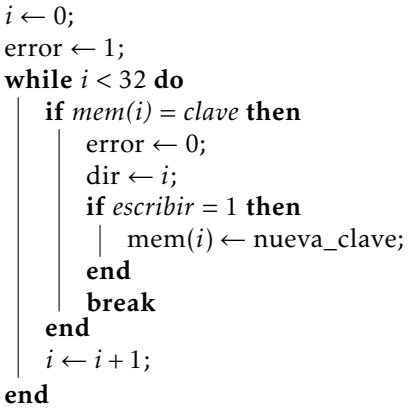
Ej. 1 — (2.0 puntos) Diseñar la ruta de datos y la unidad de control (diagrama ASM y tabla de salidas) de un sistema algorítmico que dado una clave (entrada `clave` de 4 bits), busque si el valor de esa clave está almacenado en una memoria RAM síncrona de 32x4 bits. Si lo está el sistema devolverá la dirección de memoria donde está almacenada la clave (salida `dir`, 5 bits). Además, si la entrada de control `escribir` vale '1', el sistema grabará en la dirección de memoria donde estaba almacenada la clave una nueva clave (entrada `nueva_clave` de 4 bits). Si la clave no está almacenada en memoria el sistema pondrá a '1' la señal `error`. El sistema no comienza a funcionar hasta que la señal `ini` se pone a '1'. En ese instante se almacenan los valores de clave y nueva_clave. Cuando se haya realizado la operación correspondiente el sistema volverá al estado inicial donde la señal `fin` es igual a '1'.




```
entity asm is
  port (clk         : in  std_logic;
        rst_n       : in  std_logic;
        ini         : in  std_logic;
        escribir    : in  std_logic;
        clave       : in  std_logic_vector(3 downto 0);
        nueva_clave : in  std_logic_vector(3 downto 0);
        fin         : out std_logic;
        dir         : out std_logic_vector(4 downto 0)
        error       : out std_logic
        );
end asm;

entity ram is
  port (clk   : in  std_logic;
        dina  : in  std_logic_vector(3 downto 0);
        addra : in  std_logic_vector(4 downto 0)
        wea   : in  std_logic;
        ena   : in  std_logic;
        douta : out std_logic_vector(3 downto 0);
        );
end ram;
```

En la ruta de datos se puede usar una memoria RAM[(1)] síncrona de un solo puerto, un contador ascendente/descendente[(2)], registros y los elementos combinacionales que se consideren necesarios. Se valorará el uso del hardware mínimo.

[(1)]: Suponer que los puertos de salida conservan el valor leído hasta que se realice una nueva lectura.

[(2)]: La definición de puertos del contador ascendente/descendente: `din`, bus dato de entrada; `ce`, habilitación del contador; `load`, señal de carga; `ud`, cuenta arriba (ud=1) o cuenta abajo (ud=0); `dout`, bus de salida.

Ej. 2 — (1.5 puntos)

1. Diseñar, utilizando puertas lógicas, una red iterativa 1D 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 de interna sólo puede tener dos bits.
2. Demostrar que la siguiente especificación es una solución del problema anterior:

celda i-ésima

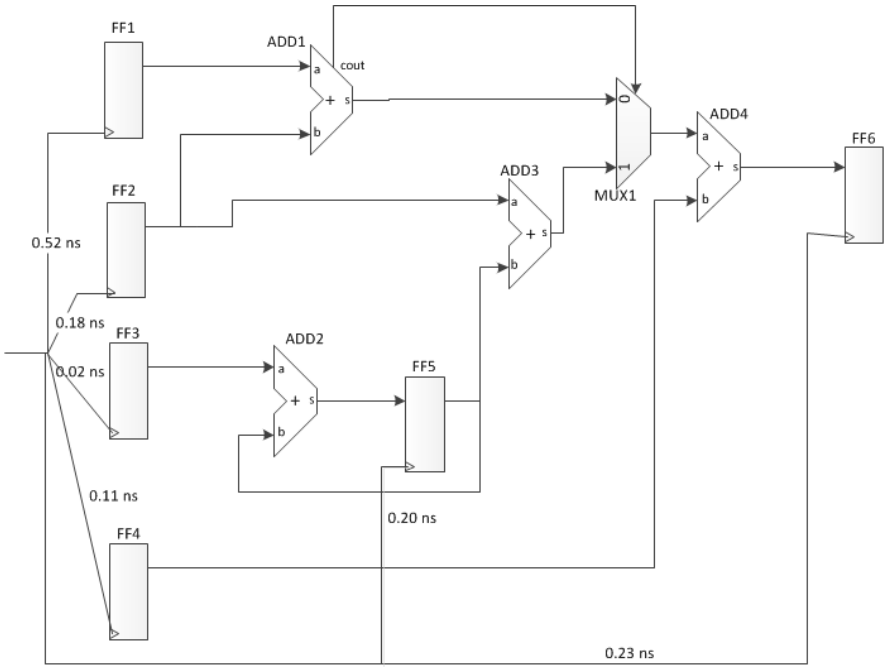
```
zi <= (xi and ci(0)) and (ci(1) and ci(2));
ci+1(0) <= xi;
ci+1(1) <=ci(0);
ci+1(2) <= ci(1);
```

