Ej. 1 — (2.0 puntos) Diseñar el diagrama ASM, la ruta de datos y la tabla de salidas de la unidad de control de un sistema algorítmico que ordene de menor a mayor los números impares de una lista de 16 enteros de 8 bits en formato binario puro. Los números están guardados de forma no ordenada en las 16 primeras palabras de una memoria SRAM síncrona de 32x8 bits. El sistema debe seleccionar los números impares y almacenarlos en orden ascendente en las líneas de memoria de la línea 16 a la 31.

El sistema tiene como entradas: `clk`, `rst` e `ini`. Y como salida una única señal `fin`. El sistema comienza a funcionar cuando la señal `ini` se pone a 1. Cuando se complete el cálculo el sistema volverá al estado inicial, en el que la señal `fin` es igual a 1.

Además de la memoria SRAM síncrona de 32x8 bits, READ-FIRST, con un solo puerto, podemos usar en la ruta de datos: contadores, registros, una única ALU con operaciones de suma y resta (r/s=0 es suma y r/s=1 es resta) y los elementos combinacionales que se necesiten.

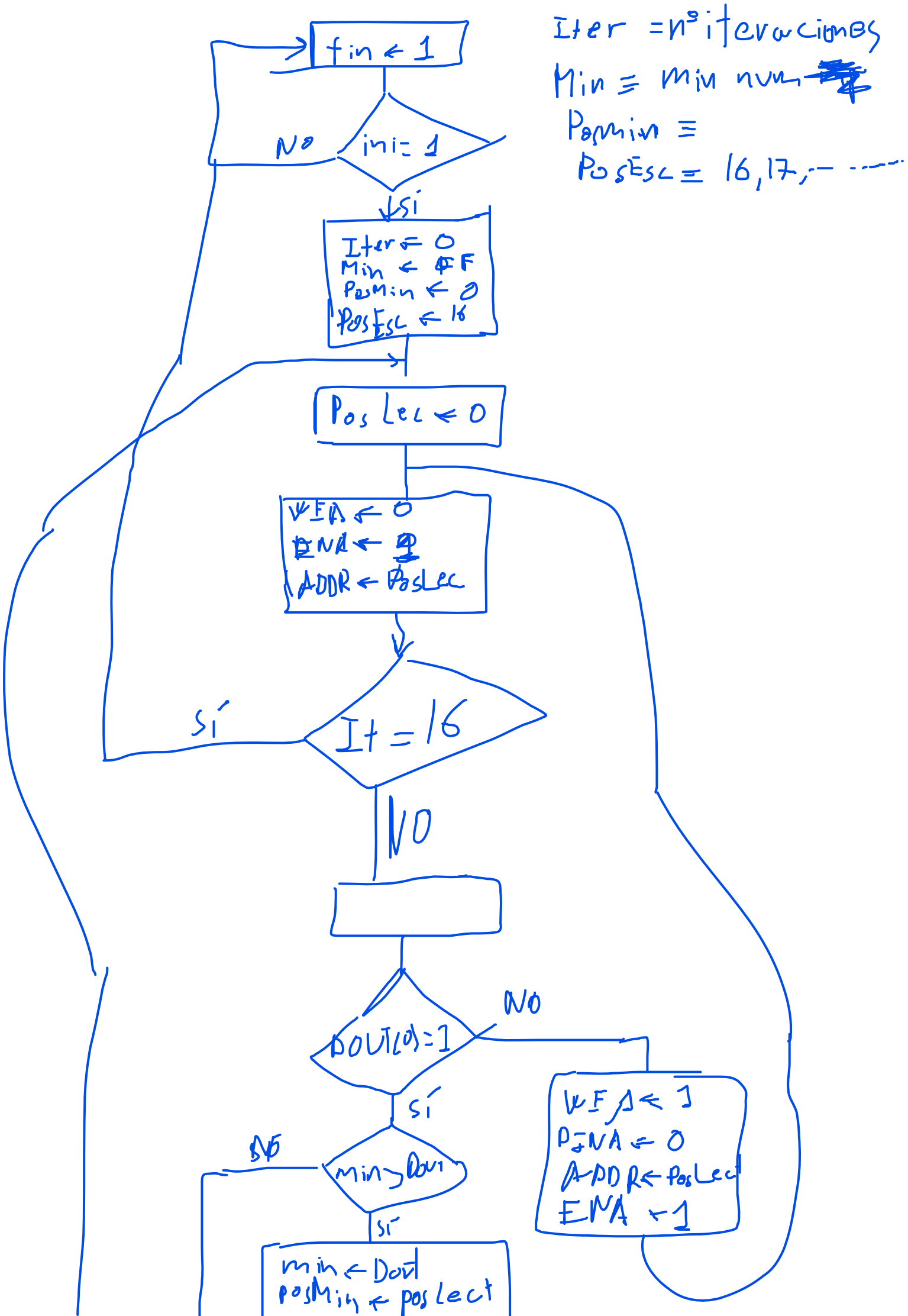
La ALU, además del resultado, tiene como salidas las señales: `zero`, resultado de la última operación es un cero, y `neg`, resultado de la última operación es negativo.

