PBL 2019-2020

Iratxe Campo Larrinaga

Asier Fernández de Lecea

## Tareas realizadas

Ejercicio de integración:

* Gestor del reloj: reinicio y contadores de centésimas, segundos, minutos y horas. **HECHO**
* Uso de interruptores para gestionar la visualización en los cuatro 7-segmentos: horas-minutos, segundos-centésimas y grados del servo. **HECHO**
* Uso de un interruptor para que el reloj vaya rápido y así ver bien la evolución de las horas. **HECHO**
* Uso de pulsadores normales (o rápidos) para la puesta en hora del reloj.
* Captura de pulsos y filtrado de los pulsadores. **HECHO**
* Control de un motor de continua mediante PWM en función de la hora del reloj. **HECHO**
* Control de un servo en función de lo elegido por el usuario. **HECHO**
* Integración del conversor binario-BCD en el diseño. **HECHO**

Primer prototipo:

* Integración de la alarma: puesta en hora, activación/desactivación de alarma. **HECHO**
* Conexión de la alarma con el motor PWM (o el servo). **HECHO**
* Uso de un pulsador para el control de visualización: según se va pulsando se van viendo la hora-minutos, los segundos-centésimas, la alarma, la posición en grados, etc. Opcional con el anterior. **HECHO**
* Hacer que en cada cambio de hora los puntos de los 7-segmentos parpadeen.

Ampliaciones al primer prototipo:

* Implementar un cronómetro en el reloj. **HECHO**
* Implementar una alarma avanzada: suena, se apaga y se reconecta al de 5 minutos. **HECHO**
* Implementar un juego de acertar tiempo.
* Descripción y uso de una pila FIFO/LIFO en el proyecto: captura de la hora a la que se apagó la alarma. **HECHO**
* Implementación de funcionalidad más compleja para el control del servo o del motor de continua: limpiaparabrisas, introducir un tiempo y que gire ese tiempo el motor DC, que el motor DC gire cada vez a más velocidad. **HECHO**
* O hacer que según pase más tiempo sin apagar la alarma vaya cambiando el comportamiento de los motores: el servo se mueve en función del tiempo que pasa, el motor DC cada vez gira más deprisa. **HECHO**
* Cambio suave de velocidad de PWM. **HECHO**
* Implementación y test de transmisión serie vía UART.
* Añadir una función inventada al diseño.

## Descripción de switches y entradas del proyecto

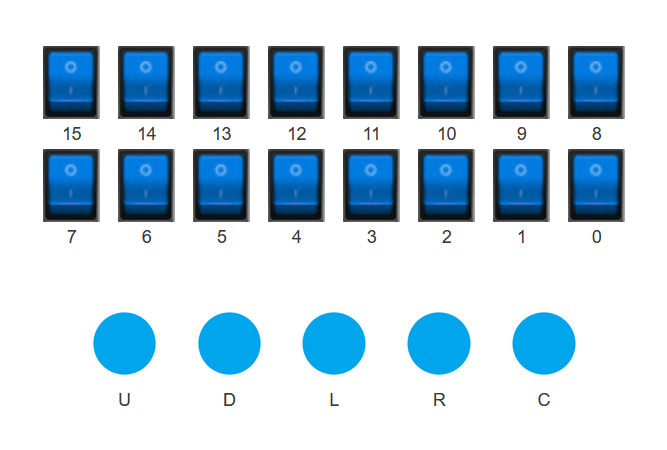
### Entradas globales comunes

**sw (15 downto 14)** **→** switches globales, cambios de modo

**sw(13)** **→** switch global, activación de la alarma (sw(13) = 0 no se ve alarma, sw(13) = 1 se ve alarma), si la alarma está “sonando” el servomotor gira en modo aspersor y los dp’s del siete-segmentos parpadean (para la siguiente entrega haremos que vayan más rápido), motor dc gira más rápido cuanto más tiempo está la alarma sonando

Modos

Activación de la alarma



### Reloj, sw(15 downto 14) a “00”

* **sw(0) →** cambiar display de hh:mm (sw(0) = 0) a ss:ss (sw(0) = 1, segundos y centésimas)
* **sw(1) →** modo fast (sw(1) = 0) o modo normal (sw(1) = 0)
* **sw(2) →** pausa del reloj (sw(2) = 1) o seguir a ritmo normal rápido (sw(2) = 0)
* **sw(3) →** puesta en hora del reloj, se pueden cambiar solo horas y minutos
  + **btnL →** cambiar las horas (reflejado por el cambio de posición del dp)
  + **btnR →** cambiar los minutos (reflejado por el cambio de posición del dp)
  + **sw(6) →** restar el valor seleccionado, si se deja pulsado durante dos segundos resta rápido
  + **sw(7) →** sumar el valor seleccionado, si se deja pulsado durante dos segundos suma rápido

**Nota 1:** alarma no se ve en este modo para no confundir los dp's de la alarma con el de selección

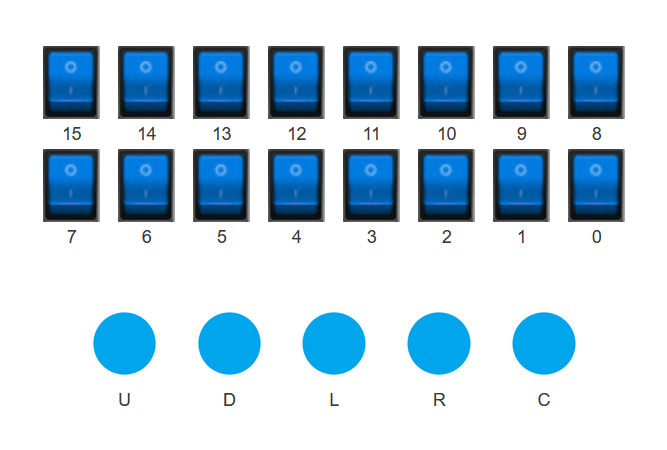
**Nota 2:** la puesta en hora tiene preferencia sobre el cambio de la alarma

* **sw(4) →** cambio de hora de la alarma, se pueden cambiar solo horas y minutos
  + **btnL →** cambiar las horas (reflejado por el cambio de posición del dp)
  + **btnR →** cambiar los minutos (reflejado por el cambio de posición del dp)
  + **sw(6) →** restar el valor seleccionado, si se deja pulsado durante dos segundos resta rápido
  + **sw(7) →** sumar el valor seleccionado, si se deja pulsado durante dos segundos suma rápido

**Nota 1:** alarma no se ve en este modo para no confundir los dp's de la alarma con el de selección

**Nota 2:** la puesta en hora tiene preferencia sobre el cambio de la alarma

* **sw(8) →** cambio de sentido de giro del dcmotor, dcmotor gira en función de la hora del reloj (23h → 100%, 0h → 0%)
* **sw(9) →** visualización del contenido de la pila
  + **sw(10) →** botón de pop para observar las horas a las que se ha pulsado el botón de “cinco minutos más” de la alarma (btnU) cuando estaba sonando la alarma, es necesario hacer una pulsación normal del switch como si fuera un botón (sw(10) a 1 y sw(10) a 0), los valores de la pila se van perdiendo con cada pulsación de sw(10) (obviamente...)
* **sw(13) 🡪** switch global, activación de la alarma (sw(13) = 0 no se ve alarma, sw(13) = 1 se ve alarma), si la alarma está “sonando” el servomotor gira en modo aspersor y los dp’s del siete-segmentos parpadean (para la siguiente entrega haremos que vayan más rápido), motor dc gira más rápido cuanto más tiempo está la alarma sonando
  + **btnU →** botón de “cinco minutos más” de la alarma, solo tiene efecto cuando la alarma está sonando (sw(13) = 1, obviamente), guarda hora en pila LIFO
  + **btnD →** botón de “cinco minutos más” de la alarma, solo tiene efecto cuando la alarma está sonando (sw(13) = 1, obviamente), no guarda hora en pila LIFO
* **btnC →** botón de reinicio del reloj, la alarma y la pila, independiente de los otros reinicios de los distintos modos (cada btnC es el reinicio del modo correspondiente)



Reinicio del reloj, alarma y pila

Cambiar a posición horas

Cambiar a posición minutos

Retraso de la alarma

Retraso de la alarma y almacenado en la pila

Visualizar

horas-minutos /

segundos-centésimas

Pausa reloj

Avanzar rápido reloj

Cambio hora alarma

Cambio hora reloj

Pop pila

Activar alarma

Cambio de sentido dcmotor

Sumar 1 o sumar rápido, depende del tiempo

Restar 1 o restar rápido, depende del tiempo

Visualizar pila

Modos

**Servomotor, sw(15 downto 14) a “01”**

* **sw(7) →** activar (sw(7) = 1) o desactivar (sw(7) = 0) el modo aspersor del servomotor, sobrescribe sw(6) (modos manuales)
  + **sw(11 downto 8) →** delimitan el tiempo que debe transcurrir entre cada movimiento del servomotor en modo aspersor

“XXX1” → un segundo entre movimientos, led(15 downto 12) a "0001"

"XX10" → dos segundos entre movimientos, led(15 downto 12) a "0010"

"X100" → tres segundos entre movimientos, led(15 downto 12) a "0100"

"1000" → cuatro segundos entre movimientos, led(15 downto 12) a "1000"

* **sw(6) →** selección de modos manuales, selectores (sw(6) = 0) o suma/resta (sw(6) = 1)

**sw(6) = 0**

* + **sw(3 downto 0) →** switches para mover el servomotor

**sw(6) = 1**

* + **sw(5) →** sumar, aumentar la posición del servomotor, si se deja pulsado durante dos segundos suma rápido
  + **sw(4) →** restar, disminuir la posición del servomotor, si se deja pulsado durante dos segundos resta rápido
* **btnC →** botón de reinicio del servomotor, independiente de los otros reinicios de los distintos modos (cada btnC es el reinicio del modo correspondiente)

Selección de modo manual

Modo aspersor

Sumar 1 o sumar rápido, depende del tiempo

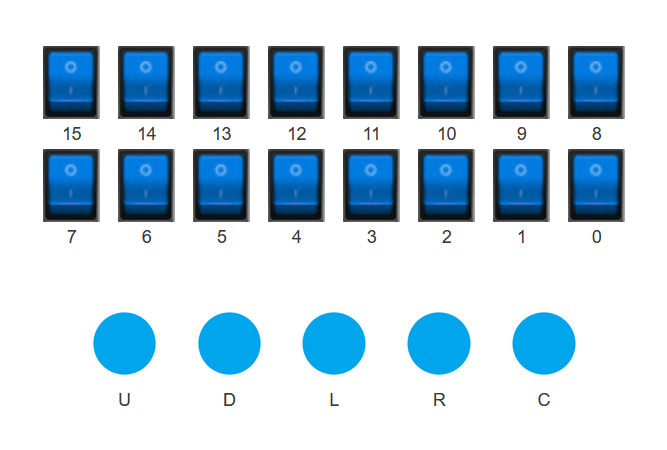
Restar 1 o restar rápido, depende del tiempo

Reinicio del servomotor

Movimiento del servomotor

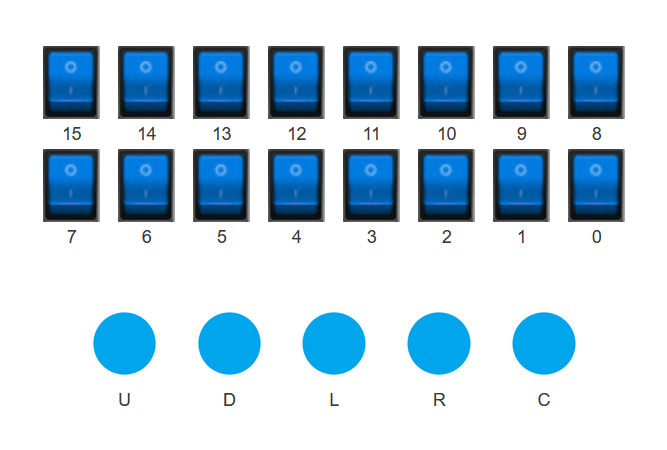
Tiempo entre cada movimiento aspersor

Modos



### Conversor binario-BCD, sw(15 downto 14) a “10”

* **sw(9 downto 0) →** selectores de introducción del dato en binario a convertir
* **sw(10) →** conversión rápida (sw(10) = 1) o lenta paso a paso (sw(10) = 0), se saca su estado en el led(14)
* **sw(11) →** inicio de la conversión, se saca su estado en el led(15) y resultado se ve en el siete-segmentos
* **btnC →** botón de reinicio del conversor, independiente de los otros reinicios de los distintos modos (cada btnC es el reinicio del modo correspondiente)



Enable conversión

Conversión rápida

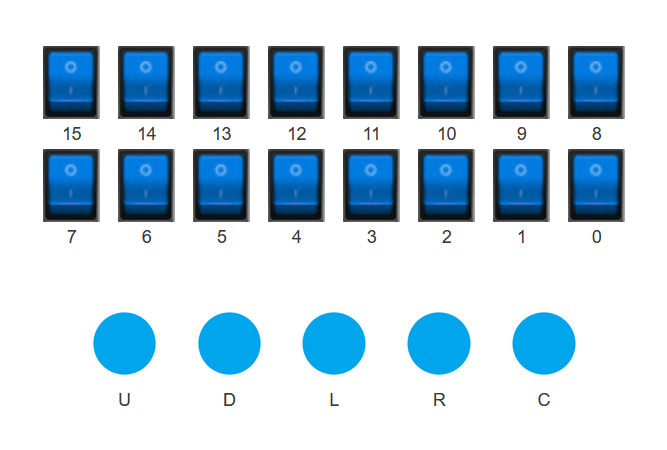
Modo

Selectores de dato binario de entrada

Reinicio del conversor

Selectores de dato binario de entrada

### Cronómetro, sw(15 downto 14) a “11”

* **sw(0) →** cambiar display de ss:ss (sw(0) = 0, segundos y centésimas) a hh:mm (sw(0) = 1)
* **btnU →** pausa del cronómetro
* **btnD →** split del cronómetro, para ver el tiempo en ss:ss y que siga contando el reloj mientras está “parado”
* **btnC →** botón de reinicio del cronómetro, independiente de los otros reinicios de los distintos modos (cada btnC es el reinicio del modo correspondiente)

Reinicio del cronómetro

Split del cronómetro

Pausa del cronómetro

Visualizar

horas-minutos /

segundos-centésimas

Modo

### Implementación VHDL

library IEEE;

use IEEE.std\_logic\_1164.all;

use IEEE.numeric\_std.all;

use IEEE.std\_logic\_unsigned.all;

entity main is

port (

clk : in std\_logic;

sw : in STD\_LOGIC\_VECTOR(15 DOWNTO 0); -- interruptores

btnU : in STD\_LOGIC; -- boton arriba

btnD : in STD\_LOGIC; -- boton abajo

btnL : in STD\_LOGIC; -- boton izquierda

btnR : in STD\_LOGIC; -- boton derecha

btnC : in STD\_LOGIC; -- boton central

led : out STD\_LOGIC\_VECTOR(15 DOWNTO 0); -- leds

seg : out STD\_LOGIC\_VECTOR(6 DOWNTO 0); -- siete segmentos

dp : out STD\_LOGIC; -- punto decimal del siete segmentos

an : out STD\_LOGIC\_VECTOR(3 DOWNTO 0); -- control de 7-seg

servo : out std\_logic;

dcmotor : out std\_logic\_vector (1 downto 0)

);

end main;

architecture Behavioral of main is

-- signals de control de modos y comunes

signal vector\_modo: std\_logic\_vector(1 downto 0);

signal sal\_mux: std\_logic\_vector (3 downto 0);

signal enable\_aux: std\_logic\_vector (3 downto 0);

signal cont\_base\_enable: integer range 0 to 100000;

