Electrónica Digital

Práctica nº 4 (primera parte)

Diseño de sistemas digitales

Trabajo a realizar en la práctica

La práctica consiste en la implementación de un sistema secuencial para la gestión del aforo de un parking (Fig. 1). Dicho sistema permite monitorizar el número de vehículos que hay sobre dos displays de 7 segmentos. Para la implementación de la aplicación, además de los displays, se disponen de dos sensores opto-electrónicos, uno en la entrada (SE) y otro en la salida (SS) del parking. Estos sensores nos informan de que ha entrado un nuevo vehículo o ha salido un vehículo respectivamente.

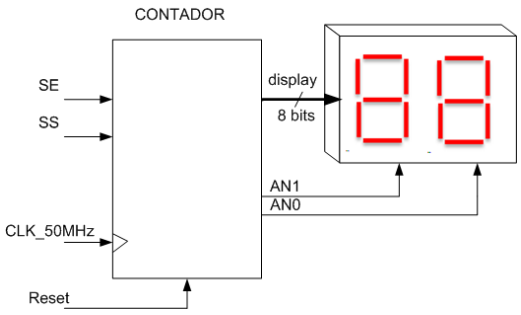

Fig. 1. Sistema a implementar.

Para el diseño del sistema se propone el esquema de bloques descrito en la Fig. 2. Dicho esquema consiste en dos etapas claramente diferenciadas, la etapa que se encarga de contar el número de vehículos presentes en el parking y la etapa de monitorización.

a) La etapa de control de ocupación del parking, está formada por los módulos M0, M1, M2 y M3:

- El módulo M0 gestiona el modo de cuenta (ascendente/descendente) en función de si se ha activado el sensor SE o el SS.

- Los módulos M1 se encargan de detectar los flancos descendentes que se producen en las señales de los sensores SE y SS. Solamente cuando se detecta un flanco descendente, procedente de las señales de uno de los dos sensores (SE, SS), se procederá a actualizar la cuenta de vehículos.
- El módulo M2 consiste en un contador ascendente/descencente que incrementa o decrementa la cuenta en función de si ha entrado o ha salido un vehículo del parking (módulo M0).
- El módulo M3 se encarga de gestionar la multiplexación de los datos a visualizar y genera la señal Enable_1kHz para evaluar el estado de los sensores SE y SS.

b) La etapa de monitorización (módulo M4) es la que nos permite visualizar sobre los dos displays, los datos relacionados con la ocupación del parking (Unidades y Decenas de vehículos). Tal y como se puede observar en la Fig. 2, está formada por un multiplexor 2x4 y un decodificador BCD-7segementos.

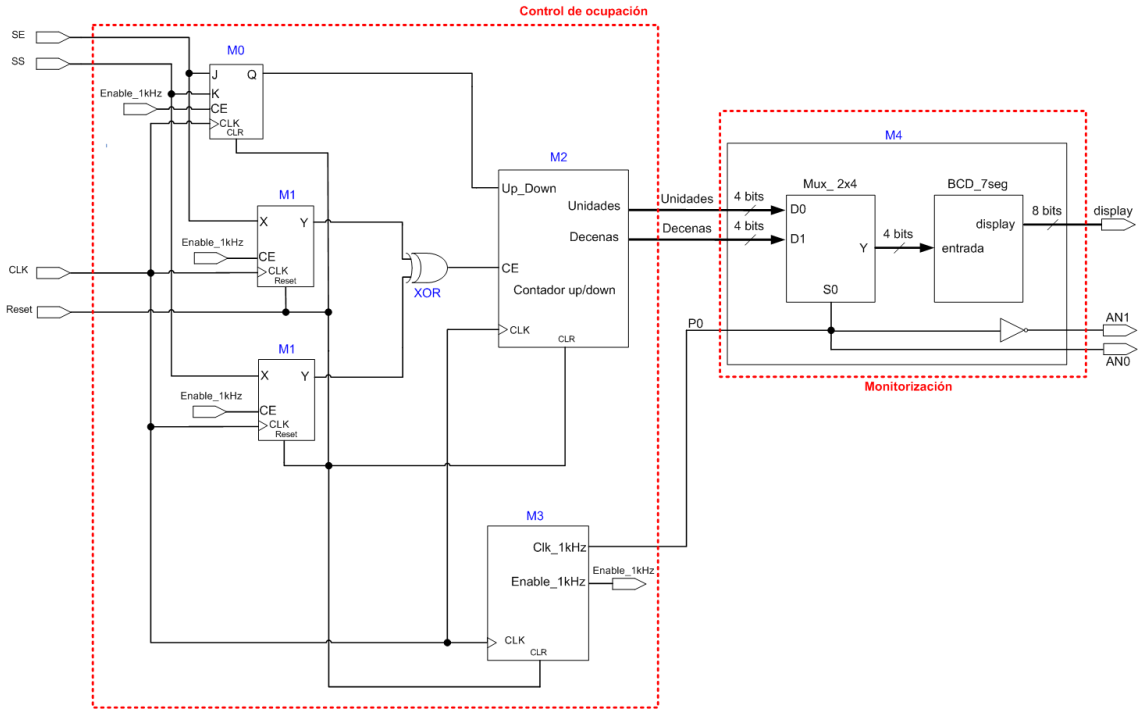

Figura 2. Esquema de bloques propuesto.

Para la implementación del sistema digital, se empleará la herramienta software WebPack ISE y la placa de evaluación de Digilent, que incorpora la FPGA XC3S200 de la serie Spartan 3 del fabricante Xilinx.