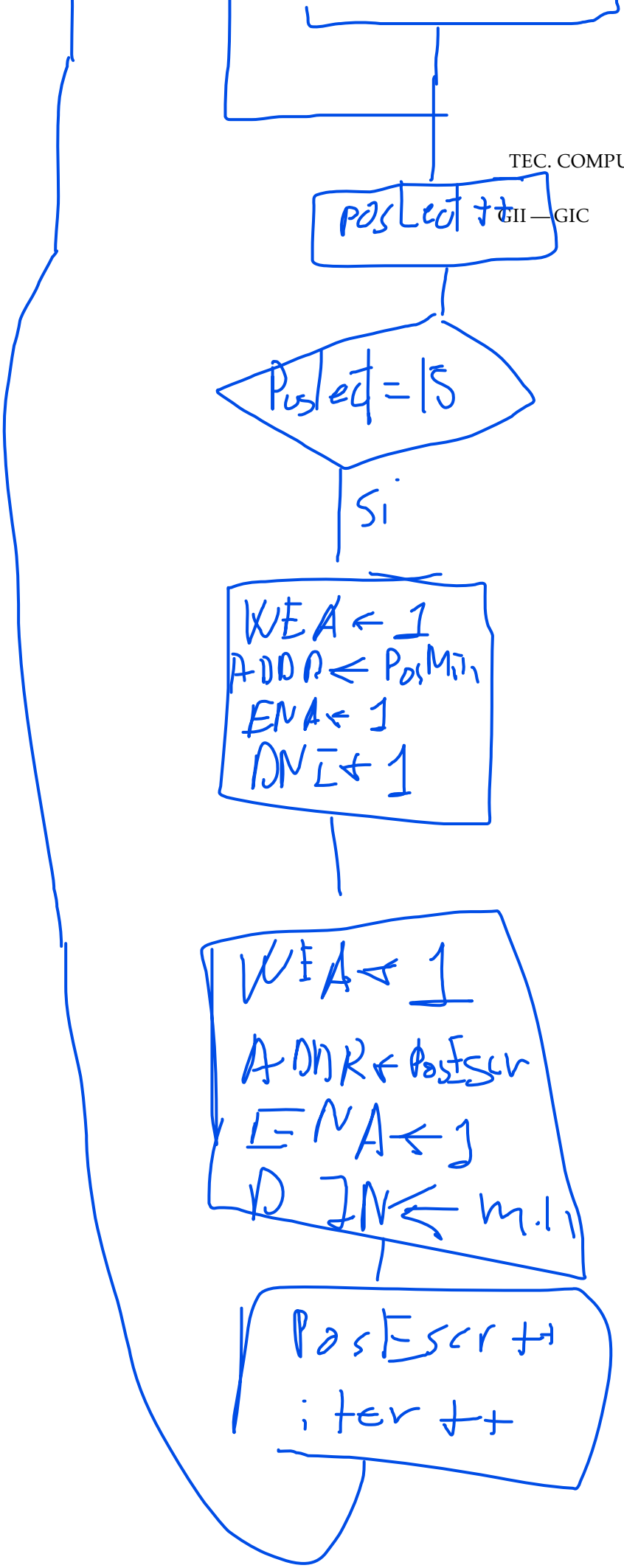
El algoritmo es el siguiente:

- Encontramos el valor mínimo de los 16 números iniciales y, si es impar, lo guardamos en la dirección 16. En la posición inicial de este mínimo colocamos un cero.
- Encontramos el valor mínimo de los 15 números restantes y, si es impar, lo guardamos en la dirección 17 (si el anterior número fue impar) o en la 16 (si el anterior número no fue impar). En la posición inicial de este mínimo colocamos un cero.
- Repetimos este proceso hasta tener ordenados todos los números en las líneas 16-31 y el valor cero en las líneas 0-15.

