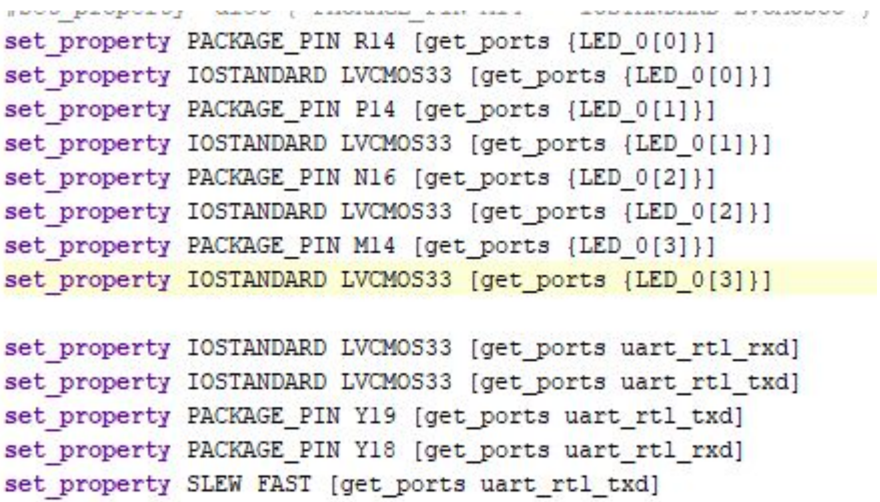
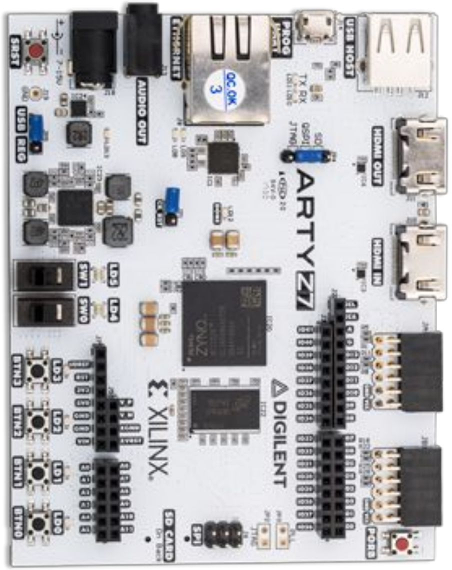
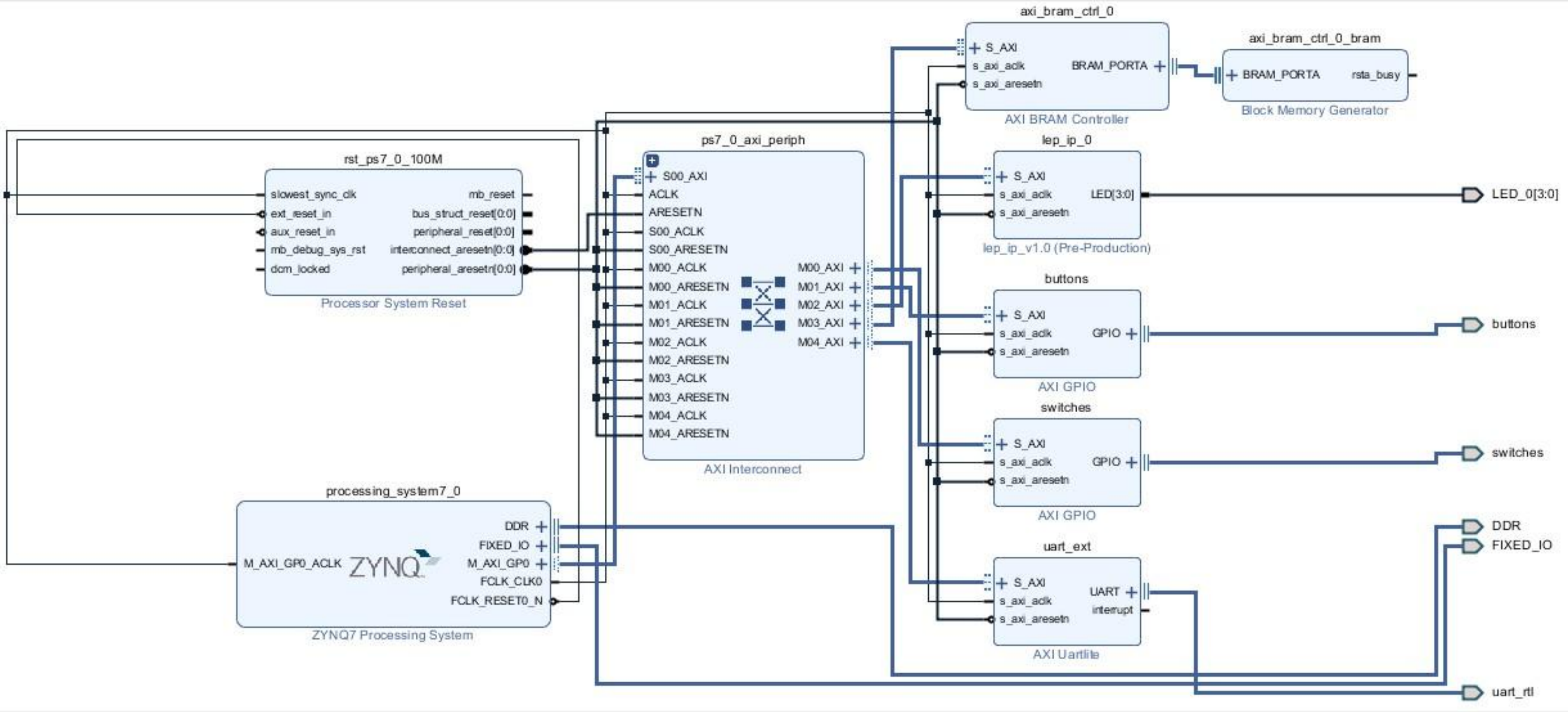
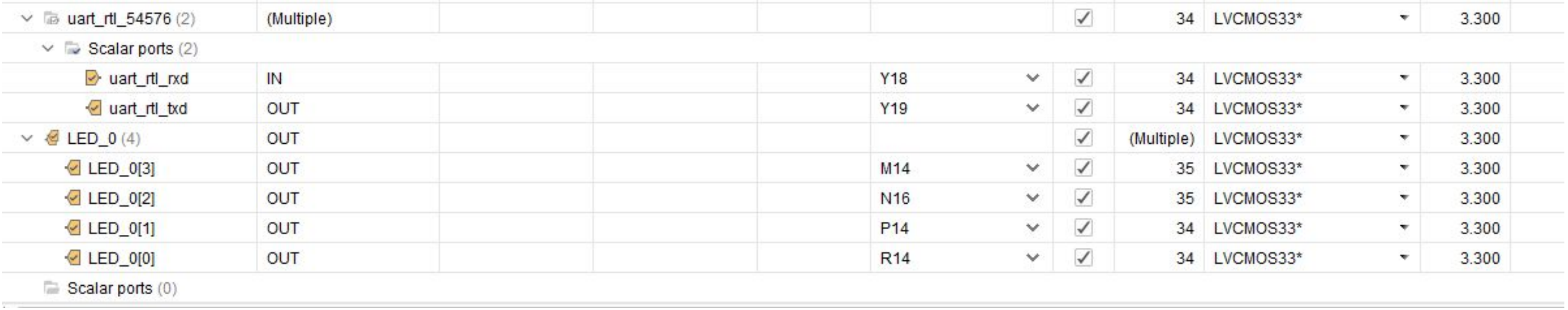
Contador ascendente y descendente por uart.



