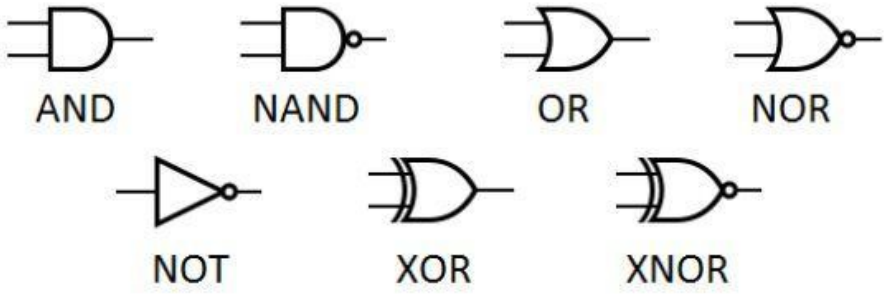
IE0323
Sistemas Digitales I

Laboratorio #7
Elementos Secuenciales

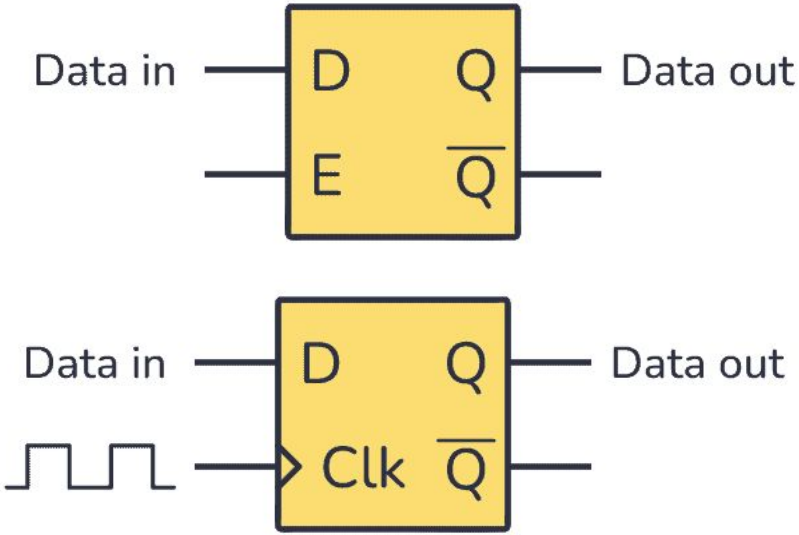
Objetivos

- Comprender el funcionamiento de los circuitos secuenciales y su diferencia con la lógica combinacional.
- Diseñar e implementar contadores síncronos y su aplicación en la visualización de números en un display de siete segmentos.
- Construir un reloj digital que mida minutos, segundos y centésimas de segundo.

Combinacional vs Secuencial



Salida depende de entradas ACTUALES



Salida depende de entradas ACTUALES Y ANTERIORES

Reloj en la DE10-Lite

- La tarjeta cuenta con 3 relojes accesibles a los/as desarrolladores
  - 1 reloj de 10MHz para conversión analógica digital **(no lo usaremos)**
  - 2 relojes de 50MHz para lógica

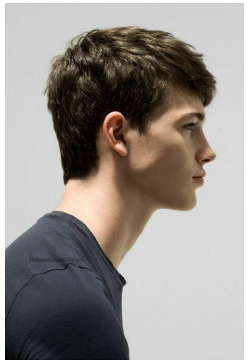
Table 3-2 Pin Assignment of Clock Inputs

Signal Name	FPGA Pin No.	Description	I/O Standard
ADC_CLK_10	PIN_N5	10 MHz clock input for ADC (Bank 3B)	3.3-V LVTTL
MAX10_CLK1_50	PIN_P11	50 MHz clock input(Bank 3B)	3.3-V LVTTL
MAX10_CLK2_50	PIN_N14	50 MHz clock input(Bank 3B)	3.3-V LVTTL