El trabajo de la práctica se organiza en dos sesiones. En la primera sesión se pretende abordar el diseño de la etapa de monitorización, mientras que en la última sesión se realizará el diseño de la etapa de control de ocupación del parking, completando la descripción del sistema a diseñar. Cada una de estas sesiones está dividida a su vez en dos apartados: a) Estudio Previo (Diseño del circuito) y b) Implementación del sistema digital sobre la FPGA XC3S200.

Nota: ***Se deberán realizar las tareas de diseño marcadas con la letra "D" seguida del número de tarea, correspondientes al apartado del estudio previo antes de la sesión de prácticas.***

1 Diseño de la etapa de monitorización.

1.1 Estudio Previo de la etapa de monitorización

En el siguiente apartado se realizará el diseño de la etapa de monitorización, descrita en la Fig. 2.

D1. Descripción en lenguaje VHDL del bloque BCD_7seg.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;


entity BCD_7seg is
    Port ( entrada : in  STD_LOGIC_VECTOR (3 downto 0);
           display : out  STD_LOGIC_VECTOR (7 downto 0));
end BCD_7seg;

architecture Behavioral of BCD_7seg is

begin

...

end Behavioral;
```

D2. Descripción en lenguaje VHDL del bloque Mux_2x4.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;


entity Mux_2x4 is
       Port(D0,D1 : in STD_LOGIC_VECTOR (3 downto 0);
            S0 : in STD_LOGIC;
            Y: out STD_LOGIC_VECTOR (3 downto 0));
end Mux_2x4;


architecture Behavioral of Mux_2x4 is

begin

...

end Behavioral;
```

D3. Descripción en lenguaje VHDL del módulo M4, correspondiente a la etapa de monitorización.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity M4 is
       port(Unidades,Decenas : in std_logic_vector (3 downto 0);
            P0: in std_logic;
            DISPLAY: out std_logic_vector (7 downto 0);
            AN0,AN1: out std_logic);
end M4;

architecture Behavioral of M4 is

component Mux_2a4 is
       Port(D0,D1 : in STD_LOGIC_VECTOR (3 downto 0);
            S0 : in STD_LOGIC;
            Y: out STD_LOGIC_VECTOR (3 downto 0));
end component;

component BCD_7seg is
       Port(entrada : in  STD_LOGIC_VECTOR (3 downto 0);
           display : out  STD_LOGIC_VECTOR (7 downto 0));
end component;

....

begin

....

end Behavioral;
```

1.2 Implementación de la etapa de monitorización sobre la FPGA XC3S200

En este apartado se describe paso a paso el proceso a seguir para la simulación de la etapa de monitorización, empleando la herramienta software WebPack ISE y realizando la descripción de dicha etapa mediante una descripción estructural.

1.2.1 Descripción mediante lenguaje VHDL

T1. Iniciar la herramienta de trabajo proporcionada por Xilinx, **"ISE Project Navigator"** , y crear un nuevo proyecto para la descripción del circuito lógico a implementar, seleccionando la opción de descripción mediante lenguaje hardware (HDL).

ISE creación de un proyecto

T2. Crear un fichero fuente en el que se realizará la descripción del circuito lógico a implementar en lenguaje VHDL. A la hora de introducir los datos referidos a las entradas y salidas del sistema, seguiremos las indicaciones de la Fig. 3.

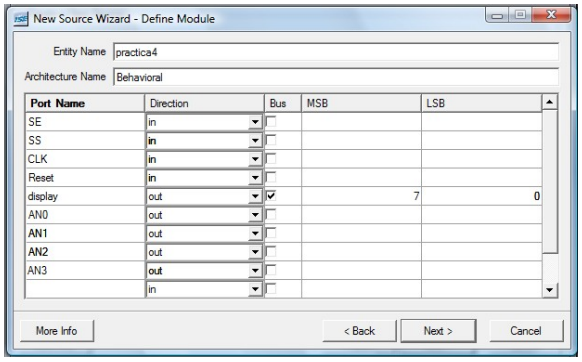

Fig. 3. Creación del fichero fuente en VHDL.

T3. Crear el componente relacionado con el módulo M4 de acuerdo con la Fig. 4.