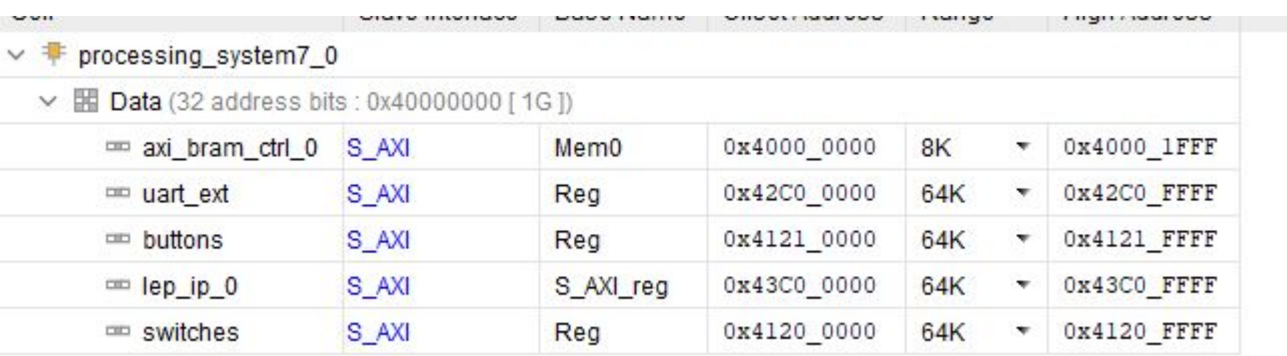
```
set_property PACKAGE_PIN R14 [get_ports {LED_0[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {LED_0[0]}]
set_property PACKAGE_PIN P14 [get_ports {LED_0[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {LED_0[1]}]
set_property PACKAGE_PIN N16 [get_ports {LED_0[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {LED_0[2]}]
set_property PACKAGE_PIN M14 [get_ports {LED_0[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {LED_0[3]}]

set_property IOSTANDARD LVCMOS33 [get_ports uart_rtl_rxd]
set_property IOSTANDARD LVCMOS33 [get_ports uart_rtl_txd]
set_property PACKAGE_PIN Y19 [get_ports uart_rtl_txd]
set_property PACKAGE_PIN Y18 [get_ports uart_rtl_rxd]
set_property SLEW FAST [get_ports uart_rtl_txd]
```




