ESCUELA SUPERIOR POLITECNICA DEL LITORAL
FACULTAD DE INGENIERIA EN ELECTRICIDAD Y COMPUTACIÓN

SISTEMAS DIGITALES II

DEBER PARA SEGUNDA EVALUACIÓN

II TÉRMINO 2015-2016

PROBLEMA # 1

Diseñe un pequeño Sistema Digital **CIRCUITO IDENTIFICADOR DE NUMEROS MINIMO Y MAXIMO**.
El circuito debe tener una entrada de **DATOS**, representada por un teclado decimal
Los números pueden estar en el rango entre 00 y 99. Para ingresar cada digito de los números, se debe activar la entrada **ENTER**. El circuito puede recibir **N** números.
La salida del circuito debe tener dos display de 7 segmentos, que inicialmente están en cero. Cada vez que se ingresa un número, su valor se presenta en los displays.
Después de ingresar los **N** números, se activa la entrada **START**.
El circuito debe analizar los números ingresados y, después, debe mostrar, por medio de los displays, el número **MIN** y el número **MAX**.
Si todos los números ingresados han sido iguales, los display no mostraran nada, es decir deben estar apagados.

Presente:

1. **Diagrama general de entradas/salidas**
2. **Diagrama de bloques de su solución**
3. **Partición Funcional**
4. **Diagrama ASM** del circuito **Controlador** debidamente documentada. (indicar todos las entradas y salidas)

Nota. Asuma que la implementación será hecha usando tecnología FPGA y lenguaje VHDL. Dimensionar adecuadamente cada componente.

PROBLEMA # 2

Se debe diseñar un sistema digital **RECEPTOR-ALMACENADOR** de datos que formará parte de un sistema de comunicación más complejo.
Un transmisor envía una trama compuesta de tres bytes, la que es recibida en forma **SERIAL** en el lado del receptor quien debe almacenarlas en una memoria **RAM (512 x 8)** hasta completar 128 tramas que serán utilizadas por otro bloque quien correrá un algoritmo de encripción de información usando un algoritmo matemático.
Para almacenar en la memoria RAM cada trama, el receptor debe considerar que debe almacenar los datos en el orden D3, D1, D2, siendo D1 el primer byte de la trama, D2 el segundo byte de la trama y D3 el tercer dato de la trama.
El transmisor envía la señal **Inicio_Trama** cada vez que le pasa una trama de tres bytes al receptor. Una vez almacenadas las 128 tramas, el receptor se queda esperando por la señal **LISTO** que le debe enviar el bloque de encripción de datos y una vez hecho esto el sistema **RECEPTOR-ALMACENADOR** vuelve a esperar por la señal **Inicio_Trama** para volver a iniciar un nuevo proceso de recepción y almacenamiento.

Presentar:

1. **Diagrama de bloques** del Sistema Digital.
2. **Partición Funcional** del Sistema Digital.
3. **Diagrama ASM** del circuito **Controlador** debidamente documentado. (indicar todos las entradas y salidas)

Nota. Asuma que la implementación será hecha usando tecnología FPGA y lenguaje VHDL. Dimensionar adecuadamente cada componente.

PROBLEMA # 3

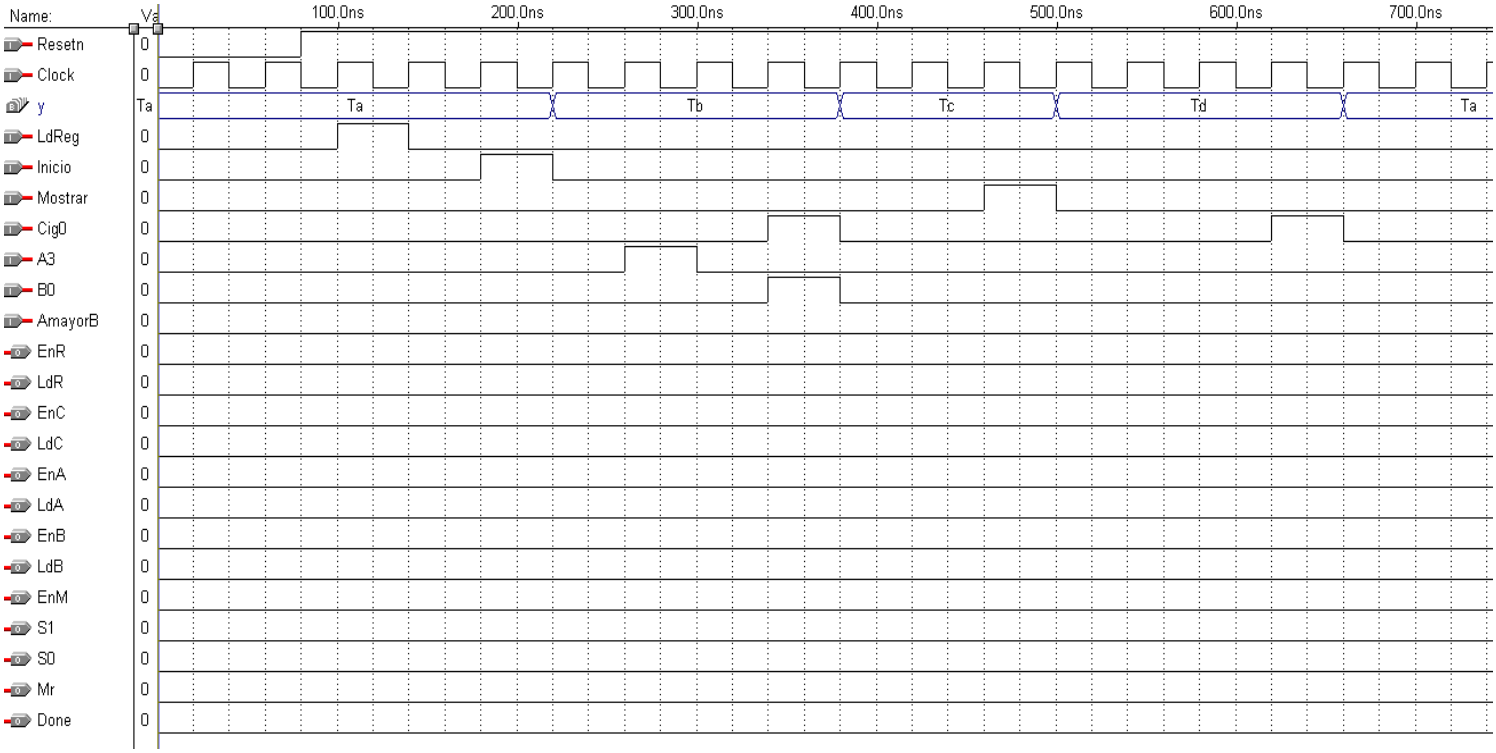
Dada la siguiente descripción en **VHDL** del funcionamiento de un **Sistema Digital**:
Presentar:

