<table>
<tr><td colspan="2" rowspan="2"> 
UNIVERSIDAD POLITÉCNICA DE MADRID
ETSIS de Telecomunicación</td><td colspan="6">APELLIDOS:</td></tr>
<tr><td colspan="4">NOMBRE:</td><td colspan="2">DNI:</td></tr>
<tr><td colspan="4">ASIGNATURA: DISEÑO DIGITAL I</td><td colspan="2">Bloque: II (Escrito)</td></tr>
<tr><td colspan="2">TITULACIÓN:</td><td colspan="2">☐ Electrónica de Comunic.
☐ Sistemas de Telecom.</td><td colspan="2">☐ Sonido e Imagen
☐ Telemática</td></tr>
<tr><td colspan="3">Fecha</td><td>Curso</td><td colspan="5">Calificación de los ejercicios</td><td>Nota Final</td></tr>
<tr><td>09</td><td>01</td><td>2017</td><td>TERCERO</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

ADVERTENCIAS PARA LA REALIZACIÓN DE LA PRIMERA PARTE DEL EXAMEN (Ejercicios 1, 2, 3 y 4)

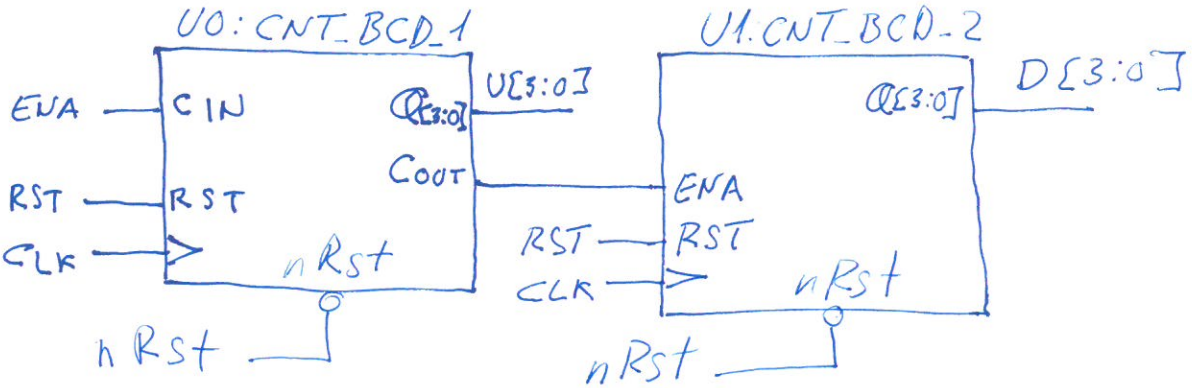
- Rellene **AHORA** los datos personales que deben figurar en esta hoja.
- Mientras dure el examen deberá exponer su D.N.I. encima de la mesa.
- **NO SE ADMITIRÁN** exámenes escritos a lapicero ni con tinta roja o verde.
- **COMPRUEBE** que su ejemplar del examen consta de **4** ejercicios en **16** páginas numeradas.
- En este examen **NO PUEDEN UTILIZARSE CALCULADORAS, LIBROS, APUNTES NI DISPOSITIVOS DE TELECOMUNICACIÓN**. Retírelos ahora de la mesa.
- La duración de esta parte del examen es de **110 minutos**.

Esta hoja se ha dejado en blanco intencionadamente

Ejercicio 1	Subsistemas digitales		
		0.5 puntos	15 minutos

Dibuje el diagrama de bloques de un contador BCD de dos dígitos con entrada de habilitación, activa a nivel alto, entrada de reset síncrono, también activa a nivel alto y entrada de reset asíncrono activa a nivel bajo. Realícelo empleando contadores BCD de un dígito y la lógica adicional que considere necesaria. Debe describir detalladamente el modelo lógico (interfaz y función) de cada uno de los bloques que utilice.

Nota: realice el conexionado utilizando etiquetas



CNT_BCD_1: CONTADOR BCD CON ENTRADA DE HABILITACIÓN (CIN) Y SALIDA DE FIN DE CUENTA DEPENDIENTE DE LA HABILITACION (COUT); ENTRADA DE RESET SÍNCRONO Y ENTRADA DE RESET ASÍNCRONO.