axi_bram_ctrl_0
S_AXI
s_axi_aclk
s_axi_aresetn
BRAM_PORTA
AXI BRAM Controller
axi_bram_ctrl_0_bram
BRAM_PORTA
rsta_busy
Block Memory Generator
rst_ps7_0_100M
slowest_sync_clk
ext_reset_in
aux_reset_in
mb_debug_sys_rst
dcm_locked
mb_reset
bus_struct_reset[0:0]
peripheral_reset[0:0]
interconnect_aresetn[0:0]
peripheral_aresetn[0:0]
Processor System Reset
ps7_0_axi_periph
S00_AXI
ACLK
ARESETN
S00_ACLK
S00_ARESETN
M00_ACLK
M00_ARESETN
M01_ACLK
M01_ARESETN
M02_ACLK
M02_ARESETN
M03_ACLK
M03_ARESETN
M04_ACLK
M04_ARESETN
M00_AXI
M01_AXI
M02_AXI
M03_AXI
M04_AXI
AXI Interconnect
lep_ip_0
S_AXI
s_axi_aclk
s_axi_aresetn
LED[3:0]
lep_ip_v1.0 (Pre-Production)
LED_0[3:0]
buttons
S_AXI
s_axi_aclk
s_axi_aresetn
GPIO
AXI GPIO
buttons
switches
S_AXI
s_axi_aclk
s_axi_aresetn
GPIO
AXI GPIO
switches
processing_system7_0
M_AXI_GP0_ACLK
ZYNQ
DDR
FIXED_IO
M_AXI_GP0
FCLK_CLK0
FCLK_RESET0_N
ZYNQ7 Processing System
DDR
FIXED_IO
uart_ext
S_AXI
s_axi_aclk
s_axi_aresetn
UART
interrupt
AXI Uartlite
uart_rtl