Reloj en la DE10-Lite

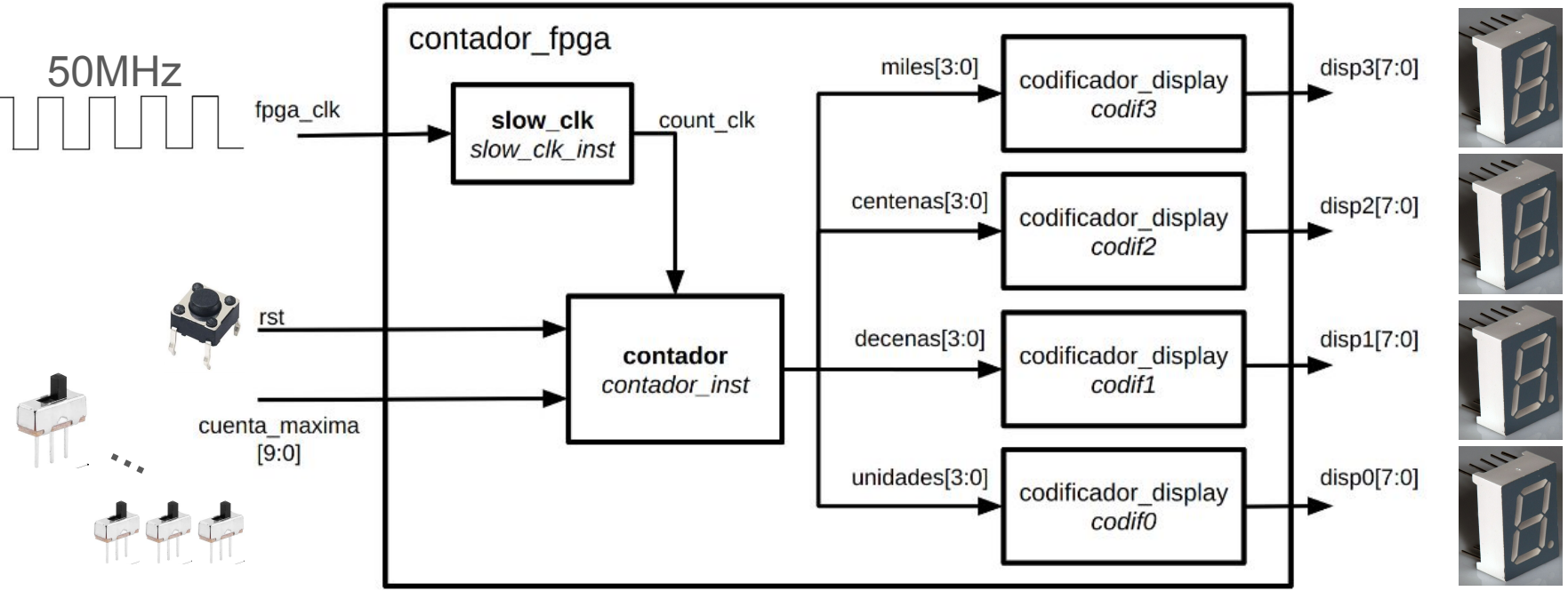
- Los relojes para lógica son de 50MHz, es decir 20ns
- El ojo humano no puede ver a más de 60Hz