```
Sentencias: Declaración de componenetes:
        COMPONENT nombre_componente
            PORT (port_list);
        END COMPONENT;
```

Fig. 4. Declaración de un componente.

T4. Instanciar el componentes M4, creando las señales internas Unidades, Decenas y P0 antes de la etiqueta de inicio de la arquitectura.

T5. Una vez hemos instanciado el componente M4, faltará describirlo. Para ello, situar el ratón encima del componente que aparece en la ventana **"Sources"**, pulsar el botón derecho del ratón y seleccionar la opción **"New Source"** (Fig. 5). Esta acción nos permitirá añadir la descripción en lenguaje VHDL del componente seleccionado, a partir de la descripción realizada en el apartado D3 del estudio previo.

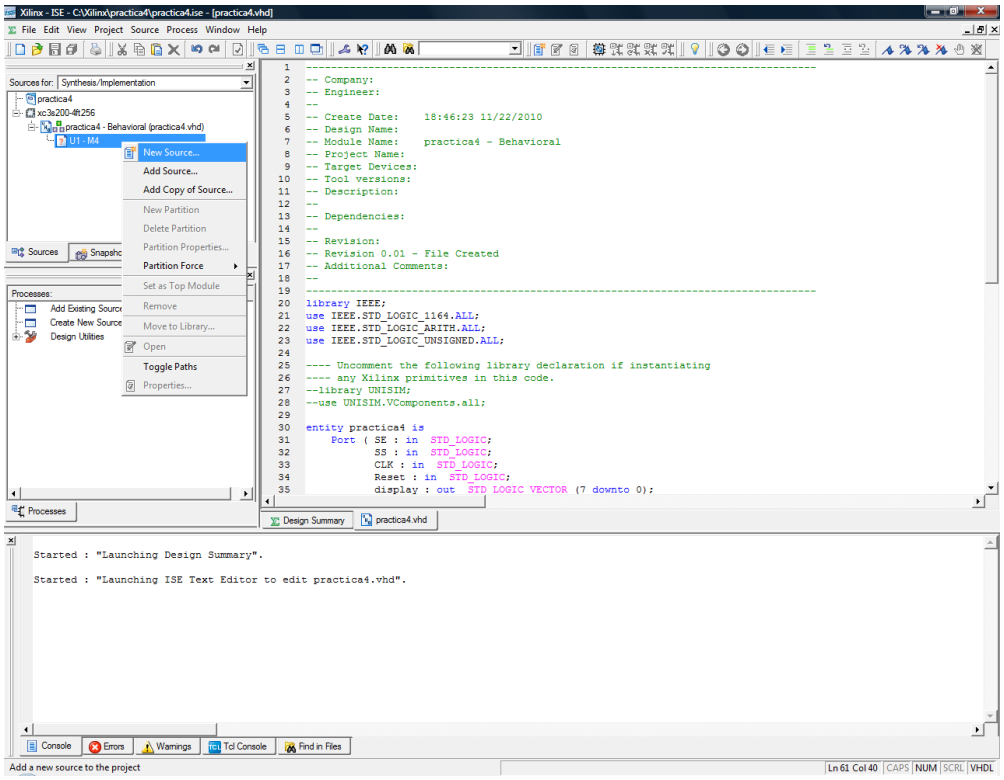

Fig. 5. Descripción de la etapa de monitorización.

T6. Una vez añadida la descripción del componente M4, faltará describir los componentes Mux2x4 y BCD_7seg, a partir de los apartados D1 y D2 del estudio previo. Así que será necesario repetir el procedimiento anterior (T5) con los componentes Mux2x4 y BCD_7seg (Fig. 6).

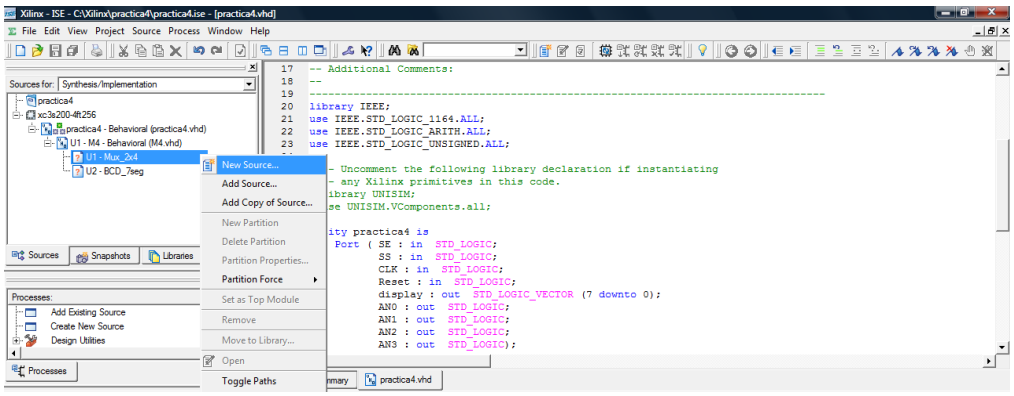

Fig. 6. Descripción de los componentes Mux2x4 y BCD_7seg.

T7. Una vez finalizada la descripción de los diferentes componentes (Mux2x4 y BCD_7seg), comprobaremos que no exista ningún error de sintaxis. Salvar los ficheros *.vhd, seleccionar el componente M4 en la ventana **"Sources",** y ejecutar la opción **"Check Syntax"** que aparece en la ventana **"Processes"**.

Si no se ha producido ningún error, el proceso de "**Check Syntax"** quedará marcado en verde. En el caso de que se hubiese producido algún error, se nos indicará la línea correspondiente, así como el error detectado para su corrección.

T8. Crear un banco de pruebas para simular el funcionamiento del componente correspondiente al módulo M4. Para ello, seleccionar el fichero fuente *.vhd correspondiente a dicho componente en la ventana **Sources** y añadir un nuevo fichero al proyecto del tipo **"VHDL Test Bench"**. Editar los estímulos que se aplicarán en las entradas, tal y como se muestra en el cronograma de la Fig. 7.

Antes de iniciar el proceso de simulación, seleccionar un tiempo se simulación de 1200 ns.

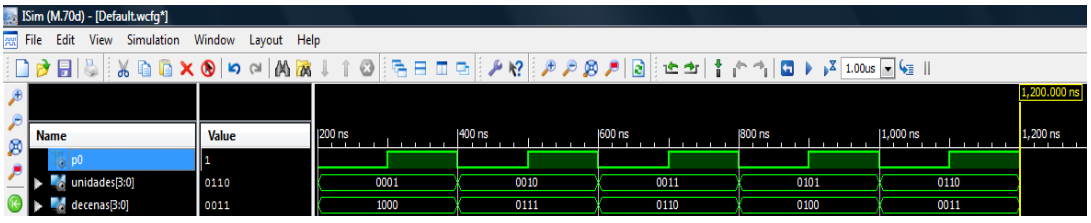

Fig. 7. Edición de los estímulos del componente M4.

ISE simulación de un diseño

T9. Comprobar si el cronograma obtenido coincide con el resultado esperado, en caso contrario rehacer el diseño del componente M4 y repetir el proceso.

Electrónica Digital

Práctica nº 4 (segunda parte)

Diseño de sistemas digitales

Trabajo a realizar en la práctica

La práctica consiste en la implementación de un sistema secuencial para la gestión del aforo de un parking (Fig. 1). Dicho sistema permite monitorizar el número de vehículos que hay sobre dos displays de 7 segmentos. Para la implementación de la aplicación, además de los displays, se disponen de dos sensores opto-electrónicos, uno en la entrada (SE) y otro en la salida (SS) del parking. Estos sensores nos informan de que ha entrado un nuevo vehículo o ha salido un vehículo respectivamente.

