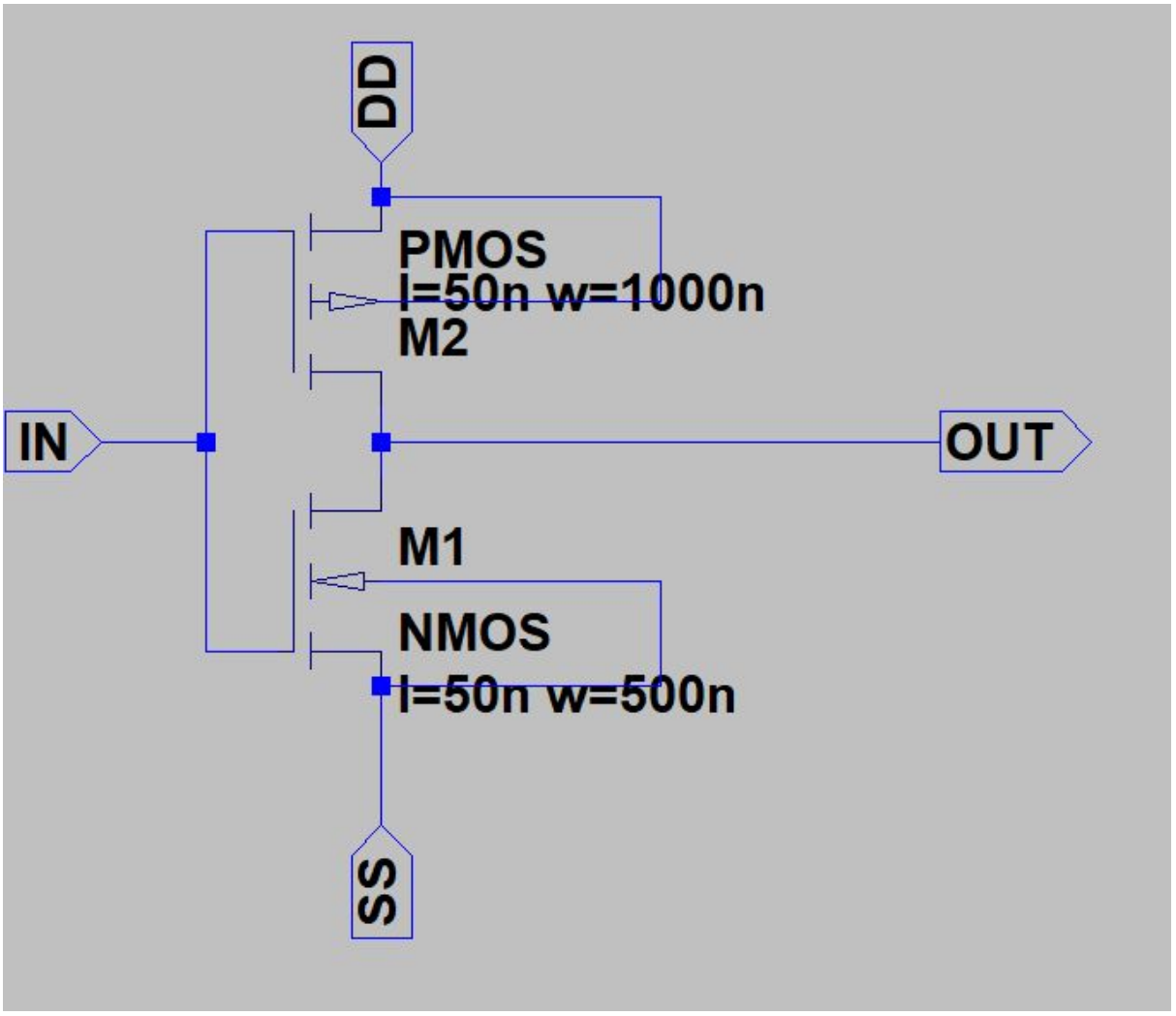
Infome Práctica 5 FuTFI

1.

Montamos el siguiente circuito (1.asc):



En este circuito, para los inversores, he definido un símbolo de la siguiente forma:



A las salidas, como indica el enunciado, he conectado 4 inversores adicionales.

Para obtener retardos de propagación simétricos, tendremos que buscar un factor de escalado apropiado para los transistores PMOS y NMOS. Para ello, partimos de que $R_P = 2R_N$ (con la misma anchura y longitud, la resistencia de un PMOS es el doble).

Vamos a jugar con la anchura, por lo que lo que buscaremos será disminuir o igualar la resistencia con respecto a un transistor de tamaño mínimo (no se puede aumentar, por eso son de tamaño mínimo).

Las anchuras las calculamos según la siguiente fórmula:

$\frac{R}{s_1} + ... + \frac{R}{s_n} \leq R$ (R, en el miembro de la izquierda, será R si es un transistor NMOS, o 2R si es PMOS, por lo que acabamos de comentar arriba).

