EXAMEN SED, Primera parte
12/2/2014

NOMBRE: .....

1) El siguiente código VHDL describe un circuito combinacional mediante dos asignaciones concurrentes:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY circuito IS
PORT ( a,b,c,d: IN STD_LOGIC_VECTOR (2 DOWNTO 0);
seleccion: IN INTEGER RANGE 0 TO 3;
enable: IN STD_LOGIC;
e: OUT STD_LOGIC_VECTOR (2 DOWNTO 0));
END circuito;

ARCHITECTURE concurrente OF circuito IS
SIGNAL interna: STD_LOGIC_VECTOR (2 DOWNTO 0);
BEGIN

WITH seleccion SELECT
interna<= a WHEN 0,
          b WHEN 1,
          c WHEN 2,
          d WHEN 3;

e<= "ZZZ" WHEN enable='0' ELSE interna;

end concurrente;
```

1.1) Describe la función que realiza el circuito:

1.2) ¿Podrían conectarse las salidas *e* de dos circuitos como éste en paralelo (es decir, uniendo entre sí las líneas de bit de salida correspondientes)?:
Razona la respuesta:

1.3) Utilizando el espacio reservado a la derecha del código anterior, sustituye las dos asignaciones concurrentes por un solo proceso, de manera que el circuito resultante sea el mismo.
NOTA: si no es necesario, puede prescindirse de la señal intermedia (***interna***) utilizada en la arquitectura 'concurrente'.

2) El siguiente código VHDL es una descripción funcional sintetizable del TIMER0 (de 8 bits) de un PIC, incluyendo el *flag* de petición de interrupción cuando se desborda (T0IF), la puesta a cero de este *flag* mediante una señal (ClearT0IF), y la posibilidad de cargar el *timer* con un valor inicial (TMR0_inicial).
No se ha incluido la función del *prescaler*.

```
library ieee;
use ieee.std_logic_1164.all;
entity timer0 is
        Port( reloj: IN std_logic; --se supone reloj de '4Tosc'
        SET_Timer0, ClearT0IF: IN std_logic;
        TMR0_inicial :    IN INTEGER range 0 to 255;
        TIMER: INOUT INTEGER range 0 to 255;
        T0IF: OUT std_logic);
end timer0;

architecture rtl of timer0 IS
begin
        process
        begin
                wait until reloj'EVENT and reloj ='1';
                if (ClearT0IF='1') then T0IF <='0'; end if;
                if (SET_Timer0='1')  then TIMER<=TMR0_inicial;
                else TIMER<=TIMER+1; end if;
                if TIMER=255 then T0IF<='1'; end if;
        end process;
end rtl;
```

2.1) Tal como se indica en su declaración, TIMER debe estar definido como puerto INOUT y no como OUT. ¿Por qué?:

2.2) Sólo hemos puesto visible el paquete *std_logic_1164*. Tendríamos que añadir el *numeric_std* o el *std_logic_arith* para que se sintetice correctamente? (razona la respuesta):

2.3) Dibuja un esquema aproximado del circuito que se infiere:

2.4) Indica cómo podríamos añadir la funcionalidad del prescaler (para este *timer* debiera tomar los valores 1, 2, 4, 8, 16, 32, 64, 128 y 256)