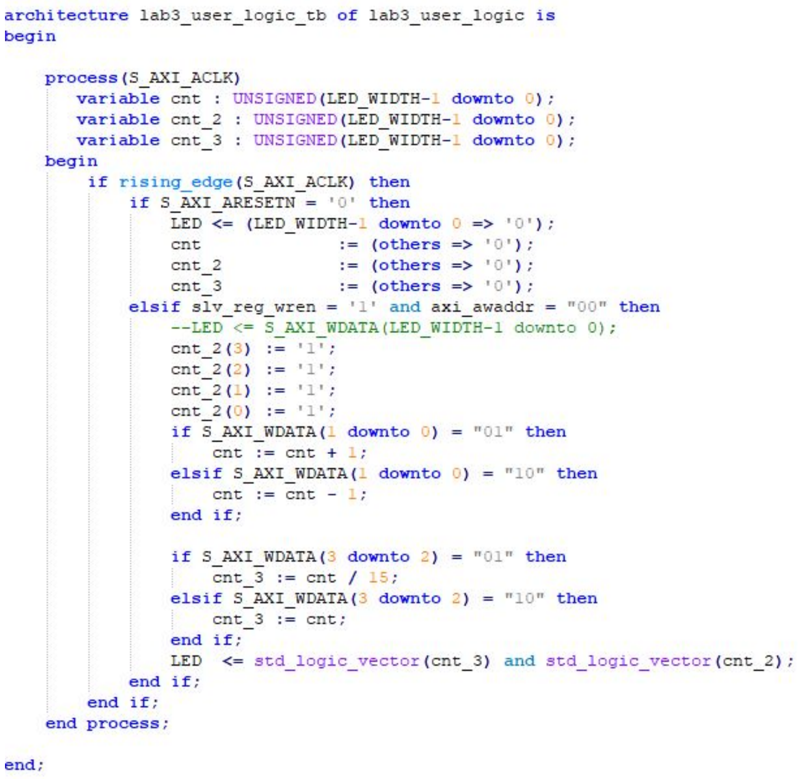


uart_rtl_54576 (2)	(Multiple)					☑	34	LVCMOS33*	3.300
Scalar ports (2)									
uart_rtl_rxd	IN				Y18	☑	34	LVCMOS33*	3.300
uart_rtl_txd	OUT				Y19	☑	34	LVCMOS33*	3.300
LED_0 (4)	OUT					☑	(Multiple)	LVCMOS33*	3.300
LED_0[3]	OUT				M14	☑	35	LVCMOS33*	3.300
LED_0[2]	OUT				N16	☑	35	LVCMOS33*	3.300
LED_0[1]	OUT				P14	☑	34	LVCMOS33*	3.300
LED_0[0]	OUT				R14	☑	34	LVCMOS33*	3.300
Scalar ports (0)									



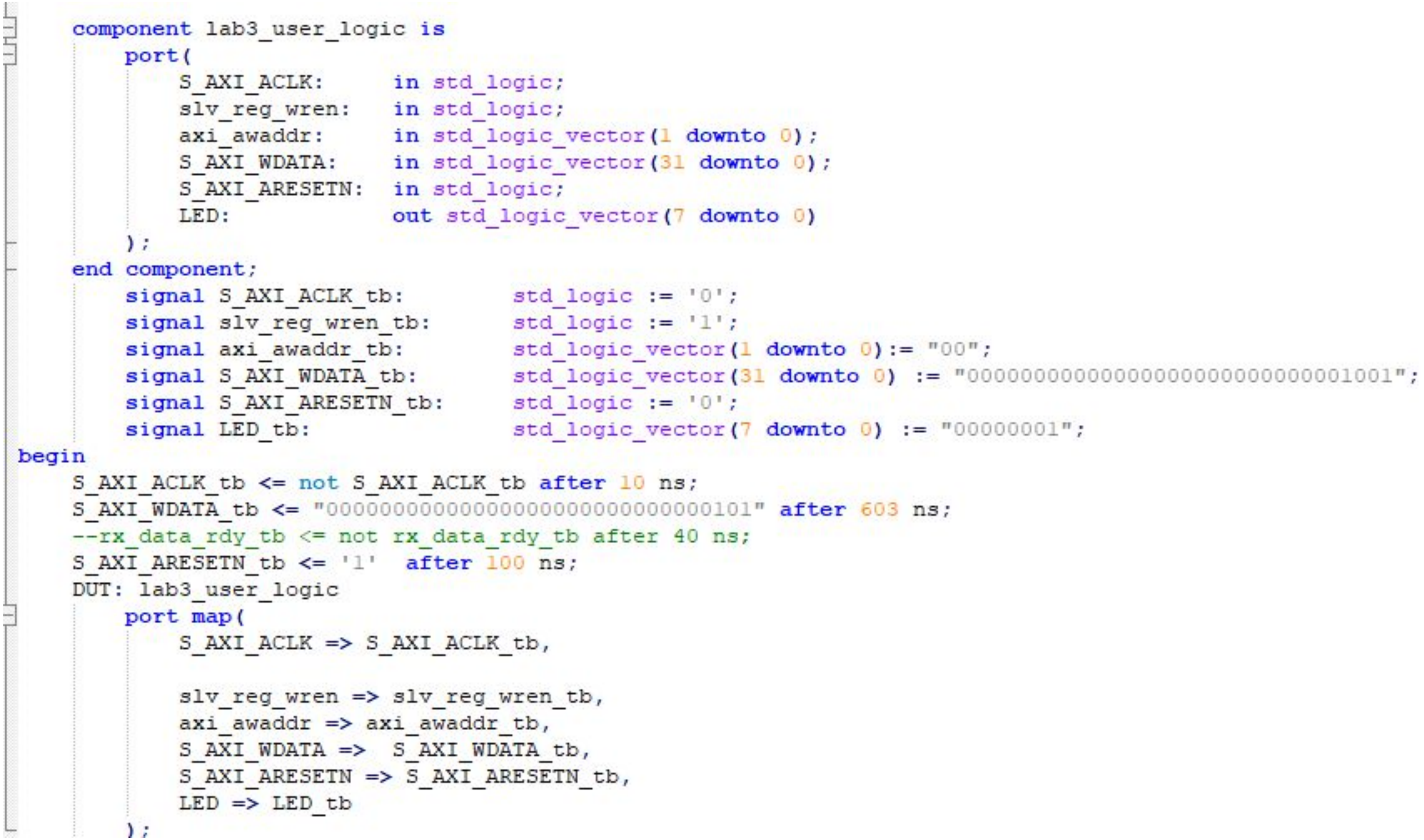
processing_system7_0					
Data (32 address bits : 0x40000000 [1G])					
axi_bram_ctrl_0	S_AXI	Mem0	0x4000_0000	8K	0x4000_1FFF
uart_ext	S_AXI	Reg	0x42C0_0000	64K	0x42C0_FFFF
buttons	S_AXI	Reg	0x4121_0000	64K	0x4121_FFFF
lep_ip_0	S_AXI	S_AXI_reg	0x43C0_0000	64K	0x43C0_FFFF
switches	S_AXI	Reg	0x4120_0000	64K	0x4120_FFFF

