

# TRABAJO PRÁCTICO FINAL MyS

**AUTOR:**  
**ING. JOSÉ ROBERTO CASTRO DELGADO**

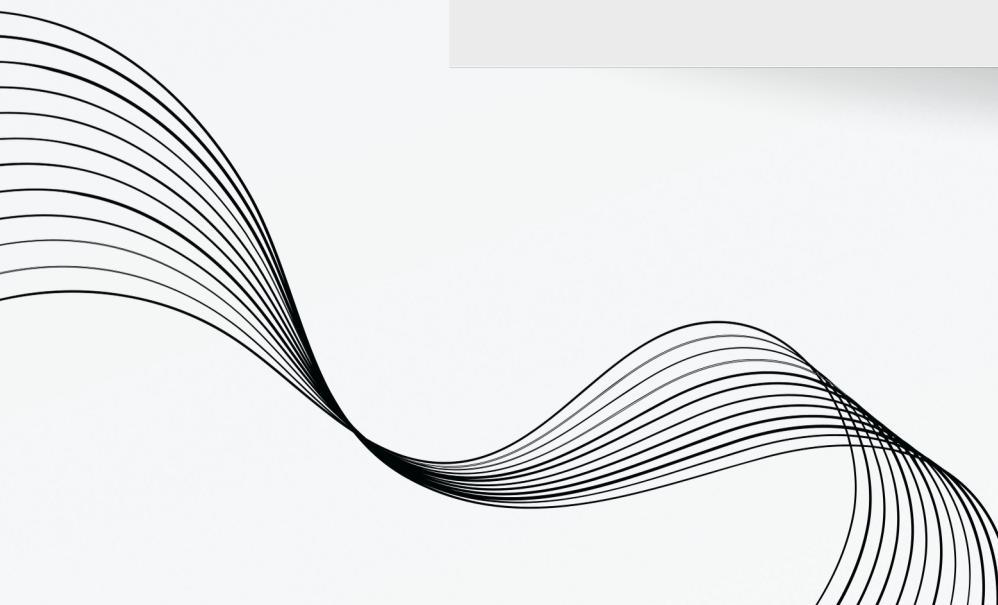
# INTRODUCCIÓN



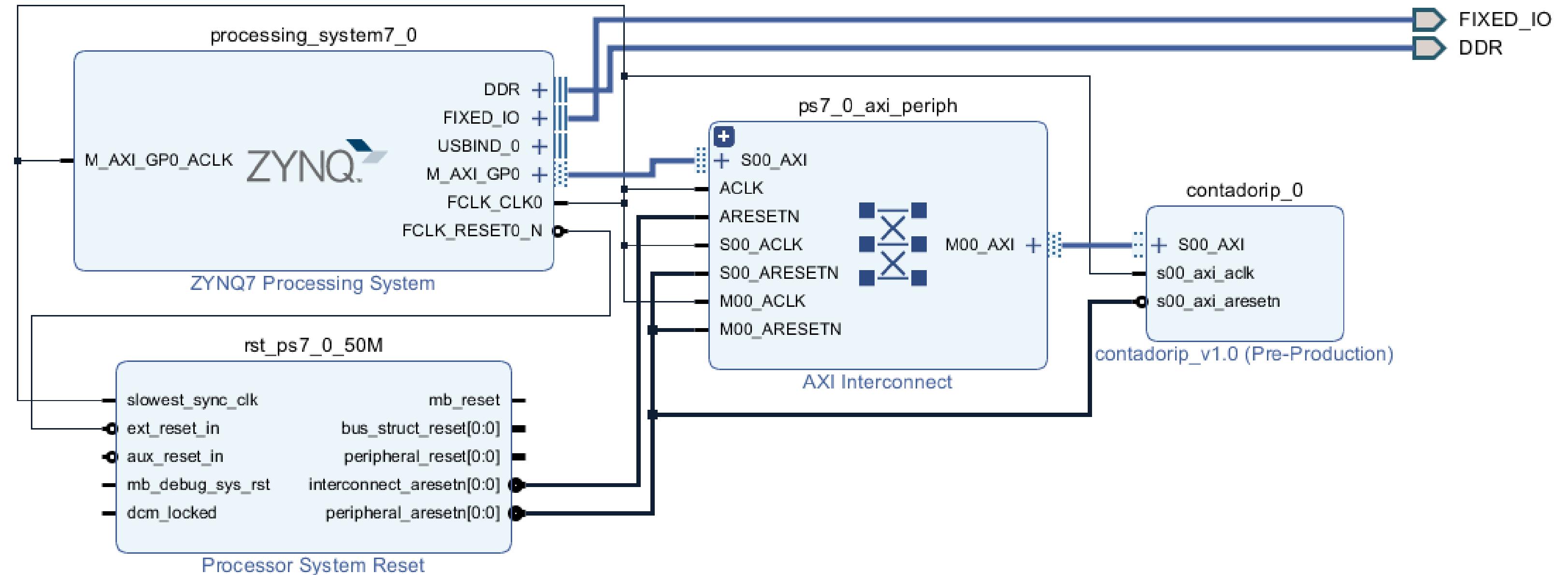
- Implementar un contador ascendente/descendente en VHDL.
- Simular el comportamiento del contador.