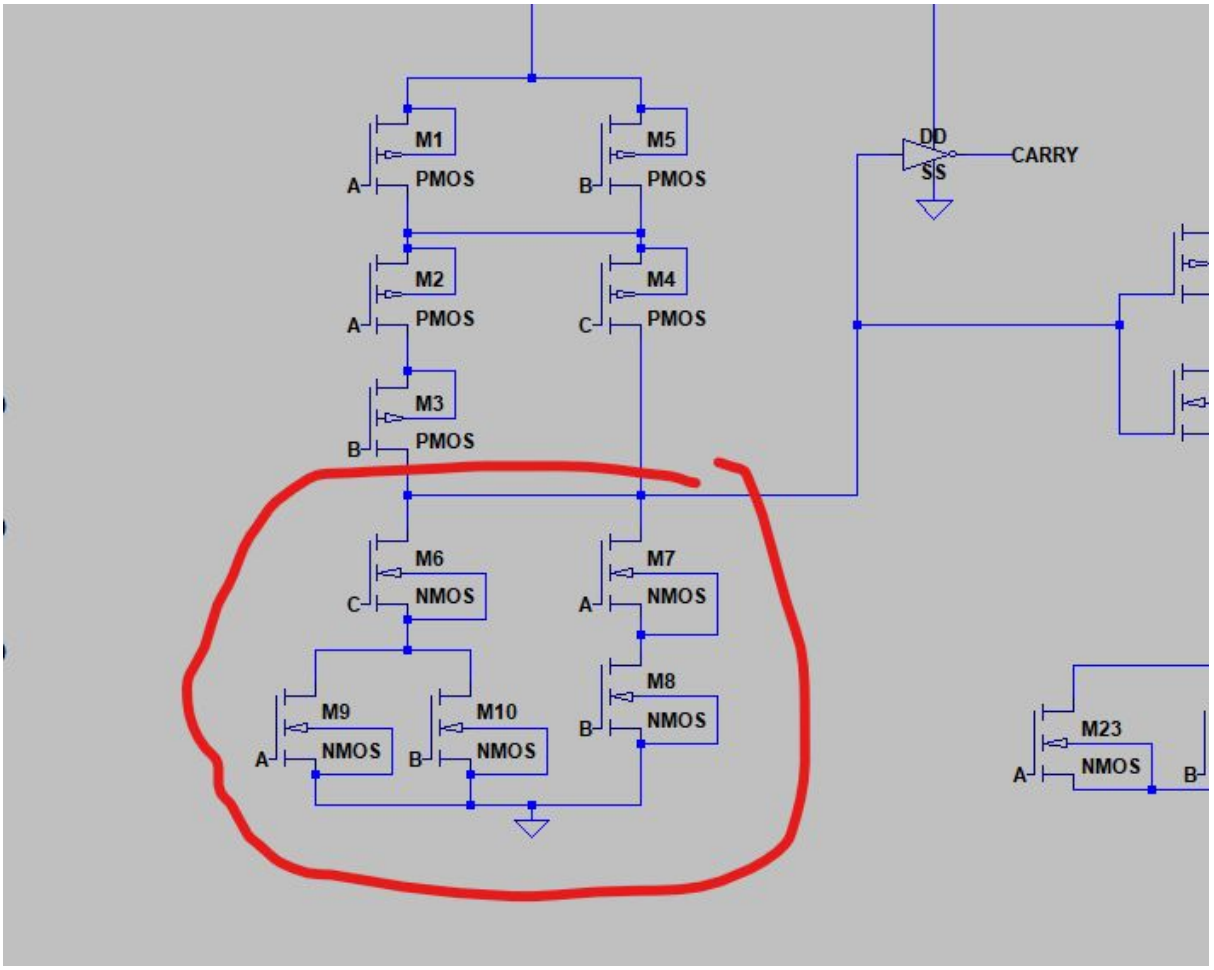
Pese a todo, en casos triviales (asociación de resistencias iguales en serie), la fórmula se puede simplificar como que s debe ser igual para todas las R (ya que son iguales):

$n\frac{R}{s} \leq R \Rightarrow n \leq s$, y en este informe vamos a coger $s = n$ en los casos en que sea posible. De no serlo, cogeremos el entero más próximo.

Como esta fórmula es directa (el factor de escalado es igual al total de transistores que están en serie), voy a ahorrarme especificar explícitamente todos los cálculos en los casos triviales. También es importante tener en cuenta que trataremos a los transistores en

paralelo que sean iguales (vemos un ejemplo justo al principio de los cálculos) como casos equivalentes. Así, de nuevo, nos ahorramos explicitar bastantes cálculos.

Empezamos por esta parte del circuito:



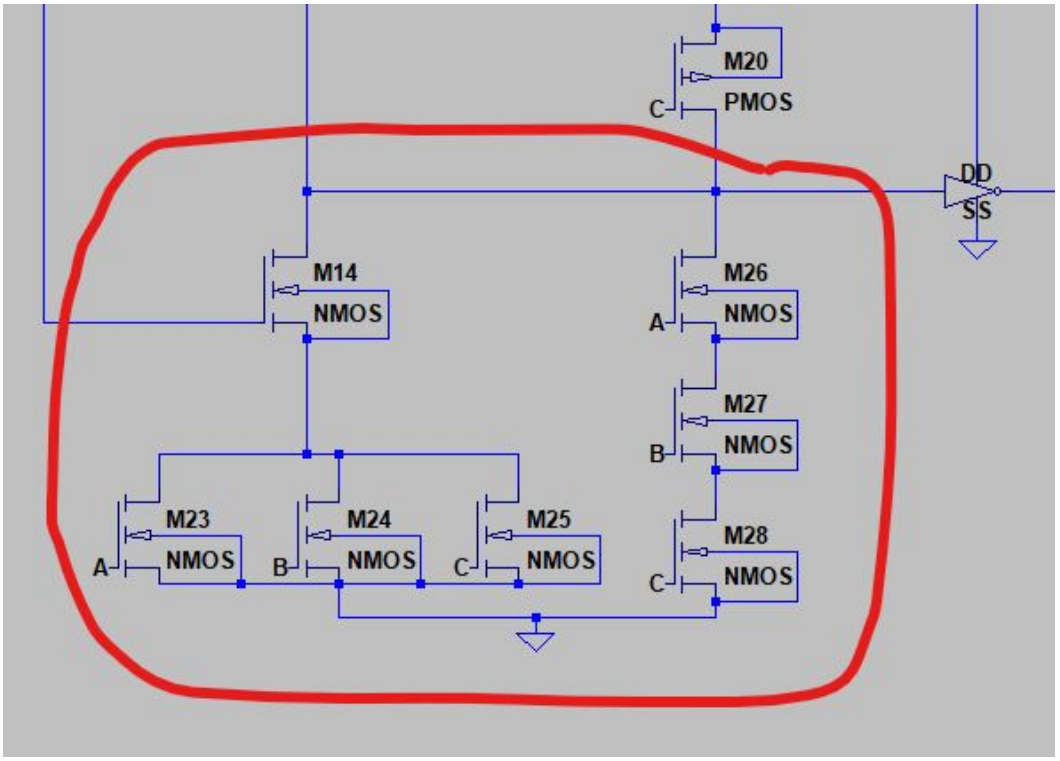
El peor caso será cuando se pongan en ON solamente A y B, A y C o B y C. Los tres casos son equivalentes, porque tendremos dos transistores en ON y el resto en OFF. Así que queremos que dos transistores sean iguales al NMOS de un inversor:

(Tomamos $R = R_N$, para abreviar)

$(R + R)/s = R \Rightarrow s = 2$

El factor de escalado de los NMOS será 2 (1000nm).

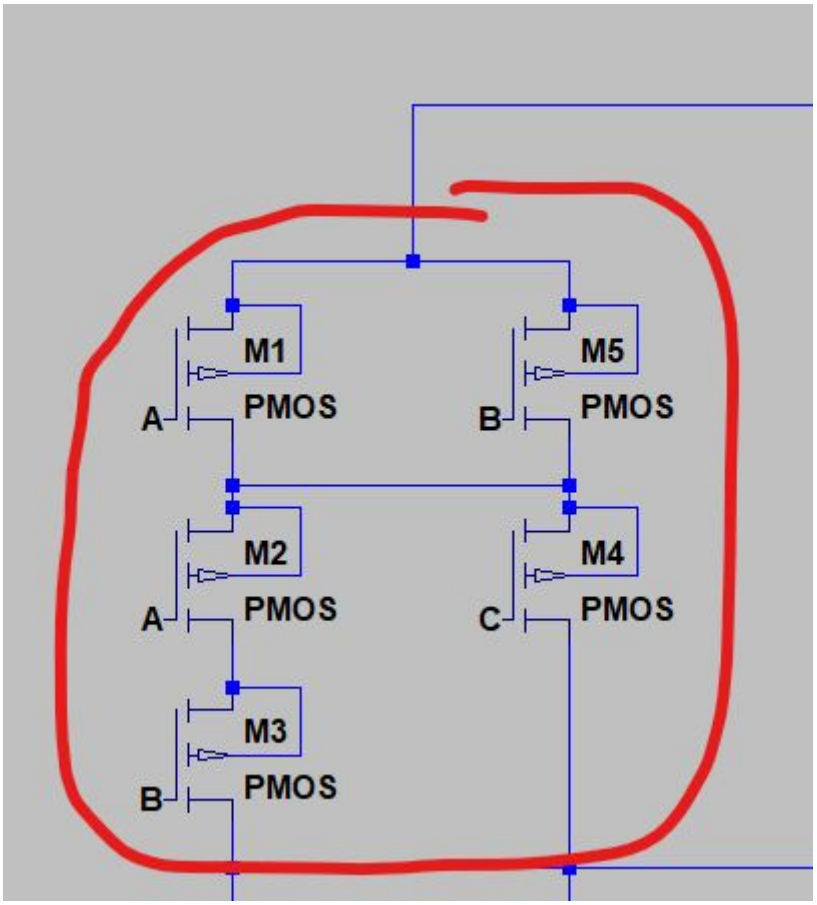
Saltamos a otra parte del circuito:



El peor caso será en el que acumulamos 3 transistores en serie, la rama de la derecha. En ese caso, guiándonos por la tabla de verdad, el NMOS de arriba a la izquiera (el negado de CARRY, valdrá 0, ya que CARRY será 1). Por eso la rama de la izquierda no estará activada. Entonces, los transistores de la derecha, deberán tener un factor de escalado de 3, con cálculos análogos a los que hicimos previamente. (1500nm)