Codigo VHDL



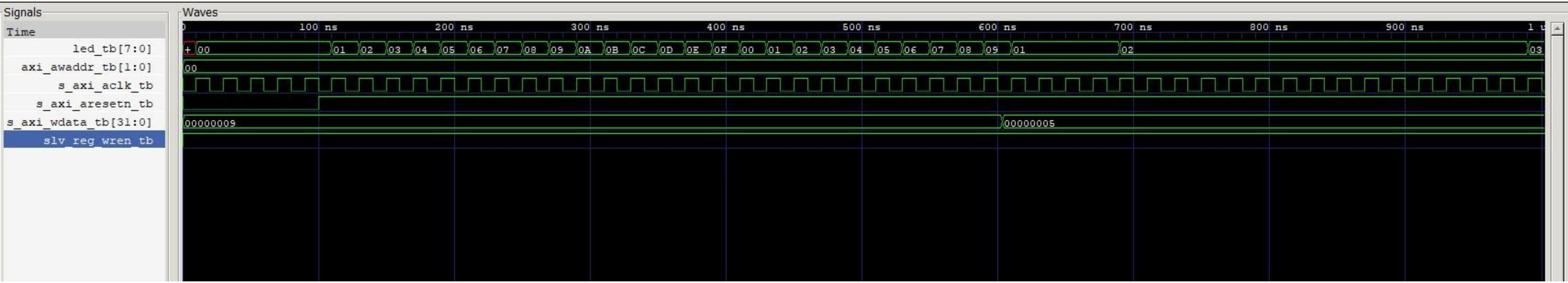
```vhdl
architecture lab3_user_logic_tb of lab3_user_logic is
begin

    process(S_AXI_ACLK)
       variable cnt : UNSIGNED(LED_WIDTH-1 downto 0);
       variable cnt_2 : UNSIGNED(LED_WIDTH-1 downto 0);
       variable cnt_3 : UNSIGNED(LED_WIDTH-1 downto 0);
    begin
        if rising_edge(S_AXI_ACLK) then
            if S_AXI_ARESETN = '0' then
                LED <= (LED_WIDTH-1 downto 0 => '0');
                cnt             := (others => '0');
                cnt_2           := (others => '0');
                cnt_3           := (others => '0');
            elsif slv_reg_wren = '1' and axi_awaddr = "00" then
                --LED <= S_AXI_WDATA(LED_WIDTH-1 downto 0);
                cnt_2(3)  := '1';
                cnt_2(2)  := '1';
                cnt_2(1)  := '1';
                cnt_2(0)  := '1';
                if S_AXI_WDATA(1 downto 0) = "01" then
                    cnt := cnt + 1;
                elsif S_AXI_WDATA(1 downto 0) = "10" then
                    cnt := cnt - 1;
                end if;

                if S_AXI_WDATA(3 downto 2) = "01" then
                    cnt_3 := cnt / 15;
                elsif S_AXI_WDATA(3 downto 2) = "10" then
                    cnt_3 := cnt;
                end if;
                LED  <= std_logic_vector(cnt_3) and std_logic_vector(cnt_2);
            end if;
        end if;
    end process;

end;
```

Registro de 32 bits				
	Contador		Led	
bits	Ascendente	Descendente	HL	LL
0	1	0	x	x
1	0	1	x	x
2	x	x	1	0
3	x	x	0	1
4-31	x	x	x	x