- Sintetizar el diseño para una FPGA.
- Establecer la comunicación con un microprocesador mediante una aplicación en C.



# DIAGRAMA DE BLOQUES



#### Entidad contador\_asc\_desc:

- Tiene cuatro señales: clk, reset, control (0 para ascendente, 1 para descendente) y count\_out (salida de 4 bits).

#### Arquitectura Behavioral:

- Se declara una señal interna count que almacena el valor del contador.

#### Proceso principal:

- Si reset es activado, el contador se reinicia a 0000.
- En cada flanco ascendente de clk, si control es 0, el contador aumenta; si es 1, el contador disminuye.

#### Salida del contador:

- El valor de count se asigna a count\_out, que es la salida del contador.

```
-- contador_asc_desc.vhd
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;

entity contador_asc_desc is
    Port (
        clk : in STD_LOGIC;
        reset : in STD_LOGIC;
        control : in STD_LOGIC; -- '0' para ascendente, '1' para descendente
        count_out : out STD_LOGIC_VECTOR(3 downto 0)
    );
end contador_asc_desc;

architecture Behavioral of contador_asc_desc is
    signal count : STD_LOGIC_VECTOR(3 downto 0) := "0000";
begin
    process(clk, reset)
    begin
        if reset = '1' then
            count <= "0000";
        elsif rising_edge(clk) then
            if control = '0' then
                count <= count + 1;
            else
                count <= count - 1;
            end if;
        end if;
    end process;

    count_out <= count;
end Behavioral;
```

- **Bibliotecas:** Se incluyen las necesarias para la configuración y control de la FPGA (xparameters.h, xil\_io.h, contadorip.h).
- **Direcciones y offsets:** Se definen las direcciones base y los registros del contador (reset, control, count\_out).
- **Estructura Contador\_Config:** Contiene la dirección base, estado de reset y modo de operación (ascendente/descendente).
- **Función Contador\_Init:** Inicializa la estructura con la dirección base, activa el reset y configura el contador en modo ascendente.

```

#include "xparameters.h"
#include "xil_io.h"
#include "contadorip.h"

#define CONTADOR_BASE_ADDR XPAR_CONTADORIP_0_S00_AXI_BASEADDR
#define REG_RESET_OFFSET 0x00
#define REG_CONTROL_OFFSET 0x04
#define REG_COUNT_OUT_OFFSET 0x08

// Configuración y estado del contador
typedef struct {
    int32_t base_addr;
    int32_t reset;           // Estado de reset (0: no activo, 1: activo)
    int32_t control;         // Modo del contador (0: ascendente, 1: descendente)
} Contador_Config;

void Contador_Init(Contador_Config *contador, int32_t base_addr) {
    contador->base_addr = base_addr;
    contador->reset = 0x01;
    contador->control = 0x00; // Modo predeterminado: ascendente
}