No se da cuenta

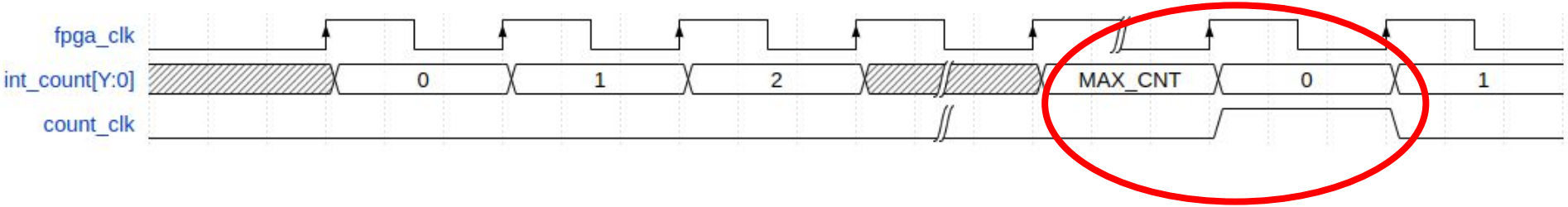


Cambia 50,000,000 de veces en un segundo

Ejemplo Introductorio – Contador



Módulo slow_clk



count_clk se activa cada MAX_CNT ciclos de fpga_clk

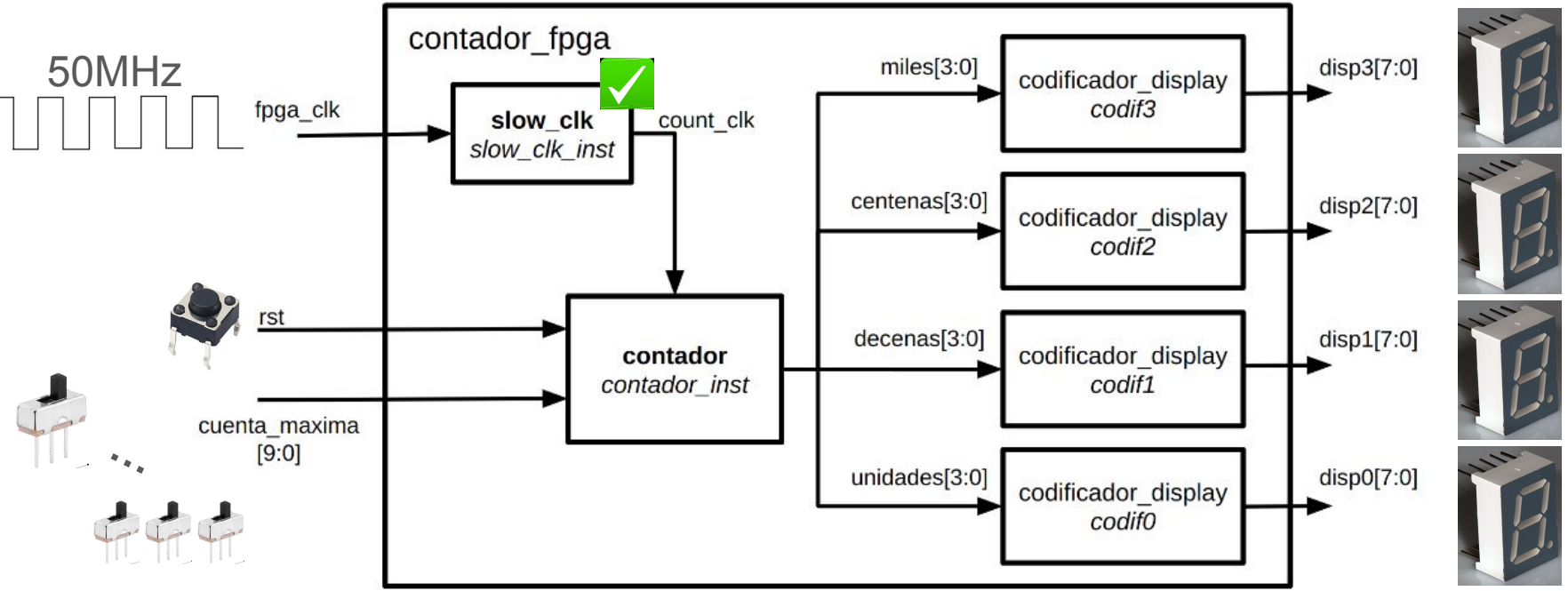
Módulo slow_clk

X es la cuenta máxima a la que llega el contador

Cada ciclo de fpga_clk, el valor de int_count se incrementa, EXCEPTO si llegamos a MAX_CNT (en ese caso reiniciamos la cuenta)

```
1  //Declaracion del modulo slow_clk
2  module slow_clk (fpga_clk, count_clk);
3    //La entrada es el clk rapido, que viene de la tarjeta
4    //Debe maperse a MAX10_CLK1_50 o MAX10_CLK2_50
5    input fpga_clk;
6    //La salida es el clk del contador, el cual es mas lento
7    output reg count_clk;
8
9    //Se define el parametro MAX_CLK
10   //como el valor maximo al que hay que reiniciar la cuenta
11   //Usted debe modificar el valor "X" por uno apropiado
12   parameter MAX_CLK = X;
13   //Mediante un log2 se calcula la cantidad de bits
14   //necesarios para almacenar la cuenta
15   parameter Y = $clog2(MAX_CLK);
16   //Se declara el conteo intermedio
17   reg [Y:0] int_count = 0;
18
19   //En cada flanco positivo del reloj
20   always @(posedge fpga_clk) begin
21     //Si la cuenta ya llego a MAX_CLK
22     if (int_count == MAX_CLK) begin
23       //Reinicie la cuenta
24       int_count <= 0;
25       //Y active el clk lento
26       count_clk <= 1;
27     end
28     //Si aun no llega a MAX_CLK
29     else begin
30       //Aumente la cuenta
31       int_count <= int_count + 1;
32       //Y mantenga (o ponga) el clk lento en 0
33       count_clk <= 0;
34     end
35   //Termina always
36   end
37 //Termina modulo
38 endmodule
```