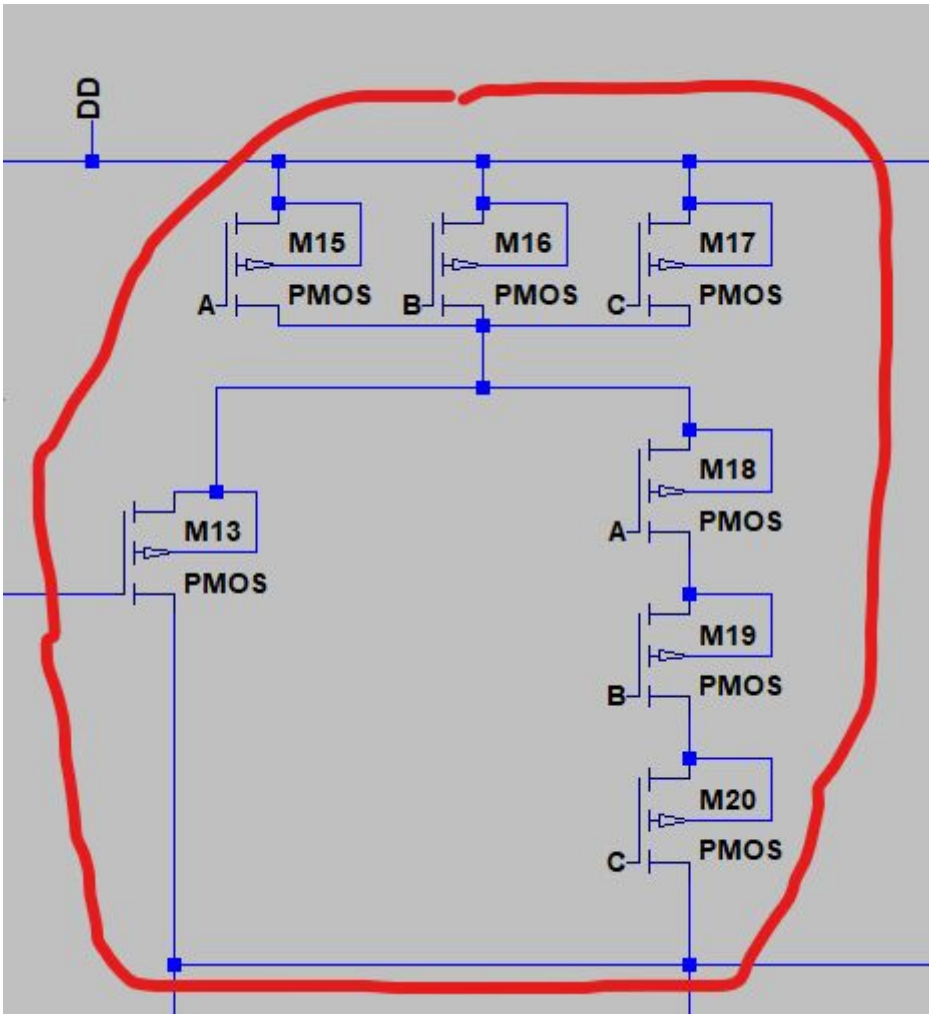
Para la rama de la izquierda, los transistores que dependen de A, B y C tendrán igual factor de escalado por estar en paralelo. Si tomamos el caso de que sólo esté activado C, y por tanto el transistor que depende del negado de CARRY también está en ON, entonces tenemos un factor de escalado de 2 (1000nm), para ambos transistores. El mismo proceso se puede aplicar a los transistores que dependen de A y B, ya que en los tres casos el transistor de arriba estará en ON. De esta forma, obtenemos que para ambas ramas, el retardo de propagación será menor o igual al de un inversor de tamaño mínimo.

Pasamos ahora a los transistores PMOS:



En esta parte del circuíto, el peor caso será el de tres transistores en serie. Por ello, tendremos un factor de escalado de 3*2 = 6 (multiplicamos por 2 el factor de escalado porque, como se ve en el Anexo 1, los transistores PMOS presentan el doble de resistencia que los PMOS para el tamaño mínimo), para los transistores que dependen de A y B.

Para el transistor que depende de C, su peor caso será con dos transistores, por lo que tendremos un primer transistor con factor de escalado 6 (lo acabamos de calcular), que afectará al factor de escalado del segundo. Este factor de escalado deberá ser de 3, ya que sustituye a los dos transistores que tendríamos en la rama de la izquierda (6/2 = 3). Esto es lógico: si dos transistores iguales en ON nos dan una resistencia concreta, estando en serie (se suman sus resistencias) y queremos conseguir esa misma resistencia con un único transistor en ON, entonces este transistor deberá tener la resistencia de la suma de dos transistores, es decir, el doble de uno de ellos. Y eso es lo mismo que dividir la anchura por la mitad.



Pasamos, finalmente, a nuestra última parte del circuito. Aquí el peor caso sería pasar de '0' a '1' (LH), con una rama de 4 transistores. El transistor que depende del negado de CARRY permanecería en OFF, si A = B = C = '1'. Por eso el factor de escalado de los transistores que dependen de A, B y C debe ser 8 (2*4).