4. **Partición Funcional** del Sistema Digital.
5. **Diagrama ASM** del circuito Controlador del Sistema Digital, indicando claramente todas las salidas que deben ser generadas.
6. **Diagramas de Tiempo** del circuito **Controlador** asumiendo las condiciones de entrada dadas. Indique claramente los nombres y la duración de cada estado (**y**).

```
library ieee;
use ieee.std_logic_1164.all;

entity tema2 is
    port(Resetn, Clock  : in std_logic;
         Inicio, LdReg, Mostrar : in std_logic;
         DataA, DataB   : in std_logic_vector(3 downto 0);
         Done, BitR     : out std_logic);
end tema2;

architecture mixta of tema2 is
    type estado is (Ta, Tb, Tc, Td);
    signal y : estado;
    signal EnA, LdA, EnB, LdB, EnR, LdR, EnC, LdC, smux, zero : std_logic;
    signal EnM, MR, AgtB, AmayorB, Cig0, vcc : std_logic;
    signal S : std_logic_vector(1 downto 0);
    signal A, B, R, Cnt, zeros4, tres : std_logic_vector(3 downto 0);

    component registro_d_i is
        port (Resetn, Clock, En, Ld, L  : in std_logic;
              Entpar        : in std_logic_vector (3 downto 0);
              Q             : buffer std_logic_vector (3 downto 0));
    end component;

    component registro_i_d is
        port (Resetn, Clock, En, Ld, R      : in std_logic;
              Entpar        : in std_logic_vector (3 downto 0);
              Q             : buffer std_logic_vector (3 downto 0));
    end component;

    component contador_down is
        port (Resetn, Clock, En, Ld : in std_logic;
              Ent           : in std_logic_vector(3 downto 0);
              Q             : buffer std_logic_vector(3 downto 0));
    end component;

    component ffD is
        port(Resetn, Clock, En, D : in std_logic;
             Q : out std_logic);
    end component;
begin
```

```
-- Circuito Controlador
MSS_transiciones: process(Resetn, Clock)
    begin
        if Resetn ='0' then y <=Ta;
        elsif Clock'event and Clock ='1' then
            case y is
                when Ta=> if Inicio ='0' then y <=Ta; else y <=Tb; end if;
                when Tb=> if Cig0 ='0' then y <=Tb; else y <=Tc; end if;
                when Tc=> if Mostrar ='0' then y <=Tc; else y <=Td; end if;
                when Td=> if Cig0 ='0' then y <=Td; else y <=Ta; end if;
            end case;
        end if;
    end process;
MSS_salidas: process(y, LdReg, AmayorB, B(0))
    begin
EnA <='0'; LdA <='0'; EnB <='0'; LdB <='0'; EnR <='0'; LdR <='0';
EnC <='0'; LdC <='0';  EnM <='0'; S <="00"; Mr <='0'; Done <='0';
    case y is
        when Ta=> EnR<='1'; LdR <='1'; EnC <='1'; LdC <='1';
            if LdReg ='1' then EnA <='1'; LdA <='1'; EnB <='1'; LdB <='1'; end if;
            if Inicio ='1' then EnM <='1'; end if;
        when Tb=> EnA <='1'; EnB <='1'; EnR <='1'; EnC <='1';
            if (A(3) ='1' and AmayorB ='0') or (A(3) ='0' and B(0) ='0')
                then S(0) <='1'; end if;
            if A(3) ='0' then S(1) <='1'; end if;
        when Tc=> Done <='1';
            if Mostrar ='1' then EnC <='1'; LdC <='1'; end if;
        when Td=> EnC <='1'; Mr <='1'; EnR <='1';
    end case;
end process;
```

```
-- Procesador de datos
zero <='0'; zeros4 <="0000"; vcc <='1';
tres <="0011";

regA: registro_d_i port map(Resetn, Clock, EnA, LdA, zero, DataA, A);
regR: registro_d_i port map(Resetn, Clock, EnR, LdR, smux, zeros4, R);
regB: registro_i_d port map(Resetn, Clock, EnB, LdB, zero, DataB, B);

ff: ffD port map(Resetn, Clock, EnM, AgtB, AmayorB);

cnt_down: contador_down port map(Resetn, Clock, EnC, LdC, tres, Cnt);

AgtB <='1' when A > B else '0';

Cig0 <='1' when Cnt ="0000" else '0';

with S select
    smux <= B(0) when "00",
            not B(0) when "01",
            vcc when "10",
            zero when others;

BitR <= R(3) when MR ='1' else 'Z';
end mixta;
```