Ejemplo Introductorio – Contador



Módulo contador

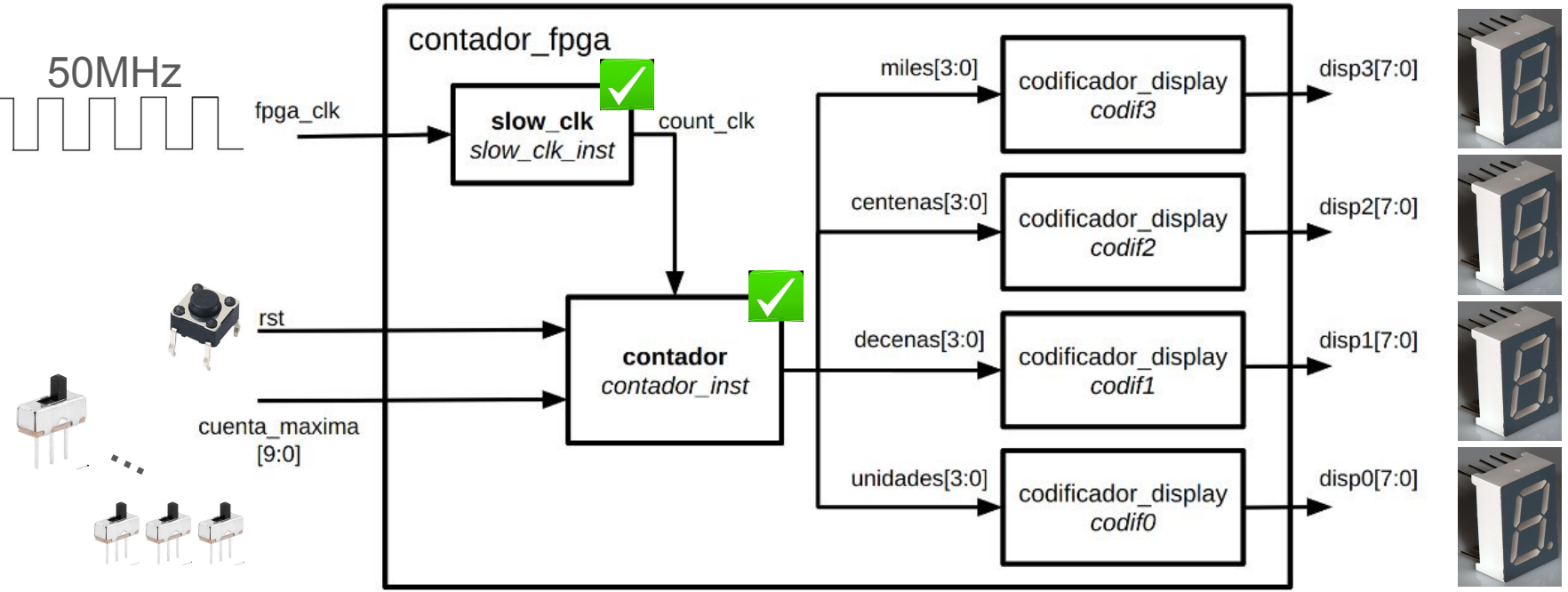
Usamos el reloj lento

Contamos de forma normal, de 1 en 1 y “en binario”

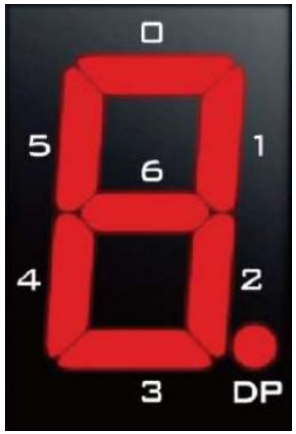
Convertimos el conteo a “unidades”, “decenas”, “centenas” y “unidades de millar”

```
module contador (rst, cuenta_maxima, count_clk,
                 unidades, decenas, centenas, miles);
  //Se definen las entradas:
  //El reset (debe maperse a Key0)
  //El clk de conteo (viene de slow_clk)
  input rst, count_clk;
  //El valor maximo a desplegar (SW9-SW0)
  input [9:0] cuenta_maxima;
  //Se definen las salidas:
   //Los registros para enviar a los displays
  //Cada uno contiene una cifra de significancia
  output reg  [3:0] unidades, decenas, centenas, miles;

 //El conteo a desplegar
  reg  [9:0] count;

  //En cada flanco positivo del reloj
  always @(posedge count_clk) begin
    //Si hay un reset, regrese a 0 el conteo
    if (rst == 0) count <= 0;
    //Si se alcanzo la cuenta maxima
    //regrese a 0 el conteo
    else if (count == cuenta_maxima) count <= 0;
    //Sino, siga contando normalmente
    else count <= count + 1;

    //Se calcula cada cifra
    //La parte entera de dividir el conteo entre 1000
    //da las unidades de millar
    miles <= count/1000;
    //Si se quitan las UM (mediante el residuo)
    //Y se divide por 100, obtenemos centenas
    centenas <= (count%1000)/100;
    //Si se quitan las centenas (mediante el residuo)
    //Y se divide por 10, obtenemos decenas
    decenas <= ((count%1000)%100)/10;
    //Si se quitan las decenas (mediante el residuo)
    //obtenemos unidades
    unidades <= ((count%1000)%100)%10;
  end
endmodule
```