-- signals del reloj

signal reloj\_inicio: std\_logic;

signal reloj\_cont\_centesimas: std\_logic\_vector (3 downto 0);

signal reloj\_cont\_decimas: std\_logic\_vector (3 downto 0);

signal reloj\_cont\_segs\_unidades: std\_logic\_vector (3 downto 0);

signal reloj\_cont\_segs\_decenas: std\_logic\_vector (3 downto 0);

signal reloj\_cont\_mins\_unidades: std\_logic\_vector (3 downto 0);

signal reloj\_cont\_mins\_decenas: std\_logic\_vector (3 downto 0);

signal reloj\_cont\_horas\_unidades: std\_logic\_vector (3 downto 0);

signal reloj\_cont\_horas\_decenas: std\_logic\_vector (3 downto 0);

signal reloj\_cont\_base: integer range 0 to 1000000;

signal reloj\_tope\_freq: integer range 0 to 1000000;

signal reloj\_select\_display\_hhmm\_ss: std\_logic;

signal reloj\_fast: std\_logic;

signal reloj\_pausa: std\_logic;

-- signals del reloj - pulsador de dedo del dp

signal reloj\_puesta\_en\_hora\_switch: std\_logic;

signal reloj\_puesta\_en\_hora\_dp\_display\_m\_h: std\_logic := '0';

signal reloj\_puesta\_en\_hora\_dp\_estado\_pulsador: std\_logic\_vector (2 downto 0);

signal reloj\_puesta\_en\_hora\_dp\_cont\_filtro: integer range 0 to 100000000;

signal reloj\_puesta\_en\_hora\_dp\_salida: std\_logic;

signal reloj\_puesta\_en\_hora\_dp\_flag\_suma: std\_logic;

signal reloj\_puesta\_en\_hora\_dp\_flag\_resta: std\_logic;

signal reloj\_puesta\_en\_hora\_dp\_btnL: std\_logic;

signal reloj\_puesta\_en\_hora\_dp\_btnR: std\_logic;

-- signals del reloj - switches de suma y resta

signal reloj\_puesta\_en\_hora\_estado\_pulsador: std\_logic\_vector (2 downto 0);

signal reloj\_puesta\_en\_hora\_cont\_filtro: integer range 0 to 500000000;

signal reloj\_puesta\_en\_hora\_flag\_salida: std\_logic;

signal reloj\_puesta\_en\_hora\_flag\_suma: std\_logic;

signal reloj\_puesta\_en\_hora\_flag\_resta: std\_logic;

signal reloj\_puesta\_en\_hora\_sumar: std\_logic;

signal reloj\_puesta\_en\_hora\_restar: std\_logic;

-- signals del reloj-dcmotor

signal reloj\_dcmotor\_cont\_horas\_unidades\_integer: integer range 0 to 9;

signal reloj\_dcmotor\_cont\_horas\_decenas\_integer: integer range 0 to 9;

signal reloj\_dcmotor\_cont\_horas\_integer\_dcmotor: integer range 0 to 100;

signal reloj\_dcmotor\_estado\_pwm: std\_logic\_vector (2 downto 0);

signal reloj\_dcmotor\_duty\_cycle: integer range 0 to 100;

signal reloj\_dcmotor\_sentido\_giro: std\_logic;

signal reloj\_dcmotor\_cont\_flancos: integer range 0 to 100000000;

signal reloj\_dcmotor\_pwm\_longitud\_pulso: integer range 0 to 100000000;

signal reloj\_dcmotor\_pwm\_longitud\_ciclo: integer range 0 to 100000000;

signal reloj\_dcmotor\_pwm\_hz: integer range 0 to 500;

signal reloj\_dcmotor\_pwm\_out: std\_logic;

-- signals del reloj-alarma

signal reloj\_alarma\_switch\_off\_on: std\_logic;

signal reloj\_alarma\_minutos\_unidades: std\_logic\_vector (3 downto 0);

signal reloj\_alarma\_minutos\_decenas : std\_logic\_vector (3 downto 0);

signal reloj\_alarma\_horas\_unidades : std\_logic\_vector (3 downto 0);

signal reloj\_alarma\_horas\_decenas : std\_logic\_vector (3 downto 0);

signal reloj\_alarma\_sonar: std\_logic;

signal reloj\_alarma\_switch\_settime: std\_logic;

signal reloj\_alarma\_led\_estado: std\_logic\_vector(1 downto 0);

signal reloj\_alarma\_led\_contflancos: integer range 0 to 100000000;

signal reloj\_alarma\_led: std\_logic;

signal reloj\_alarma\_snooze\_estado\_pulsador: std\_logic\_vector (2 downto 0);

signal reloj\_alarma\_snooze\_cont\_filtro: integer range 0 to 100005;

signal reloj\_alarma\_snooze\_btnU: std\_logic;

signal reloj\_alarma\_snooze\_btnD: std\_logic;

signal reloj\_alarma\_snooze\_salida: std\_logic;

signal reloj\_alarma\_snooze\_salida\_suma: std\_logic;

signal reloj\_alarma\_snooze\_salida\_resta: std\_logic;

-- signals del servomotor

signal servo\_inicio: std\_logic;

signal servo\_estado\_servo: std\_logic\_vector (1 downto 0);

signal servo\_selector\_aspersor\_mode: std\_logic;

signal servo\_selector\_input\_mode: std\_logic;

signal servo\_aspersor\_cont: std\_logic\_vector(3 downto 0);

signal servo\_selector\_switches: std\_logic\_vector (3 downto 0);

signal servo\_grados: integer range 0 to 180;

signal servo\_cont\_flancos: integer range 0 to 2000000;

signal servo\_pwm\_longitud\_pulso: integer range 0 to 2000000;

-- signals del servomotor-reloj

signal servo\_segundos\_offset: std\_logic\_vector(3 downto 0);

signal servo\_suma\_o\_resta: std\_logic := '0';

signal servo\_cont\_base: integer range 0 to 400000000;

signal servo\_tope\_freq: integer range 0 to 400000000;

-- signals del servomotor-pulsador

signal servo\_estado\_pulsador: std\_logic\_vector (2 downto 0);

signal servo\_cont\_filtro: integer range 0 to 500000000;

signal servo\_salida: std\_logic;

signal servo\_flag\_suma: std\_logic;

signal servo\_flag\_resta: std\_logic;

signal servo\_freq\_min: integer range 0 to 100000000;

signal servo\_contador\_centenas: std\_logic\_vector (3 downto 0);

signal servo\_contador\_decenas: std\_logic\_vector (3 downto 0);

signal servo\_botonMas: std\_logic;

signal servo\_botonMenos: std\_logic;

-- signals del servomotor-pulsador-pwm

signal servo\_contador\_decenas\_integer: integer range 0 to 9;

signal servo\_contador\_centenas\_integer: integer range 0 to 9;

signal servo\_numero\_int: integer range 0 to 200;

-- signals del conversor binario-BCD

signal binario\_bcd\_inicio: std\_logic;

signal binario\_bcd\_binario: std\_logic\_vector (9 downto 0);

signal binario\_bcd\_enable: std\_logic;

signal binario\_bcd\_fin: std\_logic;

signal binario\_bcd\_vector: std\_logic\_vector (25 downto 0);

signal binario\_bcd\_estado\_conversion: std\_logic\_vector (1 downto 0);

signal binario\_bcd\_contador\_desplazamientos: integer range 0 to 9;

signal binario\_bcd\_unidades: std\_logic\_vector (3 downto 0);

signal binario\_bcd\_decenas: std\_logic\_vector (3 downto 0);

signal binario\_bcd\_centenas: std\_logic\_vector (3 downto 0);

signal binario\_bcd\_millar: std\_logic\_vector (3 downto 0);

-- signals del conversor binario-BCD - reloj

signal binario\_bcd\_cont\_base\_enable: integer range 0 to 100000;

signal binario\_bcd\_cont: integer range 0 to 100000000;

signal binario\_bcd\_tope\_freq: integer range 0 to 400000000;

signal binario\_bcd\_modo\_lento\_rapido: std\_logic;

-- signals del cronometro

signal cronom\_inicio: std\_logic;

signal cronom\_cont\_centesimas: std\_logic\_vector (3 downto 0);

signal cronom\_cont\_decimas: std\_logic\_vector (3 downto 0);

signal cronom\_cont\_segs\_unidades: std\_logic\_vector (3 downto 0);

signal cronom\_cont\_segs\_decenas: std\_logic\_vector (3 downto 0);

signal cronom\_cont\_mins\_unidades: std\_logic\_vector (3 downto 0);

signal cronom\_cont\_mins\_decenas: std\_logic\_vector (3 downto 0);

signal cronom\_cont\_horas\_unidades: std\_logic\_vector (3 downto 0);

signal cronom\_cont\_horas\_decenas: std\_logic\_vector (3 downto 0);

signal cronom\_cont\_base: integer range 0 to 1000000;

signal cronom\_tope\_freq: integer range 0 to 1000000;

signal cronom\_pausa: std\_logic;

signal cronom\_select\_display\_hhmm\_ss: std\_logic;

signal cronom\_split: std\_logic;

signal cronom\_cont\_centesimas\_temp: std\_logic\_vector (3 downto 0);

signal cronom\_cont\_decimas\_temp: std\_logic\_vector (3 downto 0);

signal cronom\_cont\_segs\_unidades\_temp: std\_logic\_vector (3 downto 0);

signal cronom\_cont\_segs\_decenas\_temp: std\_logic\_vector (3 downto 0);

-- signals del cronometro - func

signal cronom\_func\_btnU\_stop: std\_logic;

signal cronom\_func\_btnD\_split: std\_logic;

signal cronom\_func\_estado\_pulsador: std\_logic\_vector (2 downto 0);

signal cronom\_func\_cont\_filtro: integer range 0 to 100000000;

signal cronom\_func\_flag\_salida: std\_logic;

signal cronom\_func\_flag\_salida\_flag\_pausa: std\_logic;

signal cronom\_func\_flag\_salida\_flag\_split: std\_logic;

signal cronom\_func\_flag\_pausa: std\_logic := '0';

signal cronom\_func\_flag\_split: std\_logic := '0';

--signals del LIFO

signal reloj\_visualizar\_pila: std\_logic;

signal reloj\_pila\_push: std\_logic;

signal reloj\_pila\_pop: std\_logic;

signal reloj\_pila\_stack\_pointer: integer range -8 to 8 :=7;

type data is array (7 downto 0) of std\_logic\_vector(15 downto 0);

signal reloj\_pila: data := (others => (others => '0'));

signal reloj\_pila\_entrada: std\_logic\_vector (15 downto 0);

signal reloj\_pila\_entrada\_hora\_decenas: std\_logic\_vector (3 downto 0);

signal reloj\_pila\_entrada\_hora\_unidades: std\_logic\_vector (3 downto 0);

signal reloj\_pila\_entrada\_min\_decenas: std\_logic\_vector (3 downto 0);

signal reloj\_pila\_entrada\_min\_unidades: std\_logic\_vector (3 downto 0);

signal reloj\_pila\_salida: std\_logic\_vector (15 downto 0) := "0000000000000000";

signal reloj\_pila\_salida\_hora\_decenas: std\_logic\_vector (3 downto 0);

signal reloj\_pila\_salida\_hora\_unidades: std\_logic\_vector (3 downto 0);

signal reloj\_pila\_salida\_min\_decenas: std\_logic\_vector (3 downto 0);

signal reloj\_pila\_salida\_min\_unidades: std\_logic\_vector (3 downto 0);

type estado is (idle, mete\_push, saca\_pop, llena, vacia, overflow, underflow);

signal reloj\_estado\_pila: estado;

signal reloj\_pila\_llena: std\_logic;

signal reloj\_pila\_vacia: std\_logic;

signal reloj\_pila\_error\_overflow: std\_logic;

signal reloj\_pila\_error\_underflow: std\_logic;

signal reloj\_pila\_estado\_push: std\_logic\_vector (1 downto 0);

signal reloj\_pila\_cont\_filtro\_push: integer range 0 to 100000;

signal reloj\_pila\_salida\_push: std\_logic;

signal reloj\_pila\_estado\_pop: std\_logic\_vector (1 downto 0);

signal reloj\_pila\_cont\_filtro\_pop: integer range 0 to 100000;

signal reloj\_pila\_salida\_pop: std\_logic;

begin

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- LOGICA DE CONTROL

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- ####################################################################

vector\_modo <= sw(15 downto 14);

reloj\_alarma\_switch\_off\_on <= sw(13);

servo\_pwm\_longitud\_pulso <= servo\_grados \* 1111 + 50000;

-- process de asignacion de switches

process(btnC, clk, vector\_modo)

begin

if btnC = '1' then

case vector\_modo is

when "00" => -- reloj

reloj\_inicio <= '1';

when "01" => -- servomotor

servo\_inicio <= '1';

when "10" => -- bin-bcd

binario\_bcd\_inicio <= '1';

when others => -- cronometro

cronom\_inicio <= '1';

end case;

elsif rising\_edge(clk) then

reloj\_inicio <= '0';

servo\_inicio <= '0';

binario\_bcd\_inicio <= '0';

cronom\_inicio <= '0';

case vector\_modo is

when "00" => -- reloj

reloj\_select\_display\_hhmm\_ss <= sw(0);

reloj\_fast <= sw(1);

reloj\_pausa <= sw(2);

reloj\_puesta\_en\_hora\_switch <= sw(3);

reloj\_alarma\_switch\_settime <= sw(4);

reloj\_puesta\_en\_hora\_restar <= sw(6);

reloj\_puesta\_en\_hora\_sumar <= sw(7);

reloj\_dcmotor\_sentido\_giro <= sw(8);

reloj\_puesta\_en\_hora\_dp\_btnL <= btnL;

reloj\_puesta\_en\_hora\_dp\_btnR <= btnR;

reloj\_dcmotor\_pwm\_hz <= 200; -- (200 Hz) -> (500000 flancos) -> (0.5 ms)

reloj\_alarma\_snooze\_btnU <= btnU;

reloj\_alarma\_snooze\_btnD <= btnD;

led(3 downto 0) <= "0000";

reloj\_dcmotor\_cont\_horas\_integer\_dcmotor <= (reloj\_dcmotor\_cont\_horas\_decenas\_integer \* 10) + reloj\_dcmotor\_cont\_horas\_unidades\_integer;

if reloj\_alarma\_sonar = '0' then

reloj\_dcmotor\_cont\_horas\_unidades\_integer <= conv\_integer(reloj\_cont\_horas\_unidades);

reloj\_dcmotor\_cont\_horas\_decenas\_integer <= conv\_integer(reloj\_cont\_horas\_decenas);

reloj\_dcmotor\_duty\_cycle <= reloj\_dcmotor\_cont\_horas\_integer\_dcmotor \* 99 / 23;

else

reloj\_dcmotor\_cont\_horas\_unidades\_integer <= conv\_integer(reloj\_cont\_segs\_unidades);

reloj\_dcmotor\_cont\_horas\_decenas\_integer <= conv\_integer(reloj\_cont\_segs\_decenas);

reloj\_dcmotor\_duty\_cycle <= reloj\_dcmotor\_cont\_horas\_integer\_dcmotor \* 99 / 59;

end if;

--led(2 downto 0) <= std\_logic\_vector(to\_unsigned(reloj\_pila\_stack\_pointer, 3));

reloj\_visualizar\_pila <= sw(9);

reloj\_pila\_pop <= sw(10);

reloj\_pila\_push <= btnU;

when "01" => -- servomotor

servo\_segundos\_offset <= sw(11 downto 8);

servo\_selector\_aspersor\_mode <= sw(7);

servo\_selector\_input\_mode <= sw(6);

servo\_botonMas <= sw(5);

servo\_botonMenos <= sw(4);

servo\_selector\_switches <= sw(3 downto 0);

servo\_freq\_min <= 100000;

servo\_contador\_decenas\_integer <= conv\_integer(servo\_contador\_decenas);

servo\_contador\_centenas\_integer <= conv\_integer(servo\_contador\_centenas);

servo\_numero\_int <= ((servo\_contador\_centenas\_integer \* 10) + servo\_contador\_decenas\_integer) \* 10;

led(3 downto 0) <= "0000";

if servo\_segundos\_offset = "XXX1" then

led(15 downto 12) <= "0001";

elsif servo\_segundos\_offset = "XX10" then

led(15 downto 12) <= "0010";

elsif servo\_segundos\_offset = "X100" then

led(15 downto 12) <= "0100";

elsif servo\_segundos\_offset = "1000" then

led(15 downto 12) <= "1000";

else

led(15 downto 12) <= "0000";

end if;

when "10" => -- bin-bcd

binario\_bcd\_binario <= sw(9 downto 0);

binario\_bcd\_enable <= sw(11);

binario\_bcd\_modo\_lento\_rapido <= sw(10);

led(15) <= binario\_bcd\_fin;

led(14) <= binario\_bcd\_modo\_lento\_rapido;