CNT_BCD_2: CONTADOR BCD CON ENTRADA DE HABILITACIÓN (ENA), ENTRADA DE RESET SÍNCRONO Y ENTRADA DE RESET ASÍNCRONO

Ejercicio 2	Subsistemas Digitales		
		0.5 puntos	10 minutos

El siguiente el código VHDL modela el funcionamiento de un subsistema complejo. Determine el tipo de subsistema, indicando sus características específicas. Además, explique detalladamente su funcionamiento. El reloj del circuito tiene una frecuencia de 100 MHz.

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

entity circuito is
port(clk:   in     std_logic;
     nRst:  in     std_logic;
     trig:  in     std_logic;
     pulse: buffer std_logic);
end entity;

architecture rtl of circuito is
  signal cuenta: std_logic_vector(25 downto 0);

begin
  process(clk, nRst)
  begin
    if nRst = '0' then
      cuenta <= (others => '0');

    elsif clk'event and clk = '1' then
      if trig = '1' and pulse = '0' then
        cuenta <= (0 => '1', others => '0');

      elsif pulse = '1' then
        if cuenta /= 50000000 then
          cuenta <= cuenta + 1;

        else
          cuenta <= (others => '0');
        end if;
      end if;
    end if;
  end process;

  pulse <= '1' when cuenta /= 0 else '0';
end rtl;
```

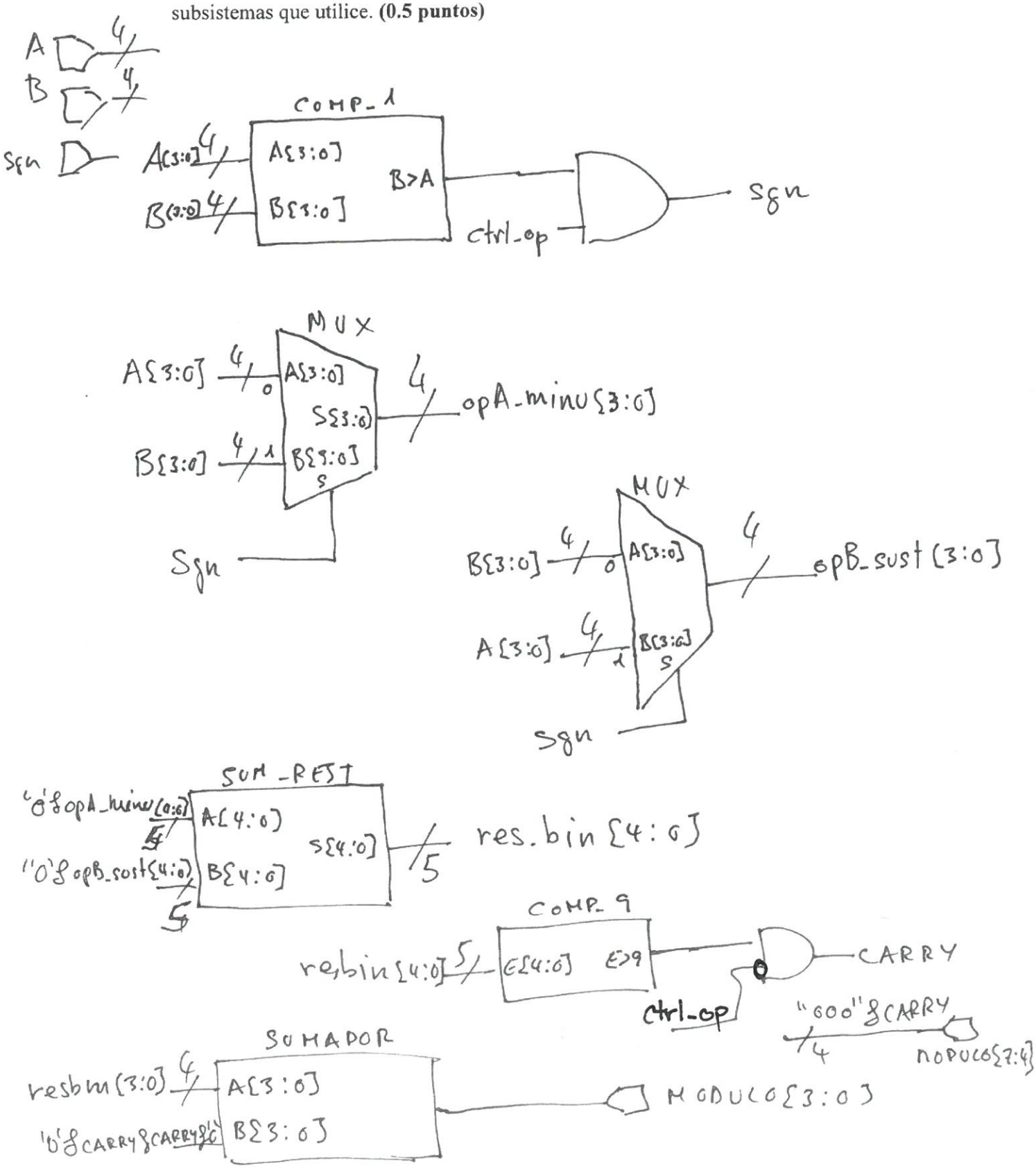
MONOESTABLE NO REDISPARABLE QUE GENERA UN PULSO DE 0.5 S. ENTRADA DE DISPARO, trig, Y SALIDA DE PULSO, pulse.

Ejercicio 3	Análisis de modelos. Diseño		
		2 puntos	40 minutos

El siguiente el código VHDL modela el funcionamiento de un determinado circuito.

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

entity circuito is
port(A:       in     std_logic_vector(3 downto 0);
     B:       in     std_logic_vector(3 downto 0);
     ctrl_op: in     std_logic;
     sgn:     buffer std_logic;
     modulo:  buffer std_logic_vector(7 downto 0));

end entity;

architecture rtl of circuito is
  signal opA_minu: std_logic_vector(3 downto 0);
  signal opB_sust: std_logic_vector(3 downto 0);

  signal res_bin:  std_logic_vector(4 downto 0);
  signal carry:    std_logic;

begin
  sgn <= '1' when ctrl_op = '1' and B > A else
         '0';

  opA_minu <= A when sgn = '0' else
              B;

  opB_sust <= B when sgn = '0' else
              A;

  res_bin <= ('0' & opA_minu) + opB_sust when ctrl_op = '0' else
             ('0' & opA_minu) + (not opB_sust) + 1;

  carry <= '1' when res_bin  > 9 and ctrl_op = '0' else
           '0';

  modulo(7 downto 4) <= "000" & carry;
  modulo(3 downto 0) <= res_bin(3 downto 0) + ('0'&carry&carry & '0');

end rtl;
```