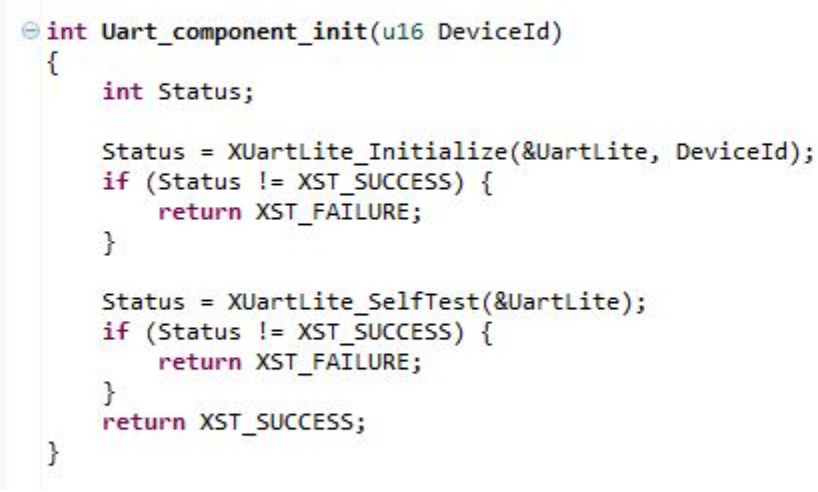


```vhdl
    component lab3_user_logic is
        port(
            S_AXI_ACLK:     in std_logic;
            slv_reg_wren:   in std_logic;
            axi_awaddr:     in std_logic_vector(1 downto 0);
            S_AXI_WDATA:    in std_logic_vector(31 downto 0);
            S_AXI_ARESETN:  in std_logic;
            LED:            out std_logic_vector(7 downto 0)
        );
    end component;
        signal S_AXI_ACLK_tb:        std_logic := '0';
        signal slv_reg_wren_tb:      std_logic := '1';
        signal axi_awaddr_tb:        std_logic_vector(1 downto 0):= "00";
        signal S_AXI_WDATA_tb:       std_logic_vector(31 downto 0)  := "00000000000000000000000000001001";
        signal S_AXI_ARESETN_tb:     std_logic := '0';
        signal LED_tb:               std_logic_vector(7 downto 0)  := "00000001";
begin
    S_AXI_ACLK_tb <= not S_AXI_ACLK_tb after 10 ns;
    S_AXI_WDATA_tb <= "00000000000000000000000000000101" after 603 ns;
    --rx_data_rdy_tb <= not rx_data_rdy_tb after 40 ns;
    S_AXI_ARESETN_tb <= '1'  after 100 ns;
    DUT: lab3_user_logic
        port map(
            S_AXI_ACLK => S_AXI_ACLK_tb,

            slv_reg_wren => slv_reg_wren_tb,
            axi_awaddr => axi_awaddr_tb,
            S_AXI_WDATA =>  S_AXI_WDATA_tb,
            S_AXI_ARESETN => S_AXI_ARESETN_tb,
            LED => LED_tb
        );
```

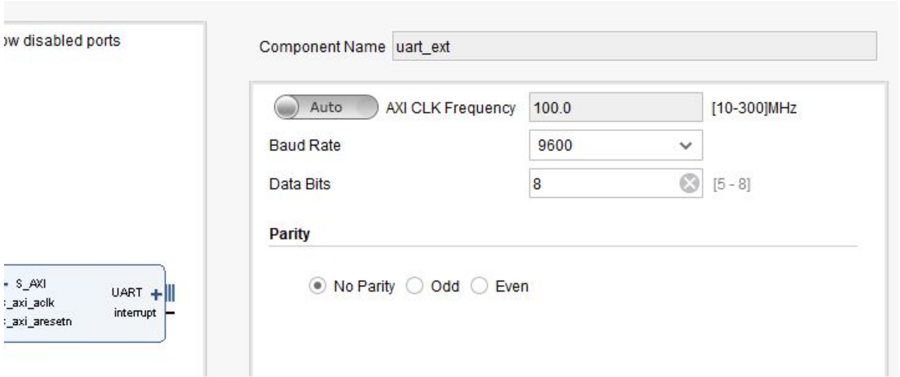
Simulación


Signals
Waves
Time
led_tb[7:0]
axi_awaddr_tb[1:0]
s_axi_aclk_tb
s_axi_aresetn_tb
s_axi_wdata_tb[31:0]
slv_reg_wren_tb
0
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns
800 ns
900 ns
1 u
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 00 01 02 03 04 05 06 07 08 09 01 02 03
00
00000009
00000005

Codigo en C.



```
int Uart_component_init(u16 DeviceId)
{
    int Status;

    Status = XUartLite_Initialize(&UartLite, DeviceId);
    if (Status != XST_SUCCESS) {
        return XST_FAILURE;
    }

    Status = XUartLite_SelfTest(&UartLite);
    if (Status != XST_SUCCESS) {
        return XST_FAILURE;
    }
    return XST_SUCCESS;
}
```