-- led(11 downto 0) <= binario\_bcd\_vector;

when others => -- cronometro

cronom\_select\_display\_hhmm\_ss <= sw(0);

cronom\_func\_btnU\_stop <= btnU;

cronom\_func\_btnD\_split <= btnD;

cronom\_tope\_freq <= 1000000;

led(3 downto 0) <= "0000";

end case;

end if;

end process;

-- process de la alarma

process(reloj\_alarma\_switch\_off\_on, reloj\_alarma\_minutos\_unidades,

reloj\_cont\_mins\_unidades, reloj\_alarma\_minutos\_decenas, reloj\_cont\_mins\_decenas,

reloj\_alarma\_horas\_unidades, reloj\_cont\_horas\_unidades, reloj\_alarma\_horas\_decenas,

reloj\_cont\_horas\_decenas)

begin

if reloj\_alarma\_switch\_off\_on = '0' then

reloj\_alarma\_sonar <= '0';

else

if reloj\_alarma\_minutos\_unidades = reloj\_cont\_mins\_unidades and

reloj\_alarma\_minutos\_decenas = reloj\_cont\_mins\_decenas and

reloj\_alarma\_horas\_unidades = reloj\_cont\_horas\_unidades and

reloj\_alarma\_horas\_decenas = reloj\_cont\_horas\_decenas

then

reloj\_alarma\_sonar <= '1';

else

reloj\_alarma\_sonar <= '0';

end if;

end if;

end process;

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- LOGICA DEL 7-SEG

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- ####################################################################

an <= enable\_aux;

-- process de conteo y modificacion del cont\_base\_enable (el que da la freq de 1kHz para cambio de seg)

process(btnC, clk)

begin

if btnC = '1' then

cont\_base\_enable <= 0;

elsif rising\_edge(clk) then

if cont\_base\_enable = 100000 then

cont\_base\_enable <= 0;

else

cont\_base\_enable <= cont\_base\_enable + 1;

end if;

end if;

end process;

-- modificacion de la signal para ir de seg en seg (rotacion izquierda 1000 veces por segundo)

process(btnC, clk)

begin

if btnC = '1' then

enable\_aux <= "1110";

elsif rising\_edge(clk) then

if cont\_base\_enable = 100000 then

enable\_aux <= enable\_aux(2 downto 0) & enable\_aux(3); -- desplazamiento a la izquierda

-- enable\_aux <= enable\_aux(0) & enable\_aux(3 downto 1); -- desplazamiento a la derecha

end if;

end if;

end process;

-- multiplexado de las entradas al 7-seg

process(enable\_aux, vector\_modo, reloj\_select\_display\_hhmm\_ss, reloj\_cont\_centesimas,

reloj\_cont\_decimas, reloj\_cont\_segs\_unidades, reloj\_cont\_segs\_decenas,

reloj\_cont\_mins\_unidades, reloj\_cont\_mins\_decenas, reloj\_cont\_horas\_unidades,

reloj\_cont\_horas\_decenas, servo\_grados, binario\_bcd\_unidades,

binario\_bcd\_decenas, binario\_bcd\_centenas, binario\_bcd\_millar, reloj\_puesta\_en\_hora\_switch, reloj\_puesta\_en\_hora\_dp\_display\_m\_h,

reloj\_alarma\_minutos\_unidades, reloj\_alarma\_minutos\_decenas, reloj\_alarma\_horas\_unidades,

reloj\_alarma\_horas\_decenas, reloj\_alarma\_sonar, reloj\_alarma\_switch\_settime, reloj\_alarma\_led,

cronom\_split, cronom\_cont\_centesimas\_temp, cronom\_cont\_decimas\_temp, cronom\_cont\_segs\_unidades\_temp,

cronom\_cont\_segs\_decenas\_temp, cronom\_select\_display\_hhmm\_ss, cronom\_cont\_mins\_unidades,

cronom\_cont\_mins\_decenas, cronom\_cont\_horas\_unidades, cronom\_cont\_horas\_decenas, cronom\_cont\_centesimas,

cronom\_cont\_decimas, cronom\_cont\_segs\_unidades, cronom\_cont\_segs\_decenas, reloj\_visualizar\_pila, reloj\_pila\_salida)

begin

if vector\_modo = "00" then -- reloj

if reloj\_puesta\_en\_hora\_switch = '1' then

case enable\_aux is

when "1110" =>

sal\_mux <= reloj\_cont\_mins\_unidades;

if reloj\_puesta\_en\_hora\_dp\_display\_m\_h = '0' then

dp <= '0';

else

dp <= '1';

end if;

when "1101" =>

sal\_mux <= reloj\_cont\_mins\_decenas;

dp <= '1';

when "1011" =>

sal\_mux <= reloj\_cont\_horas\_unidades;

if reloj\_puesta\_en\_hora\_dp\_display\_m\_h = '1' then

dp <= '0';

else

dp <= '1';

end if;

when "0111" =>

sal\_mux <= reloj\_cont\_horas\_decenas;

dp <= '1';

when others =>

sal\_mux <= "0000";

dp <= '1';

end case;

elsif reloj\_alarma\_switch\_settime = '1' then

case enable\_aux is

when "1110" =>

sal\_mux <= reloj\_alarma\_minutos\_unidades;

if reloj\_puesta\_en\_hora\_dp\_display\_m\_h = '0' then

dp <= '0';

else

dp <= '1';

end if;

when "1101" =>

sal\_mux <= reloj\_alarma\_minutos\_decenas;

dp <= '1';

when "1011" =>

sal\_mux <= reloj\_alarma\_horas\_unidades;

if reloj\_puesta\_en\_hora\_dp\_display\_m\_h = '1' then

dp <= '0';

else

dp <= '1';

end if;

when "0111" =>

sal\_mux <= reloj\_alarma\_horas\_decenas;

dp <= '1';

when others =>

sal\_mux <= "0000";

dp <= '1';

end case;

elsif reloj\_visualizar\_pila = '1' then

case enable\_aux is

when "1110" =>

sal\_mux <= reloj\_pila\_salida(3 downto 0);

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1101" =>

sal\_mux <= reloj\_pila\_salida(7 downto 4);

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1011" =>

sal\_mux <= reloj\_pila\_salida(11 downto 8);

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "0111" =>

sal\_mux <= reloj\_pila\_salida(15 downto 12);

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when others =>

sal\_mux <= "0000";

dp <= '1';

end case;

elsif reloj\_select\_display\_hhmm\_ss = '0' then

case enable\_aux is

when "1110" =>

sal\_mux <= reloj\_cont\_mins\_unidades;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1101" =>

sal\_mux <= reloj\_cont\_mins\_decenas;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1011" =>

sal\_mux <= reloj\_cont\_horas\_unidades;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "0111" =>

sal\_mux <= reloj\_cont\_horas\_decenas;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when others =>

sal\_mux <= "0000";

dp <= '1';

end case;

else

case enable\_aux is

when "1110" =>

sal\_mux <= reloj\_cont\_centesimas;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1101" =>

sal\_mux <= reloj\_cont\_decimas;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1011" =>

sal\_mux <= reloj\_cont\_segs\_unidades;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "0111" =>

sal\_mux <= reloj\_cont\_segs\_decenas;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when others =>

sal\_mux <= "0000";

dp <= '1';

end case;

end if;

elsif vector\_modo = "01" then -- servomotor

if servo\_grados < 100 then

case enable\_aux is

when "0111" =>

sal\_mux <= "1111";

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1011" =>

sal\_mux <= std\_logic\_vector(to\_unsigned(servo\_grados / 100, 4));

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1101" =>

sal\_mux <= std\_logic\_vector(to\_unsigned(servo\_grados / 10, 4));

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1110" =>

sal\_mux <= "0000";

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when others =>

sal\_mux <= "1111";

dp <= '1';

end case;

else

case enable\_aux is

when "0111" =>

sal\_mux <= "1111";

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1011" =>

sal\_mux <= std\_logic\_vector(to\_unsigned(servo\_grados / 100, 4));

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1101" =>

sal\_mux <= std\_logic\_vector(to\_unsigned((servo\_grados / 10) - 10, 4));

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1110" =>

sal\_mux <= "0000";

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when others =>

sal\_mux <= "1111";

dp <= '1';

end case;

end if;

elsif vector\_modo = "10" then -- conversor bin-bcd

case enable\_aux is

when "0111" =>

sal\_mux <= binario\_bcd\_millar;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1011" =>

sal\_mux <= binario\_bcd\_centenas;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1101" =>

sal\_mux <= binario\_bcd\_decenas;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1110" =>

sal\_mux <= binario\_bcd\_unidades;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when others =>

sal\_mux <= "0000";

dp <= '1';

end case;

else -- cronometro

if cronom\_split = '1' then

case enable\_aux is

when "1110" =>

sal\_mux <= cronom\_cont\_centesimas\_temp;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1101" =>

sal\_mux <= cronom\_cont\_decimas\_temp;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1011" =>

sal\_mux <= cronom\_cont\_segs\_unidades\_temp;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "0111" =>

sal\_mux <= cronom\_cont\_segs\_decenas\_temp;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when others =>

sal\_mux <= "0000";

dp <= '1';

end case;

elsif cronom\_select\_display\_hhmm\_ss = '1' then

case enable\_aux is

when "1110" =>

sal\_mux <= cronom\_cont\_mins\_unidades;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1101" =>

sal\_mux <= cronom\_cont\_mins\_decenas;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1011" =>

sal\_mux <= cronom\_cont\_horas\_unidades;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "0111" =>

sal\_mux <= cronom\_cont\_horas\_decenas;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when others =>

sal\_mux <= "0000";

dp <= '1';

end case;

else

case enable\_aux is

when "1110" =>

sal\_mux <= cronom\_cont\_centesimas;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1101" =>

sal\_mux <= cronom\_cont\_decimas;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "1011" =>

sal\_mux <= cronom\_cont\_segs\_unidades;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when "0111" =>

sal\_mux <= cronom\_cont\_segs\_decenas;

if reloj\_alarma\_sonar = '1' then

dp <= reloj\_alarma\_led;

else

dp <= '1';

end if;

when others =>

sal\_mux <= "0000";

dp <= '1';

end case;

end if;

end if;

end process;

-- proceso de display de diferentes valores en diferentes siete\_segs

process(sal\_mux)

begin

case sal\_mux is

when "0000" => seg <= "0000001";

when "0001" => seg <= "1001111";

when "0010" => seg <= "0010010";

when "0011" => seg <= "0000110";

when "0100" => seg <= "1001100";

when "0101" => seg <= "0100100";

when "0110" => seg <= "1100000";

when "0111" => seg <= "0001111";

when "1000" => seg <= "0000000";

when "1001" => seg <= "0001100";

when others => seg <= "1111111";

end case;

end process;

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- LOGICA DEL RELOJ

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- process de reloj

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_cont\_base <= 0;

elsif rising\_edge(clk) then

if reloj\_cont\_base = reloj\_tope\_freq then

reloj\_cont\_base <= 0;

else

reloj\_cont\_base <= reloj\_cont\_base + 1;

end if;

end if;

end process;

-- process de cambio de vel.

process(reloj\_fast)

begin

if reloj\_fast = '0' then

reloj\_tope\_freq <= 1000000;

else

reloj\_tope\_freq <= 500;

end if;

end process;

-- process de monitoreo de segundos.centesimas

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_cont\_centesimas <= "0000";

elsif rising\_edge(clk) then

if reloj\_pausa = '0' and reloj\_puesta\_en\_hora\_switch = '0' then

-- si ha pasado una segundos.centesimas

if reloj\_cont\_base = reloj\_tope\_freq then

if reloj\_cont\_centesimas = "1001" then

reloj\_cont\_centesimas <= "0000";

else

reloj\_cont\_centesimas <= reloj\_cont\_centesimas + 1;

end if;

end if;

elsif reloj\_pausa = '1' then

-- hacer cosas

elsif reloj\_puesta\_en\_hora\_switch = '1' then

-- hacer cosas

end if;

end if;

end process;

-- process de monitoreo de segundos.decimas

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_cont\_decimas <= "0000";

elsif rising\_edge(clk) then

if reloj\_pausa = '0' and reloj\_puesta\_en\_hora\_switch = '0' then

-- si ha pasado una segundos.centesimas y las segundos.centesimas = 9

if reloj\_cont\_base = reloj\_tope\_freq and reloj\_cont\_centesimas = "1001" then

if reloj\_cont\_decimas = "1001" then

reloj\_cont\_decimas <= "0000";

else

reloj\_cont\_decimas <= reloj\_cont\_decimas + 1;

end if;

end if;

elsif reloj\_pausa = '1' then

-- hacer cosas

elsif reloj\_puesta\_en\_hora\_switch = '1' then

-- hacer cosas

end if;

end if;

end process;

-- process de monitoreo de segundos.unidades

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_cont\_segs\_unidades <= "0000";

elsif rising\_edge(clk) then

if reloj\_pausa = '0' and reloj\_puesta\_en\_hora\_switch = '0' then

-- si ha pasado una segundos.centesimas y las segundos.centesimas = 9 y las segundos.decimas = 9

if reloj\_cont\_base = reloj\_tope\_freq and reloj\_cont\_centesimas = "1001" and reloj\_cont\_decimas = "1001" then

if reloj\_cont\_segs\_unidades = "1001" then

reloj\_cont\_segs\_unidades <= "0000";

else

reloj\_cont\_segs\_unidades <= reloj\_cont\_segs\_unidades + 1;

end if;

end if;

elsif reloj\_pausa = '1' then

-- hacer cosas

elsif reloj\_puesta\_en\_hora\_switch = '1' then

-- hacer cosas

end if;

end if;

end process;

-- process de monitoreo de segundos.decenas

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_cont\_segs\_decenas <= "0000";

elsif rising\_edge(clk) then

if reloj\_pausa = '0' and reloj\_puesta\_en\_hora\_switch = '0' then

-- si ha pasado una segundos.centesimas y las segundos.centesimas = 9 y las segundos.decimas = 9 y las segundos.unidades = 9

if reloj\_cont\_base = reloj\_tope\_freq and reloj\_cont\_centesimas = "1001" and reloj\_cont\_decimas = "1001" and reloj\_cont\_segs\_unidades = "1001" then

if reloj\_cont\_segs\_decenas = "0101" then

reloj\_cont\_segs\_decenas <= "0000";

else

reloj\_cont\_segs\_decenas <= reloj\_cont\_segs\_decenas + 1;

end if;

end if;

elsif reloj\_pausa = '1' then

-- hacer cosas

elsif reloj\_puesta\_en\_hora\_switch = '1' then

-- hacer cosas

end if;

end if;

end process;

-- process de monitoreo de minutos.unidades

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_cont\_mins\_unidades <= "0000";

elsif rising\_edge(clk) then

if reloj\_pausa = '0' and reloj\_puesta\_en\_hora\_switch = '0' then

-- si ha pasado una segundos.centesimas y las segundos.centesimas = 9 y las segundos.decimas = 9 y las segundos.unidades = 9 y