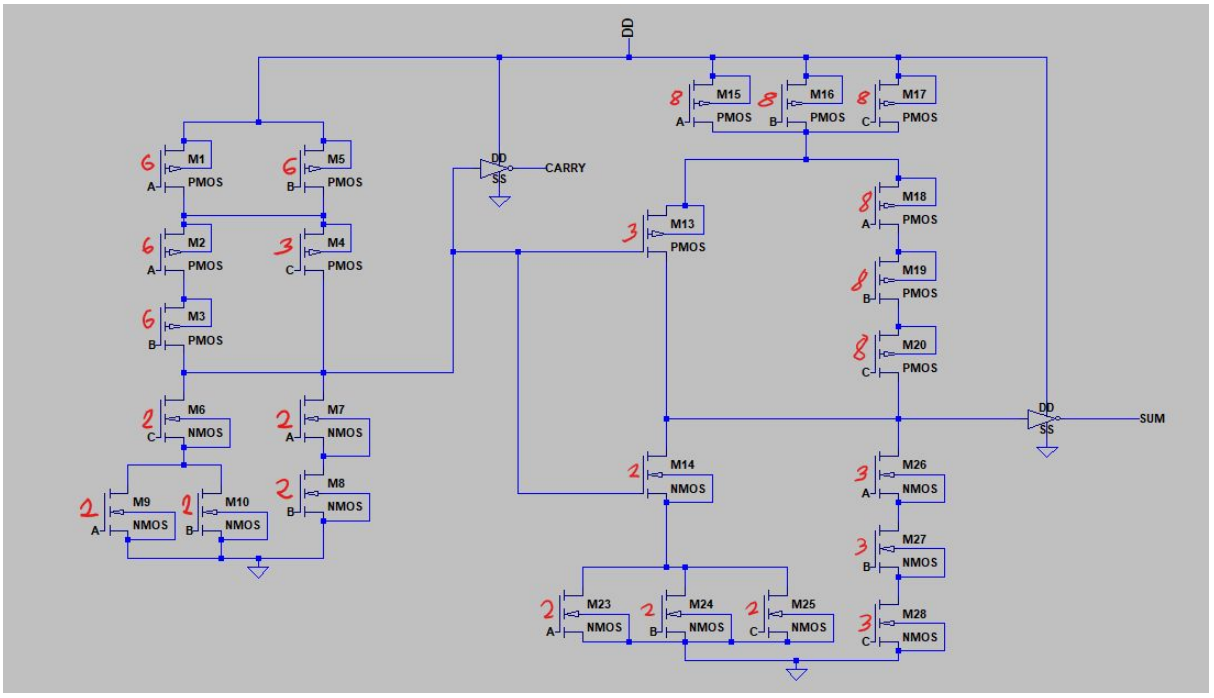
Si tomamos la rama de la izquierda, tenemos un primer transistor (de los de arriba que están en paralelo) que ya sabemos que tiene factor de escalado 8. Por tanto, queremos conseguir que:

$$\frac{2R}{8} + \frac{2R}{s} \leq R \Rightarrow \frac{2R}{s} \leq R - \frac{2R}{8} \Rightarrow \frac{2R}{R-\frac{2R}{8}} \leq s (teniendo\ en\ cuenta\ que\ s > 0)$$

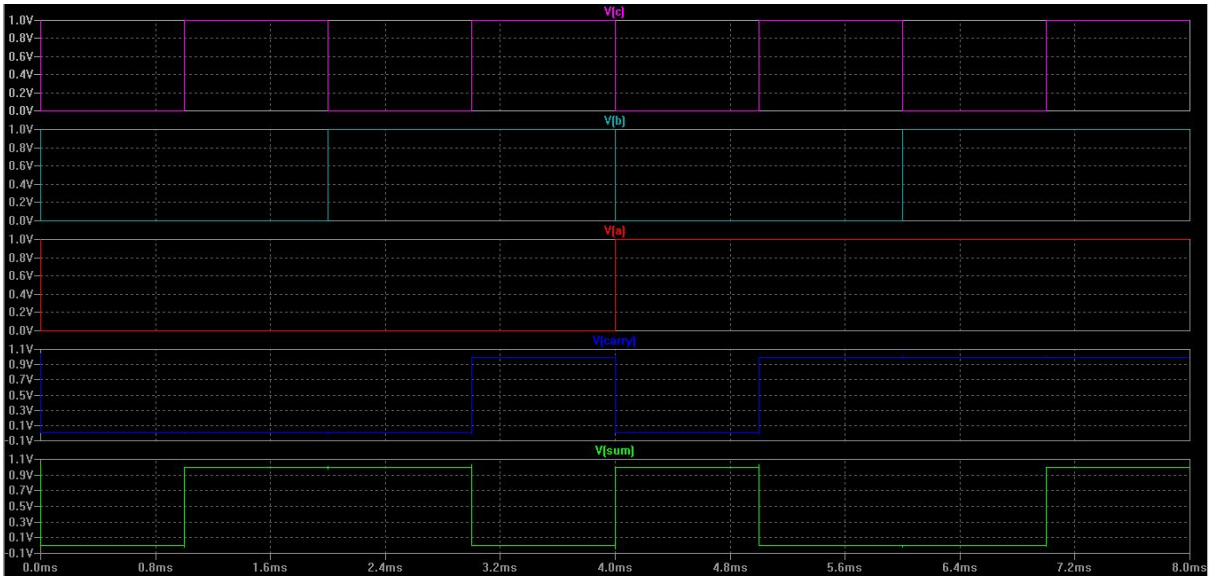
$$\Rightarrow \frac{2R}{\frac{6R}{8}} \leq s \Rightarrow \frac{16}{6} \leq s \Rightarrow 2.66 \leq s$$

Y como queremos un factor de escalado entero, cogemos 3 para seguir cumpliendo la inecuación.

