

# Periféricos I

---

## E/S básicas



**Lucas Leiva, Elías Todorovich – INTIA**  
**{lleiva, etodorov}@exa.unicen.edu.ar**  
**UNICEN – Septiembre 2012**

## Objetivos

---

- Implementar controladores básicos de entrada/salida con lógica programable
  - Switches
  - Botones
  - Leds
  - Displays de 7 segmentos
- Comprender el flujo de diseño utilizando herramientas EDA

# Introducción

## ¿Por qué usamos periféricos?

Rta: porque todo sistema embebido necesita comunicarse con el exterior

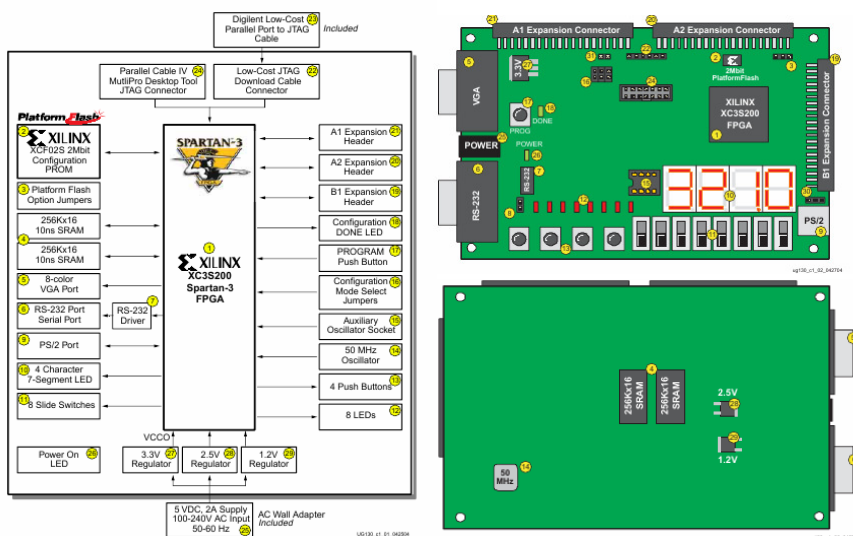
## ¿Cómo se usan?

Rta: descrito en la hoja de datos del periférico

## ¿Qué pasos son necesarios para su uso?

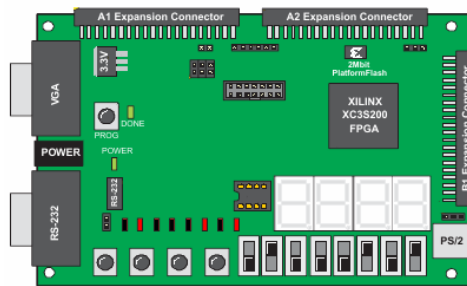
- Determinar E/S
- Analizar señales de E/S (tiempos, frecuencias, sincronizaciones, estado activo, etc.)
- Verificar mapeo de puertos FPGA – periférico
- Implementar controlador

# Spartan 3 Starter Kit



## Problema 1: "Encender leds"

*"Implementar un diseño que presente el estado de los switches en los leds"*



Periféricos I

5

## Leds y switches

### Switches

■ Up: 1, Down: 0

Switch	SW7	SW6	SW5	SW4	SW3	SW2	SW1	SW0
FPGA Pin	K13	K14	J13	J14	H13	H14	G12	F12

### Leds

■ Encendido: 1,  
apagado: 0

Led	LD7	LD6	LD5	LD4	LD3	LD2	LD1	LD0
FPGA Pin	P11	P12	N12	P13	N14	L12	P14	K12

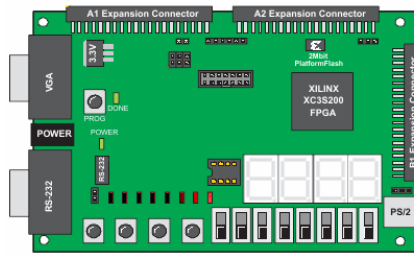
Periféricos I

6

## Problema 2: "Divisiones de frecuencia"

---

*Implementar un diseño que permita encender:*



- LD0 = 100 Hz
- LD1 = 20 Hz
- LD2 = 5 Hz
- LD3 = 1 Hz

## Consideraciones

---

- Reset
  - Botón pulsado: 1
  - BTN3 FPGA Pin: L14
- Clock
  - Frecuencia 50 MHz (período 20 ns)
  - Clk FPGA Pin: T9