-- las segundos.decenas = 5

if reloj\_cont\_base = reloj\_tope\_freq and reloj\_cont\_centesimas = "1001" and reloj\_cont\_decimas = "1001" and reloj\_cont\_segs\_unidades = "1001" and reloj\_cont\_segs\_decenas = "0101" then

if reloj\_cont\_mins\_unidades = "1001" then

reloj\_cont\_mins\_unidades <= "0000";

else

reloj\_cont\_mins\_unidades <= reloj\_cont\_mins\_unidades + 1;

end if;

end if;

elsif reloj\_pausa = '1' then

-- hacer cosas

elsif reloj\_puesta\_en\_hora\_switch = '1' then

if reloj\_puesta\_en\_hora\_dp\_display\_m\_h = '0' then

if reloj\_puesta\_en\_hora\_flag\_salida = '1' then

if reloj\_puesta\_en\_hora\_flag\_suma = '1' then

if reloj\_cont\_mins\_unidades = 9 and reloj\_cont\_mins\_decenas < 5 then

reloj\_cont\_mins\_unidades <= "0000";

elsif reloj\_cont\_mins\_unidades = 9 and reloj\_cont\_mins\_decenas = 5 then

reloj\_cont\_mins\_unidades <= "1001";

else

reloj\_cont\_mins\_unidades <= reloj\_cont\_mins\_unidades + 1;

end if;

elsif reloj\_puesta\_en\_hora\_flag\_resta = '1' then

if reloj\_cont\_mins\_unidades = 0 and reloj\_cont\_mins\_decenas > 0 then

reloj\_cont\_mins\_unidades <= "1001";

elsif reloj\_cont\_mins\_unidades = 0 and reloj\_cont\_mins\_decenas = 0 then

reloj\_cont\_mins\_unidades <= "0000";

else

reloj\_cont\_mins\_unidades <= reloj\_cont\_mins\_unidades - 1;

end if;

end if;

end if;

end if;

end if;

end if;

end process;

-- process de monitoreo de minutos.decenas

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_cont\_mins\_decenas <= "0000";

elsif rising\_edge(clk) then

if reloj\_pausa = '0' and reloj\_puesta\_en\_hora\_switch = '0' then

-- si ha pasado una segundos.centesimas y las segundos.centesimas = 9 y las segundos.decimas = 9 y las segundos.unidades = 9 y

-- las segundos.decenas = 5 y las minutos.unidades = 9

if reloj\_cont\_base = reloj\_tope\_freq and reloj\_cont\_centesimas = "1001" and reloj\_cont\_decimas = "1001" and reloj\_cont\_segs\_unidades = "1001" and reloj\_cont\_segs\_decenas = "0101" and reloj\_cont\_mins\_unidades = "1001" then

if reloj\_cont\_mins\_decenas = "0101" then

reloj\_cont\_mins\_decenas <= "0000";

else

reloj\_cont\_mins\_decenas <= reloj\_cont\_mins\_decenas + 1;

end if;

end if;

elsif reloj\_pausa = '1' then

-- hacer cosas

elsif reloj\_puesta\_en\_hora\_switch = '1' then

if reloj\_puesta\_en\_hora\_dp\_display\_m\_h = '0' then

if reloj\_puesta\_en\_hora\_flag\_salida = '1' then

if reloj\_puesta\_en\_hora\_flag\_suma = '1' and reloj\_cont\_mins\_unidades = 9 then

if reloj\_cont\_mins\_decenas = 5 then

reloj\_cont\_mins\_decenas <= "0101";

else

reloj\_cont\_mins\_decenas <= reloj\_cont\_mins\_decenas + 1;

end if;

elsif reloj\_puesta\_en\_hora\_flag\_resta = '1' and reloj\_cont\_mins\_unidades = 0 then

if reloj\_cont\_mins\_decenas = 0 then

reloj\_cont\_mins\_decenas <= "0000";

else

reloj\_cont\_mins\_decenas <= reloj\_cont\_mins\_decenas - 1;

end if;

end if;

end if;

end if;

end if;

end if;

end process;

-- process de monitoreo de horas.unidades

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_cont\_horas\_unidades <= "0000";

elsif rising\_edge(clk) then

if reloj\_pausa = '0' and reloj\_puesta\_en\_hora\_switch = '0' then

-- si ha pasado una segundos.centesimas y las segundos.centesimas = 9 y las segundos.decimas = 9 y las segundos.unidades = 9 y

-- las segundos.decenas = 5 y las minutos.unidades = 9 y las minutos.decenas = 5

if reloj\_cont\_base = reloj\_tope\_freq and reloj\_cont\_centesimas = "1001" and reloj\_cont\_decimas = "1001" and reloj\_cont\_segs\_unidades = "1001" and reloj\_cont\_segs\_decenas = "0101" and reloj\_cont\_mins\_unidades = "1001" and reloj\_cont\_mins\_decenas = "0101" then

if reloj\_cont\_horas\_unidades = "1001" then

reloj\_cont\_horas\_unidades <= "0000";

elsif reloj\_cont\_horas\_unidades = "0011" and reloj\_cont\_horas\_decenas = "0010" then

reloj\_cont\_horas\_unidades <= "0000";

else

reloj\_cont\_horas\_unidades <= reloj\_cont\_horas\_unidades + 1;

end if;

end if;

elsif reloj\_pausa = '1' then

-- hacer cosas

elsif reloj\_puesta\_en\_hora\_switch = '1' then

if reloj\_puesta\_en\_hora\_dp\_display\_m\_h = '1' then

if reloj\_puesta\_en\_hora\_flag\_salida = '1' then

if reloj\_puesta\_en\_hora\_flag\_suma = '1' then

if reloj\_cont\_horas\_unidades = 9 and reloj\_cont\_horas\_decenas < 2 then

reloj\_cont\_horas\_unidades <= "0000";

elsif reloj\_cont\_horas\_unidades = 3 and reloj\_cont\_horas\_decenas = 2 then

reloj\_cont\_horas\_unidades <= "0011";

else

reloj\_cont\_horas\_unidades <= reloj\_cont\_horas\_unidades + 1;

end if;

elsif reloj\_puesta\_en\_hora\_flag\_resta = '1' then

if reloj\_cont\_horas\_unidades = 0 and reloj\_cont\_horas\_decenas > 0 then

reloj\_cont\_horas\_unidades <= "1001";

elsif reloj\_cont\_horas\_unidades = 0 and reloj\_cont\_horas\_decenas = 0 then

reloj\_cont\_horas\_unidades <= "0000";

else

reloj\_cont\_horas\_unidades <= reloj\_cont\_horas\_unidades - 1;

end if;

end if;

end if;

end if;

end if;

end if;

end process;

-- process de monitoreo de horas.decenas

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_cont\_horas\_decenas <= "0000";

elsif rising\_edge(clk) then

if reloj\_pausa = '0' and reloj\_puesta\_en\_hora\_switch = '0' then

-- si ha pasado una segundos.centesimas y las segundos.centesimas = 9 y las segundos.decimas = 9 y las segundos.unidades = 9 y

-- las segundos.decenas = 5 y las minutos.unidades = 9 y las minutos.decenas = 5 y las horas.unidades = 9 or 3

if reloj\_cont\_base = reloj\_tope\_freq and reloj\_cont\_centesimas = "1001" and reloj\_cont\_decimas = "1001" and reloj\_cont\_segs\_unidades = "1001" and reloj\_cont\_segs\_decenas = "0101" and reloj\_cont\_mins\_unidades = "1001" and reloj\_cont\_mins\_decenas = "0101" and (reloj\_cont\_horas\_unidades = "1001" or reloj\_cont\_horas\_unidades = "0011") then

if reloj\_cont\_horas\_decenas = "0010" then

reloj\_cont\_horas\_decenas <= "0000";

elsif reloj\_cont\_horas\_unidades = "1001" then

reloj\_cont\_horas\_decenas <= reloj\_cont\_horas\_decenas + 1;

end if;

end if;

elsif reloj\_pausa = '1' then

-- hacer cosas

elsif reloj\_puesta\_en\_hora\_switch = '1' then

if reloj\_puesta\_en\_hora\_dp\_display\_m\_h = '1' then

if reloj\_puesta\_en\_hora\_flag\_salida = '1' then

if reloj\_puesta\_en\_hora\_flag\_suma = '1' and reloj\_cont\_horas\_unidades = 9 then

if reloj\_cont\_horas\_decenas = 2 then

reloj\_cont\_horas\_decenas <= "0010";

else

reloj\_cont\_horas\_decenas <= reloj\_cont\_horas\_decenas + 1;

end if;

elsif reloj\_puesta\_en\_hora\_flag\_resta = '1' and reloj\_cont\_horas\_unidades = 0 then

if reloj\_cont\_horas\_decenas = 0 then

reloj\_cont\_horas\_decenas <= "0000";

else

reloj\_cont\_horas\_decenas <= reloj\_cont\_horas\_decenas - 1;

end if;

end if;

end if;

end if;

end if;

end if;

end process;

-- process de monitoreo de alarma.minutos.unidades

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_alarma\_minutos\_unidades <= "0000";

elsif rising\_edge(clk) then

if reloj\_alarma\_switch\_settime = '1' and reloj\_puesta\_en\_hora\_switch = '0' then

if reloj\_puesta\_en\_hora\_dp\_display\_m\_h = '0' then

if reloj\_puesta\_en\_hora\_flag\_salida = '1' then

if reloj\_puesta\_en\_hora\_flag\_suma = '1' then

if reloj\_alarma\_minutos\_unidades = 9 and reloj\_alarma\_minutos\_decenas < 5 then

reloj\_alarma\_minutos\_unidades <= "0000";

elsif reloj\_alarma\_minutos\_unidades = 9 and reloj\_alarma\_minutos\_decenas = 5 then

reloj\_alarma\_minutos\_unidades <= "1001";

else

reloj\_alarma\_minutos\_unidades <= reloj\_alarma\_minutos\_unidades + 1;

end if;

elsif reloj\_puesta\_en\_hora\_flag\_resta = '1' then

if reloj\_alarma\_minutos\_unidades = 0 and reloj\_alarma\_minutos\_decenas > 0 then

reloj\_alarma\_minutos\_unidades <= "1001";

elsif reloj\_alarma\_minutos\_unidades = 0 and reloj\_alarma\_minutos\_decenas = 0 then

reloj\_alarma\_minutos\_unidades <= "0000";

else

reloj\_alarma\_minutos\_unidades <= reloj\_alarma\_minutos\_unidades - 1;

end if;

end if;

end if;

end if;

elsif reloj\_alarma\_sonar = '1' then

if reloj\_alarma\_snooze\_salida = '1' then

if reloj\_alarma\_snooze\_salida\_suma = '1' or reloj\_alarma\_snooze\_salida\_resta = '1' then

if reloj\_alarma\_minutos\_unidades >= 5 then

reloj\_alarma\_minutos\_unidades <= reloj\_alarma\_minutos\_unidades - 5;

elsif reloj\_alarma\_minutos\_unidades < 5 then

reloj\_alarma\_minutos\_unidades <= reloj\_alarma\_minutos\_unidades + 5;

end if;

end if;

end if;

end if;

end if;

end process;

-- process de monitoreo de alarma.minutos.decenas

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_alarma\_minutos\_decenas <= "0000";

elsif rising\_edge(clk) then

if reloj\_alarma\_switch\_settime = '1' and reloj\_puesta\_en\_hora\_switch = '0' then

if reloj\_puesta\_en\_hora\_dp\_display\_m\_h = '0' then

if reloj\_puesta\_en\_hora\_flag\_salida = '1' then

if reloj\_puesta\_en\_hora\_flag\_suma = '1' and reloj\_alarma\_minutos\_unidades = 9 then

if reloj\_alarma\_minutos\_decenas = 5 then

reloj\_alarma\_minutos\_decenas <= "0101";

else

reloj\_alarma\_minutos\_decenas <= reloj\_alarma\_minutos\_decenas + 1;

end if;

elsif reloj\_puesta\_en\_hora\_flag\_resta = '1' and reloj\_alarma\_minutos\_unidades = 0 then

if reloj\_alarma\_minutos\_decenas = 0 then

reloj\_alarma\_minutos\_decenas <= "0000";

else

reloj\_alarma\_minutos\_decenas <= reloj\_alarma\_minutos\_decenas - 1;

end if;

end if;

end if;

end if;

elsif reloj\_alarma\_sonar = '1' then

if reloj\_alarma\_snooze\_salida = '1' then

if reloj\_alarma\_snooze\_salida\_suma = '1' or reloj\_alarma\_snooze\_salida\_resta = '1' then

if reloj\_alarma\_minutos\_unidades >= 5 then

if reloj\_alarma\_minutos\_decenas < 5 then

reloj\_alarma\_minutos\_decenas <= reloj\_alarma\_minutos\_decenas + 1;

else

reloj\_alarma\_minutos\_decenas <= "0000";

end if;

end if;

end if;

end if;

end if;

end if;

end process;

-- process de monitoreo de alarma.horas.unidades

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_alarma\_horas\_unidades <= "0000";

elsif rising\_edge(clk) then

if reloj\_alarma\_switch\_settime = '1' and reloj\_puesta\_en\_hora\_switch = '0' then

if reloj\_puesta\_en\_hora\_dp\_display\_m\_h = '1' then

if reloj\_puesta\_en\_hora\_flag\_salida = '1' then

if reloj\_puesta\_en\_hora\_flag\_suma = '1' then

if reloj\_alarma\_horas\_unidades = 9 and reloj\_alarma\_horas\_decenas < 2 then

reloj\_alarma\_horas\_unidades <= "0000";

elsif reloj\_alarma\_horas\_unidades = 3 and reloj\_alarma\_horas\_decenas = 2 then

reloj\_alarma\_horas\_unidades <= "0011";

else

reloj\_alarma\_horas\_unidades <= reloj\_alarma\_horas\_unidades + 1;

end if;

elsif reloj\_puesta\_en\_hora\_flag\_resta = '1' then

if reloj\_alarma\_horas\_unidades = 0 and reloj\_alarma\_horas\_decenas > 0 then

reloj\_alarma\_horas\_unidades <= "1001";

elsif reloj\_alarma\_horas\_unidades = 0 and reloj\_alarma\_horas\_decenas = 0 then

reloj\_alarma\_horas\_unidades <= "0000";

else

reloj\_alarma\_horas\_unidades <= reloj\_alarma\_horas\_unidades - 1;

end if;

end if;

end if;

end if;

elsif reloj\_alarma\_sonar = '1' then

if reloj\_alarma\_snooze\_salida = '1' then

if reloj\_alarma\_snooze\_salida\_suma = '1' or reloj\_alarma\_snooze\_salida\_resta = '1' then

if reloj\_alarma\_minutos\_unidades >= 5 then

if reloj\_alarma\_minutos\_decenas = 5 then

if reloj\_alarma\_horas\_decenas < 2 then

if reloj\_alarma\_horas\_unidades < 9 then

reloj\_alarma\_horas\_unidades <= reloj\_alarma\_horas\_unidades + 1;

else

reloj\_alarma\_horas\_unidades <= "0000";

end if;

else

if reloj\_alarma\_horas\_unidades < 3 then

reloj\_alarma\_horas\_unidades <= reloj\_alarma\_horas\_unidades + 1;

else

reloj\_alarma\_horas\_unidades <= "0000";

end if;

end if;

end if;

end if;

end if;

end if;

end if;

end if;

end process;

-- process de monitoreo de alarma.horas.decenas

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_alarma\_horas\_decenas <= "0000";

elsif rising\_edge(clk) then

if reloj\_alarma\_switch\_settime = '1' and reloj\_puesta\_en\_hora\_switch = '0' then

if reloj\_puesta\_en\_hora\_dp\_display\_m\_h = '1' then

if reloj\_puesta\_en\_hora\_flag\_salida = '1' then

if reloj\_puesta\_en\_hora\_flag\_suma = '1' and reloj\_alarma\_horas\_unidades = 9 then

if reloj\_alarma\_horas\_decenas = 2 then

reloj\_alarma\_horas\_decenas <= "0010";

else

reloj\_alarma\_horas\_decenas <= reloj\_alarma\_horas\_decenas + 1;

end if;

elsif reloj\_puesta\_en\_hora\_flag\_resta = '1' and reloj\_alarma\_horas\_unidades = 0 then

if reloj\_alarma\_horas\_decenas = 0 then

reloj\_alarma\_horas\_decenas <= "0000";

else

reloj\_alarma\_horas\_decenas <= reloj\_alarma\_horas\_decenas - 1;

end if;

end if;

end if;

end if;

elsif reloj\_alarma\_sonar = '1' then

if reloj\_alarma\_snooze\_salida = '1' then

if reloj\_alarma\_snooze\_salida\_suma = '1' or reloj\_alarma\_snooze\_salida\_resta = '1' then

if reloj\_alarma\_minutos\_unidades >= 5 then

if reloj\_alarma\_minutos\_decenas = 5 then

if reloj\_alarma\_horas\_decenas < 2 then

if reloj\_alarma\_horas\_unidades = 9 then

reloj\_alarma\_horas\_decenas <= reloj\_alarma\_horas\_decenas + 1;

end if;

else

if reloj\_alarma\_horas\_unidades = 3 then

reloj\_alarma\_horas\_decenas <= "0000";

end if;

end if;

end if;

end if;

end if;

end if;

end if;

end if;

end process;