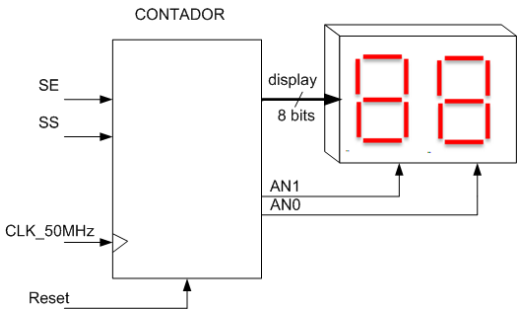

Fig. 1. Sistema a implementar.

Para el diseño del sistema se propone el esquema de bloques descrito en la Fig. 2. Dicho esquema consiste en dos etapas claramente diferenciadas, la etapa que se encarga de contar el número de vehículos presentes en el parking y la etapa de monitorización.

a) La etapa de control de ocupación del parking, está formada por los módulos M0, M1, M2 y M3:

- El módulo M0 gestiona el modo de cuenta (ascendente/descendente) en función de si se ha activado el sensor SE o el SS.

- Los módulos M1 se encargan de detectar los flancos descendentes que se producen en las señales de los sensores SE y SS. Solamente cuando se detecta un flanco descendente, procedente de las señales de uno de los dos sensores (SE, SS), se procederá a actualizar la cuenta de vehículos.
- El módulo M2 consiste en un contador ascendente/descencente que incrementa o decrementa la cuenta en función de si ha entrado o ha salido un vehículo del parking (módulo M0).
- El módulo M3 se encarga de gestionar la multiplexación de los datos a visualizar y genera la señal Enable_1kHz para evaluar el estado de los sensores SE y SS.

b) La etapa de monitorización (módulo M4) es la que nos permite visualizar sobre los dos displays, los datos relacionados con la ocupación del parking (Unidades y Decenas de vehículos). Tal y como se puede observar en la Fig. 2, está formada por un multiplexor 2x4 y un decodificador BCD-7segementos.

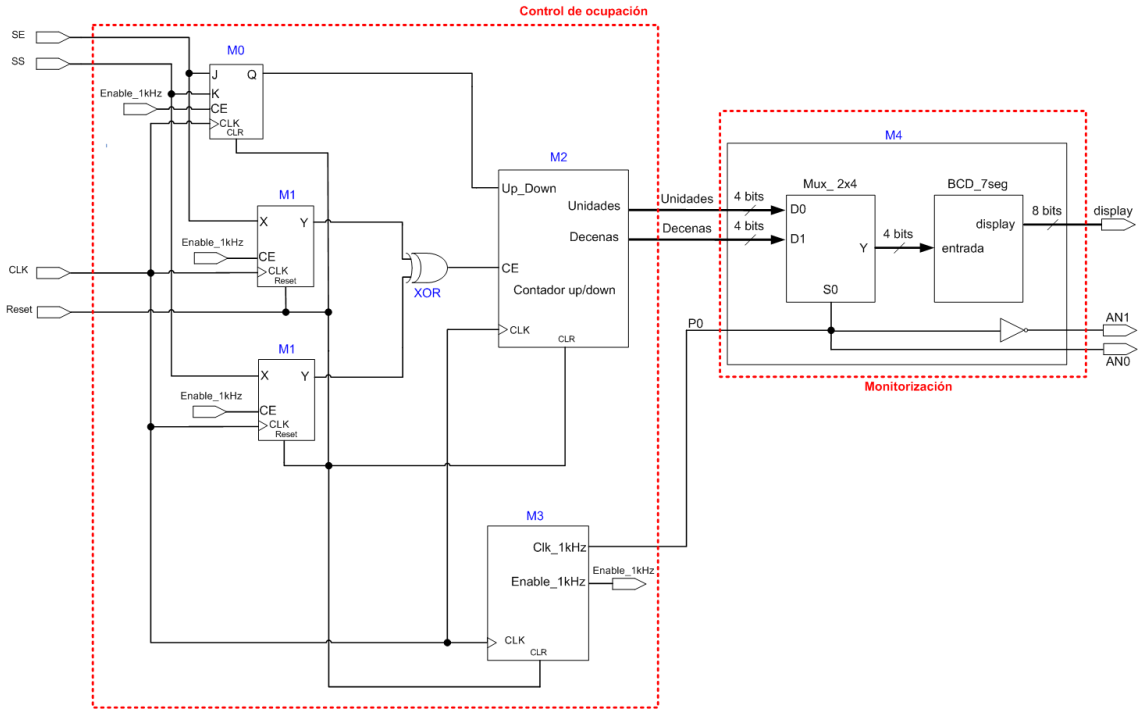

Figura 2. Esquema de bloques propuesto.

Para la implementación del sistema digital, se empleará la herramienta software WebPack ISE y la placa de evaluación de Digilent, que incorpora la FPGA XC3S200 de la serie Spartan 3 del fabricante Xilinx.

El trabajo de la práctica se organiza en dos sesiones. En la primera sesión se pretende abordar el diseño de la etapa de monitorización, mientras que en la última sesión se realizará el diseño de la etapa de control de ocupación del parking, completando la descripción del sistema a diseñar. Cada una de estas sesiones está dividida a su vez en dos apartados: a) Estudio Previo (Diseño del circuito) y b) Implementación del sistema digital sobre la FPGA XC3S200.

Nota: ***Se deberán realizar las tareas de diseño marcadas con la letra "D" seguida del número de tarea, correspondientes al apartado del estudio previo antes de la sesión de prácticas.***

2 Diseño de la etapa de control de ocupación del parking

2.1 Estudio Previo de la etapa de control de ocupación del parking

En el siguiente apartado se realizará el diseño de la etapa de control de la ocupación del parking, descrita en la Fig. 2.

D1. Descripción en lenguaje VHDL del módulo M0, que consiste en un flip flop J-K con entrada de clock enable (Fig. 3).

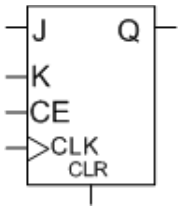

Figura 3. Flip flop J-K con entrada de clock enable (CE).

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;


entity M0 is
       port(J,K,CE,CLK,CLR : in std_logic;
            Q: out std_logic);
end M0;

architecture Behavioral of M0 is

....

begin
....

end Behavioral;
```