Ejemplo Introductorio – Contador



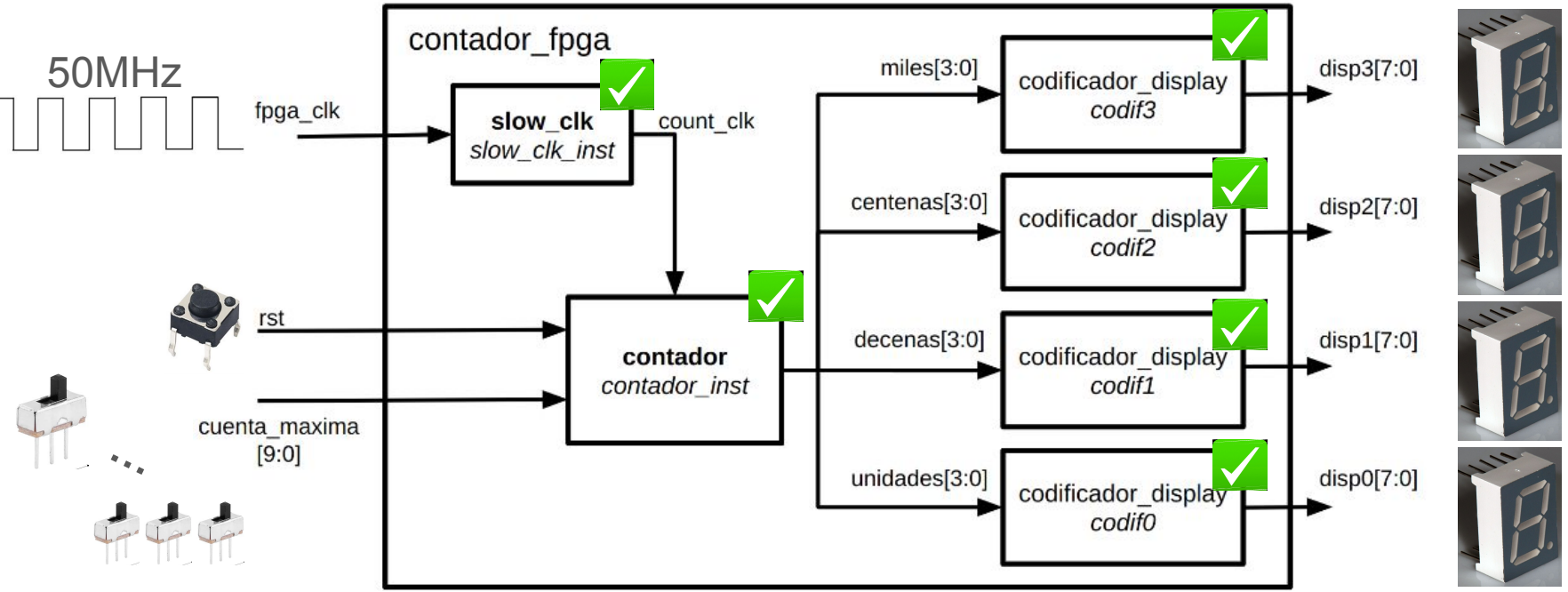
Módulo codificador_display



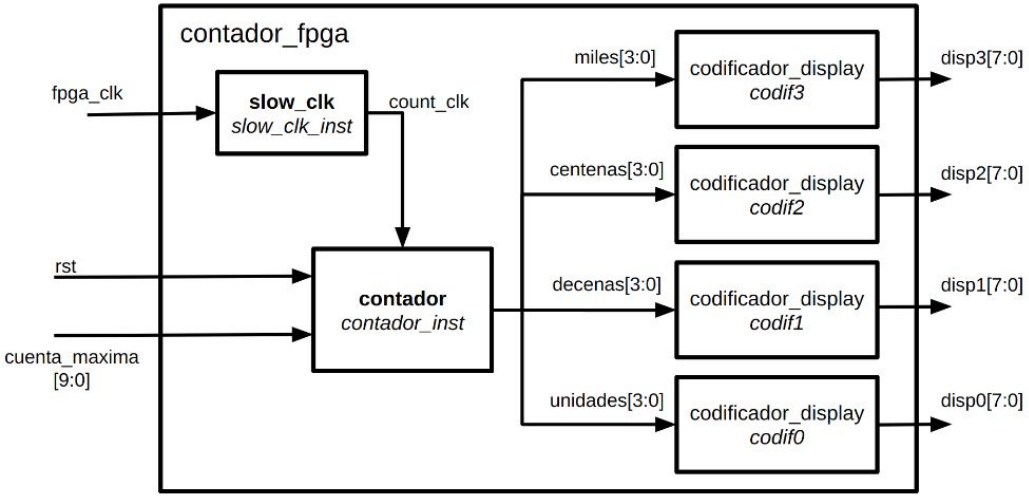

```
//Declaracion del modulo
//Este modulo se instanciara 4 veces
//Una para cada cifra significativa
//Pero solo lo escribimos una vez
//Luego solo lo instanciamos varias veces
//Cada una con entradas disintas
module codificador_display (bin, seg);
  //La entrada es el numero en binario
  //producido por el modulo contador
  input       [3:0] bin;
  //La salida es el bus de 8 bits (7seg + punto decimal)
  //Con cuales segmentos activar (en bajo)
  output reg  [7:0] seg;

  //Cada vez que cambie la entrada
  always @(bin) begin
    //Si la entrada es un 0
    if (bin == 0) seg = 8'b00111111;
    //Si la entrada es un 1
    else if (bin == 1) seg = 8'b00000110;
    //Encarguese usted de hacer los demas
    //...
  //Termina always
  end
//Termina modulo
endmodule
```