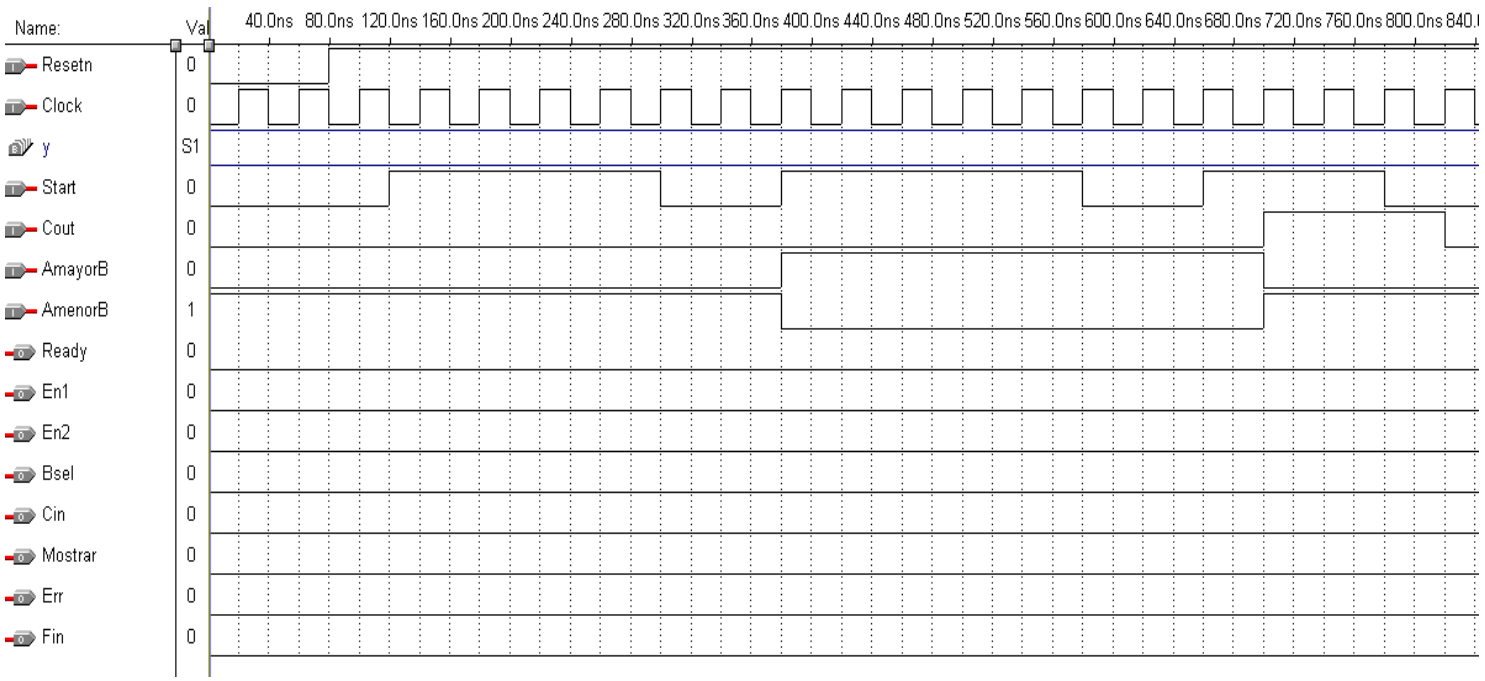
PROBLEMA # 4

Dada la siguiente descripción en **VHDL** del funcionamiento de un **Sistema Digital Presentar:**

1. **Partición Funcional** del Sistema Digital.
2. **Diagrama ASM** del circuito Controlador del Sistema Digital, indicando claramente todas las salidas que deben ser generadas.
3. **Diagramas de Tiempo** del circuito **Controlador** asumiendo las condiciones de entrada dadas. Indique claramente los nombres y la duración de cada estado (**y**).

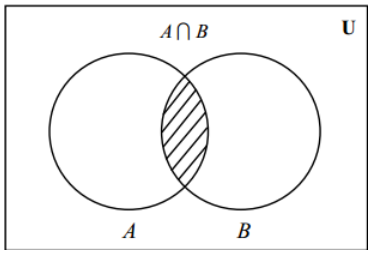
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.Std_logic_unsigned.all;

entity problema2_enero2009_new is
    port(Resetn, Clock  : in std_logic;
         Start          : in std_logic;
         N1, N2         : in std_logic_vector (3 downto 0);
         Ready, Err, Fin: out std_logic;
         Salida         : out std_logic_vector (3 downto 0));
end problema2_enero2009_new;

architecture mixta of problema2_enero2009_new is

    component registro_sost
        port(Resetn, Clock, En  : in std_logic;
             EntPar : in std_logic_vector (3 downto 0);
             Q      : out std_logic_vector (3 downto 0));
    end component;

    type estado is (S1, S2, S3, S4, S5);
    signal y: estado;
    signal En1, En2, Mostrar, Bsel, Cin, Cout : std_logic;
    signal AmayorB, AmenorB : std_logic;
    signal SN1, SN2, MN2 : std_logic_vector (3 downto 0);
    signal SR : std_logic_vector (4 downto 0);
begin