-- ####################################################################

-- ####################################################################

-- LOGICA DEL MOTOR DC

-- ####################################################################

-- ####################################################################

-- process del sentido de giro del motor

process(reloj\_dcmotor\_sentido\_giro, reloj\_dcmotor\_pwm\_out)

begin

if reloj\_dcmotor\_sentido\_giro = '0' then

dcmotor <= "0" & reloj\_dcmotor\_pwm\_out;

else

dcmotor <= reloj\_dcmotor\_pwm\_out & "0";

end if;

end process;

-- process de definicion de reloj\_dcmotor\_pwm\_longitud\_ciclo como (frecuencia base de

-- la fpga) / (reloj\_dcmotor\_pwm\_hz)

process(reloj\_dcmotor\_pwm\_hz)

begin

reloj\_dcmotor\_pwm\_longitud\_ciclo <= 100000000 / reloj\_dcmotor\_pwm\_hz;

end process;

-- process de definicion de reloj\_dcmotor\_pwm\_longitud\_pulso como porcentaje X de

-- reloj\_dcmotor\_pwm\_longitud\_ciclo en funciรณn del valor de reloj\_dcmotor\_duty\_cycle

process(reloj\_dcmotor\_pwm\_longitud\_ciclo, reloj\_dcmotor\_duty\_cycle)

begin

reloj\_dcmotor\_pwm\_longitud\_pulso <= reloj\_dcmotor\_pwm\_longitud\_ciclo \* reloj\_dcmotor\_duty\_cycle / 100;

end process;

-- process del automata del pwm

process(clk, reloj\_inicio)

begin

if reloj\_inicio = '1' then

reloj\_dcmotor\_estado\_pwm <= "000";

reloj\_dcmotor\_cont\_flancos <= 0;

elsif rising\_edge(clk) then

case reloj\_dcmotor\_estado\_pwm is

when "000" =>

reloj\_dcmotor\_cont\_flancos <= 0;

if reloj\_dcmotor\_duty\_cycle /= 0 then

reloj\_dcmotor\_estado\_pwm <= "001";

else

reloj\_dcmotor\_estado\_pwm <= "100";

end if;

when "001" =>

reloj\_dcmotor\_cont\_flancos <= 1;

reloj\_dcmotor\_estado\_pwm <= "010";

when "010" =>

reloj\_dcmotor\_cont\_flancos <= reloj\_dcmotor\_cont\_flancos + 1;

if reloj\_dcmotor\_cont\_flancos < reloj\_dcmotor\_pwm\_longitud\_pulso then

reloj\_dcmotor\_estado\_pwm <= "010";

else

if reloj\_dcmotor\_duty\_cycle /= 100 then

reloj\_dcmotor\_estado\_pwm <= "011";

else

reloj\_dcmotor\_estado\_pwm <= "001";

end if;

end if;

when "011" =>

reloj\_dcmotor\_cont\_flancos <= reloj\_dcmotor\_cont\_flancos + 1;

if reloj\_dcmotor\_cont\_flancos < reloj\_dcmotor\_pwm\_longitud\_ciclo then

reloj\_dcmotor\_estado\_pwm <= "011";

else

if reloj\_dcmotor\_duty\_cycle /= 0 then

reloj\_dcmotor\_estado\_pwm <= "001";

else

reloj\_dcmotor\_estado\_pwm <= "100";

end if;

end if;

when "100" =>

reloj\_dcmotor\_cont\_flancos <= 1;

reloj\_dcmotor\_estado\_pwm <= "011";

when others =>

reloj\_dcmotor\_cont\_flancos <= 0;

reloj\_dcmotor\_estado\_pwm <= "000";

end case;

end if;

end process;

-- process de las salidas del pwm

process(reloj\_dcmotor\_estado\_pwm)

begin

case reloj\_dcmotor\_estado\_pwm is

when "000" => reloj\_dcmotor\_pwm\_out <= '0';

when "001" => reloj\_dcmotor\_pwm\_out <= '1';

when "010" => reloj\_dcmotor\_pwm\_out <= '1';

when "011" => reloj\_dcmotor\_pwm\_out <= '0';

when "100" => reloj\_dcmotor\_pwm\_out <= '0';

when others => reloj\_dcmotor\_pwm\_out <= '0';

end case;

end process;

-- ####################################################################

-- ####################################################################

-- LOGICA DEL PULSADOR DE DEDO DP

-- ####################################################################

-- ####################################################################

-- process del automata del pulsador

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "000";

reloj\_puesta\_en\_hora\_dp\_cont\_filtro <= 0;

elsif rising\_edge(clk) then

case reloj\_puesta\_en\_hora\_dp\_estado\_pulsador is

when "000" => -- INICIO

reloj\_puesta\_en\_hora\_dp\_cont\_filtro <= 0;

if reloj\_puesta\_en\_hora\_dp\_btnL = '1' or reloj\_puesta\_en\_hora\_dp\_btnR = '1' then

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "001";

else

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "000";

end if;

when "001" => -- FILTRADO

reloj\_puesta\_en\_hora\_dp\_cont\_filtro <= reloj\_puesta\_en\_hora\_dp\_cont\_filtro + 1;

if (reloj\_puesta\_en\_hora\_dp\_btnL = '1' or reloj\_puesta\_en\_hora\_dp\_btnR = '1') and reloj\_puesta\_en\_hora\_dp\_cont\_filtro < 100000 then

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "001";

elsif (reloj\_puesta\_en\_hora\_dp\_btnL = '1' or reloj\_puesta\_en\_hora\_dp\_btnR = '1') and reloj\_puesta\_en\_hora\_dp\_cont\_filtro = 100000 then

if reloj\_puesta\_en\_hora\_dp\_btnL = '1'then

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "010";

elsif reloj\_puesta\_en\_hora\_dp\_btnR = '1' then

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "100";

end if;

else

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "000";

end if;

when "010" => -- UNO +

reloj\_puesta\_en\_hora\_dp\_cont\_filtro <= 0;

if reloj\_puesta\_en\_hora\_dp\_btnL = '1' then

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "010";

else

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "011";

end if;

when "011" => -- SUMA

reloj\_puesta\_en\_hora\_dp\_cont\_filtro <= 0;

if reloj\_puesta\_en\_hora\_dp\_btnL = '1' then

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "001";

else

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "000";

end if;

when "100" => -- UNO -

reloj\_puesta\_en\_hora\_dp\_cont\_filtro <= 0;

if reloj\_puesta\_en\_hora\_dp\_btnR = '1' then

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "100";

else

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "101";

end if;

when "101" => -- RESTA

reloj\_puesta\_en\_hora\_dp\_cont\_filtro <= 0;

if reloj\_puesta\_en\_hora\_dp\_btnR = '1' then

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "001";

else

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "000";

end if;

when others =>

reloj\_puesta\_en\_hora\_dp\_cont\_filtro <= 0;

reloj\_puesta\_en\_hora\_dp\_estado\_pulsador <= "000";

end case;

end if;

end process;

-- process de las salidas del pulsador

process(reloj\_puesta\_en\_hora\_dp\_estado\_pulsador)

begin

case reloj\_puesta\_en\_hora\_dp\_estado\_pulsador is

when "000" =>

reloj\_puesta\_en\_hora\_dp\_salida <= '0';

reloj\_puesta\_en\_hora\_dp\_flag\_suma <= '0';

reloj\_puesta\_en\_hora\_dp\_flag\_resta <= '0';

when "001" =>

reloj\_puesta\_en\_hora\_dp\_salida <= '0';

reloj\_puesta\_en\_hora\_dp\_flag\_suma <= '0';

reloj\_puesta\_en\_hora\_dp\_flag\_resta <= '0';

when "010" =>

reloj\_puesta\_en\_hora\_dp\_salida <= '0';

reloj\_puesta\_en\_hora\_dp\_flag\_suma <= '0';

reloj\_puesta\_en\_hora\_dp\_flag\_resta <= '0';

when "011" =>

reloj\_puesta\_en\_hora\_dp\_salida <= '1';

reloj\_puesta\_en\_hora\_dp\_flag\_suma <= '1';

reloj\_puesta\_en\_hora\_dp\_flag\_resta <= '0';

when "100" =>

reloj\_puesta\_en\_hora\_dp\_salida <= '0';

reloj\_puesta\_en\_hora\_dp\_flag\_suma <= '0';

reloj\_puesta\_en\_hora\_dp\_flag\_resta <= '0';

when "101" =>

reloj\_puesta\_en\_hora\_dp\_salida <= '1';

reloj\_puesta\_en\_hora\_dp\_flag\_suma <= '0';

reloj\_puesta\_en\_hora\_dp\_flag\_resta <= '1';

when others =>

reloj\_puesta\_en\_hora\_dp\_salida <= '0';

reloj\_puesta\_en\_hora\_dp\_flag\_suma <= '0';

reloj\_puesta\_en\_hora\_dp\_flag\_resta <= '0';

end case;

end process;

-- process de sumar/restar el reloj\_puesta\_en\_hora\_dp\_display\_m\_h

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_puesta\_en\_hora\_dp\_display\_m\_h <= '0';

elsif rising\_edge(clk) then

if reloj\_puesta\_en\_hora\_dp\_salida = '1' then

if reloj\_puesta\_en\_hora\_dp\_flag\_suma = '1' then

-- sumar si esta a 0

if reloj\_puesta\_en\_hora\_dp\_display\_m\_h = '0' then

reloj\_puesta\_en\_hora\_dp\_display\_m\_h <= '1';

end if;

elsif reloj\_puesta\_en\_hora\_dp\_flag\_resta = '1' then

-- restar si esta a 1

if reloj\_puesta\_en\_hora\_dp\_display\_m\_h = '1' then

reloj\_puesta\_en\_hora\_dp\_display\_m\_h <= '0';

end if;

end if;

end if;

end if;

end process;

-- ####################################################################

-- ####################################################################

-- LOGICA DEL RELOJ - PULSADOR - SUMAR - RESTAR

-- ####################################################################

-- ####################################################################

-- process del automata pulsador dedo

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "000";

reloj\_puesta\_en\_hora\_cont\_filtro <= 0;

elsif rising\_edge(clk) then

case reloj\_puesta\_en\_hora\_estado\_pulsador is

when "000" => -- INICIO

reloj\_puesta\_en\_hora\_cont\_filtro <= 0;

if reloj\_puesta\_en\_hora\_sumar = '1' or reloj\_puesta\_en\_hora\_restar = '1' then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "001";

else

reloj\_puesta\_en\_hora\_estado\_pulsador <= "000";

end if;

when "001" => -- FILTRADO

reloj\_puesta\_en\_hora\_cont\_filtro <= reloj\_puesta\_en\_hora\_cont\_filtro + 1;

if (reloj\_puesta\_en\_hora\_sumar = '1' or reloj\_puesta\_en\_hora\_restar = '1') and reloj\_puesta\_en\_hora\_cont\_filtro < 100000 then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "001";

elsif (reloj\_puesta\_en\_hora\_sumar = '1' or reloj\_puesta\_en\_hora\_restar = '1') and reloj\_puesta\_en\_hora\_cont\_filtro = 100000 then

if reloj\_puesta\_en\_hora\_sumar = '1'then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "010";

elsif reloj\_puesta\_en\_hora\_restar = '1' then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "100";

end if;

else

reloj\_puesta\_en\_hora\_estado\_pulsador <= "000";

end if;

when "010" => -- UNO +

reloj\_puesta\_en\_hora\_cont\_filtro <= reloj\_puesta\_en\_hora\_cont\_filtro + 1;

if reloj\_puesta\_en\_hora\_sumar = '1' and reloj\_puesta\_en\_hora\_cont\_filtro < 200000000 then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "010";

elsif reloj\_puesta\_en\_hora\_sumar = '1' and reloj\_puesta\_en\_hora\_cont\_filtro = 200000000 then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "110";

elsif reloj\_puesta\_en\_hora\_sumar = '0' then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "011";

end if;

when "011" => -- SUMA

reloj\_puesta\_en\_hora\_cont\_filtro <= 0;

if reloj\_puesta\_en\_hora\_sumar = '1' then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "001";

else

reloj\_puesta\_en\_hora\_estado\_pulsador <= "000";

end if;

when "100" => -- UNO -

reloj\_puesta\_en\_hora\_cont\_filtro <= reloj\_puesta\_en\_hora\_cont\_filtro + 1;

if reloj\_puesta\_en\_hora\_restar = '1' and reloj\_puesta\_en\_hora\_cont\_filtro < 200000000 then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "100";

elsif reloj\_puesta\_en\_hora\_restar = '1' and reloj\_puesta\_en\_hora\_cont\_filtro = 200000000 then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "110";

elsif reloj\_puesta\_en\_hora\_restar = '0' then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "101";

end if;

when "101" => -- RESTA

reloj\_puesta\_en\_hora\_cont\_filtro <= 0;

if reloj\_puesta\_en\_hora\_restar = '1' then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "001";

else

reloj\_puesta\_en\_hora\_estado\_pulsador <= "000";

end if;

when "110" => -- START\_RAPIDO

reloj\_puesta\_en\_hora\_cont\_filtro <= 0;

if reloj\_puesta\_en\_hora\_sumar = '1' or reloj\_puesta\_en\_hora\_restar = '1' then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "111";

elsif reloj\_puesta\_en\_hora\_sumar = '0' and reloj\_puesta\_en\_hora\_restar = '0' then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "000";

end if;

when others => -- RAPIDO

reloj\_puesta\_en\_hora\_cont\_filtro <= reloj\_puesta\_en\_hora\_cont\_filtro + 1;

if (reloj\_puesta\_en\_hora\_sumar = '1' or reloj\_puesta\_en\_hora\_restar = '1') and reloj\_puesta\_en\_hora\_cont\_filtro < 20000000 then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "111";

elsif (reloj\_puesta\_en\_hora\_sumar = '1' or reloj\_puesta\_en\_hora\_restar = '1') and reloj\_puesta\_en\_hora\_cont\_filtro = 20000000 then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "110";

elsif reloj\_puesta\_en\_hora\_sumar = '0' and reloj\_puesta\_en\_hora\_restar = '0' then

reloj\_puesta\_en\_hora\_estado\_pulsador <= "000";

end if;

end case;

end if;

end process;

-- process de las salidas pulsador dedo

process(reloj\_puesta\_en\_hora\_estado\_pulsador, reloj\_puesta\_en\_hora\_sumar, reloj\_puesta\_en\_hora\_restar, reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_puesta\_en\_hora\_flag\_salida <= '0';

reloj\_puesta\_en\_hora\_flag\_suma <= '0';

reloj\_puesta\_en\_hora\_flag\_resta <= '0';

elsif rising\_edge(clk) then

case reloj\_puesta\_en\_hora\_estado\_pulsador is

when "000" =>

reloj\_puesta\_en\_hora\_flag\_salida <= '0';

reloj\_puesta\_en\_hora\_flag\_suma <= '0';

reloj\_puesta\_en\_hora\_flag\_resta <= '0';

when "001" =>

reloj\_puesta\_en\_hora\_flag\_salida <= '0';

reloj\_puesta\_en\_hora\_flag\_suma <= '0';

reloj\_puesta\_en\_hora\_flag\_resta <= '0';

when "010" =>

reloj\_puesta\_en\_hora\_flag\_salida <= '0';

reloj\_puesta\_en\_hora\_flag\_suma <= '0';

reloj\_puesta\_en\_hora\_flag\_resta <= '0';

when "011" =>

reloj\_puesta\_en\_hora\_flag\_salida <= '1';

reloj\_puesta\_en\_hora\_flag\_suma <= '1';

reloj\_puesta\_en\_hora\_flag\_resta <= '0';

when "100" =>

reloj\_puesta\_en\_hora\_flag\_salida <= '0';

reloj\_puesta\_en\_hora\_flag\_suma <= '0';

reloj\_puesta\_en\_hora\_flag\_resta <= '0';

when "101" =>

reloj\_puesta\_en\_hora\_flag\_salida <= '1';

reloj\_puesta\_en\_hora\_flag\_suma <= '0';

reloj\_puesta\_en\_hora\_flag\_resta <= '1';

when "110" =>

reloj\_puesta\_en\_hora\_flag\_salida <= '1';

if reloj\_puesta\_en\_hora\_sumar = '1' then

reloj\_puesta\_en\_hora\_flag\_suma <= '1';

reloj\_puesta\_en\_hora\_flag\_resta <= '0';

elsif reloj\_puesta\_en\_hora\_restar = '1' then

reloj\_puesta\_en\_hora\_flag\_suma <= '0';

reloj\_puesta\_en\_hora\_flag\_resta <= '1';

end if;

when others =>

reloj\_puesta\_en\_hora\_flag\_salida <= '0';

reloj\_puesta\_en\_hora\_flag\_suma <= '0';

reloj\_puesta\_en\_hora\_flag\_resta <= '0';

end case;

end if;

end process;