Component Name uart_ext

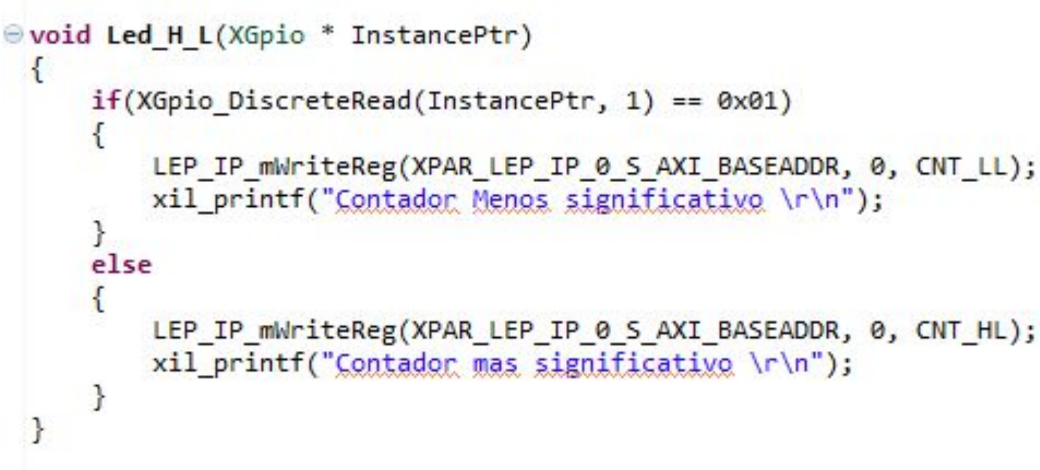
AXI CLK Frequency 100.0 [10-300]MHz

Baud Rate 9600

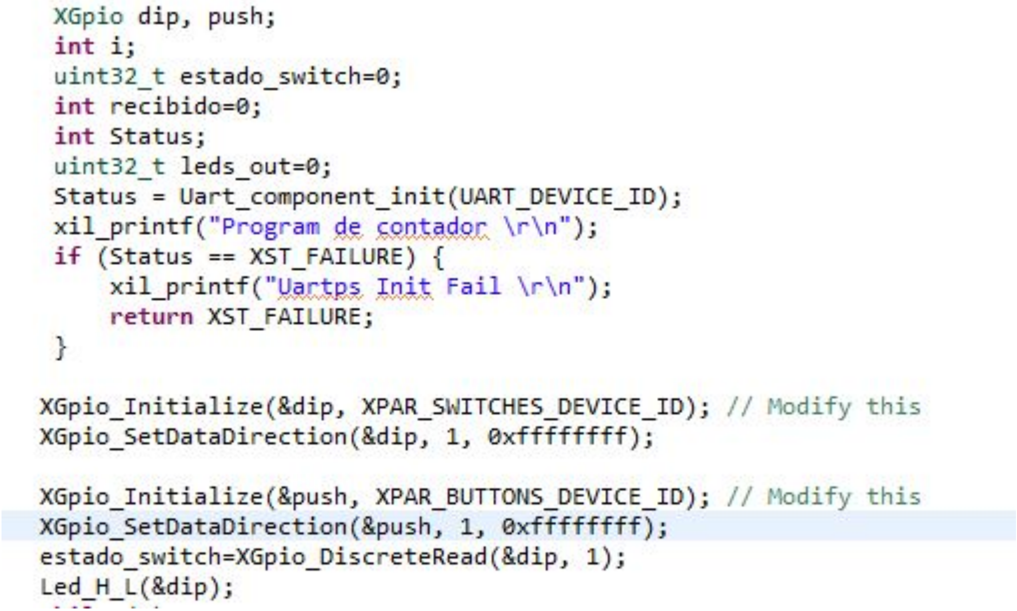
Data Bits 8 [5 - 8]