-- Controlador
MSS_trans: process(Resetn, Clock)
begin
    if Resetn ='0' then y <=S1;
        elsif Clock'event and Clock ='1' then
        case y is
            when S1=> if Start ='0' then y <=S1; else y <=S2; end if;
            when S2=> if AmayorB='1' then y <=S3;
                        elsif AmenorB ='1' then y <=S4; else y <=S1; end if;
            when S3=> y <=S5;
            when S4=> if Cout ='0' then y <=S5; else y <=S1; end if;
            when S5=> if Start ='0' then y <=S1; else y <=S5; end if;
        end case; end if; end process;
```

```
MSS_salidas: process(y, Start, AmenorB, Cout)
begin
Ready <='0'; Err <='0'; Fin <='0'; Mostrar <='0';
Bsel <='0'; En1<='0'; En2 <='0'; Cin <='0';
    case y is
        when S1=> Ready <='1';
            if Start ='0' then En1 <='1'; En2 <='1'; end if;
        when S2=>
        when S3=> Bsel <='1'; Cin <='1'; Mostrar <='1';
        when S4=> if Cout ='1' then Err <='1';
                    else Mostrar <='1'; end if;
        when S5=> Fin <='1';
    end case; end process;

-- Procesador de datos

    reg1: registro_sost port map(Resetn, Clock, En1, N1, SN1);
    reg2: registro_sost port map(Resetn, Clock, En2, N2, SN2);
    reg3: registro_sost port map(Resetn, Clock, Mostrar, SR(3 downto 0), Salida);

    AmayorB <='1' when SN1 > SN2 else '0';
    AmenorB <='1' when SN1 < SN2 else '0';


    SR <= SN1 + MN2 + Cin;
    Cout <= SR(4);

    MN2 <= SN2 when Bsel ='0' else not SN2;