## Solución

---

- ❑ Implementar contador que cambie el estado del led al alcanzar el valor requerido.
- ❑ Ej: led 0 = 100 Hz
  - período 10 ms o 10.000.000 ns
  - $10.000.000 \text{ ns} / 2 = 5.000.000 \text{ ns}$ 
    - ❑ led activo e inactivo (50% del ciclo)
  - $5.000.000 \text{ ns} / 20 \text{ ns} = 250.000 \text{ ciclos de clk}$

## Solución en VHDL

---

```
signal led0_counter: std_logic_vector(17 downto 0);
signal led0: std_logic;

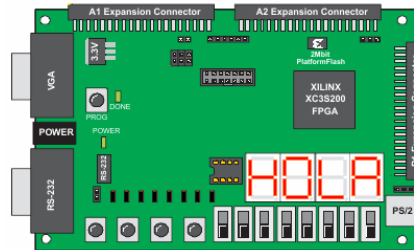
begin

led0_toggle: process (clk, reset)
begin
  if reset = '0' then
    led0_counter <= (others => '0');
    led0 <= '0';
  elsif rising_edge(clk) then
    if led0_counter = x"3D090" then -- x"3D090" = 250000
      led0_counter <= (others => '0');
      led0 <= not led0;
    else
      led0_counter <= led0_counter + 1;
    end if;
  end if;
end process;

leds(0) <= led0;
```

## Problema 3: "Hola Mundo"

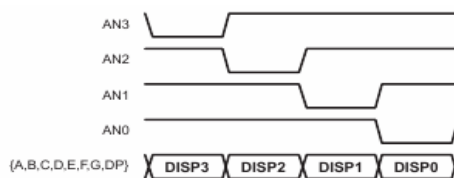
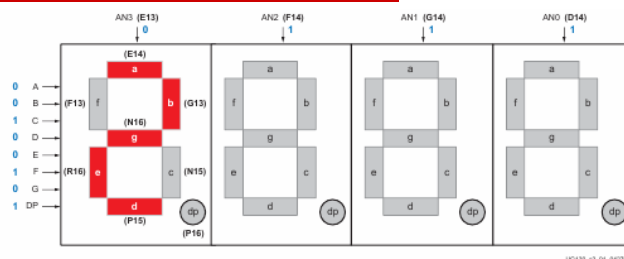
"Implementar un diseño que presente la palabra "HOLA" en los displays de 7 segmentos"



Periféricos I

11

## Display de 7 segmentos



Activo: 0  
Inactivo: 1

Periféricos I

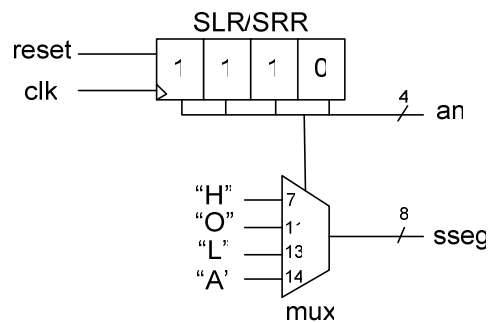
12

# Configuración

Segmento	FPGA Pin
A	E14
B	G13
C	N15
D	P15
E	R16
F	F13
G	N16
DP	P16

Anodo	FPGA Pin
AN0	D14
AN1	G14
AN2	F14
AN3	E13

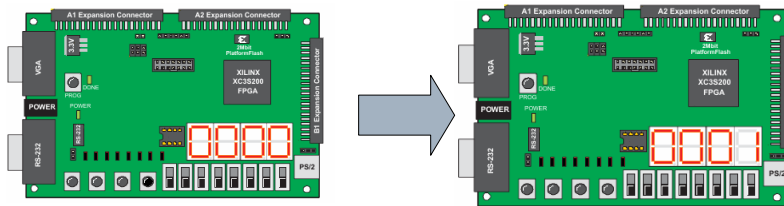
# Solución



- ❑ Registro desplazamiento inicializado en "1110"
- ❑ Multiplexor con las entradas "H", "O", "L" y "A"