Ahora que hemos calculado las anchuras de todos los transistores, nos queda el siguiente diseño (las anchuras están indicadas en rojo):



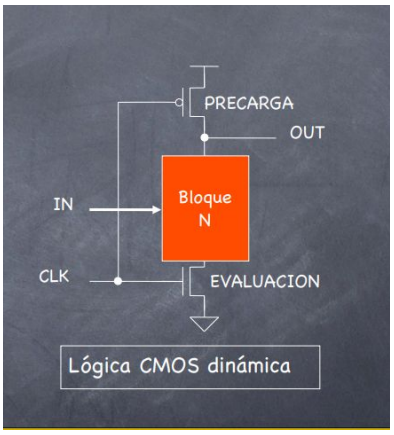
Simulando en LTSPICE, obtenemos las siguientes gráficas:



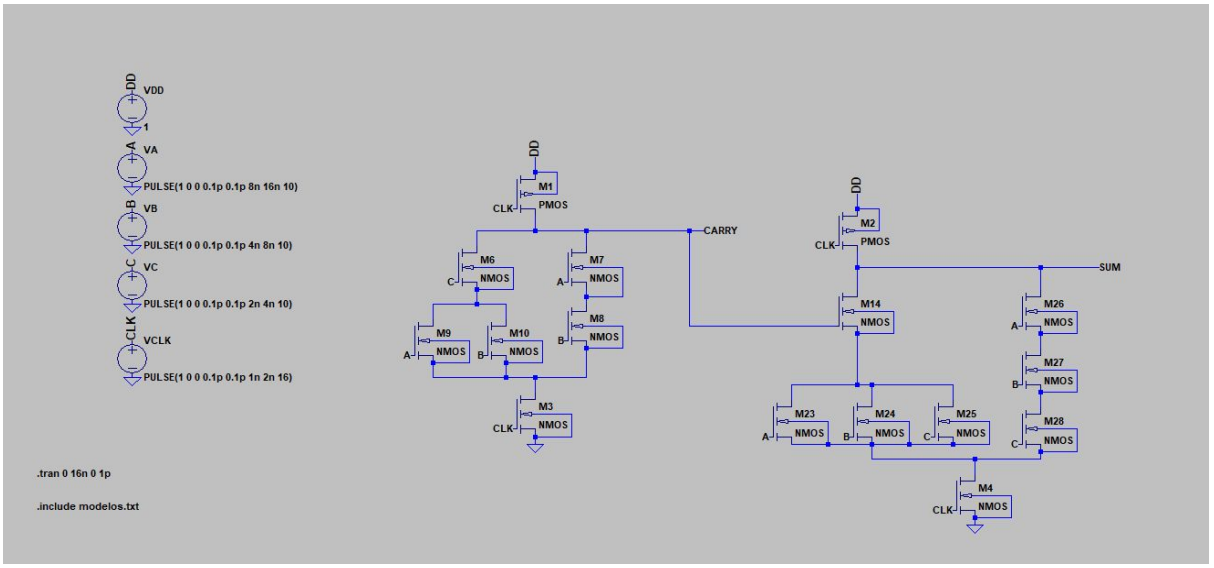
Como vemos, CARRY y SUM coinciden con los valores que deseábamos en la tabla de verdad. Si nos fijamos en detalle, se pueden apreciar pequeños picos en el voltaje al cambiar de un estado a otro, pero son casi despreciables (producidos por la superposición de estados al cambiar de uno a otro).

2.