end mixta;
```



PROBLEMA # 5

Diseñar un **Sistema Digital** que permita hacer una operación entre conjuntos de acuerdo al gráfico mostrado. Los conjuntos A y B son conjuntos de números cuyos elementos están en el intervalo [0, 255]. El ingreso de los elementos en cada conjunto es byte por byte.



U: Conjunto de números de 8 bits.
A: Conjunto de números de 8 bits, con un número de elementos ≤ 10.
B: Conjunto de números de 8 bits, con un número de elementos ≤ 10.
A∩B: Números en común de 8 bits de los conjuntos A y B.

SEÑALES

InNúmero.- Es una señal de 8 bits donde el usuario coloca cada elemento a ingresar en los conjuntos A y B.
IngresoA.- Señal que permite al usuario hacer el ingreso de los elementos del conjunto A. (Considere que el número debe estar presente en la entrada InNúmero).
IngresoB.- Señal que permite al usuario hacer el ingreso de los elementos del conjunto B. (Considere que el número debe estar presente en la entrada InNúmero).
LedIngreso.- LED indicador que todavía el usuario puede ingresar elementos. Este LED puede apagarse si el usuario presiona la tecla **FinIngreso** o si el usuario ingresó el número máximo de elementos en cada conjunto (A, B).
FinIngreso.- Señal que el usuario puede usar para indicar al Sistema que ya no desea ingresar más elementos en los conjuntos A y B.
Start.- Luego de ingresar correctamente los elementos en cada conjunto el usuario deberá presionar la tecla **Start** para empezar ejecutar el algoritmo.
OK.- Este LED indica que el Sistema terminó de encontrar los elementos de la intersección entre los conjuntos A y B.
A∩B.- Esta señal de 8 bits muestra número a número con un intervalo de 1 segundo cada uno de los elementos de la intersección entre los conjuntos A y B. Luego que el sistema termina de mostrar todos los elementos de la intersección, regresa al estado inicial para empezar nuevamente con el proceso.

Presente:

a) Diagrama de bloques (5 puntos)
b) Partición Funcional con cada señal claramente identificada (10 puntos)
c) Diagrama ASM del controlador (5 puntos)

PROBLEMA # 6

Un circuito **Data Logger** es un circuito que almacena información que luego puede ser descargada a una computadora para analizar los datos.
Una empresa que cultiva **ROSAS** para **San Valentín** pide a los estudiantes de Sistemas Digitales 2 diseñar un **Sistema para administrar su Data Logger**.

El Data Logger tiene una memoria de **4Gx32** y se le conecta un sensor de temperatura y uno de humedad.
