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. Complete las declaraciones de las señales indicadas en la arquitectura
3. Dibuje el hardware resultante de la síntesis lógica del proceso P2. ¿Cuántos biestables serán necesarios?
4. Dibuje el diagrama de estados de la máquina de estados que describen P3 y P4

```
library IEEE;
use IEEE.std_logic_1164.all;
entity RS232_Tx is
  generic(
    gDataWidth    : natural := 8;  -- bits for Data
    gTotalNumBit  : natural := 11; -- WordWidth= Start(1)+Data(8)+Parity(1)+Stop(1)
    gBaudRate     : natural := 57600;      -- bps
    gClkFrequency : natural := 100000000); -- Hz
  port(
    Clk           : in  std_logic;
    ResetN        : in  std_logic;
    RSDataToSend  : in  std_logic_vector(gDataWidth - 1 downto 0);
    EnableSend    : in  std_logic;
    TxD           : out std_logic;
    TxBusy        : out std_logic
  );
end RS232_Tx;
architecture Behavioural of RS232_Tx is
  constant StartBit        : std_logic := '0';
  constant StopBit         : std_logic := '1';
  ------------------------------------------------------------------
  constant cBaudRateCycles : natural := gClkFrequency/gBaudRate;
  signal EnaBaudRate       : ____________________;
  signal ClrBaudRate       : ____________________;
  signal CountBaudRate     : ____________________;
  signal EndBaudRate       : ____________________;
  ------------------------------------------------------------------
  signal EnaNumBitsTx      : std_logic;
  signal ClrNumBitsTx      : std_logic;
  signal CountNumBitsTx    : natural range 0 to gTotalNumBit-1;
  signal EndNumBitsTx      : std_logic;
  ------------------------------------------------------------------
  type tStateTx is (Idle, Sending);
  signal CurrentState      : tStateTx;
  signal NextState         : TstateTx;
  signal RegDataTx         : std_logic_vector(gTotalNumBit-1 downto 0);
  signal ClrTx             : std_logic;
  signal ParityBit         : std_logic;
begin
  TxD          <= RegDataTx(0);
  TxBusy       <= NOT(ClrTx);
  ------------------------------------------------------------------------------
  P1: process(                    )
    variable aux_Parity: std_logic;
  begin
    aux_Parity:= '0';
       for I in RSDataToSend'range loop
      aux_Parity:= aux_Parity XOR RSDataToSend(I);
    end loop;
    ParityBit     <= aux_Parity;
  end process P1;
  ------------------------------------------------------------------------------
  P2: process(                        )
  begin
```

```
  if ResetN = '0' then
    RegDataTx <= (others => '1');
  elsif Clk'event and Clk = '1'  then
    if ClrTx = '1' then
       RegDataTx <= (others => '1');
    elsif EnableSend = '1' then
      RegDataTx <= StopBit & ParityBit & RSDataToSend & StartBit;
    elsif EnaNumBitsTx = '1' then
      RegDataTx <= '1' & RegDataTx(gTotalNumBit-1 downto 1);
    end if;
  end if;
end process P2;
-------------------------------------------------------------------------------
P3: process(                              )
begin
  if ResetN = '0' then
    CurrentState <= Idle;
  elsif Clk'event and Clk = '1' then
    CurrentState <= NextState;
  end if;
end process P3;
-------------------------------------------------------------------------------
P4: process(                                      )
begin
  ClrTx        <= '0';
  ClrBaudRate  <= '0';
  ClrNumBitsTx <= '0';
  EnaBaudRate   <= '0';
  case CurrentState is
    when Idle =>
      ClrBaudRate  <= '1';
      ClrNumBitsTx <= '1';
      ClrTx         <= '1';
      if EnableSend = '1' then
        NextState <= Sending;
        ClrTx  <= '0';
      else
        NextState <= Idle;
      end if;
    when Sending =>
      EnaBaudRate  <= '1';
      if EndNumBitsTx = '1'  then
        NextState <= Idle;
      else
        NextState <= Sending;
      end if;
    when others =>
      NextState <= Idle;
  end case;
end process P4;
-------------------------------------------------------------------------------
P5: process(                                    )
begin
  if ResetN = '0' then
    CountBaudRate <= 0;
  elsif Clk'event and Clk = '1' then
    if ClrBaudRate = '1' then
        CountBaudRate <= 0;
    elsif  EnaBaudRate = '1' then
       if CountBaudRate = cBaudRateCycles - 1 then
        CountBaudRate <= 0;
       else
        CountBaudRate <= CountBaudRate + 1;
       end if;
    end if;
```

```
    end if;
  end process P5;
  EndBaudRate <= '1' when ((CountBaudRate = cBaudRateCycles - 1) AND
                           (EnaBaudRate = '1'))
                 else '0';
  ------------------------------------------------------------------------------
  P6: EnaNumBitsTx <= EndBaudRate;
  process(                                  )
  begin
    if ResetN = '0' then
      CountNumBitsTx <= 0;
    elsif Clk'event and Clk = '1' then
      if ClrNumBitsTx = '1' then
          CountNumBitsTx <= 0;
      elsif  EnaNumBitsTx = '1' then
        if CountNumBitsTx = gTotalNumBit - 1 then
          CountNumBitsTx <= 0;
         else
          CountNumBitsTx <= CountNumBitsTx + 1;
        end if;
       end if;
    end if;
  end process P6;
  EndNumBitsTx <= '1' when ((CountNumBitsTx = gTotalNumBit - 1) AND
                            (EnaNumBitsTx = '1'))
                  else '0';
end Behavioural;
```