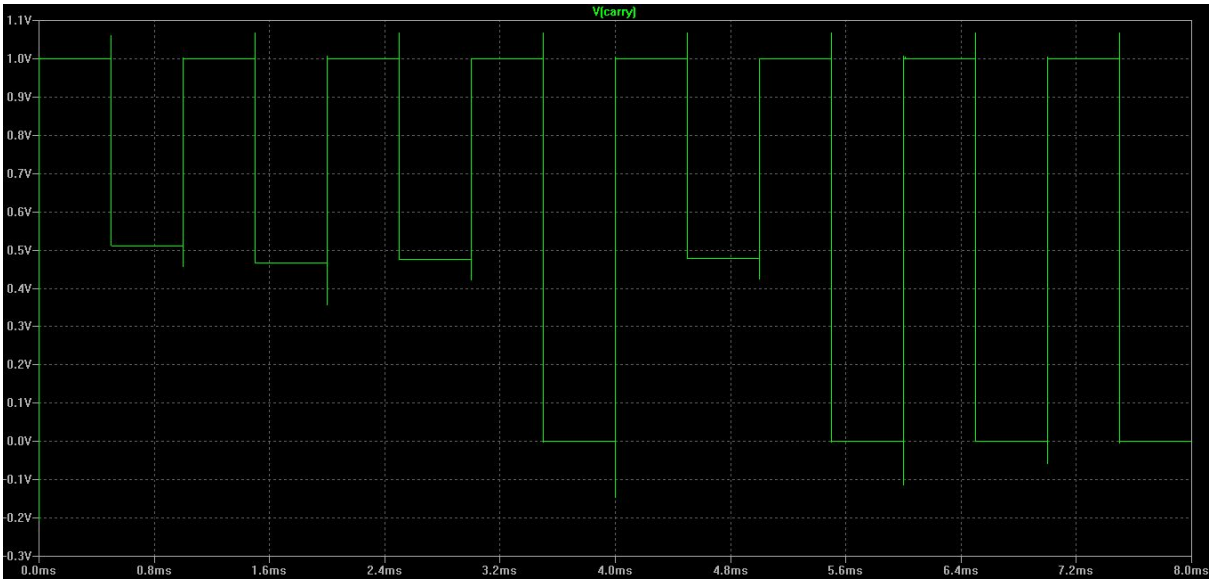
Para implementar las funciones mediante lógica dinámica, conservamos los bloques PDN de la implementación en CMOS complementario, y añadimos debajo del mismo un transistor NMOS adicional que esté controlado por el ciclo de reloj CLK (para controlar el paso de precarga-evaluación). Sobre el bloque PDN, introducimos un PMOS que también esté controlado por CLK y que nos cargue un '1' lógico en la salida, según el esquema a la derecha (sacado de los apuntes del tema 7).



El circuíto resulta (2.asc):



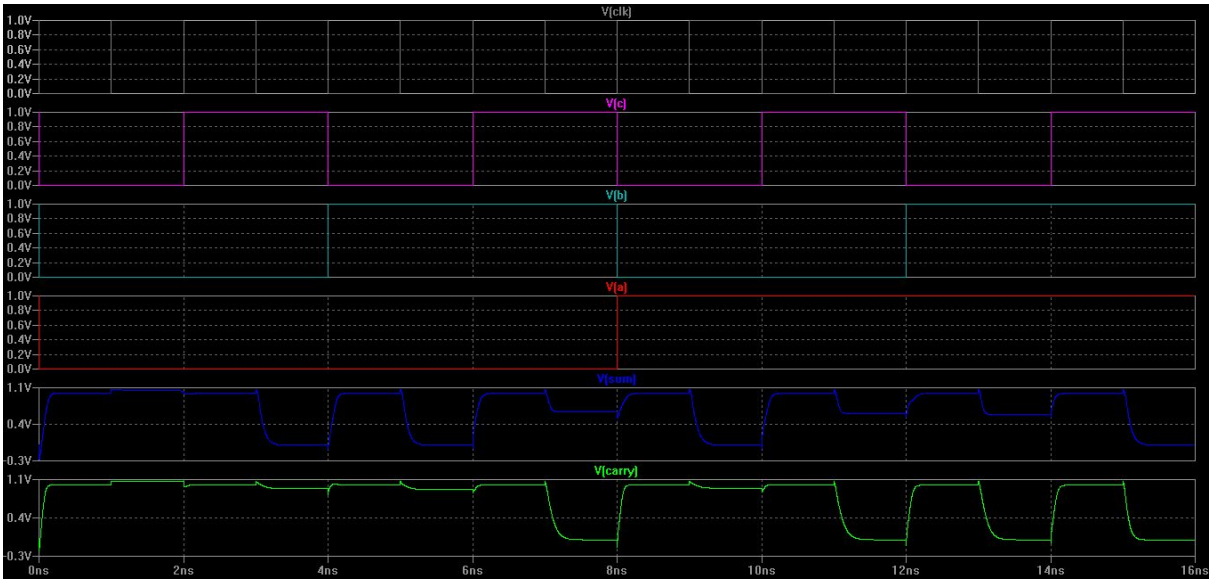
Como vemos en la gráfica inferior, pese a todo, si mantenemos los periodos de reloj en el orden de los milisegundos, aunque obtengamos un resultado coherente con los valores de la tabla de verdad, los '1's lógicos están muy mal definidos (tanto que se pueden interpretar como '0'). Esto es debido a la fuga de la carga por el bloque PDN. La solución será disminuír el tiempo de evaluación (incrementar la frecuencia); al fin y al cabo, la lógica CMOS dinámica está pensada para ser muy rápida.



Cambiando los tiempos al orden de los nanosegundos:

Ahora sí se puede ver que tenemos una salida bien definida.

Representamos las señales de reloj y las salidas en una gráfica:

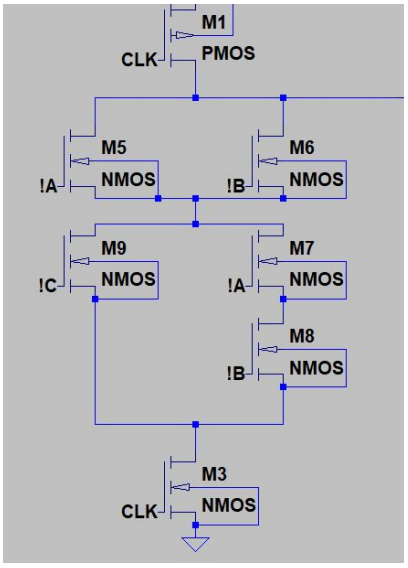


Vemos que los resultados son coherentes (son la negada de la tabla de verdad). Lo que sí que pasa, es que la señal de SUM está bastante mal definida. Esto es porque como en el momento de la precarga la salida del negado de $CARRY$ es por defecto 1, entonces vamos a tener el bloque PUN de la suma en ON sí o sí en el momento de la evaluación. Esto va a producir una descarga (parcial) del 1 que tenemos acumulado en la precarga de SUM, que dejará de descargarse cuando se evalúe correctamente $CARRY$.