1. Dibuje el diagrama de bloques del circuito modelado. Para realizarlo puede emplear cualquier subsistema digital, simple o complejo, con una funcionalidad conocida. Describa detalladamente el modelo lógico (interfaz y función) de cada uno de los subsistemas que utilice. **(0.5 puntos)**



COMP_1 : COMPARADOR BINARIO DE 4 BITS. ACTIVA LA SALIDA CUANDO B > A

MUX : SELECTOR DE DATOS DE DOS CANALES DE 4 BITS

SUM_REST : SUMADOR - RESTADOR DE 5 BITS

COMP 9 : COMPARADOR CON UNA CONSTANTE. ACTIVA LA SALIDA CUANDO LA ENTRADA, DE 5 BITS, ES MAYOR QUE 9

SUMADOR : SUMADOR DE 4 BITS

2. Explique detalladamente el funcionamiento del circuito, teniendo en cuenta que las entradas **A** y **B** son dígitos BCD. **(0.9 puntos)**

EL CIRCUITO SUMA O RESTA DOS NUMEROS BCD DE UN DÍGITO Y EXPRESA LA SALIDA EN CÓDIGO SIGNO + MAGNITUD, EXPRESANDO LA MAGNITUD (MÓDULO) EN BCD CON DOS DÍGITOS

3. Escriba el fragmento de código que habría que añadir para modelar una salida adicional del circuito, **sal_ca2**, que entregue el resultado en complemento a 2. Indique además el número de bits que debe tener dicha salida. **(0.6 puntos)**

Nº de bits de la salida **sal_ca2**:

Rango de salida entre −9 y +18, 6 bits

Código:

```
sal_ca2 <= ("00" & A) + ("00" & B) when ctrl_op = '0'
           else ("00" & A) + ("11" & (not B)) + 1;
```

Ejercicio 4	Análisis de modelos. Diseño		
		3 puntos	45 minutos

El siguiente el código VHDL modela el funcionamiento de un circuito que controla el nivel de ocupación del aforo de un local.