D2. Descripción estructural en lenguaje VHDL del módulo M2. Este módulo consiste en dos contadores BCD reversibles, tal como se indica en la Fig. 4.

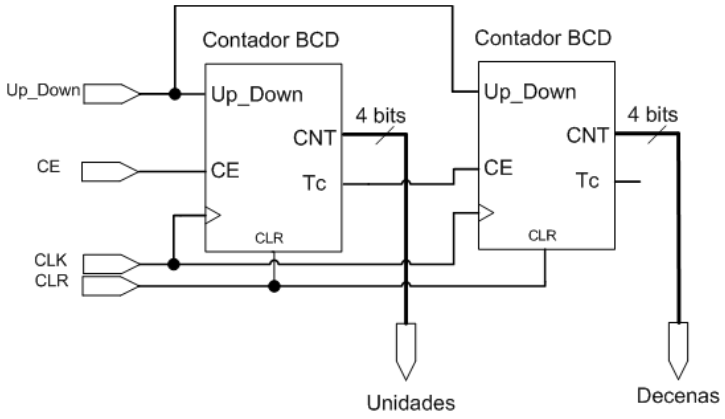

Figura 4. Módulo M2.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity M2 is
       port(UP_DOWN,CE,CLK,CLR: in std_logic;
            Unidades,Decenas : out std_logic_vector (3 downto 0));
end M2;

architecture Behavioral of M2 is

component Contador_BCD is
       port(UP_DOWN,CE,CLK,CLR: in std_logic;
            Tc: out std_logic;
            CNT : out std_logic_vector (3 downto 0));
end component;

....
begin

....
end Behavioral;
```

2.2 Implementación de la etapa de control de ocupación del parking sobre la FPGA XC3S200

En este apartado se describe paso a paso el proceso a seguir para la simulación de la etapa de control de ocupación del parking, empleando la herramienta software WebPack ISE y realizando la descripción de dicha etapa mediante una descripción estructural.

T1. Iniciar la herramienta de trabajo proporcionada por Xilinx, **"ISE Project Navigator"** , y abrir el proyecto que se creó en la sesión anterior (**"File->Open Project"**).

T2. Completar la arquitectura del fichero *.vhd que describe el funcionamiento del sistema a implementar, utilizando una descripción estructural. Para ello, crear los componentes relacionados con los módulos M0, M1, M2 y M3. A la hora de crear los componentes M1 y M3 seguir la sintaxis que se muestra en la Fig. 5.

```
component M1
      port(X,CE,CLK,CLR: in std_logic;
           Y: out std_logic);
end component;