Faltan casos, ustedes deberán completarlos

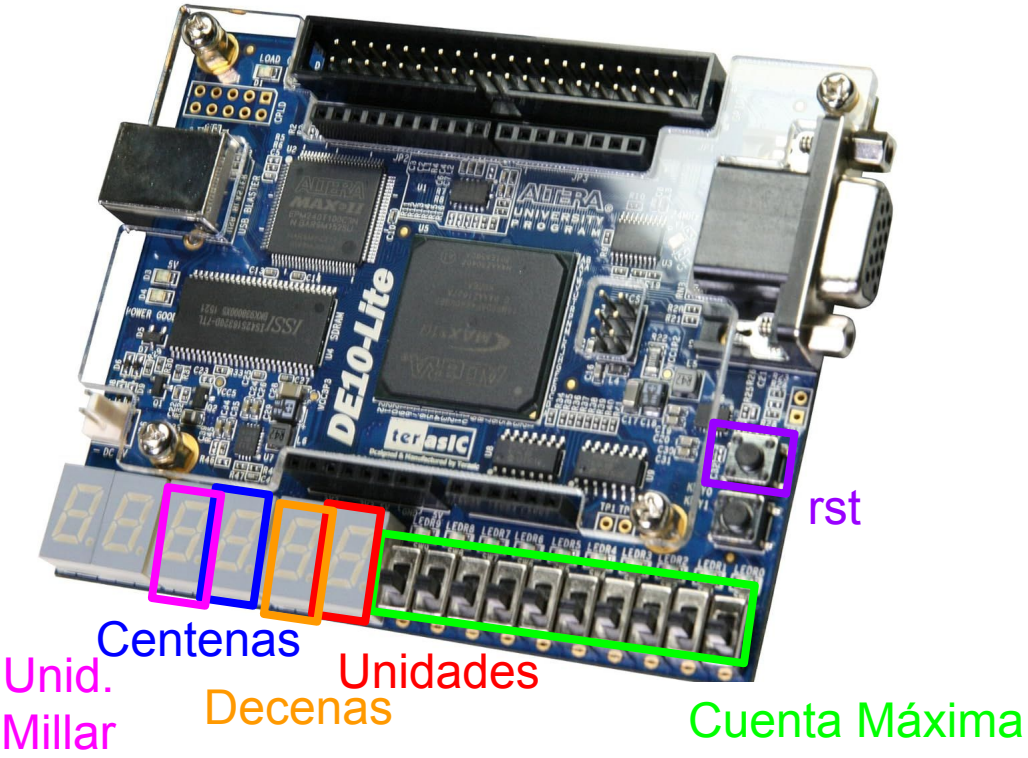
Ejemplo Introductorio – Contador



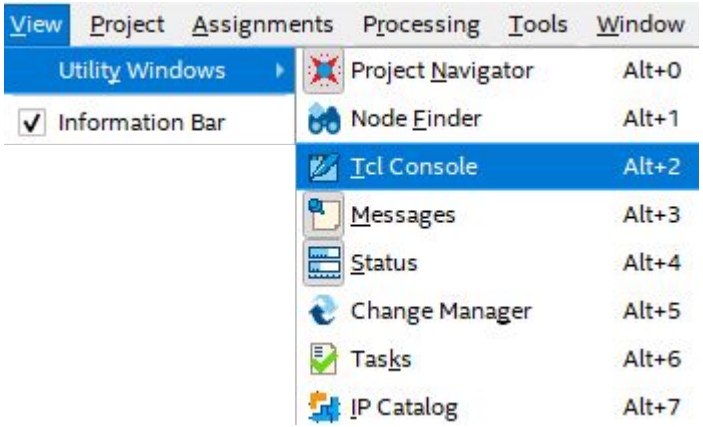
Módulo top

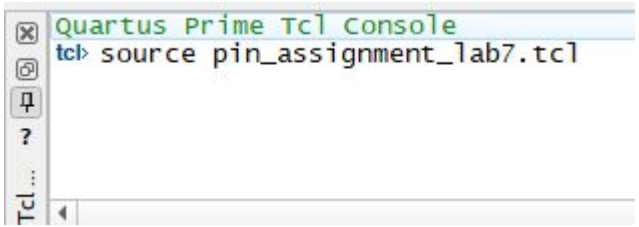



```
module contador_fpga (rst, cuenta_maxima, fpga_clk,
                      disp0, disp1, disp2, disp3);
  //Se definen las entradas
  //El fpga_clk
  //Debe maperse a MAX10_CLK1_50 o MAX10_CLK2_50
  //Y el rst (debe mapearse a Key0)
  input rst, fpga_clk;
  //La cuenta maxima (SW9-SW0)
  input [9:0] cuenta_maxima;
  //Las salidas a los displays (HEX3-HEX0)
  output wire  [7:0] disp0, disp1, disp2, disp3;

  //Se declaran interconexiones entre modulos
  wire count_clk;
  wire [3:0] unidades, decenas, centenas, miles;

  //Se instancia el modulo slow_clk
  //Y se conectan las entradas y salidas apropiadas
  slow_clk slow_clk_inst (.fpga_clk(fpga_clk),
           .count_clk(count_clk));

  //Se instancia el modulo contador
  //Y se conectan las entradas y salidas apropiadas
  contador contador_inst (.rst(rst),
           .cuenta_maxima(cuenta_maxima),
           .count_clk(count_clk), .unidades(unidades),
           .decenas(decenas), .centenas(centenas),