Ejercicio 2 (50 min, 6 pto)

Se quiere diseñar un circuito digital que procesa datos, realizando la función exponencial. El circuito deberá ejecutar la operación e^X. Para diseñar el circuito se va a utilizar su representación en serie de Taylor, truncando en el cuarto elemento.

$$Y(X) = 1 + X + \frac{X^2}{2!} + \frac{X^3}{3!} \qquad (1)$$

Para que esta aproximación sea válida, los datos de entrada (X) deben estar en el rango entre -1 y +1. Para poder implementar el algoritmo utilizando números enteros, se realiza un cambio de variable (Z=X*128), de modo que Z sea un número entero y se represente con 8 bits (en CA2). Nótese que con este cambio se descarta el valor +1 en el rango de X. Con este cambio, la ecuación queda:

$$Y(Z) = 1 + \frac{Z}{2^7} + \frac{Z^2}{2 \cdot 2^{14}} + \frac{Z^3}{6 \cdot 2^{21}} \qquad (2)$$

Con el rango de entradas que se desea, el resultado de la función resulta entre e^{-1} y $e^{127/128}$. Se quiere expresar dicho resultado como un número entero de 10 bits, de modo que hay que multiplicar la ecuación anterior por 128 (2^7), quedando la ecuación que se quiere implementar mediante un circuito:

$$W(Z) = 2^7 \left(1 + \frac{Z}{2^7} + \frac{Z^2}{2 \cdot 2^{14}} + \frac{Z^3}{6 \cdot 2^{21}}\right) = 2^7 + Z + \frac{Z^2}{2^8} + \frac{Z^3}{3 \cdot 2^{15}} \qquad (3)$$

El sistema cuenta con un reloj (Clk) de 100 kHz para realizar las operaciones, una señal de inicialización asíncrona (Reset) y un Enable de carga del dato de entrada, que se activa durante un ciclo de reloj para que el circuito cargue el dato de entrada (frecuencia de muestreo 1kHz). La entidad del circuito es:

```
entity EXP_23 is
  port (
    Clk        : in  std_logic;
    Reset      : in  std_logic;
    Enable     : in  std_logic;
    Data_in    : in  signed(7 downto 0);  -- Z
    Data_out   : out signed(9 downto 0)); -- W(Z)
end EXP_23;
```

Para todas las preguntas, razone sus respuestas.

1. Compruebe los rangos de las variables y las funciones que se van a utilizar, es decir:
   Si el rango de la variable Z es [-128, 127] (número entero en CA2 con 8 bits):
   a. Especifique el rango de la variable X que se corresponde con el rango de Z
   b. Especifique los rangos de Y(Z) y W(Z) según el rango de Z
   c. Compruebe que todos ellos están dentro de las especificaciones (Z es signed de 8 bits y W(Z) es signed de 10 bits).
2. Exprese la función W(Z) de modo que se pueda realizar con números enteros, y sólo con divisiones entre potencias de 2. Los coeficientes necesarios deben ser de 8 bits en complemento a 2.
3. Se quiere hacer una arquitectura paralela.
   a. Dibújela en nivel de transferencia de registros (RTL)
   b. Determine razonadamente el número de bits de cada sumando de la ecuación.
   c. Indique el número de bits que se necesitan para representar el resultado final, sin despreciar ningún dígito. Si el resultado final debe tener 10 bits, indique los bits que se deben despreciar.
4. Asumir que la entrada se captura con la señal Enable, y la salida está registrada, disponible en el ciclo de reloj siguiente.
   a. Describa la declaración de las señales que se necesitan para almacenar los resultados parciales y final
   b. Describa el proceso secuencial que registra el dato de entrada
   c. Describa las operaciones intermedias mediante asignaciones concurrentes o un proceso combinacional
   d. Describa el proceso secuencial que registra el resultado final.
5. Si cada multiplicador tarda 40 ns en producir un resultado y cada sumador 20 ns. Indique el retardo máximo del circuito y la frecuencia necesaria para la señal de reloj. ¿Cómo modificaría la arquitectura del apartado anterior para triplicar la frecuencia de reloj?