Parity

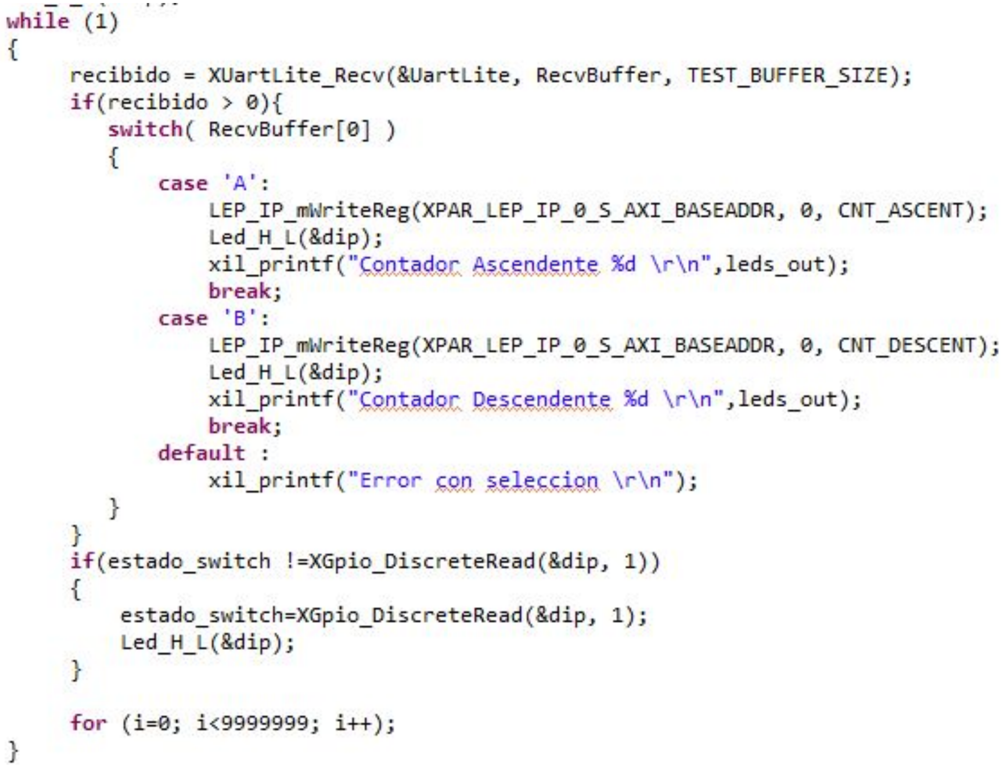
● No Parity ○ Odd ○ Even



```c
void Led_H_L(XGpio * InstancePtr)
{
    if(XGpio_DiscreteRead(InstancePtr, 1) == 0x01)
    {
        LEP_IP_mWriteReg(XPAR_LEP_IP_0_S_AXI_BASEADDR, 0, CNT_LL);
        xil_printf("Contador Menos significativo \r\n");
    }
    else
    {
        LEP_IP_mWriteReg(XPAR_LEP_IP_0_S_AXI_BASEADDR, 0, CNT_HL);
        xil_printf("Contador mas significativo \r\n");
    }
}
```



```
 XGpio dip, push;
 int i;
 uint32_t estado_switch=0;
 int recibido=0;
 int Status;
 uint32_t leds_out=0;
 Status = Uart_component_init(UART_DEVICE_ID);
 xil_printf("Program de contador \r\n");
 if (Status == XST_FAILURE) {
     xil_printf("Uartps Init Fail \r\n");
     return XST_FAILURE;
 }

XGpio_Initialize(&dip, XPAR_SWITCHES_DEVICE_ID); // Modify this
XGpio_SetDataDirection(&dip, 1, 0xffffffff);

XGpio_Initialize(&push, XPAR_BUTTONS_DEVICE_ID); // Modify this
XGpio_SetDataDirection(&push, 1, 0xffffffff);
estado_switch=XGpio_DiscreteRead(&dip, 1);
Led_H_L(&dip);
```



```c
while (1)
{
     recibido = XUartLite_Recv(&UartLite, RecvBuffer, TEST_BUFFER_SIZE);
     if(recibido > 0){
        switch( RecvBuffer[0] )
        {
            case 'A':
                LEP_IP_mWriteReg(XPAR_LEP_IP_0_S_AXI_BASEADDR, 0, CNT_ASCENT);
                Led_H_L(&dip);
                xil_printf("Contador Ascendente %d \r\n",leds_out);
                break;
            case 'B':
                LEP_IP_mWriteReg(XPAR_LEP_IP_0_S_AXI_BASEADDR, 0, CNT_DESCENT);
                Led_H_L(&dip);
                xil_printf("Contador Descendente %d \r\n",leds_out);
                break;
            default :
                xil_printf("Error con seleccion \r\n");
        }
     }
     if(estado_switch !=XGpio_DiscreteRead(&dip, 1))
     {
         estado_switch=XGpio_DiscreteRead(&dip, 1);
         Led_H_L(&dip);
     }

     for (i=0; i<9999999; i++);
}
```