-- ####################################################################

-- ####################################################################

-- LED DE LA ALARMA

-- ####################################################################

-- ####################################################################

-- process del automata de la alarma del reloj

process(clk, reloj\_inicio, reloj\_alarma\_sonar)

begin

if reloj\_inicio = '1' then

reloj\_alarma\_led\_estado <= "00";

reloj\_alarma\_led\_contflancos <= 0;

elsif rising\_edge(clk) then

if reloj\_alarma\_sonar = '1' then

case reloj\_alarma\_led\_estado is

when "00" =>

reloj\_alarma\_led\_contflancos <= 0;

reloj\_alarma\_led\_estado <= "01";

when "01" =>

reloj\_alarma\_led\_contflancos <= 1;

reloj\_alarma\_led\_estado <= "10";

when "10" =>

reloj\_alarma\_led\_contflancos <= reloj\_alarma\_led\_contflancos + 1;

if reloj\_alarma\_led\_contflancos = 50000000 then

reloj\_alarma\_led\_estado <= "11";

else

reloj\_alarma\_led\_estado <= "10";

end if;

when others =>

reloj\_alarma\_led\_contflancos <= reloj\_alarma\_led\_contflancos + 1;

if reloj\_alarma\_led\_contflancos = 100000000 then

reloj\_alarma\_led\_estado <= "01";

else

reloj\_alarma\_led\_estado <= "11";

end if;

end case;

end if;

end if;

end process;

-- process de salidas del led de la alarma del reloj

process(reloj\_alarma\_led\_estado, reloj\_alarma\_sonar)

begin

if reloj\_alarma\_sonar = '1' then

case reloj\_alarma\_led\_estado is

when "00" => reloj\_alarma\_led <= '0';

when "01" => reloj\_alarma\_led <= '1';

when "10" => reloj\_alarma\_led <= '1';

when others => reloj\_alarma\_led <= '0';

end case;

else

reloj\_alarma\_led <= '0';

end if;

end process;

-- ####################################################################

-- ####################################################################

-- LOGICA DE SNOOZE

-- ####################################################################

-- ####################################################################

-- process del automata del pulsador de snooze

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_alarma\_snooze\_estado\_pulsador <= "000";

reloj\_alarma\_snooze\_cont\_filtro <= 0;

elsif rising\_edge(clk) then

case reloj\_alarma\_snooze\_estado\_pulsador is

when "000" => -- INICIO

reloj\_alarma\_snooze\_cont\_filtro <= 0;

if reloj\_alarma\_snooze\_btnU = '1' or reloj\_alarma\_snooze\_btnD = '1' then

reloj\_alarma\_snooze\_estado\_pulsador <= "001";

else

reloj\_alarma\_snooze\_estado\_pulsador <= "000";

end if;

when "001" => -- FILTRADO

reloj\_alarma\_snooze\_cont\_filtro <= reloj\_alarma\_snooze\_cont\_filtro + 1;

if (reloj\_alarma\_snooze\_btnU = '1' or reloj\_alarma\_snooze\_btnD = '1') and reloj\_alarma\_snooze\_cont\_filtro < 100000 then

reloj\_alarma\_snooze\_estado\_pulsador <= "001";

elsif (reloj\_alarma\_snooze\_btnU = '1' or reloj\_alarma\_snooze\_btnD = '1') and reloj\_alarma\_snooze\_cont\_filtro = 100000 then

if reloj\_alarma\_snooze\_btnU = '1'then

reloj\_alarma\_snooze\_estado\_pulsador <= "010";

elsif reloj\_alarma\_snooze\_btnD = '1' then

reloj\_alarma\_snooze\_estado\_pulsador <= "100";

end if;

else

reloj\_alarma\_snooze\_estado\_pulsador <= "000";

end if;

when "010" => -- UNO +

reloj\_alarma\_snooze\_cont\_filtro <= 0;

if reloj\_alarma\_snooze\_btnU = '1' then

reloj\_alarma\_snooze\_estado\_pulsador <= "010";

else

reloj\_alarma\_snooze\_estado\_pulsador <= "011";

end if;

when "011" => -- SUMA

reloj\_alarma\_snooze\_cont\_filtro <= 0;

if reloj\_alarma\_snooze\_btnU = '1' then

reloj\_alarma\_snooze\_estado\_pulsador <= "001";

else

reloj\_alarma\_snooze\_estado\_pulsador <= "000";

end if;

when "100" => -- UNO -

reloj\_alarma\_snooze\_cont\_filtro <= 0;

if reloj\_alarma\_snooze\_btnD = '1' then

reloj\_alarma\_snooze\_estado\_pulsador <= "100";

else

reloj\_alarma\_snooze\_estado\_pulsador <= "101";

end if;

when "101" => -- RESTA

reloj\_alarma\_snooze\_cont\_filtro <= 0;

if reloj\_alarma\_snooze\_btnD = '1' then

reloj\_alarma\_snooze\_estado\_pulsador <= "001";

else

reloj\_alarma\_snooze\_estado\_pulsador <= "000";

end if;

when others =>

reloj\_alarma\_snooze\_cont\_filtro <= 0;

reloj\_alarma\_snooze\_estado\_pulsador <= "000";

end case;

end if;

end process;

-- process de las salidas del pulsador de snooze

process(reloj\_alarma\_snooze\_estado\_pulsador, clk, reloj\_inicio)

begin

if reloj\_inicio = '1' then

reloj\_alarma\_snooze\_salida <= '0';

reloj\_alarma\_snooze\_salida\_suma <= '0';

reloj\_alarma\_snooze\_salida\_resta <= '0';

elsif rising\_edge(clk) then

case reloj\_alarma\_snooze\_estado\_pulsador is

when "000" =>

reloj\_alarma\_snooze\_salida <= '0';

reloj\_alarma\_snooze\_salida\_suma <= '0';

reloj\_alarma\_snooze\_salida\_resta <= '0';

when "001" =>

reloj\_alarma\_snooze\_salida <= '0';

reloj\_alarma\_snooze\_salida\_suma <= '0';

reloj\_alarma\_snooze\_salida\_resta <= '0';

when "010" =>

reloj\_alarma\_snooze\_salida <= '0';

reloj\_alarma\_snooze\_salida\_suma <= '0';

reloj\_alarma\_snooze\_salida\_resta <= '0';

when "011" =>

reloj\_alarma\_snooze\_salida <= '1';

reloj\_alarma\_snooze\_salida\_suma <= '1';

reloj\_alarma\_snooze\_salida\_resta <= '0';

when "100" =>

reloj\_alarma\_snooze\_salida <= '0';

reloj\_alarma\_snooze\_salida\_suma <= '0';

reloj\_alarma\_snooze\_salida\_resta <= '0';

when "101" =>

reloj\_alarma\_snooze\_salida <= '1';

reloj\_alarma\_snooze\_salida\_suma <= '0';

reloj\_alarma\_snooze\_salida\_resta <= '1';

when others =>

reloj\_alarma\_snooze\_salida <= '0';

reloj\_alarma\_snooze\_salida\_suma <= '0';

reloj\_alarma\_snooze\_salida\_resta <= '0';

end case;

end if;

end process;

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- LOGICA DEL SERVOMOTOR

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- process de designacion de servo\_grados por tiempo/switches/dedo

process(servo\_selector\_aspersor\_mode, servo\_aspersor\_cont,

servo\_selector\_input\_mode, servo\_selector\_switches, servo\_suma\_o\_resta,

servo\_numero\_int, reloj\_alarma\_sonar)

begin

-- servo\_grados por aspersor

if servo\_selector\_aspersor\_mode = '1' or reloj\_alarma\_sonar = '1' then

case servo\_aspersor\_cont is

when "0000" => servo\_grados <= 10;

when "0001" => servo\_grados <= 20;

when "0010" => servo\_grados <= 30;

when "0011" => servo\_grados <= 40;

when "0100" => servo\_grados <= 50;

when "0101" => servo\_grados <= 60;

when "0110" => servo\_grados <= 70;

when "0111" => servo\_grados <= 80;

when "1000" => servo\_grados <= 90;

when "1001" => servo\_grados <= 100;

when "1010" => servo\_grados <= 110;

when "1011" => servo\_grados <= 120;

when "1100" => servo\_grados <= 130;

when "1101" => servo\_grados <= 140;

when "1110" => servo\_grados <= 150;

when others => servo\_grados <= 170;

end case;

-- servo\_grados por switches

elsif servo\_selector\_input\_mode = '0' then

case servo\_selector\_switches is

when "0000" => servo\_grados <= 10;

when "0001" => servo\_grados <= 20;

when "0010" => servo\_grados <= 30;

when "0011" => servo\_grados <= 40;

when "0100" => servo\_grados <= 50;

when "0101" => servo\_grados <= 60;

when "0110" => servo\_grados <= 70;

when "0111" => servo\_grados <= 80;

when "1000" => servo\_grados <= 90;

when "1001" => servo\_grados <= 100;

when "1010" => servo\_grados <= 110;

when "1011" => servo\_grados <= 120;

when "1100" => servo\_grados <= 130;

when "1101" => servo\_grados <= 140;

when "1110" => servo\_grados <= 150;

when others => servo\_grados <= 170;

end case;

-- servo\_grados por dedo

else

servo\_grados <= servo\_numero\_int;

end if;

end process;

-- process del automata del pwm del servo

process(clk, servo\_inicio)

begin

if servo\_inicio = '1' then

servo\_estado\_servo <= "00";

servo\_cont\_flancos <= 0;

elsif rising\_edge(clk) then

case servo\_estado\_servo is

when "00" =>

servo\_cont\_flancos <= 0;

servo\_estado\_servo <= "01";

when "01" =>

servo\_cont\_flancos <= 1;

servo\_estado\_servo <= "10";

when "10" =>

servo\_cont\_flancos <= servo\_cont\_flancos + 1;

if servo\_cont\_flancos = servo\_pwm\_longitud\_pulso then

servo\_estado\_servo <= "11";

else

servo\_estado\_servo <= "10";

end if;

when others =>

servo\_cont\_flancos <= servo\_cont\_flancos + 1;

if servo\_cont\_flancos = 2000000 then

servo\_estado\_servo <= "01";

else

servo\_estado\_servo <= "11";

end if;

end case;

end if;

end process;

-- process de salidas del servo

process(servo\_estado\_servo)

begin

case servo\_estado\_servo is

when "00" => servo <= '0';

when "01" => servo <= '1';

when "10" => servo <= '1';

when others => servo <= '0';

end case;

end process;

-- ####################################################################

-- ####################################################################

-- LOGICA DEL SERVO-RELOJ

-- ####################################################################

-- ####################################################################

-- proceso de reloj

process(servo\_inicio, clk)

begin

if servo\_inicio = '1' then

servo\_cont\_base <= 0;

elsif rising\_edge(clk) then

if servo\_cont\_base = servo\_tope\_freq then

servo\_cont\_base <= 0;

else

servo\_cont\_base <= servo\_cont\_base + 1;

end if;

end if;

end process;

-- process de cambio de vel.

process(servo\_segundos\_offset)

begin

if servo\_segundos\_offset = "XXX1" then

servo\_tope\_freq <= 100000000;

elsif servo\_segundos\_offset = "XX10" then

servo\_tope\_freq <= 200000000;

elsif servo\_segundos\_offset = "X100" then

servo\_tope\_freq <= 300000000;

elsif servo\_segundos\_offset = "1000" then

servo\_tope\_freq <= 400000000;

else

servo\_tope\_freq <= 100000000;

end if;

end process;

-- process de cambio de servo\_aspersor\_cont

process(servo\_inicio, clk)

begin

if servo\_inicio = '1' then

servo\_suma\_o\_resta <= '0';

servo\_aspersor\_cont <= "0000";

elsif rising\_edge(clk) then

if servo\_selector\_aspersor\_mode = '1' or reloj\_alarma\_sonar = '1' then

if servo\_cont\_base = servo\_tope\_freq then

if servo\_aspersor\_cont = "1111" then

servo\_suma\_o\_resta <= '1';

elsif servo\_aspersor\_cont = "0000" then

servo\_suma\_o\_resta <= '0';

end if;

if servo\_suma\_o\_resta = '0' and servo\_aspersor\_cont /= "1111" then

servo\_aspersor\_cont <= servo\_aspersor\_cont + 1;

elsif servo\_suma\_o\_resta = '1' and servo\_aspersor\_cont /= "0000" then

servo\_aspersor\_cont <= servo\_aspersor\_cont - 1;

end if;

end if;

else

servo\_suma\_o\_resta <= '0';

servo\_aspersor\_cont <= "0000";

end if;

end if;

end process;

-- ####################################################################

-- ####################################################################

-- LOGICA DEL SERVO-PULSADOR

-- ####################################################################

-- ####################################################################

-- process del automata pulsador dedo

process(servo\_inicio, clk)

begin

if servo\_inicio = '1' then

servo\_estado\_pulsador <= "000";

servo\_cont\_filtro <= 0;

elsif rising\_edge(clk) then

case servo\_estado\_pulsador is

when "000" => -- INICIO

servo\_cont\_filtro <= 0;

if servo\_botonMas = '1' or servo\_botonMenos = '1' then

servo\_estado\_pulsador <= "001";

else

servo\_estado\_pulsador <= "000";

end if;

when "001" => -- FILTRADO

servo\_cont\_filtro <= servo\_cont\_filtro + 1;

if (servo\_botonMas = '1' or servo\_botonMenos = '1') and servo\_cont\_filtro < servo\_freq\_min then

servo\_estado\_pulsador <= "001";

elsif (servo\_botonMas = '1' or servo\_botonMenos = '1') and servo\_cont\_filtro = servo\_freq\_min then

if servo\_botonMas = '1'then

servo\_estado\_pulsador <= "010";

elsif servo\_botonMenos = '1' then

servo\_estado\_pulsador <= "100";

end if;

else

servo\_estado\_pulsador <= "000";

end if;

when "010" => -- UNO +

servo\_cont\_filtro <= servo\_cont\_filtro + 1;

if servo\_botonMas = '1' and servo\_cont\_filtro < 200000000 then

servo\_estado\_pulsador <= "010";

elsif servo\_botonMas = '1' and servo\_cont\_filtro = 200000000 then

servo\_estado\_pulsador <= "110";

elsif servo\_botonMas = '0' then

servo\_estado\_pulsador <= "011";

end if;

when "011" => -- SUMA

servo\_cont\_filtro <= 0;

if servo\_botonMas = '1' then

servo\_estado\_pulsador <= "001";

else

servo\_estado\_pulsador <= "000";

end if;

when "100" => -- UNO -

servo\_cont\_filtro <= servo\_cont\_filtro + 1;

if servo\_botonMenos = '1' and servo\_cont\_filtro < 200000000 then

servo\_estado\_pulsador <= "100";

elsif servo\_botonMenos = '1' and servo\_cont\_filtro = 200000000 then

servo\_estado\_pulsador <= "110";

elsif servo\_botonMenos = '0' then

servo\_estado\_pulsador <= "101";

end if;

when "101" => -- RESTA

servo\_cont\_filtro <= 0;

if servo\_botonMenos = '1' then

servo\_estado\_pulsador <= "001";

else

servo\_estado\_pulsador <= "000";

end if;

when "110" => -- START\_RAPIDO

servo\_cont\_filtro <= 0;

if servo\_botonMas = '1' or servo\_botonMenos = '1' then

servo\_estado\_pulsador <= "111";

elsif servo\_botonMas = '0' and servo\_botonMenos = '0' then

servo\_estado\_pulsador <= "000";

end if;

when others => -- RAPIDO

servo\_cont\_filtro <= servo\_cont\_filtro + 1;

if (servo\_botonMas = '1' or servo\_botonMenos = '1') and servo\_cont\_filtro < 20000000 then

servo\_estado\_pulsador <= "111";

elsif (servo\_botonMas = '1' or servo\_botonMenos = '1') and servo\_cont\_filtro = 20000000 then

servo\_estado\_pulsador <= "110";

elsif servo\_botonMas = '0' and servo\_botonMenos = '0' then

servo\_estado\_pulsador <= "000";

end if;

end case;

end if;

end process;