A continuación, se presenta un ejemplo con una memoria de 16 líneas en el que se ordena una lista de 8 números.

Inicial		Final	
0	4	0	0
1	7	1	0
2	3	2	0
3	6	3	0
4	1	4	0
5	9	5	0
6	8	6	0
7	4	7	0
8	0	8	1
9	0	9	3
10	0	10	7
11	0	11	9
12	0	12	0
13	0	13	0
14	0	14	0
15	0	15	0

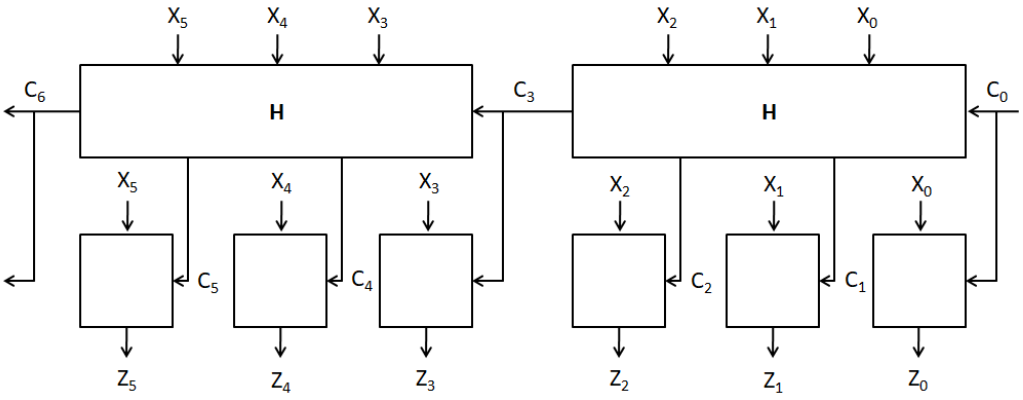

Iter = nº iteraciones
Min ≡ min num
PosMin ≡
PosEsc ≡ 16, 17, ...
fin ← 1
No
ini = 1
Sí
Iter ← 0
Min ← FF
PosMin ← 0
PosEsc ← 16
Pos Lec ← 0
WFA ← 0
ENA ← 0
ADDR ← PosLec
Sí
It = 16
NO
DOUT[0] = 1
NO
Sí
WFA ← 1
ENA ← 0
ADDR ← PosLec
ENA ← 1
NO
min > Dout
Sí
min ← Dout
posMin ← poslect



Falla Iter++
y PosLect ≤ 15

Ej. 2 — (1.50 puntos) Diseñar utilizando puertas lógicas:

1. Diseñar, utilizando puertas lógicas, una red iterativa que dado un vector de entrada, X, de n bits genere una salida, Z, de n bits de forma que cada uno de los bits de la salida indique si hay cuatro unos seguidos en el vector de entrada. Se permite solapamiento. La única restricción que se impone al diseño es que la señal interna sólo puede tener dos bits.
2. Una red iterativa con anticipación de operandos, tal y como se observa en la siguiente figura, con la misma funcionalidad y restricciones que la especificación del apartado anterior.



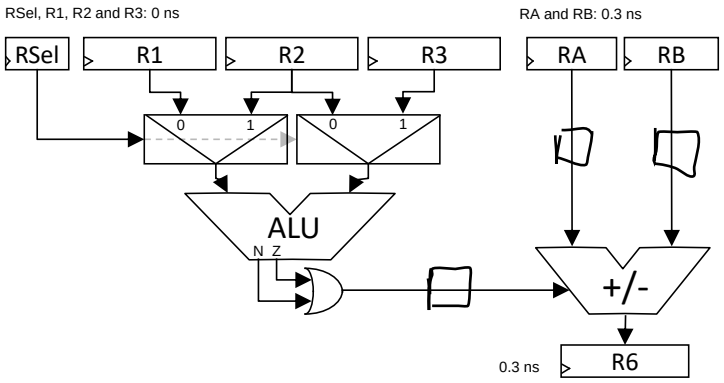
3. Suponiendo que todas las puertas de dos entradas tienen un retardo t, las puertas de tres entradas un retardo $2 \cdot t$ y las puertas de cuatro entradas tienen un retardo $3 \cdot t$, calcular el retardo del camino crítico en el diseño del apartado anterior para $n = 15$. En el caso de haber usado inversores, suponer que su retardo es despreciable.