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

entity circuito_2 is
port(clk:      in std_logic;
     nRst:     in std_logic;
     entrada:  in std_logic;
     salida:   in std_logic;
     aforo:    buffer std_logic_vector(6 downto 0);
     alarma:   buffer std_logic);
end entity;

architecture rtl of circuito_2 is
  signal ent_reg:         std_logic; -- Declaración de señales
  signal sal_reg:         std_logic;
  signal pulso_ent:       std_logic;
  signal pulso_sal:       std_logic;
  signal ena_cnt:         std_logic;
  signal u_dw_cnt:        std_logic;
  signal ena_rst_alarma:  std_logic;
  signal cnt_alarma:      std_logic_vector(16 downto 0);
  signal fdc:             std_logic;
  constant fdc_alarma:    natural := 100000;

begin
  process(nRst, clk) -- Conformadores de pulsos (Fragmento 1)
  begin
    if nRst = '0' then
      ent_reg <= '0';

    elsif clk'event and clk = '1' then
      ent_reg <= entrada;

    end if;
  end process;

  process(nRst, clk)
  begin
    if nRst = '0' then
      sal_reg <= '0';

    elsif clk'event and clk = '1' then
      sal_reg <= salida;

    end if;
  end process;

  pulso_ent <= '1' when entrada = '1' and ent_reg = '0' else
               '0';

  pulso_sal <= '1' when salida = '1'  and sal_reg = '0' else
               '0';
  -- Fin de Conformadores de pulsos (Fin del fragmento 1)
```

```
    -- Control del contador de aforo y contador de aforo (Fragmento 2)
    ena_cnt <= '1' when pulso_ent /= pulso_sal else
               '0';

    u_dw_cnt <= '0' when pulso_ent = '1' else
                '1' when pulso_sal = '1';

    process(nRst, clk)
    begin
      if nRst = '0' then
        aforo <= (others => '0');

      elsif clk'event and clk = '1' then
        if ena_cnt = '1' then
          if u_dw_cnt = '0' then
            aforo <= aforo + 1;

          else
            aforo <= aforo - 1;

          end if;
        end if;
      end if;
    end process;
    -- Fin del Control del contador de aforo y contador de aforo

    -- Control de alarma y generación de alarma (Fragmento 3)
    ena_rst_alarma <= '1' when aforo = 100 else
                      '0';

    process(nRst, clk)
    begin
      if nRst = '0' then
        cnt_alarma <= (0 => '1', others => '0');

      elsif clk'event and clk = '1' then
        if ena_rst_alarma = '1' then
          if fdc = '1' then
            cnt_alarma <= (0 => '1', others => '0');

          else
            cnt_alarma <= cnt_alarma + 1;

          end if;

        else
          cnt_alarma <= (0 => '1', others => '0');

        end if;
      end if;
    end process;

    fdc <= '1' when cnt_alarma = fdc_alarma else
           '0';

    alarma <= '1' when cnt_alarma > fdc_alarma/2 else
              '0';
    -- Fin del control de alarma y generación de alarma

end rtl;
```

El circuito recibe dos señales procedentes de dos sensores que se activan cuando una persona entra (**entrada**) o abandona el local (**salida**). Dichas señales son pulsos, activos a nivel alto y de una duración indeterminada –depende del tiempo que el paso de una persona active el sensor. Estos dos pulsos se conforman (**fragmento 1** del código) para convertirlos en pulsos con un ancho de un periodo del reloj del circuito.

Los pulsos conformados (**pulso_ent** y **pulso_sal**) controlan la cuenta del aforo ocupado generando dos señales de control de cuenta: una señal de habilitación (**ena_cnt_aforo**) y una señal de control del sentido de cuenta (**u_dw_cnt**). La generación de estas dos señales y el modelo del contador del aforo constituyen el **fragmento 2** del código del modelo.

El número de personas dentro del local se indica mediante la salida **aforo**. El aforo máximo es de 100 personas y cuando se alcanza dicho nivel de ocupación se impide la entrada de más clientes, por lo que mientras se mantenga dicha ocupación no podrá llegar ninguna señal procedente del sensor de entrada. Mientras el local se mantenga lleno se generará una señal de alarma (salida **alarma**). El circuito que genera dicha señal se modela en el **fragmento 3**.

1. Identifique los subsistemas que componen el sistema de control de aforo, detallando completamente las características de cada uno de ellos. (**0.3 puntos**)

CONFORMADORES DE PULSOS: CONVIERTEN UN PULSO DE DURACIÓN INDETERMINADA EN UN PULSO CON UNA DURACIÓN DE UN PERIODO DE RELOJ (AUTÓMATA DE MEALY).

CONTADOR REVERSIBLE DE 7 BITS: CONTADOR CON ENTRADA DE HABILITACIÓN, ENTRADA DE CONTROL DEL SENTIDO DE CUENTA Y ENTRADA DE RESET ASÍNCRONO. TIENE SALIDA DE FIN DE CUENTA INDEPENDIENTE DE LA HABILITACIÓN. EL MÓDULO DEL CONTADOR ES 100.

TIMER: TEMPORIZADOR CON ENTRADA DE HABILITACIÓN QUE GENERA UNA SEÑAL CUADRADA PERIÓDICA CON UNA FRECUENCIA IGUAL A LA DEL RELOJ DIVIDIDA POR 100.000 Y UN CICLO DE TRABAJO DEL 50%. CUANDO LA ENTRADA DE HABILITACIÓN ESTÁ DESACTIVADA, RESETEA AL TIMER.

2. Proponga un estilo de codificación alternativo que permita mejorar la calidad del código del **fragmento 1**. **(0.3 puntos)**

   Nota: La propuesta debe incluir una justificación clara y completa de la misma y, además, debe venir acompañada del código que la materializa.

