Condiciones de entrega:

Horario y duración:

a. La hora de inicio es 8.30 hs y la de finalización es 11.30hs
b. Todas las entregas realizadas fuera de término no serán tenidas en cuenta para su corrección

Entrega:

a. Se utilizará el repositorio individual utilizado para la entrega de los trabajos prácticos
b. Todos los archivos .vhd y resoluciones de los ejercicios junto con el tema del parcial deben ser subidos a una carpeta con el nombre primerParcial. El nombre de los archivos vhd deben coincidir con el nombre de la entidad.
c. Se recomienda hacer un commit cada 30 minutos.
d. Al finalizar el parcial debe hacer el commit al repositorio.

Condiciones generales

a. Cualquier diseño que infiera un latch o tenga errores de síntesis será invalido.
b. Cualquier diseño que coloque una señal que no sea clock en un rising_edge o falling_edge será invalido
c. Todos los diseños se realizan con el siguiente dispositivo xc7z010clg400-1
d. Implemente un solo proyecto para todos los ejercicios.
e. Todos los ejercicios deben tener su testbench que demuestre su funcionamiento.
f. Serán mejor considerados los ejercicios que utilicen menos recursos.
g. Todos los resets son sincrónicos activos alto.

Parte práctica:

1. (2.5 Puntos)Realice la descripción de un circuito en VHDL que recibe un dato de 13 bits en el flanco ascendente de la señal sck y realice una compresión destructiva a 8 bits según la siguiente tabla. Se sabe que la frecuencia de la señal sck es de 1MHz

Entrada													Salida							
S	0	0	0	0	0	0	0	0	A	B	C	D	S	0	0	0	A	B	C	D
S	0	0	0	0	0	0	1	A	B	C	D	-	S	0	0	1	A	B	C	D
S	0	0	0	0	0	1	A	B	C	D	-	-	S	0	1	0	A	B	C	D
S	0	0	0	0	1	A	B	C	D	-	-	-	S	0	1	1	A	B	C	D
S	0	0	0	1	A	B	C	D	-	-	-	-	S	1	0	0	A	B	C	D
S	0	0	1	A	B	C	D	-	-	-	-	-	S	1	0	1	A	B	C	D
S	0	1	A	B	C	D	-	-	-	-	-	-	S	1	1	0	A	B	C	D
S	1	A	B	C	D	-	-	-	-	-	-	-	S	1	1	1	A	B	C	D

```
entity companding_A is
       Port ( clk : in std_logic;
              rst : in std_logic;
              sck : in std_logic;
              entrada : in STD_LOGIC_VECTOR (12 downto 0);
              salida : out STD_LOGIC_VECTOR (7 downto 0));
end companding_A;
```

Se pide:

a. Un diagrama en bloques del VHDL implementado
b. La descripción en VHDL.
c. Testbench que demuestre su funcionamiento.
d. Verifique post implementation que el sistema funciona con un clock de 100MHz Indique los slacks resultantes.

2. (2.5 Puntos)Nos han pedido implementar una modificación del algoritmo de la aproximación del cálculo de la inversa de la raíz cuadrada utilizado en el Quake III Arena, nuestra función calcula un valor aproximado al doble de la inversa de la raíz cuadrada. la función que ejecuta ese cálculo es la siguiente

```
float Q_rsqrt(float number)
{
  long i;
  float x2, y;

  i  = * ( long * ) &y;        // evil floating point bit level hacking
  i  = 0x5f3759df - ( i >> 1 );               // what the fuck?
  y  = * ( float * ) &i;
  y = 2 * y;
  return y;
}
```

La entidad del componente a implementar es:

```
entity quakeIII_invroot is
   Port ( number: in std_logic_vector (31 downto 0);
          y: in std_logic_vector (31 downto 0));
end quakeIII_invroot;
```

Tanto la entrada number como la salida y están expresadas en punto flotante. Siendo el bit 31 el signo, los bits 30 al 23 el exponente y los bits del 22 al cero la mantisa

Se pide:

a. Un diagrama en bloques del VHDL implementado
b. La descripción en VHDL.

3. (2.5 puntos)Implemente un sistema de detección de llamadas de auxilio, las cuales son transmitidas en caracteres ascii en el flanco ascendente de la señal sck. Al detectar la secuencia de caracteres SOS (0x53 0x4F 0x53) el sistema colocará su salida en 1 por un ciclo de clock del sistema. El reset del sistema es activo alto y el clock del sistema es activo flanco ascendente y la señal sck tiene una frecuencia de 1MHz

Se pide:

a. Realizar un diagrama de bloques o de estados según la solución planteada.
b. Realizar la descripción en VHDL de la solución planteada.
c. Testbench que demuestre su funcionamiento.
d. Verifique post implementation que el sistema funciona con un clock de 100MHz Indique los slacks resultantes.

```
entity detectorAuxilio is
    Port ( clk : in std_logic;
           rst : in std_logic;
           entrada: in std_logic_vector (7 downto 0);
           sck : in std_logic;
           alarma: out std_logic);
end detectorAuxilio;
```

4. (2.5 puntos)Dado el siguiente código en VHDL dibuje el circuito resultante suponga que stOpA es 0 y stOpB es 1

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

entity sumadorSecuencial is
    Port ( clk : in std_logic;
           rst : in std_logic;
           ena : in std_logic;
           entrada : in std_logic_vector (7 downto 0);
           Salida : out std_logic_vector (7 downto 0)
           );
end sumadorSecuencial;

architecture Behavioral of sumadorSecuencial is

    type state_type is (stOpA, stOpB);
    signal state, next_state : state_type;

    signal r0, r1 : std_logic_vector (7 downto 0);
    signal enaR0, enaR1 : std_logic;

begin

process (clk)
begin
    if (rising_edge (clk)) then
        if (rst = '1') then
            state <= stOpA;
        else
            state <= next_state;
            if (enaR0 = '1') then
                 r0 <= entrada;
            end if;
            if (enaR1 = '1') then
                 r1 <= entrada;
            end if;
        end if;
    end if;
end process;
```

```
process (state, ena)
begin
    enaR0 <= '0';
    enaR1 <= '0';
    next_state <= state;
    case (state) is
        when stOpA =>
            if (ena = '1') then
                enaR0 <= '1';
                next_state <= stOpB;
            end if;

        when stOpB =>
            if (ena = '1') then
                enaR1 <= '1';
                next_state <= stOpA;
            end if;

        when others =>
            next_state <= stOpA;
    end case;

end process;

    salida <= std_logic_vector (signed (r0) + signed (r1));

end Behavioral;
```