-- process de las salidas pulsador dedo

process(servo\_estado\_pulsador, servo\_botonMas, servo\_botonMenos, servo\_inicio, clk)

begin

if servo\_inicio = '1' then

servo\_salida <= '0';

servo\_flag\_suma <= '0';

servo\_flag\_resta <= '0';

elsif rising\_edge(clk) then

case servo\_estado\_pulsador is

when "000" =>

servo\_salida <= '0';

servo\_flag\_suma <= '0';

servo\_flag\_resta <= '0';

when "001" =>

servo\_salida <= '0';

servo\_flag\_suma <= '0';

servo\_flag\_resta <= '0';

when "010" =>

servo\_salida <= '0';

servo\_flag\_suma <= '0';

servo\_flag\_resta <= '0';

when "011" =>

servo\_salida <= '1';

servo\_flag\_suma <= '1';

servo\_flag\_resta <= '0';

when "100" =>

servo\_salida <= '0';

servo\_flag\_suma <= '0';

servo\_flag\_resta <= '0';

when "101" =>

servo\_salida <= '1';

servo\_flag\_suma <= '0';

servo\_flag\_resta <= '1';

when "110" =>

servo\_salida <= '1';

if servo\_botonMas = '1' then

servo\_flag\_suma <= '1';

servo\_flag\_resta <= '0';

elsif servo\_botonMenos = '1' then

servo\_flag\_suma <= '0';

servo\_flag\_resta <= '1';

end if;

when others =>

servo\_salida <= '0';

servo\_flag\_suma <= '0';

servo\_flag\_resta <= '0';

end case;

end if;

end process;

-- process de sumar/restar decenas

process(servo\_inicio, clk)

begin

if servo\_inicio = '1' then

servo\_contador\_decenas <= "0001";

elsif rising\_edge(clk) then

if servo\_salida = '1' then

if servo\_flag\_suma = '1' then

if servo\_contador\_decenas = 7 and servo\_contador\_centenas = 1 then

servo\_contador\_decenas <= "0111";

elsif servo\_contador\_decenas = 9 then

servo\_contador\_decenas <= "0000";

else

servo\_contador\_decenas <= servo\_contador\_decenas + 1;

end if;

elsif servo\_flag\_resta = '1' then

if servo\_contador\_decenas = 1 and servo\_contador\_centenas = 0 then

servo\_contador\_decenas <= "0001";

elsif servo\_contador\_decenas = 0 then

servo\_contador\_decenas <= "1001";

else

servo\_contador\_decenas <= servo\_contador\_decenas - 1;

end if;

end if;

end if;

end if;

end process;

-- process de sumar/restar centenas

process(servo\_inicio, clk)

begin

if servo\_inicio = '1' then

servo\_contador\_centenas <= "0000";

elsif rising\_edge(clk) then

if servo\_salida = '1' then

if servo\_flag\_suma = '1' then

if servo\_contador\_decenas = 9 then

servo\_contador\_centenas <= servo\_contador\_centenas + 1;

end if;

elsif servo\_flag\_resta = '1' then

if servo\_contador\_centenas = 1 and servo\_contador\_decenas = 0 then

servo\_contador\_centenas <= servo\_contador\_centenas - 1;

end if;

end if;

end if;

end if;

end process;

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- LOGICA DE CONVERSION

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- process del automata de la conversion

process(clk, binario\_bcd\_inicio)

begin

if binario\_bcd\_inicio = '1' then

binario\_bcd\_vector <= "00000000000000000000000000";

binario\_bcd\_estado\_conversion <= "00";

binario\_bcd\_contador\_desplazamientos <= 0;

binario\_bcd\_unidades <= "0000";

binario\_bcd\_decenas <= "0000";

binario\_bcd\_centenas <= "0000";

binario\_bcd\_millar <= "0000";

binario\_bcd\_fin <= '0';

elsif rising\_edge(clk) then

if binario\_bcd\_cont = 0 and binario\_bcd\_fin = '0' then

case binario\_bcd\_estado\_conversion is

-- start

when "00" =>

binario\_bcd\_contador\_desplazamientos <= 0;

binario\_bcd\_vector <= "0000000000000000" & binario\_bcd\_binario;

if binario\_bcd\_enable = '1' then

binario\_bcd\_estado\_conversion <= "01";

else

binario\_bcd\_estado\_conversion <= "00";

end if;

binario\_bcd\_fin <= '0';

-- despl

when "01" =>

binario\_bcd\_contador\_desplazamientos <= binario\_bcd\_contador\_desplazamientos + 1;

binario\_bcd\_vector <= binario\_bcd\_vector(24 downto 0) & '0';

if binario\_bcd\_contador\_desplazamientos < 9 then

binario\_bcd\_estado\_conversion <= "10";

else

binario\_bcd\_estado\_conversion <= "11";

end if;

binario\_bcd\_fin <= '0';

-- ¿sumar+3?

when "10" =>

binario\_bcd\_contador\_desplazamientos <= binario\_bcd\_contador\_desplazamientos;

if binario\_bcd\_vector(13 downto 10) > 4 then

binario\_bcd\_vector(13 downto 10) <= binario\_bcd\_vector(13 downto 10) + "0011";

end if;

if binario\_bcd\_vector(17 downto 14) > 4 then

binario\_bcd\_vector(17 downto 14) <= binario\_bcd\_vector(17 downto 14) + "0011";

end if;

if binario\_bcd\_vector(21 downto 18) > 4 then

binario\_bcd\_vector(21 downto 18) <= binario\_bcd\_vector(21 downto 18) + "0011";

end if;

if binario\_bcd\_vector(25 downto 22) > 4 then

binario\_bcd\_vector(25 downto 22) <= binario\_bcd\_vector(25 downto 22) + "0011";

end if;

binario\_bcd\_estado\_conversion <= "01";

binario\_bcd\_fin <= '0';

-- final

when others =>

binario\_bcd\_contador\_desplazamientos <= binario\_bcd\_contador\_desplazamientos;

binario\_bcd\_vector <= binario\_bcd\_vector;

binario\_bcd\_estado\_conversion <= "00";

binario\_bcd\_fin <= '1';

binario\_bcd\_unidades <= binario\_bcd\_vector(13 downto 10);

binario\_bcd\_decenas <= binario\_bcd\_vector(17 downto 14);

binario\_bcd\_centenas <= binario\_bcd\_vector(21 downto 18);

binario\_bcd\_millar <= binario\_bcd\_vector(25 downto 22);

end case;

end if;

end if;

end process;

-- process de conteo de segundos

process(clk, binario\_bcd\_inicio)

begin

if binario\_bcd\_inicio = '1' then

binario\_bcd\_cont <= 0;

elsif rising\_edge(clk) then

if binario\_bcd\_cont = binario\_bcd\_tope\_freq then

binario\_bcd\_cont <= 0;

else

binario\_bcd\_cont <= binario\_bcd\_cont + 1;

end if;

end if;

end process;

-- process de cambio de vel.

process(binario\_bcd\_modo\_lento\_rapido)

begin

if binario\_bcd\_modo\_lento\_rapido = '1' then

binario\_bcd\_tope\_freq <= 0;

else

binario\_bcd\_tope\_freq <= 50000000;

end if;

end process;

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- LOGICA DEL CRONOMETRO

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- process de variables temporales para el split del cronometro

process(cronom\_inicio, clk)

begin

if cronom\_inicio = '1' then

cronom\_cont\_centesimas\_temp <= "0000";

cronom\_cont\_decimas\_temp <= "0000";

cronom\_cont\_segs\_unidades\_temp <= "0000";

cronom\_cont\_segs\_decenas\_temp <= "0000";

elsif rising\_edge(clk) then

if cronom\_split = '0' then

cronom\_cont\_centesimas\_temp <= cronom\_cont\_centesimas;

cronom\_cont\_decimas\_temp <= cronom\_cont\_decimas;

cronom\_cont\_segs\_unidades\_temp <= cronom\_cont\_segs\_unidades;

cronom\_cont\_segs\_decenas\_temp <= cronom\_cont\_segs\_decenas;

end if;

end if;

end process;

-- process de reloj del cronometro

process(cronom\_inicio, clk)

begin

if cronom\_inicio = '1' then

cronom\_cont\_base <= 0;

elsif rising\_edge(clk) then

if cronom\_cont\_base = cronom\_tope\_freq then

cronom\_cont\_base <= 0;

else

cronom\_cont\_base <= cronom\_cont\_base + 1;

end if;

end if;

end process;

-- process de monitoreo de segundos.centesimas

process(cronom\_inicio, clk)

begin

if cronom\_inicio = '1' then

cronom\_cont\_centesimas <= "0000";

elsif rising\_edge(clk) then

if cronom\_pausa = '0' then

-- si ha pasado una segundos.centesimas

if cronom\_cont\_base = cronom\_tope\_freq then

if cronom\_cont\_centesimas = "1001" then

cronom\_cont\_centesimas <= "0000";

else

cronom\_cont\_centesimas <= cronom\_cont\_centesimas + 1;

end if;

end if;

elsif cronom\_pausa = '1' then

-- hacer cosas

end if;

end if;

end process;

-- process de monitoreo de segundos.decimas

process(cronom\_inicio, clk)

begin

if cronom\_inicio = '1' then

cronom\_cont\_decimas <= "0000";

elsif rising\_edge(clk) then

if cronom\_pausa = '0' then

-- si ha pasado una segundos.centesimas y las segundos.centesimas = 9

if cronom\_cont\_base = cronom\_tope\_freq and cronom\_cont\_centesimas = "1001" then

if cronom\_cont\_decimas = "1001" then

cronom\_cont\_decimas <= "0000";

else

cronom\_cont\_decimas <= cronom\_cont\_decimas + 1;

end if;

end if;

elsif cronom\_pausa = '1' then

-- hacer cosas

end if;

end if;

end process;

-- process de monitoreo de segundos.unidades

process(cronom\_inicio, clk)

begin

if cronom\_inicio = '1' then

cronom\_cont\_segs\_unidades <= "0000";

elsif rising\_edge(clk) then

if cronom\_pausa = '0' then

-- si ha pasado una segundos.centesimas y las segundos.centesimas = 9 y las segundos.decimas = 9

if cronom\_cont\_base = cronom\_tope\_freq and cronom\_cont\_centesimas = "1001" and cronom\_cont\_decimas = "1001" then

if cronom\_cont\_segs\_unidades = "1001" then

cronom\_cont\_segs\_unidades <= "0000";

else

cronom\_cont\_segs\_unidades <= cronom\_cont\_segs\_unidades + 1;

end if;

end if;

elsif cronom\_pausa = '1' then

-- hacer cosas

end if;

end if;

end process;

-- process de monitoreo de segundos.decenas

process(cronom\_inicio, clk)

begin

if cronom\_inicio = '1' then

cronom\_cont\_segs\_decenas <= "0000";

elsif rising\_edge(clk) then

if cronom\_pausa = '0' then

-- si ha pasado una segundos.centesimas y las segundos.centesimas = 9 y las segundos.decimas = 9 y las segundos.unidades = 9

if cronom\_cont\_base = cronom\_tope\_freq and cronom\_cont\_centesimas = "1001" and cronom\_cont\_decimas = "1001" and cronom\_cont\_segs\_unidades = "1001" then

if cronom\_cont\_segs\_decenas = "0101" then

cronom\_cont\_segs\_decenas <= "0000";

else

cronom\_cont\_segs\_decenas <= cronom\_cont\_segs\_decenas + 1;

end if;

end if;

elsif cronom\_pausa = '1' then

-- hacer cosas

end if;

end if;

end process;

-- process de monitoreo de minutos.unidades

process(cronom\_inicio, clk)

begin

if cronom\_inicio = '1' then

cronom\_cont\_mins\_unidades <= "0000";

elsif rising\_edge(clk) then

if cronom\_pausa = '0' then

-- si ha pasado una segundos.centesimas y las segundos.centesimas = 9 y las segundos.decimas = 9 y las segundos.unidades = 9 y

-- las segundos.decenas = 5

if cronom\_cont\_base = cronom\_tope\_freq and cronom\_cont\_centesimas = "1001" and cronom\_cont\_decimas = "1001" and cronom\_cont\_segs\_unidades = "1001" and cronom\_cont\_segs\_decenas = "0101" then

if cronom\_cont\_mins\_unidades = "1001" then

cronom\_cont\_mins\_unidades <= "0000";

else

cronom\_cont\_mins\_unidades <= cronom\_cont\_mins\_unidades + 1;

end if;

end if;

elsif cronom\_pausa = '1' then

-- hacer cosas

end if;

end if;

end process;

-- process de monitoreo de minutos.decenas

process(cronom\_inicio, clk)

begin

if cronom\_inicio = '1' then

cronom\_cont\_mins\_decenas <= "0000";

elsif rising\_edge(clk) then

if cronom\_pausa = '0' then

-- si ha pasado una segundos.centesimas y las segundos.centesimas = 9 y las segundos.decimas = 9 y las segundos.unidades = 9 y

-- las segundos.decenas = 5 y las minutos.unidades = 9

if cronom\_cont\_base = cronom\_tope\_freq and cronom\_cont\_centesimas = "1001" and cronom\_cont\_decimas = "1001" and cronom\_cont\_segs\_unidades = "1001" and cronom\_cont\_segs\_decenas = "0101" and cronom\_cont\_mins\_unidades = "1001" then

if cronom\_cont\_mins\_decenas = "0101" then

cronom\_cont\_mins\_decenas <= "0000";

else

cronom\_cont\_mins\_decenas <= cronom\_cont\_mins\_decenas + 1;

end if;

end if;

elsif cronom\_pausa = '1' then

-- hacer cosas

end if;

end if;

end process;

-- process de monitoreo de horas.unidades

process(cronom\_inicio, clk)

begin

if cronom\_inicio = '1' then

cronom\_cont\_horas\_unidades <= "0000";

elsif rising\_edge(clk) then

if cronom\_pausa = '0' then

-- si ha pasado una segundos.centesimas y las segundos.centesimas = 9 y las segundos.decimas = 9 y las segundos.unidades = 9 y

-- las segundos.decenas = 5 y las minutos.unidades = 9 y las minutos.decenas = 5

if cronom\_cont\_base = cronom\_tope\_freq and cronom\_cont\_centesimas = "1001" and cronom\_cont\_decimas = "1001" and cronom\_cont\_segs\_unidades = "1001" and cronom\_cont\_segs\_decenas = "0101" and cronom\_cont\_mins\_unidades = "1001" and cronom\_cont\_mins\_decenas = "0101" then

if cronom\_cont\_horas\_unidades = "1001" then

cronom\_cont\_horas\_unidades <= "0000";

elsif cronom\_cont\_horas\_unidades = "0011" and cronom\_cont\_horas\_decenas = "0010" then

cronom\_cont\_horas\_unidades <= "0000";

else

cronom\_cont\_horas\_unidades <= cronom\_cont\_horas\_unidades + 1;

end if;

end if;

elsif cronom\_pausa = '1' then

-- hacer cosas

end if;

end if;

end process;

-- process de monitoreo de horas.decenas

process(cronom\_inicio, clk)

begin

if cronom\_inicio = '1' then

cronom\_cont\_horas\_decenas <= "0000";

elsif rising\_edge(clk) then

if cronom\_pausa = '0' then

-- si ha pasado una segundos.centesimas y las segundos.centesimas = 9 y las segundos.decimas = 9 y las segundos.unidades = 9 y