Ej. 3 — (1.50 puntos) En el circuito de la figura, los valores de propagación de sus elementos combinacionales son los siguientes:

- Retardo(+/-) = 3 ns
- Retardo(Multiplexor) = 0.5 ns
- Retardo(Puerta AND) = 0.1 ns

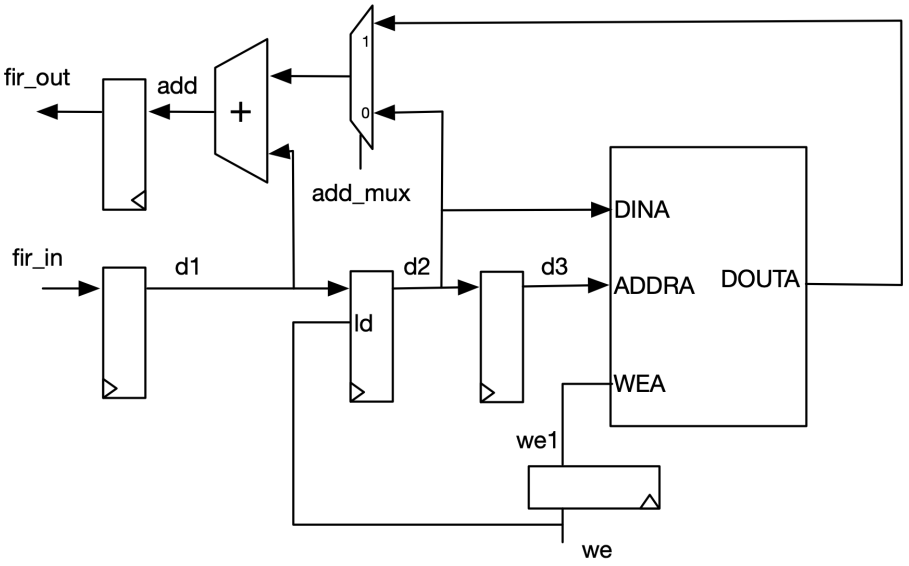
Los valores de las líneas de reloj corresponden al retardo de propagación desde la fuente de reloj hasta el registro. También téngase en cuenta que la salida de RSel es la entrada de selección de ambos multiplexores. Los parámetros de los componentes secuenciales son los siguientes:

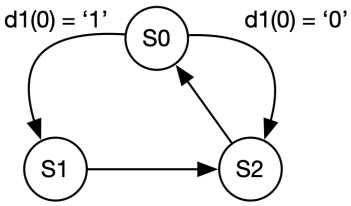
- tclk-2-q = 0.2ns
- tsetup = 0.1ns
- thold = 0.15ns



1. Indicar cuál es el camino crítico (o cuáles son, en caso de haber varios equivalentes). Calcular el máximo retardo de la ALU para que el circuito pueda funcionar correctamente a una frecuencia de reloj de 250 MHz. 0,4ns
2. Si el retardo de la ALU fuera 3 ns, ¿funcionaría este circuito a una frecuencia de reloj de 250 MHz? Justifíquese la respuesta. Si la respuesta fuese NO, usar segmentación de tal manera que esto fuera posible. Se asumirá que los retardos de propagación de las líneas de reloj desde la fuente hasta los nuevos registros es 0 ns.
3. Comentar si hay algún efecto colateral como consecuencia de la segmentación realizada en el apartado anterior. Si fuera así, indicar cómo se puede resolver. Violación hold/Buffers

Ej. 4 — (1.50 puntos) Completar el cronograma correspondiente al siguiente sistema e indicar el contenido final de la memoria SRAM. Las entradas al sistema son `fir_in`, `clk` y `rst` (no está presente en el esquema) y `fir_out` es la salida; el resto de señales son internas según tabla adjunta. Considerar que la memoria es síncrona y funciona en modo READ_FIRST y ALWAYS_ENABLE.





SRAM

ADDR	DATA	DATA FINAL
0x00	0x03	0x01
0x01	0x07	0x4
0x02	0x05	
0x03	0x09	
0x04	0x06	0x1
0x05	0x02	
0x06	0x03	
0x07	0x0C	

Tabla de salidas UC

State	add_mux	we
S0	1	0
S1	0	0
S2	0	1