```

- Contador\_SetReset: Configura el estado de reset del contador y lo escribe en el registro correspondiente de la FPGA.
- Contador\_SetControl: Establece el modo del contador (ascendente o descendente) y lo escribe en el registro de control de la FPGA.
- Contador\_ReadCount: Lee el valor actual del contador desde la FPGA y lo muestra, limitándose a los 4 bits menos significativos.

```

void Contador_SetReset(Contador_Config *contador, int32_t reset) {
    contador->reset = reset;
    Xil_Out32(contador->base_addr + REG_RESET_OFFSET, contador->reset);
    xil_printf("Reset set to: %d\r\n", contador->reset);
}

void Contador_SetControl(Contador_Config *contador, int32_t control) {
    contador->control = control;
    Xil_Out32(contador->base_addr + REG_CONTROL_OFFSET, contador->control);
    xil_printf("Control mode set to: %s\r\n", control == 0 ? "Ascending" : "Descending");
}

int32_t Contador_ReadCount(Contador_Config *contador) {
    int32_t count = Xil_In32(contador->base_addr + REG_COUNT_OUT_OFFSET);
    xil_printf("Current count value: 0x%02X\r\n", count & 0xF); // Solo los 4 bits
    return count & 0xF;
}

```

- Inicialización y configuración:
  - Se inicializa el contador con la dirección base.
  - Se desactiva el reset y se configura el contador en modo ascendente.
- Bucle principal:
  - Lee y muestra el valor del contador en cada ciclo.
  - Simula un retardo antes de cambiar el modo del contador (ascendente/descendente).
  - Cambio de modo:
  - Alterna entre los modos ascendente y descendente de manera continua.

```
int main(void) {
    xil_printf("-- Inicio del Contador IP Core --\r\n");

    Contador_Config contador;

    // Inicializar el contador
    Contador_Init(&contador, CONTADOR_BASE_ADDR);

    // Configurar el contador
    Contador_SetReset(&contador, 0x00); // Desactiva reset
    Contador_SetControl(&contador, 0x00); // Configura modo ascendente

    while (1) {
        // Leer y mostrar el valor del contador
        Contador_ReadCount(&contador);

        // Simular cambio de modo cada cierto tiempo
        for (volatile int i = 0; i < 9999999; i++); // Retardo

        Contador_SetControl(&contador, contador.control ^ 0x01); // Cambiar modo (ascendente)
    }

    xil_printf("-- Fin del Contador IP Core --\r\n");
    return 0;
}
```

# RECURSOS POR MÓDULO

| Nombre               | Slice LUTs | Slice Registers | Bonded IOPADs | BUFGCTRL |
|----------------------|------------|-----------------|---------------|----------|
| design_1_wrapper     | 608        | 809             | 130           | 1        |
| design_1_i           | 608        | 809             | 0             | 1        |
| contadorip_0         | 57         | 169             | 0             | 0        |
| processing_system7_0 | 112        | 0               | 0             | 0        |
| ps7_0_axi_periph     | 420        | 600             | 0             | 0        |
| rst_ps7_0_50M        | 19         | 40              | 0             | 0        |

# RESUMEN DE UTILIZACION DE RECURSOS

| Recurso | Utilización | Disponible | Porcentaje de Utilización |
|---------|-------------|------------|---------------------------|
| LUT     | 608         | 17600      | 3.45%                     |
| LUTRAM  | 66          | 6000       | 1.10%                     |
| FF      | 809         | 35200      | 2.30%                     |

# SIMULACIÓN



FACULTAD DE INGENIERÍA

**¡GRACIAS!**