El sistema debe leer la temperatura y la humedad en intervalos de 10 minutos y almacenar en la memoria de la siguiente forma:
En la primera casilla de memoria, el día y la hora (palabra de 32 bits), siguiente casilla la temperatura y en la tercera el valor de la humedad relativa del suelo (palabras de 32 bits cada una), hasta que se llene la memoria o se presione la tecla **INFORME.**
El sistema estará grabando por semanas incluso, hasta que se presione la tecla **INFORME**. Una vez hecho esto se visualizará en displays de siete segmentos los eventos en que la temperatura estuvo más alta en cada día, es decir mostrará la terna (día/hora, temperatura, humedad relativa) en secuencia pero solo de un valor por día (aquel cuya temperatura sea la más alta del día). Para pasar de un valor a otro de la terna debe utilizarse una tecla **MOSTAR,** asimismo de una terna a otra. Si la memoria se llena antes de que la tecla **INFORME** sea presionada el circuito debe enviar una señal de **ALARMA** y dejar de grabar. Una vez mostrado todas las ternas el circuito debe enviar una señal **SIN_DATOS,** la que también se podría encender si se presiona la tecla **INFORME** que se haya guardado el primer dato.

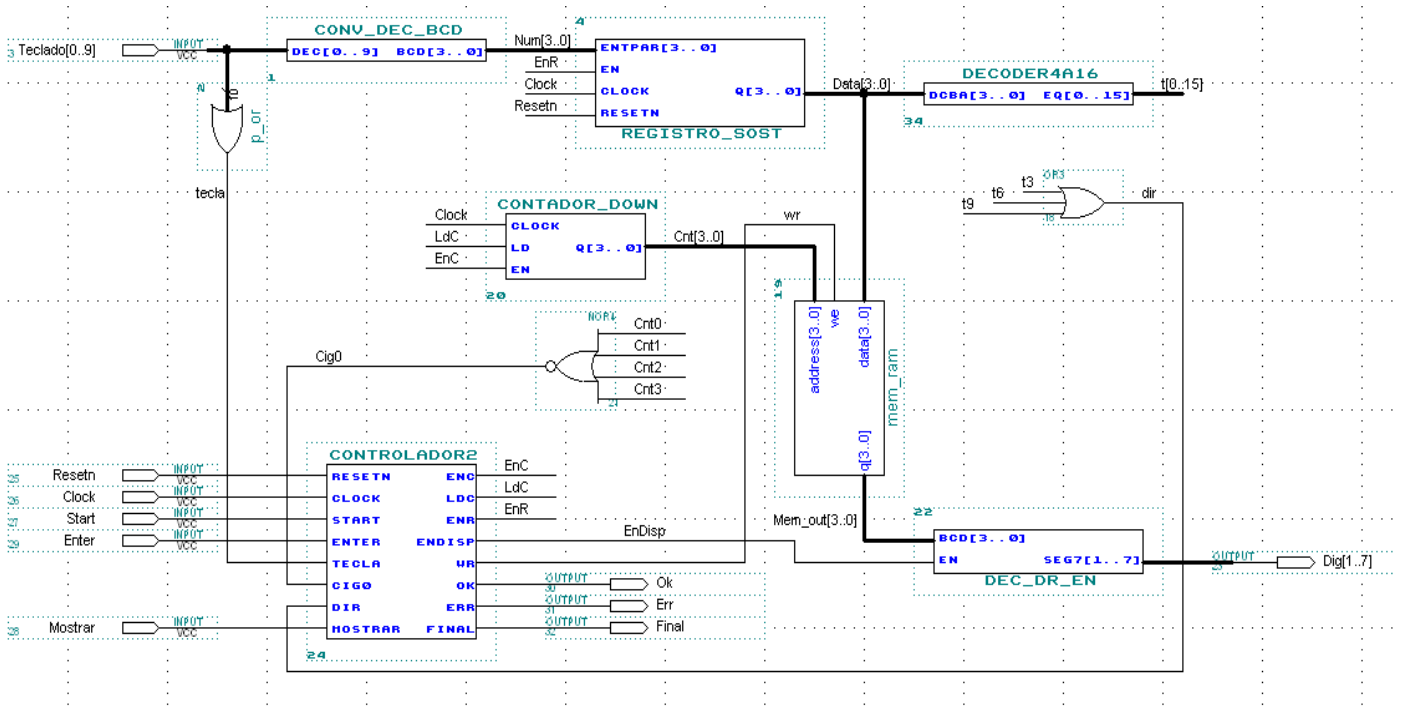
Presente:

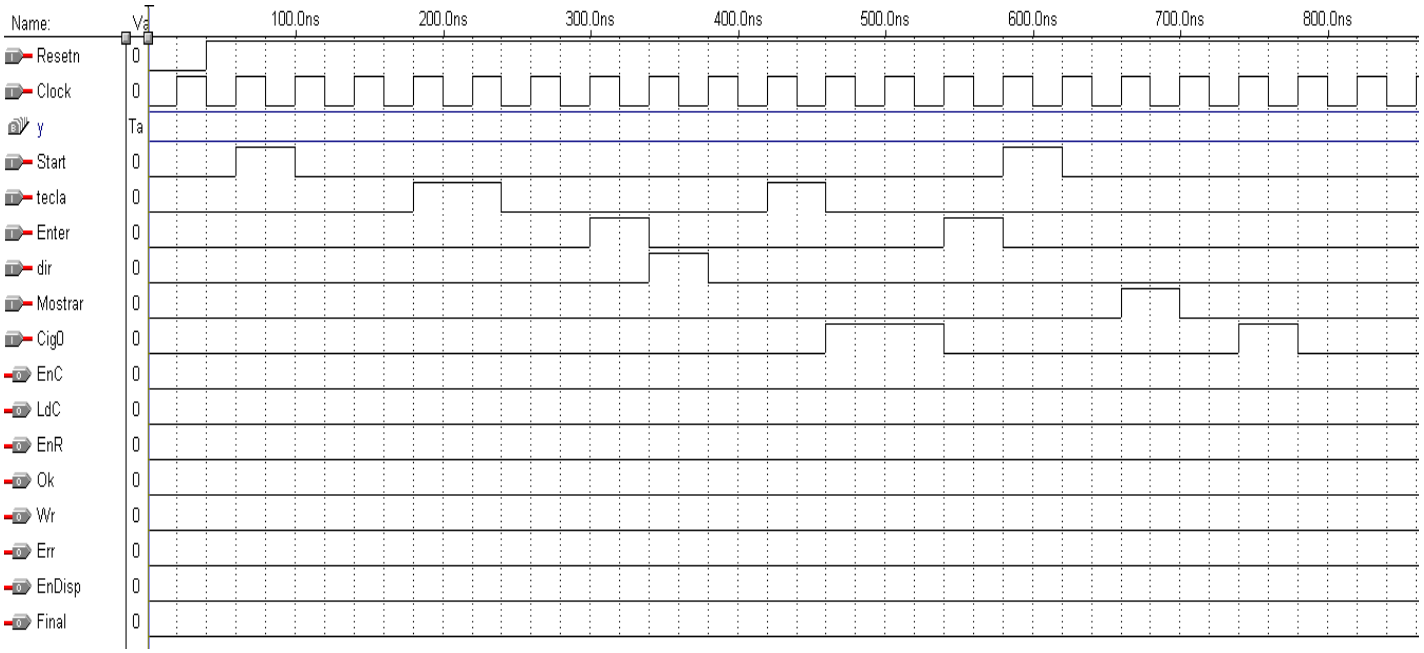
1. **Diagrama general de entradas/salidas** (**2/30**)
2. **Diagrama de bloques del sistema digital** (**8/30**)
3. **Partición Funcional** (**10/30**)
4. **Diagrama ASM** del circuito **Controlador** debidamente documentada. (indicar todos las entradas y salidas) (**10/30**)

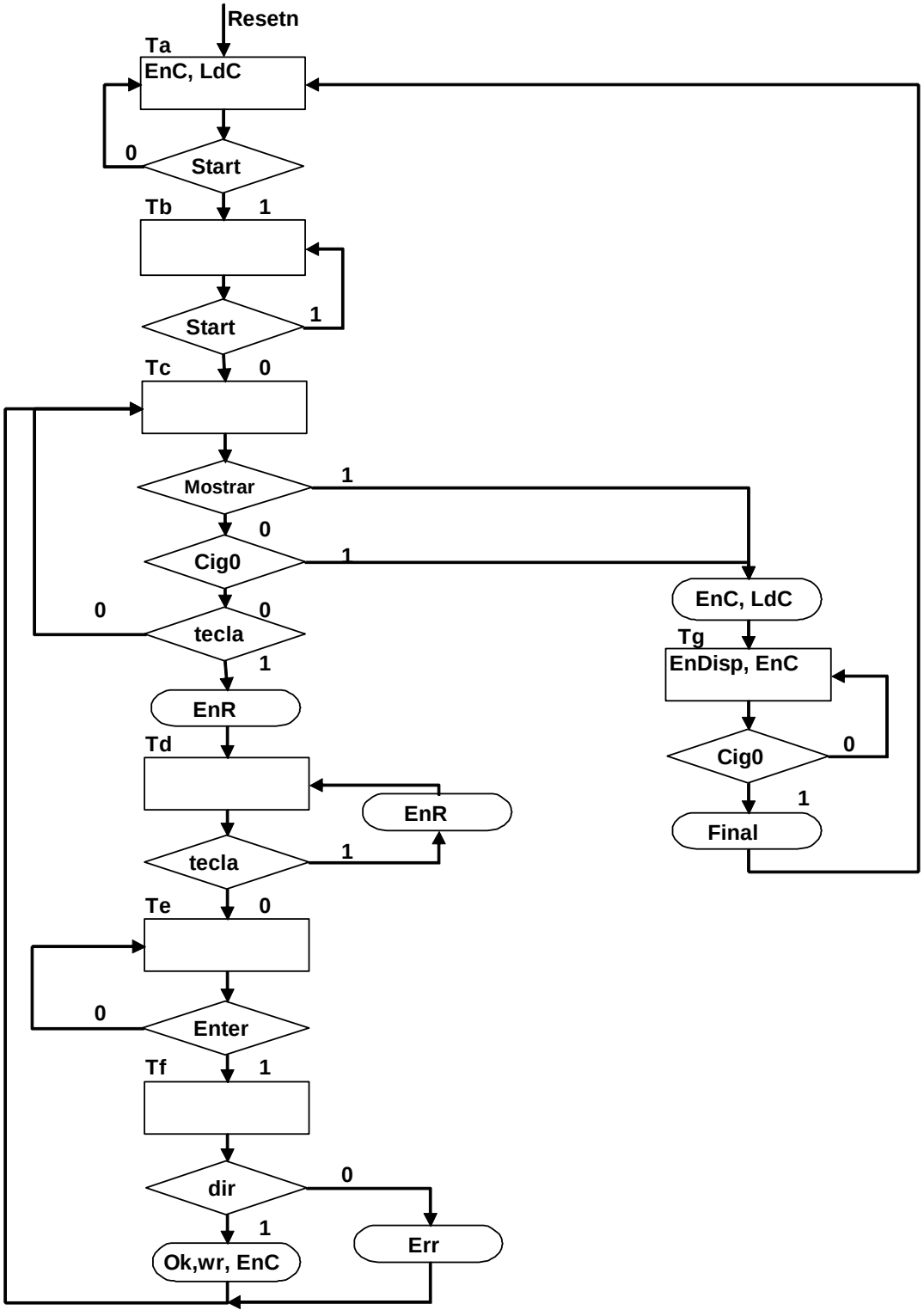
Nota. Asuma que la implementación será hecha usando tecnología FPGA y lenguaje VHDL. Dimensionar adecuadamente cada componente.

PROBLEMA # 7

Para el siguiente Sistema Digital, se muestran la **Partición Funcional** y el **Diagrama ASM** del circuito Controlador.






Resetn
Ta
EnC, LdC
0
Start
Tb
1
1
Start
Tc
0
Mostrar
1
0
Cig0
1
0
0
tecla
1
EnR
Td
EnR
tecla
1
Te
0
0
Enter
Tf
1
dir
0
1
Ok,wr, EnC
Err
EnC, LdC
Tg
EnDisp, EnC
Cig0
0
1
Final

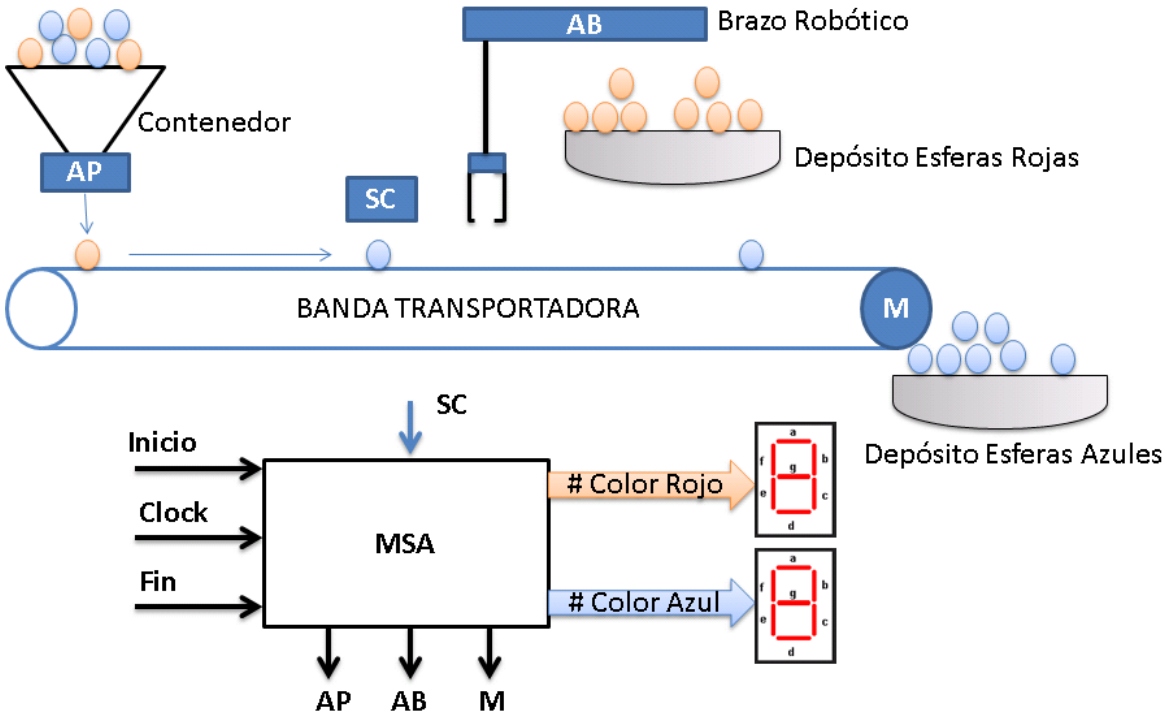
Presente:

1. Descripción del Sistema en un solo programa en **VHDL** usando las declaraciones ***process – case*** para describir las Transiciones de Estados y las Salidas del **Controlador,** y la **descripción estructural** para el **Procesador de Datos**.
   Asuma que dispone de archivos **.vhd** en la misma carpeta de Trabajo para todos los sub-circuitos que forman parte del Sistema Digital **excepto para el Controlador y para las puertas lógicas**.