component M3
      port (CLK,CLR : in std_logic;
            Enable_1kHz,clk_1kHz: out std_logic);
end component;
```

Fig. 5. Declaración de los componentes M1 y M3.

T3. Instanciar cada uno de los componentes, interconectándolos entre si, tal y como se muestra en el esquema de la Fig. 2.

T4. Asignar el valor '1' a las salidas AN2 y AN3 dentro de la arquitectura y después de la etiqueta de inicio (begin), para deshabilitar los dos displays que no se van a utilizar.

T5. Una vez tenemos la descripción del sistema completo, falta describir cada uno de los componentes que se han creado. En primer lugar se añadirá la descripción de los bloques M1 y M3. Para ello, situar el ratón encima de cualquier componente que aparece en la ventana **"Sources"**, pulsar el botón derecho del ratón y seleccionar la opción **"New Source"**. Esta

acción nos permitirá añadir la descripción en lenguaje VHDL del componente seleccionado a partir de los ficheros M1.vhd y M3.vhd, disponibles en el campus digital.

T6. A continuación, se añadirán los componentes que se han diseñado en el apartado 2.1 del estudio previo, M0, y M2. Para ello, situar el ratón encima de cualquier componente que aparece en la ventana **"Sources"**, pulsar el botón derecho del ratón y seleccionar la opción **"New Source"**. Esta acción nos permitirá añadir la descripción en lenguaje VHDL del componente seleccionado a partir de la descripción realizada en el estudio previo.

T7. Una vez finalizada la descripción de los diferentes componentes, comprobaremos que no exista ningún error de sintaxis. Salvar los ficheros *.vhd, ir seleccionando cada uno de los componentes en la ventana **"Sources"** y ejecutar la opción **"Check Syntax"** que aparece en la ventana **"Processes"**.

Si no se ha producido ningún error, el proceso de "**Check Syntax"** quedará marcado en verde. En el caso de que se hubiese producido algún error, se nos indicará la línea correspondiente, así como el error detectado para su corrección.

T8. Crear un banco de pruebas para simular el funcionamiento del componente M0. Para ello, seleccionar el fichero fuente *.vhd correspondiente a dicho componente en la ventana **Sources** y añadir un nuevo fichero al proyecto del tipo **"Test Bench Waveform"**.

En este caso, como el circuito a simular es secuencial, se generará una señal de reloj con un periodo de 20 ns. Así que se deberá asignar el valor de 20 ns a la constante "**CLK_period"**, tal y como se muestra en la plantilla de creación del banco de pruebas de la Fig. 6.

Respecto a la asignación de los estímulos que se aplicarán en las entradas, se seguirá el cronograma de la Fig. 7. En este caso, se aplicarán cambios en las señales de entrada cada 100 ns. Para ello, completar la plantilla de creación del banco de pruebas con los parámetros que se reflejan en la Fig 6, y añadir los estímulos de las entradas en las líneas marcadas con puntos suspensivos.

Antes de iniciar el proceso de simulación, se deberá de seleccionar un tiempo se simulación de 400 ns.

ISE simulación de un diseño

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;
ENTITY test_M0 IS
END test_M0;

ARCHITECTURE behavior OF test_M0 IS

    -- Component Declaration for the Unit Under Test (UUT)
     COMPONENT M0
    PORT(
         J : IN  std_logic;
         K : IN  std_logic;
         CE : IN  std_logic;
         CLK : IN  std_logic;
         CLR : IN  std_logic;
         Q : OUT  std_logic
        );
    END COMPONENT;

   --Inputs
   signal J : std_logic := '0';
   signal K : std_logic := '0';
   signal CE : std_logic := '0';
   signal CLK : std_logic := '0';
   signal CLR : std_logic := '0';

       --Outputs
   signal Q : std_logic;

   -- Clock period definitions
   constant CLK_period : time := 20 ns;

BEGIN
        -- Instantiate the Unit Under Test (UUT)
   uut: M0 PORT MAP (
          J => J,
          K => K,
          CE => CE,
          CLK => CLK,
          CLR => CLR,
          Q => Q
        );

   -- Clock process definitions
   CLK_process :process
   begin
                CLK <= '0'
                wait for CLK_period/2;
                CLK <= '1';
                wait for CLK_period/2;
   end process;
    -- Stimulus process
   stim_proc: process
   begin
      --Completar los estímulos
       ...
      wait for 100 ns;
      --Completar los estímulos
       ...
      wait for 100 ns;
      --Completar los estímulos
       ...
      wait for 100 ns;
      --Completar los estímulos
       ...

      wait;
   end process;
END;
```

Fig. 6. Especificación de los estímulos de las entradas del componente M0.

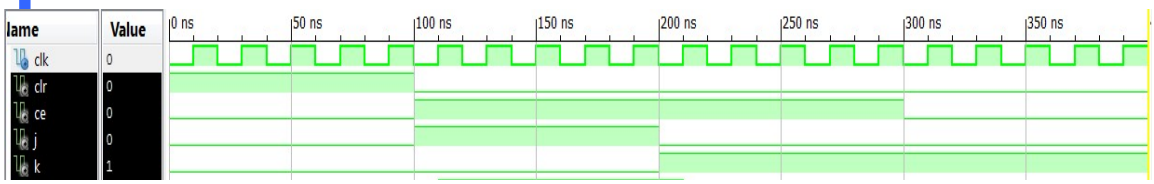

Fig. 7. Edición de los estímulos de las entradas del componente M0.

T9. Simular el funcionamiento del componente M0 y comprobar si el cronograma obtenido coincide con el resultado esperado, en caso contrario, rehacer el diseño del componente y repetir el proceso.

Electrónica Digital

Práctica nº 4 (tercera parte)

Diseño de sistemas digitales

Trabajo a realizar en la práctica

La práctica consiste en la implementación de un sistema secuencial para la gestión del aforo de un parking (Fig. 1). Dicho sistema permite monitorizar el número de vehículos que hay sobre dos displays de 7 segmentos. Para la implementación de la aplicación, además de los displays, se disponen de dos sensores opto-electrónicos, uno en la entrada (SE) y otro en la salida (SS) del parking. Estos sensores nos informan de que ha entrado un nuevo vehículo o ha salido un vehículo respectivamente.

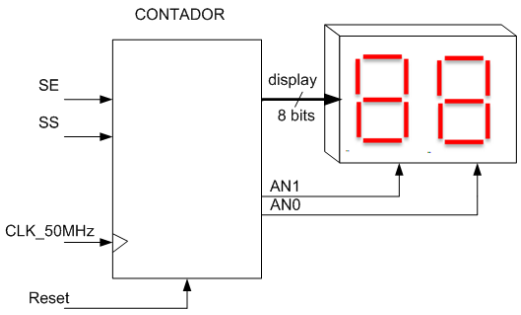

Fig. 1. Sistema a implementar.

Para el diseño del sistema se propone el esquema de bloques descrito en la Fig. 2. Dicho esquema consiste en dos etapas claramente diferenciadas, la etapa que se encarga de contar el número de vehículos presentes en el parking y la etapa de monitorización.

a) La etapa de control de ocupación del parking, está formada por los módulos M0, M1, M2 y M3:

- El módulo M0 gestiona el modo de cuenta (ascendente/descendente) en función de si se ha activado el sensor SE o el SS.