-- las segundos.decenas = 5 y las minutos.unidades = 9 y las minutos.decenas = 5 y las horas.unidades = 9 or 3

if cronom\_cont\_base = cronom\_tope\_freq and cronom\_cont\_centesimas = "1001" and cronom\_cont\_decimas = "1001" and cronom\_cont\_segs\_unidades = "1001" and cronom\_cont\_segs\_decenas = "0101" and cronom\_cont\_mins\_unidades = "1001" and cronom\_cont\_mins\_decenas = "0101" and (cronom\_cont\_horas\_unidades = "1001" or cronom\_cont\_horas\_unidades = "0011") then

if cronom\_cont\_horas\_decenas = "0010" then

cronom\_cont\_horas\_decenas <= "0000";

elsif cronom\_cont\_horas\_unidades = "1001" then

cronom\_cont\_horas\_decenas <= cronom\_cont\_horas\_decenas + 1;

end if;

end if;

elsif cronom\_pausa = '1' then

-- hacer cosas

end if;

end if;

end process;

-- ####################################################################

-- ####################################################################

-- LOGICA DEL PULSADOR DE DEDO STOP SPLIT

-- ####################################################################

-- ####################################################################

-- process del automata del pulsador

process(cronom\_inicio, clk)

begin

if cronom\_inicio = '1' then

cronom\_func\_estado\_pulsador <= "000";

cronom\_func\_cont\_filtro <= 0;

elsif rising\_edge(clk) then

case cronom\_func\_estado\_pulsador is

when "000" => -- INICIO

cronom\_func\_cont\_filtro <= 0;

if cronom\_func\_btnU\_stop = '1' or cronom\_func\_btnD\_split = '1' then

cronom\_func\_estado\_pulsador <= "001";

else

cronom\_func\_estado\_pulsador <= "000";

end if;

when "001" => -- FILTRADO

cronom\_func\_cont\_filtro <= cronom\_func\_cont\_filtro + 1;

if (cronom\_func\_btnU\_stop = '1' or cronom\_func\_btnD\_split = '1') and cronom\_func\_cont\_filtro < 100000 then

cronom\_func\_estado\_pulsador <= "001";

elsif (cronom\_func\_btnU\_stop = '1' or cronom\_func\_btnD\_split = '1') and cronom\_func\_cont\_filtro = 100000 then

if cronom\_func\_btnU\_stop = '1'then

cronom\_func\_estado\_pulsador <= "010";

elsif cronom\_func\_btnD\_split = '1' then

cronom\_func\_estado\_pulsador <= "100";

end if;

else

cronom\_func\_estado\_pulsador <= "000";

end if;

when "010" => -- UNO +

cronom\_func\_cont\_filtro <= 0;

if cronom\_func\_btnU\_stop = '1' then

cronom\_func\_estado\_pulsador <= "010";

else

cronom\_func\_estado\_pulsador <= "011";

end if;

when "011" => -- SUMA

cronom\_func\_cont\_filtro <= 0;

if cronom\_func\_btnU\_stop = '1' then

cronom\_func\_estado\_pulsador <= "001";

else

cronom\_func\_estado\_pulsador <= "000";

end if;

when "100" => -- UNO -

cronom\_func\_cont\_filtro <= 0;

if cronom\_func\_btnD\_split = '1' then

cronom\_func\_estado\_pulsador <= "100";

else

cronom\_func\_estado\_pulsador <= "101";

end if;

when "101" => -- RESTA

cronom\_func\_cont\_filtro <= 0;

if cronom\_func\_btnD\_split = '1' then

cronom\_func\_estado\_pulsador <= "001";

else

cronom\_func\_estado\_pulsador <= "000";

end if;

when others =>

cronom\_func\_cont\_filtro <= 0;

cronom\_func\_estado\_pulsador <= "000";

end case;

end if;

end process;

-- process de las salidas del pulsador

process(cronom\_func\_estado\_pulsador)

begin

case cronom\_func\_estado\_pulsador is

when "000" =>

cronom\_func\_flag\_salida <= '0';

cronom\_func\_flag\_salida\_flag\_pausa <= '0';

cronom\_func\_flag\_salida\_flag\_split <= '0';

when "001" =>

cronom\_func\_flag\_salida <= '0';

cronom\_func\_flag\_salida\_flag\_pausa <= '0';

cronom\_func\_flag\_salida\_flag\_split <= '0';

when "010" =>

cronom\_func\_flag\_salida <= '0';

cronom\_func\_flag\_salida\_flag\_pausa <= '0';

cronom\_func\_flag\_salida\_flag\_split <= '0';

when "011" =>

cronom\_func\_flag\_salida <= '1';

cronom\_func\_flag\_salida\_flag\_pausa <= '1';

cronom\_func\_flag\_salida\_flag\_split <= '0';

when "100" =>

cronom\_func\_flag\_salida <= '0';

cronom\_func\_flag\_salida\_flag\_pausa <= '0';

cronom\_func\_flag\_salida\_flag\_split <= '0';

when "101" =>

cronom\_func\_flag\_salida <= '1';

cronom\_func\_flag\_salida\_flag\_pausa <= '0';

cronom\_func\_flag\_salida\_flag\_split <= '1';

when others =>

cronom\_func\_flag\_salida <= '0';

cronom\_func\_flag\_salida\_flag\_pausa <= '0';

cronom\_func\_flag\_salida\_flag\_split <= '0';

end case;

end process;

-- process de sumar/restar el cronom\_func\_flag\_pausa

process(cronom\_inicio, clk)

begin

if cronom\_inicio = '1' then

cronom\_func\_flag\_pausa <= '0';

elsif rising\_edge(clk) then

if cronom\_func\_flag\_salida = '1' then

if cronom\_func\_flag\_salida\_flag\_pausa = '1' then

if cronom\_func\_flag\_pausa = '0' then

cronom\_func\_flag\_pausa <= '1';

cronom\_pausa <= '1';

elsif cronom\_func\_flag\_pausa = '1' then

cronom\_func\_flag\_pausa <= '0';

cronom\_pausa <= '0';

end if;

elsif cronom\_func\_flag\_salida\_flag\_split = '1' then

if cronom\_func\_flag\_split = '0' then

cronom\_func\_flag\_split <= '1';

cronom\_split <= '1';

elsif cronom\_func\_flag\_split = '1' then

cronom\_func\_flag\_split <= '0';

cronom\_split <= '0';

end if;

end if;

end if;

end if;

end process;

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- LOGICA DE LA PILA

-- ####################################################################

-- ####################################################################

-- ####################################################################

-- ####################################################################

reloj\_pila\_entrada <= reloj\_pila\_entrada\_hora\_decenas &

reloj\_pila\_entrada\_hora\_unidades &

reloj\_pila\_entrada\_min\_decenas &

reloj\_pila\_entrada\_min\_unidades;

process(reloj\_inicio, clk)

begin

if reloj\_inicio = '1' then

reloj\_pila\_entrada\_min\_unidades <= "0000";

reloj\_pila\_entrada\_min\_decenas <= "0000";

reloj\_pila\_entrada\_hora\_unidades <= "0000";

reloj\_pila\_entrada\_hora\_decenas <= "0000";

elsif rising\_edge(clk) then

if reloj\_pila\_push = '1' and reloj\_alarma\_sonar = '1' then

reloj\_pila\_entrada\_min\_unidades <= reloj\_alarma\_minutos\_unidades;

reloj\_pila\_entrada\_min\_decenas <= reloj\_alarma\_minutos\_decenas;

reloj\_pila\_entrada\_hora\_unidades <= reloj\_alarma\_horas\_unidades;

reloj\_pila\_entrada\_hora\_decenas <= reloj\_alarma\_horas\_decenas;

end if;

end if;

end process;

-- AUTOMATA DE LA PILA

process(clk, reloj\_inicio)

begin

if reloj\_inicio = '1' then

reloj\_estado\_pila <= vacia;

reloj\_pila\_stack\_pointer <= 7;

reloj\_pila\_salida <= "0000000000000000";

reloj\_pila <= ("0000000000000000", "0000000000000000", "0000000000000000", "0000000000000000",

"0000000000000000", "0000000000000000", "0000000000000000", "0000000000000000");

elsif rising\_edge(clk) then

case reloj\_estado\_pila is

when idle =>

if reloj\_pila\_salida\_push='1' then

reloj\_estado\_pila <= mete\_push;

reloj\_pila(reloj\_pila\_stack\_pointer) <= reloj\_pila\_entrada;

elsif reloj\_pila\_salida\_pop = '1' then

reloj\_estado\_pila <= saca\_pop;

reloj\_pila\_stack\_pointer <= reloj\_pila\_stack\_pointer + 1;

end if;

when mete\_push =>

reloj\_pila\_stack\_pointer <= reloj\_pila\_stack\_pointer - 1;

if reloj\_pila\_stack\_pointer /= 0 then

reloj\_estado\_pila <= idle;

else

reloj\_estado\_pila <= llena;

end if;

when saca\_pop =>

reloj\_pila\_salida <= reloj\_pila(reloj\_pila\_stack\_pointer);

if reloj\_pila\_stack\_pointer /= 7 then

reloj\_estado\_pila <= idle;

else

reloj\_estado\_pila <= vacia;

end if;

when llena =>

if reloj\_pila\_salida\_pop = '1' then

reloj\_estado\_pila <= saca\_pop;

reloj\_pila\_stack\_pointer <= reloj\_pila\_stack\_pointer + 1;

elsif reloj\_pila\_salida\_push = '1' then

reloj\_estado\_pila <= overflow;

end if;

when vacia =>

if reloj\_pila\_salida\_push = '1' then

reloj\_estado\_pila <= mete\_push;

reloj\_pila(reloj\_pila\_stack\_pointer) <= reloj\_pila\_entrada;

elsif reloj\_pila\_salida\_pop = '1' then

reloj\_estado\_pila <= underflow;

end if;

when overflow =>

if reloj\_pila\_salida\_pop = '1' then

reloj\_estado\_pila <= saca\_pop;

reloj\_pila\_stack\_pointer <= reloj\_pila\_stack\_pointer + 1;

end if;

when underflow =>

if reloj\_pila\_salida\_push = '1' then

reloj\_estado\_pila <= mete\_push;

reloj\_pila(reloj\_pila\_stack\_pointer) <= reloj\_pila\_entrada;

end if;

when others =>

reloj\_pila\_salida <= "0000000000000000";

reloj\_estado\_pila <= idle;

reloj\_pila\_stack\_pointer <= 7;

end case;

end if;

end process;

process(reloj\_estado\_pila)

begin

case reloj\_estado\_pila is

when idle =>

reloj\_pila\_llena <= '0';

reloj\_pila\_vacia <= '0';

reloj\_pila\_error\_overflow <= '0';

reloj\_pila\_error\_underflow <= '0';

when mete\_push =>

reloj\_pila\_llena <= '0';

reloj\_pila\_vacia <= '0';

reloj\_pila\_error\_overflow <= '0';

reloj\_pila\_error\_underflow <= '0';

when saca\_pop =>

reloj\_pila\_llena <= '0';

reloj\_pila\_vacia <= '0';

reloj\_pila\_error\_overflow <= '0';

reloj\_pila\_error\_underflow <= '0';

when llena =>

reloj\_pila\_llena <= '1';

reloj\_pila\_vacia <= '0';

reloj\_pila\_error\_overflow <= '0';

reloj\_pila\_error\_underflow <= '0';

when vacia =>

reloj\_pila\_llena <= '0';

reloj\_pila\_vacia <= '1';

reloj\_pila\_error\_overflow <= '0';

reloj\_pila\_error\_underflow <= '0';

when overflow =>

reloj\_pila\_llena <= '0';

reloj\_pila\_vacia <= '0';

reloj\_pila\_error\_overflow <= '1';

reloj\_pila\_error\_underflow <= '0';

when underflow =>

reloj\_pila\_llena <= '0';

reloj\_pila\_vacia <= '0';

reloj\_pila\_error\_overflow <= '0';

reloj\_pila\_error\_underflow <= '1';

when others =>

reloj\_pila\_llena <= '0';

reloj\_pila\_vacia <= '0';

reloj\_pila\_error\_overflow <= '0';

reloj\_pila\_error\_underflow <= '0';

end case;

end process;

-- FIN DEL AUTOMATA DE LA PILA

-- DETECCIÃ“N Y FILTRADO DE PULSO DE PUSH

process(clk, reloj\_inicio)

begin

if reloj\_inicio = '1' then

reloj\_pila\_estado\_push <= "00";

reloj\_pila\_cont\_filtro\_push <= 0;

elsif rising\_edge(clk) then

case reloj\_pila\_estado\_push is

when "00" =>

reloj\_pila\_cont\_filtro\_push <= 0;

if reloj\_pila\_push = '0' then

reloj\_pila\_estado\_push <= "00";

else

reloj\_pila\_estado\_push <= "01";

end if;

when "01" =>

reloj\_pila\_cont\_filtro\_push <= reloj\_pila\_cont\_filtro\_push+1;

if reloj\_pila\_push = '1' and reloj\_pila\_cont\_filtro\_push < 100000 then

reloj\_pila\_estado\_push <= "01";

elsif reloj\_pila\_push = '1' and reloj\_pila\_cont\_filtro\_push = 100000 then

reloj\_pila\_estado\_push <= "10";

else --pulsador = '0'

reloj\_pila\_estado\_push <= "00";

end if;

when "10" =>

reloj\_pila\_cont\_filtro\_push <= 0;

if reloj\_pila\_push = '1' then

reloj\_pila\_estado\_push <= "10";

else

if reloj\_alarma\_sonar = '1' then

reloj\_pila\_estado\_push <= "11";

else

reloj\_pila\_estado\_push <= "00";

end if;

end if;

when others =>

reloj\_pila\_cont\_filtro\_push <= 0;

reloj\_pila\_estado\_push <= "00";

end case;

end if;

end process;

process(reloj\_pila\_estado\_push)

begin

case reloj\_pila\_estado\_push is

when "00" => reloj\_pila\_salida\_push <= '0';

when "01" => reloj\_pila\_salida\_push <= '0';

when "10" => reloj\_pila\_salida\_push <= '0';

when others => reloj\_pila\_salida\_push <= '1';

end case;

end process;

-- DETECCIÃ“N Y FILTRADO DE PULSO DE POP

process(clk, reloj\_inicio)

begin

if reloj\_inicio = '1' then

reloj\_pila\_estado\_pop <= "00";

elsif rising\_edge(clk) then

case reloj\_pila\_estado\_pop is

when "00" =>

reloj\_pila\_cont\_filtro\_pop <= 0;

if reloj\_pila\_pop = '0' then

reloj\_pila\_estado\_pop <= "00";

else

reloj\_pila\_estado\_pop <= "01";

end if;

when "01" =>

reloj\_pila\_cont\_filtro\_pop <= reloj\_pila\_cont\_filtro\_pop+1;

if reloj\_pila\_pop = '1' and reloj\_pila\_cont\_filtro\_pop < 100000 then

reloj\_pila\_estado\_pop <= "01";

elsif reloj\_pila\_pop = '1' and reloj\_pila\_cont\_filtro\_pop = 100000 then

reloj\_pila\_estado\_pop <= "10";

else --pulsador = '0'

reloj\_pila\_estado\_pop <= "00";

end if;

when "10" =>

reloj\_pila\_cont\_filtro\_pop <= 0;

if reloj\_pila\_pop = '1' then

reloj\_pila\_estado\_pop <= "10";

else

reloj\_pila\_estado\_pop <= "11";

end if;

when others =>

reloj\_pila\_cont\_filtro\_pop <= 0;

reloj\_pila\_estado\_pop <= "00";

end case;

end if;

end process;

process(reloj\_pila\_estado\_pop)

begin

case reloj\_pila\_estado\_pop is

when "00" => reloj\_pila\_salida\_pop <= '0';

when "01" => reloj\_pila\_salida\_pop <= '0';

when "10" => reloj\_pila\_salida\_pop <= '0';

when others => reloj\_pila\_salida\_pop <= '1';

end case;

end process;

end Behavioral;