   Así mismo suponga que el orden de las entradas en la declaración ***port*** de los sub-circuitos es similar (de izquierda a derecha y de arriba abajo) al del Diagrama Esquemático presentado.

2. Grafique los **Diagramas de Tiempo** del circuito **Controlador** asumiendo las condiciones de entrada dadas. Indique claramente los nombres de los estados (**y**) que corresponden a cada periodo de reloj.

PROBLEMA # 8

Se tienen un proceso de clasificación de esferas por color, esferas que inicialmente se encuentran almacenadas en un **contenedor** que en la salida de las esferas tiene acoplado un **Actuador de Paso (AP)** que permite pasar una sola esfera por vez. Luego cada esfera cae a la **banda transportadora** hasta llegar al **sensor de color (SC)** y si es la esfera roja un **actuador de brazo robótico (AB)** que levanta la lleva al **depósito de esferas rojas**, caso contrario avanza por la banda transportadora hasta llegar al **depósito de esferas azules**.
Se quiere realizar un sistema Digital que luego de presionar y soltar el botón **INICIO**, accione el **Motor (M)** de la **banda transportadora**, luego activará el **actuador de paso** para permitir que caiga una esfera por vez a la banda (la válvula deberá ser activada por 5 segundos por su respuesta mecánica). Una vez que una esfera está en la banda transportadora ésta pasará por el **sensor de color** el cual determinará el destino final de la esfera. Si la esfera es azul, la banda sigue trabajando hasta llevar a ésta al contenedor de esferas azules que queda al final de la banda transportadora. Si el color de la esfera es rojo, la banda transportadora deberá detenerse por 15 segundos que es el tiempo estimado que el brazo robótico demora en ***bajar - tomar a esfera - llevar la esfera - dejar la esfera - regresar a su posición original***. El **actuador de paso** se activará automáticamente si el sensor de color detecta que es una esfera azul, pero si es una esfera roja el actuador de paso se activará luego de que el brazo robótico llegue a su posición inicial. Adicionalmente el sistema permitirá ***visualizar en dos Display*** la cantidad de esferas rojas y azules en tiempo real. El sistema regresará a su estado inicial cuando el operador presiona el botón de **FIN** o cuando uno de los dos depósitos alcance la cantidad de nueve esferas.



Presentar:

1. **Diagrama de bloques** del Sistema Digital.
2. **Partición Funcional** del Sistema Digital.