Esto lo podemos solucionar de varias formas (se recogen en el tema 7). Yo he elegido modificar el bloque PDN de $CARRY$ para que funcione con lógica dominó, como se indica en el enunciado. Entonces, tendremos que modificar la función lógica para que sea la negada de la negada (y que, al invertirla a la salida, tengamos la negada).

Si $\overline{CARRY}$ es $\overline{A \cdot B + C \cdot (A + B)}$, esto se simplifica a:

$\Rightarrow \overline{(A \cdot B)} \cdot \overline{(C \cdot (A + B))} \Rightarrow (\overline{A} + \overline{B}) \cdot (\overline{C} + (\overline{A} \cdot \overline{B}))$.



Podemos implementar esta función negando las señales de A, B y C, como se ve a la izquierda.

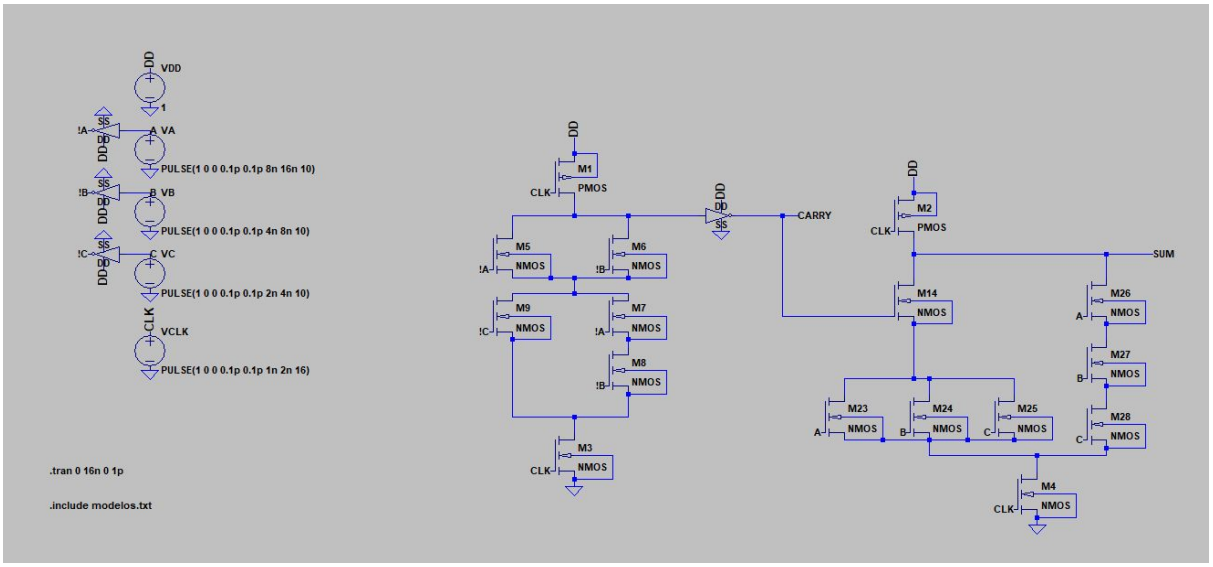
Las anchuras de los transistores tendremos que volver a calcularlas con el fin de que el retardo de propagación sea igual o menor que el de un inversor, en el peor caso.

Aquí, nuestro peor caso es tener 3 transistores en serie, por lo que para los que dependen de !A y !B, elegiremos un factor de escalado $s = 3$ (1500nm). El otro caso que tenemos es el de !A o !B y !C en serie (dos transistores en serie). Partiendo de que !A y !B tienen factor de escalado $s = 3$:

$\frac{R}{3} + \frac{R}{s} \leq R \Rightarrow \frac{R}{s} \leq \frac{2}{3}R \Rightarrow \frac{1}{s} \leq \frac{2}{3} \Rightarrow s \leq 1.5$.

Para coger un factor de escalado entero, cogemos $s = 1$ (500nm). Este será el factor de escalado de !C.

En este momento tendremos como resultado el negado de esta expresión, ya que la estamos aplicando a una PDN (cuando la función valga ‘1’, la salida será ‘0’ porque el bloque PDN conduce). Así que en estos momentos tenemos $\overline{\overline{CARRY}} = CARRY$. Si lo volvemos a negar (poniendo un inversor a la salida, que haga que la salida de la función sea ‘0’ durante la precarga), tendremos nuestra implementación por lógica dominó completamente funcional, ya que ahora seguimos teniendo $\overline{CARRY}$, pero con un ‘0’ por defecto.



Simulando, obtenemos:

Que son los mismos resultados de antes, pero con la función SUM esta vez sí bien definida (se queda en torno a 1V cuando en la tabla de verdad tenemos un 0, porque la función tiene la salida negada).