```
process (clk, nRst)
begin
  if nRst = '0' then
    ent_reg <= '0';
    sal_reg <= '0';
  elsif clk'event and clk = '1' then
      ent_reg <= entrada;
      sal_reg <= salida;
  end if
end process;
```

Simplifica el código manteniendo o mejorando su inteligibilidad.

3. En el **fragmento 2** del código del modelo hay una infracción a las reglas de modelado de sistemas digitales simples.
   a. Identifique y explique la infracción cometida (**0.2 puntos**)

Infracción en el modelado combinacional de u_dw_cnt.
No se especifica su valor cuando pulso_cnt y pulso_sal valen Ø. Modela un latch.

   b. Corrija el error (escriba el código de sustitución) (**0.1 puntos**)

```
u_dw_cnt <= '0' when pulso_cnt='1' else
            '1' when pulso_sal='1' else
            'X';
```

   c. Explique el efecto que dicha infracción tendría sobre el funcionamiento del modelo en una simulación (**0.2 puntos**)

Ninguno. No se observará ningún efecto anómalo porque, aunque la señal u_dw_cnt valdrá en ocasiones Ø y en otras 1 cuando pulso_cnt y pulso_sal valgan Ø, en este caso el contador no está habilitado y dicho valor no resulta relevante.

4. Deduzca, analizando el **fragmento 3** del código, las características de la señal de alarma. El reloj del circuito es de 1 Mhz. (**0.4 puntos**)

FRECUENCIA: 10 Hz

CICLO DE TRABAJO: 50%

5. Escriba el fragmento de código que permitiría codificar en una nueva salida, **nivel**, el grado de ocupación del local de acuerdo con la siguiente escala: **vacío, baja ocupación** (entre 1 y 30 personas), **ocupación media** (entre 30 y 70 personas), **ocupación alta** (entre 70 y 99 personas) y **lleno**. Indique el código elegido para representar la información. (**0.4 puntos**)

```
nivel <= "000" when aforo = 0 else
        "001" when aforo>0 and aforo<31 else
        "010" when aforo>30 and aforo<71 else
        "011" when aforo>70 and aforo<100 else
        "100";
```

000 vacío
001 baja ocupación
010 ocupación media
011 ocupación alta
100 lleno

6. Enumere los cambios (no debe escribir código) que habría que realizar en el código para modelar un sistema con un funcionamiento idéntico, pero en el que el aforo máximo pudiera programarse con cualquier valor comprendido entre 50 y 5000 personas. **(0.4 puntos)**

1.- Convertir el contador de aforo en un contador de módulo programable de 13 bits:

- aforo std_logic_vector (12 downto 0)
- Añadir un comparador que tenga como entrada la cuenta de aforo y una nueva entrada (aforo_prog). Su salida será la alarma (fin de cuenta) del circuito.

2.- Añadir una nueva entrada, aforo_prg, para especificar el aforo programado.

7. Explique los cambios (no debe escribir código) que habría que realizar para que la señal de alarma consistiese en un único pulso con una duración de un minuto. (**0.7 puntos**)

Habría que añadir al circuito un monoestable no redisparable que genere pulsos de 1 minuto y cuyo disparo sea la señal de alarma tras ser conformada para generar un pulso con una duración de 1 ciclo de reloj