Con las condiciones de contorno:

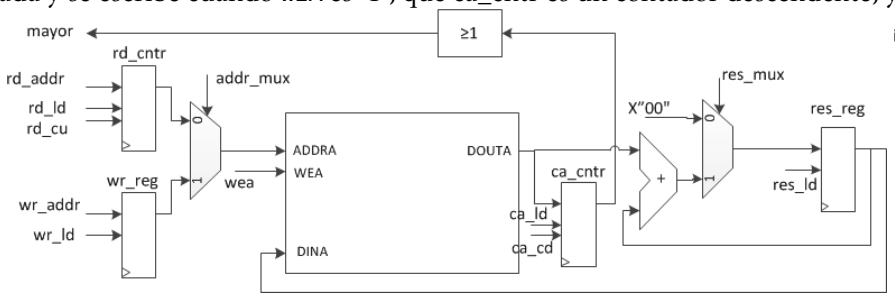
```
c0(0) <= '0';
c0(1) <= '0';
c0(2) <= '0';
```

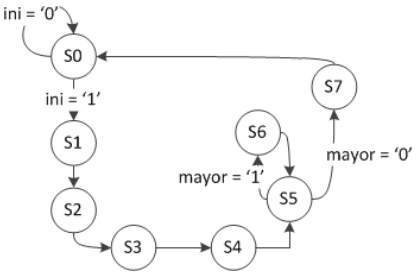
3. Si se considera que cada puerta lógica de dos entradas tiene un retardo de 1 ns (y el retardo de las puertas NOT es despreciable), ¿Cuál será el retardo para una red de n bits tanto para el apartado 1. como para el apartado 2.?

Ej. 3 — (1.5 puntos) En el circuito de la figura los valores de propagación de los componentes son los siguientes: ADD(a→s) = 1,56 ns, ADD(b→s) = 1,40 ns, ADD(a→ c_{out}) = 1,70 ns, ADD(b→ c_{out}) = 1,64 ns, MUX(0→out) = 0,75 ns, MUX(1→out) = 0,70 ns, MUX(sel→out) = 0,25 ns . Los valores en las líneas de reloj son el retardo de propagación desde la fuente de reloj. Los parámetros de los registros son: $t_{clk \to q}$ = 0,12 ns, t_{setup} = 0,10 ns y t_{hold} = 0,05 ns. (1) Calcular los márgenes de setup en los registros de destino si la frecuencia de reloj fuese 250 MHz. ¿Habría violaciones de setup? (2) ¿Cuál sería la frecuencia de reloj máxima a la que podría trabajar este circuito? (3) Con su estructura actual, el circuito tendría violaciones de setup. Para que el circuito pudiese trabajar a 250 MHz sería necesario segmentarlo. Indicar dónde se deberían introducir los registros de segmentación y calcular los nuevos margen de setup para el camino o los caminos que presentaban violaciones de setup. Supóngase que el retardo de reloj para los nuevos FF es de 0,22 ns.