## Problema 4: "Contador de pulsos"

*"Implementar un diseño que visualice en los displays de 7 segmento el número de pulsaciones (en hexadecimal) realizadas en el botón 0"*

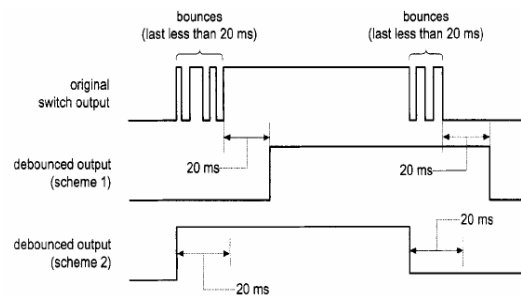


Periféricos I

15

## Rebotes del pulsador

- ❑ Todas las señales de entrada poseen rebote
- ❑ Los botones poseen un rebote de entre 15 y 20 ms
- ❑ Genera pulsos que incrementarán el contador



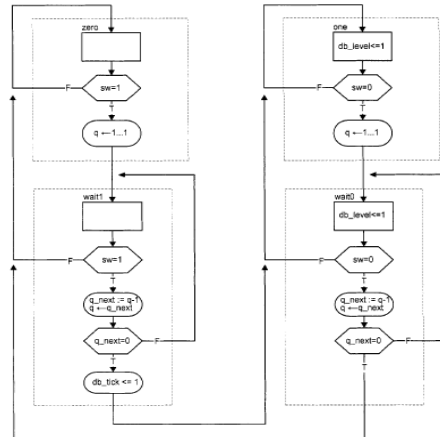
Periféricos I

16



## Solución anti-rebote (I)

- Implementación con una máquina de estados



Periféricos I

17

## Solución antirebote (II)

- Si los rebotes se producen en 20 ms
  - Muestrear la señal de entrada en 4 intervalos (5 ms)
  - Si todas las muestras fueron 0, entonces el dato es 0
  - Si todas las muestras fueron 1, entonces el dato es 1

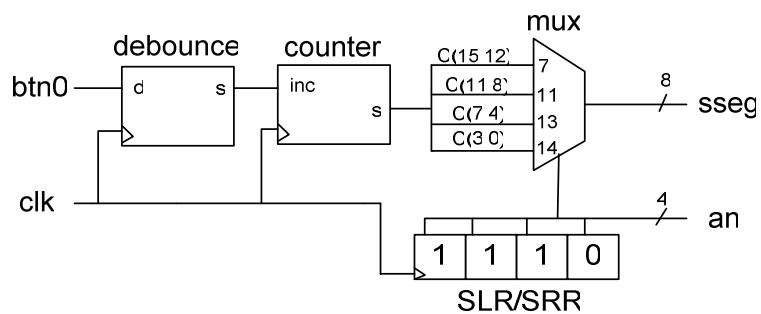
Periféricos I

18

# Implementación de anti-rebote

```
debounce:process(clk,rst)
  variable deb_buf : std_logic_vector(3 downto 0);
begin
  if rst = '1' then
    deb_buf := "0000";
    btn_deb <= '0';
    cycle_counter <= (others => '0');
  elsif rising_edge(clk) then
    if cycle_counter = x"3D090" then -- alcanzo 250000 ciclos = 5ms
      if deb_buf = "0000" then
        btn_deb <= '0'; -- hubo 4 valores en 0 => el dato es 0
      elsif deb_buf = "1111" then
        btn_deb <= '1'; -- hubo 4 valores en 1 => el dato es 1
      end if;
      deb_buf := deb_buf(2 downto 0) & rx; -- actualiza el buffer
      cycle_counter <= (others => '0'); -- reset del contador de 5ms
    else
      cycle_counter = cycle_counter + 1;
    end if;
  end if;
end process;
```

# Solución al problema 4



## Problemas adicionales

---

- Realizar las modificaciones que permitan presentar el contenido del contador de pulsos en decimal
- Implementar una calculadora, que permita operar los datos ingresados desde los switches, con las siguientes funciones
  - BTN0: carga el dato ingresado en los switches
  - BTN1: suma
  - BTN2: resta
  - BTN3: clear