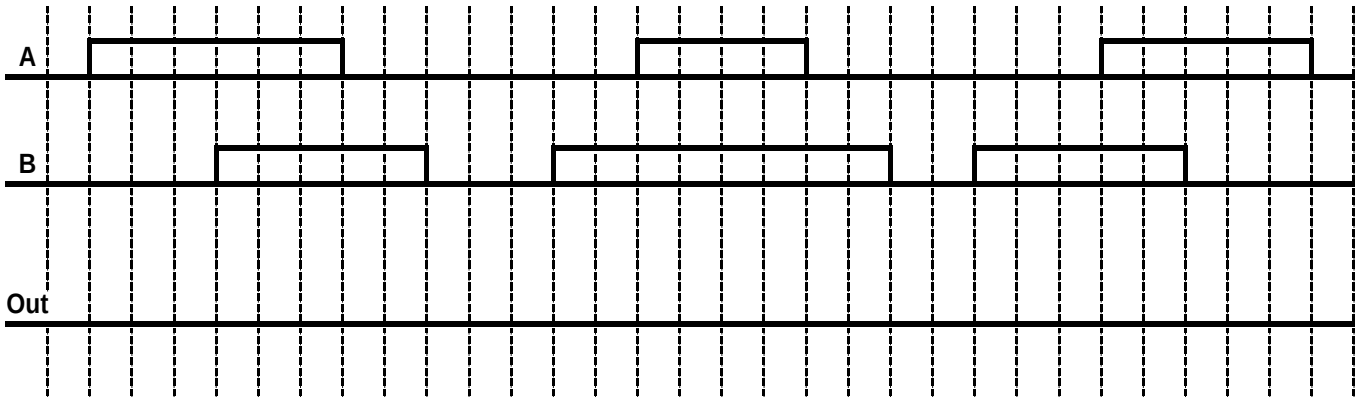
3. **Diagrama ASM** del controlador.

PROBLEMA # 9

Diseñar en modo Fundamental una **MSA** (Maquina Secuencial Asincrónica) que funciona de la siguiente manera.
La **MSA** tiene dos entradas **A** y **B** y una salida **Out**.
Inicialmente las entradas **A** y **B** y la salida **Out** son igual **0**.
Cuando una de las entradas (**A** o **B**) se hace igual **1**, la salida **Out** también se hace igual a **1**.
Si, mientras una de las entradas se mantiene igual a **1**, la otra se hace igual a **1** también, la salida **Out** se mantiene igual a **1**.
Si, luego, una de las entradas se hace igual a **0**, la salida **Out** se hace igual a **0** también y se mantiene igual a **0** aun cuando esta entrada se hace igual a **1** de nuevo, hasta que **MSA** regresa al estado inicial.

Presente:

1. Diagrama de Estados Primitivo (Formato: **A B / Out).** Mapa de Estados Primitivo. Tabla de Implicantes.
2. Diagrama de Estados Reducido. Mapa de asignación de Código de Estados.
3. Mapa de Excitación. Mapas para las salidas Y_1 y Y_0 y para la salida Out.
4. Diagrama de tiempo para la salida **Out** asumiendo valores de las entradas **A** y **B** dados. Indica claramente los periodos de tiempo correspondiente a cada estado de su Diagrama de Estados Reducido.
5. Indique si su circuito corre riesgo de tener los **Hazard Estáticos** o no. Como se puede evitar.



PROBLEMA # 10

Considere el controlador asíncrono para un automóvil de juguete. El sistema tiene dos teclas **ROJO** y **NEGRO** y dos salidas **Z1** y **Z2**.

Z1	Z2	Acción
0	0	Para
0	1	Gira izquierda
1	0	Gira derecha
1	1	Línea Recta

Con las teclas sin presionar el automóvil no se mueve. Si cualquiera de las dos teclas se presiona el automóvil se mueve en línea recta. Cuando la tecla roja se presiona y luego la negra el automóvil gira a la derecha y si se libera cualquiera el automóvil sigue en línea recta. Cuando la tecla negra se presiona primero y luego la roja el automóvil gira a la izquierda y si se libera cualquiera el automóvil sigue en línea recta.

Presentar:

1. Diagrama de Estados Primitivo (Formato: **R N / Z1 Z2).** Mapa de Estados Primitivo (**6/21**)
2. Tabla de Implicantes. Diagrama de Estados Reducido (**5/21**)
3. Mapa de asignación de Código de Estados. Mapa de Excitación. Mapas para las variables estado siguiente y mapas para las salidas **Z1 y Z2** (**9/21**)
4. Indique si su circuito corre riesgo de tener **Hazard Estáticos** o no. Como los podría evitar. (**1/21**)