Ej. 4 — (1.5 puntos) Completar el cronograma correspondiente al siguiente sistema e indicar el contenido final de la memoria. Las entradas al sistema son `rd_addr`, `wr_addr` e `ini`; la salida es `fin`; el resto de señales son señales de estado o control, estas últimas generadas por la unidad de control según tabla adjunta. Considerar que la memoria es síncrona, está siempre habilitada y se escribe cuando `WEA` es '1'; que ca_cntr es un contador descendente; y que rd_cntr es un contador ascendente.



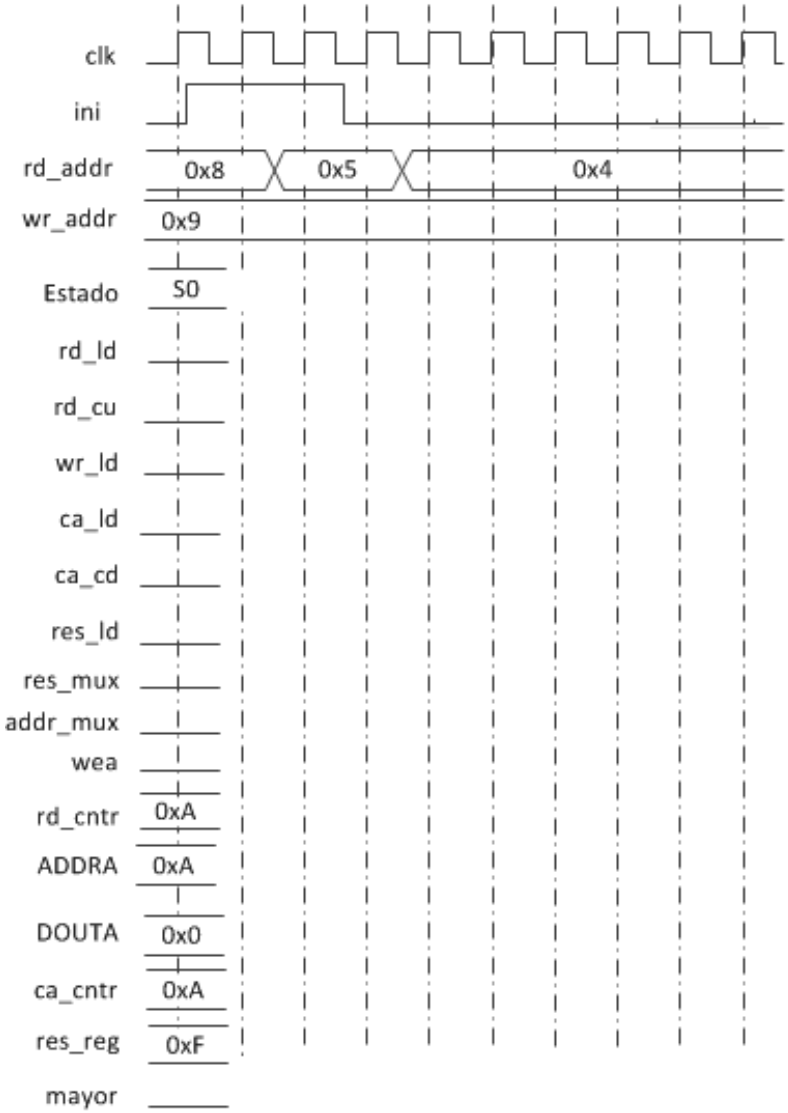


Estado	rd_ld	rd_cu	wr_ld	ca_ld	ca_cd	res_ld	res_mux	addr_mux	wea	fin
S0	0	0	0	0	0	0	0	0	0	1
S1	1	0	1	0	0	1	0	0	0	0
S2	0	0	0	0	0	0	0	0	0	0
S3	0	1	0	1	0	0	0	0	0	0
S4	0	0	0	0	0	0	0	0	0	0
S5	0	0	0	0	0	0	0	0	0	0
S6	0	0	0	0	1	1	1	0	0	0
S7	0	0	0	0	0	0	0	1	1	0

Señales de control

Dirección	Dato
0x04	0x04
0x05	0x01
0x06	0x07
0x07	0x37
0x08	0x11
0x09	0x55
0x0A	0x00
0x0B	0x00

Contenido de la memoria



Adjunto se copia de nuevo el cronograma del Ej.4– por si cometierais algún error en el cronograma original.