- Los módulos M1 se encargan de detectar los flancos descendentes que se producen en las señales de los sensores SE y SS. Solamente cuando se detecta un flanco descendente, procedente de las señales de uno de los dos sensores (SE, SS), se procederá a actualizar la cuenta de vehículos.
- El módulo M2 consiste en un contador ascendente/descencente que incrementa o decrementa la cuenta en función de si ha entrado o ha salido un vehículo del parking (módulo M0).
- El módulo M3 se encarga de gestionar la multiplexación de los datos a visualizar y genera la señal Enable_1kHz para evaluar el estado de los sensores SE y SS.

b) La etapa de monitorización (módulo M4) es la que nos permite visualizar sobre los dos displays, los datos relacionados con la ocupación del parking (Unidades y Decenas de vehículos). Tal y como se puede observar en la Fig. 2, está formada por un multiplexor 2x4 y un decodificador BCD-7segementos.

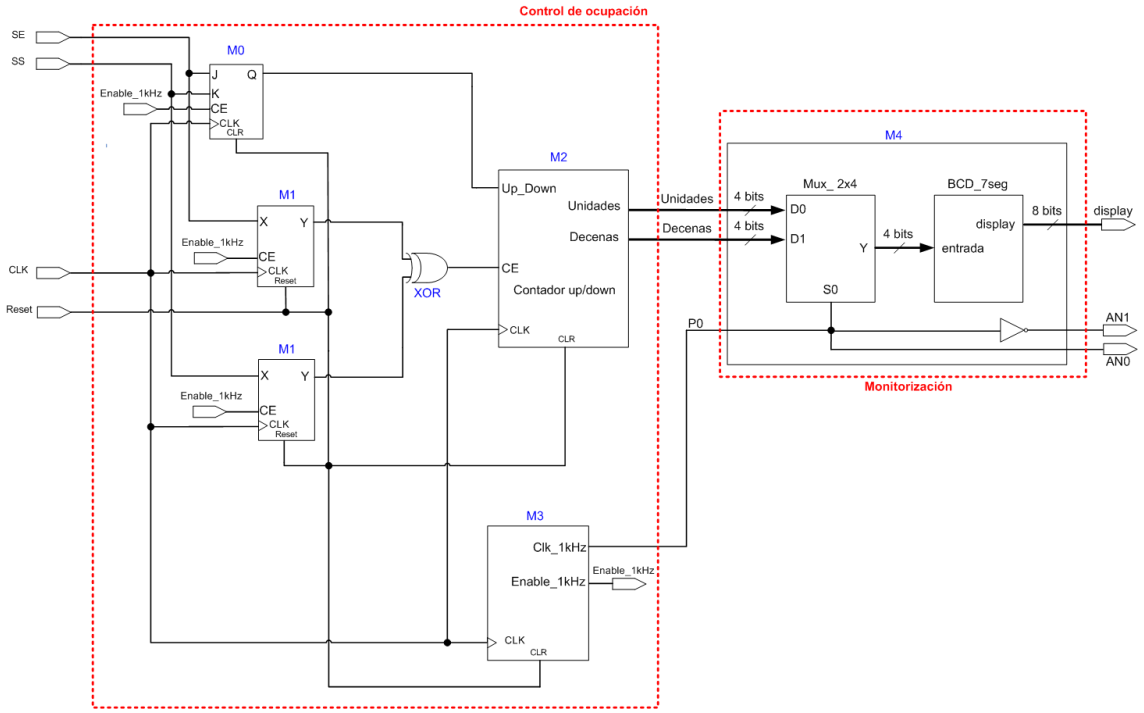

Figura 2. Esquema de bloques propuesto.

Para la implementación del sistema digital, se empleará la herramienta software WebPack ISE y la placa de evaluación de Digilent, que incorpora la FPGA XC3S200 de la serie Spartan 3 del fabricante Xilinx.

El trabajo de la práctica se organiza en dos sesiones. En la primera sesión se pretende abordar el diseño de la etapa de monitorización, mientras que en la última sesión se realizará el diseño de la etapa de control de ocupación del parking, completando la descripción del sistema a diseñar. Cada una de estas sesiones está dividida a su vez en dos apartados: a) Estudio Previo (Diseño del circuito) y b) Implementación del sistema digital sobre la FPGA XC3S200.

Nota: ***Se deberán realizar las tareas de diseño marcadas con la letra "D" seguida del número de tarea, correspondientes al apartado del estudio previo antes de la sesión de prácticas.***

3 Diseño del sistema del sistema de gestión del aforo de un parking

3.1 Estudio Previo correspondiente al diseño del módulo M2

En el siguiente apartado se finaliza el diseño del módulo M2 de la etapa de control de la ocupación del parking, descrita en la Fig. 2.

D1. Descripción en lenguaje VHDL del contador BCD reversible de la Fig. 9, con entrada de enable (CE) y salida de finalización de cuenta (Tc).

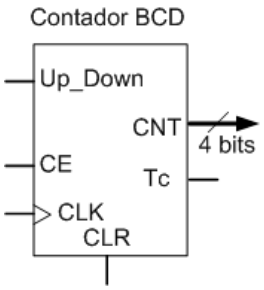

Figura 3. Contador BCD reversible con entrada de clock enable (CE) y salida de finalización de cuenta (Tc).