           .miles(miles));

  //Se instancian los codificadores a 7seg
  //Se instancia 4, uno por cifra
  //Y se conectan las entradas y salidas apropiadas
  codificador_display codif0 (.bin(unidades),
                              .seg(disp0));
  codificador_display codif1 (.bin(decenas),
                              .seg(disp1));
  codificador_display codif2 (.bin(centenas),
                              .seg(disp2));
  codificador_display codif3 (.bin(miles),
                              .seg(disp3));
endmodule
```


DE10-Lite
rst
Unid.
Millar
Centenas
Decenas
Unidades
Cuenta Máxima

Archivo de asignación de pines






```
#CLK AND RST
set_location_assignment PIN_B8 -to rst
set_location_assignment PIN_P11 -to fpga_clk

#HEX0
set_location_assignment PIN_C14 -to disp0[0]
set_location_assignment PIN_E15 -to disp0[1]
set_location_assignment PIN_C15 -to disp0[2]
set_location_assignment PIN_C16 -to disp0[3]
set_location_assignment PIN_E16 -to disp0[4]
set_location_assignment PIN_D17 -to disp0[5]
set_location_assignment PIN_C17 -to disp0[6]
set_location_assignment PIN_D15 -to disp0[7]

...
```

Trabajo en Clase – Temporizador