Ejercicio 1 (40 min, 4 pto).

Para el circuito digital descrito en VHDL, se pide:

1. Complete las listas de sensibilidad para que las simulaciones y las síntesis lógica sean correctas y coherentes
2. Dibuje el hardware resultante de la síntesis lógica del proceso P3. ¿Cuántos biestables serán necesarios?
3. Dibuje el diagrama de estados de la máquina de estados que describen P1 y P2

```
library IEEE;
use IEEE.std_logic_1164.all;

entity CIRCUITO is
  port (
    Clk    : in  std_logic;
    Reset  : in  std_logic;
    Valid  : in  std_logic;
    IsOpen : out std_logic);
end CIRCUITO;

architecture BEHAVIORAL of CIRCUITO is
  type     tStates is (s0, ok1, ok2, err);
  signal   CurrentState : tStates;
  signal   NextState    : tStates;
  constant cTopTimer    : natural := 18;
  signal   Timer1       : natural range 0 to cTopTimer;
  signal   EnaT         : std_logic;
begin

  P1: process(                                )
  begin
    if Reset = '1' then
      CurrentState <= s0;
    elsif Clk'event and Clk = '1' then
      CurrentState <= NextState;
    end if;
  end process P1;

  P2: process(                                )
  begin
    EnaT <= '1';
    IsOpen <= '0';
    case CurrentState is
      when s0 =>
        if Valid = '1' AND Timer1 = cTopTimer then
          NextState <= ok1;
        else
          NextState <= s0;
        end if;
```

```
      when ok1 =>
        if Valid = '0' then
          NextState <= s0;
        elsif Timer1 = cTopTimer then
          NextState <= ok2;
        else
          NextState <= ok1;
        end if;
      when ok2 =>
        IsOpen <= '1';
        if Valid = '0' then
          NextState <= err;
        elsif Timer1 = cTopTimer then
          NextState <= s0;
        else
          NextState <= ok2;
        end if;
      when err =>
        EnaT <= '0';
        if Valid = '1' then
          NextState <= s0;
        else
          NextState <= err;
        end if;
    end case;
  end process P2;

  P3: process(                              )
  begin
    if Reset = '1' then
      Timer1 <= 0;
    elsif Clk'event and Clk = '1' then
      if EnaT = '1' then
        if Timer1 = cTopTimer then
          Timer1 <= 0;
        else
          Timer1 <= Timer1 + 1;
        end if;
      end if;
    end if;
  end process P3;
end BEHAVIORAL;
```

Ejercicio 2 (50 min, 6 pto)

Se quiere diseñar un circuito digital que procese una señal procedente de un sensor de infrarrojos. Esta señal tiene una frecuencia de 100 Hz y debe multiplicarse por una referencia senoidal (también de 100 Hz) y posteriormente filtrarse para quedarse con la componente continua del producto. El sistema cuenta con un reloj (CLK) de 100 kHz para realizar las operaciones. La entidad del circuito es:

```
entity LOCK-IN is
  port (
    Clk         : in  std_logic;
    Reset       : in  std_logic;
    ADC_ready   : in  std_logic;
    ADC_in      : in  unsigned(13 downto 0);
    Ref_in      : in  signed(13 downto 0);
    Data_out    : out signed(__ downto 0) );
end LOCK-IN;
```

La frecuencia de muestreo es de 1kHz, por lo que nos llega una muestra cada 1 ms. La señal tiene 14 bits y procede de un convertidor analógico digital (ADC) que presenta la señal en binario natural. Sin embargo, esta señal debe convertirse a un valor entero (complemento a dos) pues el ADC la presenta centrada en el valor medio del fondo de escala (el valor 0 de la señal es el valor 8192 del ADC). Para adquirir cada señal procedente del ADC son necesarios 200 µs; cuando el dato está convertido, la entrada ADC_ready se activa durante un ciclo de reloj.

Una vez convertida la señal de entrada a complemento a 2, se multiplica por la referencia senoidal (también entrada de 14 bits, siempre disponible) y al producto se le aplica un filtro FIR de 8 etapas. Los coeficientes de este filtro tienen 6 bits y son configurables, con lo que no cabe optimización del resultado final. Se puede acceder a los coeficientes como una matriz constante llamada aC(0 to 7) y que ya está declarada en un paquete.

Para todas las preguntas, razone sus respuestas.

1. Indique de cuántos ciclos de reloj CLK se dispone entre muestra y muestra de la señal.
2. Indique de cuántos ciclos de reloj CLK se dispone para el procesamiento de datos, una vez adquirido el dato del ADC.
3. Escriba la sentencia de asignación concurrente o el proceso combinacional que convierte el dato procedente del ADC (binario natural) a complemento a 2.
4. Indique el número de bits que tendrá el resultado final, si no queremos truncar ninguna operación intermedia.
5. Se quiere hacer una arquitectura paralela para el procesamiento de la señal de entrada (truncando el resultado de cada operación intermedia a 24 bits). Asumir que la entrada se captura con la señal ADC_ready, y la salida se captura en el ciclo de reloj siguiente.
    a. Describa el proceso secuencial que registra los datos de entrada al filtro FIR
    b. Describa el filtro FIR mediante asignaciones concurrentes o un proceso combinacional
    c. Describa el proceso secuencial que registra la salida del filtro FIR. Genere la señal de carga necesaria.
6. Si sólo se dispone de un bloque multiplicador, dibuje ruta de datos de la arquitectura serie que habría que utilizar. ¿Cuántos ciclos de reloj son necesarios? ¿Hay suficientes, de acuerdo con su repuesta al apartado 2?