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity Contador_BCD is
       port(UP_DOWN,CE,CLK,CLR: in std_logic;
            Tc: out std_logic;
            CNT : out std_logic_vector (3 downto 0));
end Contador_BCD;

architecture Behavioral of Contador_BCD is

signal cnt_temp: std_logic_vector (3 downto 0);
begin

process(CLK,CLR)--Contador BCD reversible
begin
....
end process;

process(CLK,CLR)--Generación de la señal Tc
begin
       if(CLR='1')then
              Tc<='0';
       elsif(CLK'event and CLK='1') then
              if(CE='1') Then
                     if(cnt_temp=9 and UP_DOWN='1') or (cnt_temp=0 and
          UP_DOWN='0')then
                                    Tc<='1';
                     else Tc<='0';
                     end if;
              else
                     Tc<='0';
              end if;
        end if;
end process;

end Behavioral;
```

3.2 Implementación del sistema de gestión del aforo de un parking sobre la FPGA XC3S200

En este apartado se describe paso a paso el proceso a seguir para finalizar la descripción, simulación y la implementación del sistema completo, empleando la herramienta software WebPack ISE.

T1. Iniciar la herramienta de trabajo proporcionada por Xilinx, **"ISE Project Navigator"** , y abrir el proyecto que se creó en la sesión anterior (**"File->Open Project"**).

T2. Para finalizar la descripción del sistema completo, falta describir el componente Contador_BCD que se ha diseñado en el apartado 3.1 del estudio previo. Para ello, situar el ratón encima de cualquier componente que aparece en la ventana **"Sources"**, pulsar el botón derecho del ratón y seleccionar la opción **"New Source"**. Esta acción nos permitirá añadir la descripción en lenguaje VHDL del componente seleccionado a partir de la descripción realizada en el estudio previo.

T3. Una vez finalizada la descripción de los diferentes componentes, comprobaremos que no exista ningún error de sintaxis. Salvar los ficheros *.vhd, ir seleccionando cada uno de los componentes en la ventana **"Sources"** y ejecutar la opción **"Check Syntax"** que aparece en la ventana **"Processes"**.

Si no se ha producido ningún error, el proceso de "Check Syntax" quedará marcado en verde. En el caso de que se hubiese producido algún error, se nos indicará la línea correspondiente, así como el error detectado para su corrección.

T4. Crear un banco de pruebas para simular el funcionamiento del componente correspondiente al módulo M2.

En este caso, se introducirán los estímulos reflejados en el cronograma de la Fig.4. Para ello, seguir la plantilla de la Fig.5 e introducir los estímulos de las entradas, completando las líneas marcadas con puntos suspensivos.

Antes de iniciar el proceso de simulación, se deberá de seleccionar un tiempo se simulación de 1020 ns.

ISE simulación de un diseño

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY test_M2 IS
END test_M2;

ARCHITECTURE behavior OF test_M2 IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT M2
    PORT(
         UP_DOWN : IN  std_logic;
         CE : IN  std_logic;
         CLK : IN  std_logic;
         CLR : IN  std_logic;
         Unidades : OUT  std_logic_vector(3 downto 0);
         Decenas : OUT  std_logic_vector(3 downto 0)
        );
    END COMPONENT;
    --Inputs
   signal UP_DOWN : std_logic := '0';
   signal CE : std_logic := '0';
   signal CLK : std_logic := '0';
   signal CLR : std_logic := '0';

       --Outputs
   signal Unidades : std_logic_vector(3 downto 0);
   signal Decenas : std_logic_vector(3 downto 0);

   -- Clock period definitions
   constant CLK_period : time := 20 ns;

BEGIN
       -- Instantiate the Unit Under Test (UUT)
   uut: M2 PORT MAP (
          UP_DOWN => UP_DOWN,
          CE => CE,
          CLK => CLK,
          CLR => CLR,
          Unidades => Unidades,
          Decenas => Decenas
        );
   -- Clock process definitions
   CLK_process :process
   begin
                CLK <= '0';
                wait for CLK_period/2;
                CLK <= '1';
                wait for CLK_period/2;
   end process;
     -- Clock process definitions
   Enable_process :process
   begin
                CE <= '0';
                wait for CLK_period;
                CE <= '1';
                wait for CLK_period;
   end process;
   -- Stimulus process
   stim_proc: process
   begin
       ...
      wait for 20 ns;
        ...
      wait for 500 ns;
        ...
      wait;
   end process;

END;
```

Fig. 4. Especificación de los estímulos de las entradas del componente M2.

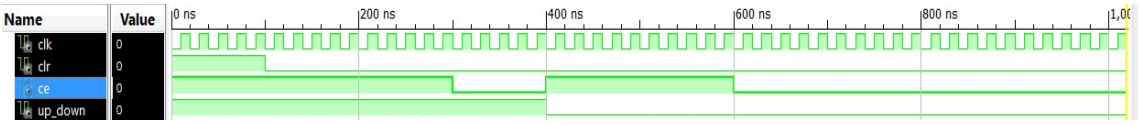

Fig. 5. Edición de los estímulos de las entradas del componente M2.

T5. Simular el funcionamiento del componente M2 y comprobar si el cronograma obtenido coincide con el resultado esperado, en caso contrario, rehacer el diseño y repetir el proceso.

T6. Una vez tenemos la descripción correcta del sistema a implementar, podemos proceder a añadir las restricciones relacionadas con la asignación de pines. Para ello, seleccionar el módulo principal (*.vhd), y en la ventana de **"Processes"** bajo el texto **"User Constrains"** seleccionar la opción **"Assign Packages Pins"**. A la hora de asignar los pines a las entradas y salidas del sistema, seguir la información presente en la Tabla I.

SEÑAL	PIN
SE	L13
SS	M13
Reset	L14
CLK	T9
display(0)	P16
display(1)	N16
display(2)	F13
display(3)	R16
display(4)	P15
display(5)	N15
display(6)	G13
display(7)	E14
AN0	D14
AN1	G14
AN2	F14
AN3	E13

Tabla I. Asignación de pines.

ISE asignación de Pines

T7. Configurar la FPGA para que implemente la aplicación diseñada.

ISE configuración de la FPGA

T8. Comprobar el correcto funcionamiento del circuito sobre la placa de evaluación.