Carrera de Especialización en
Sistemas Embebidos

Módulo SPI - VHDL

Ing. Guillermo Luis Castiglioni

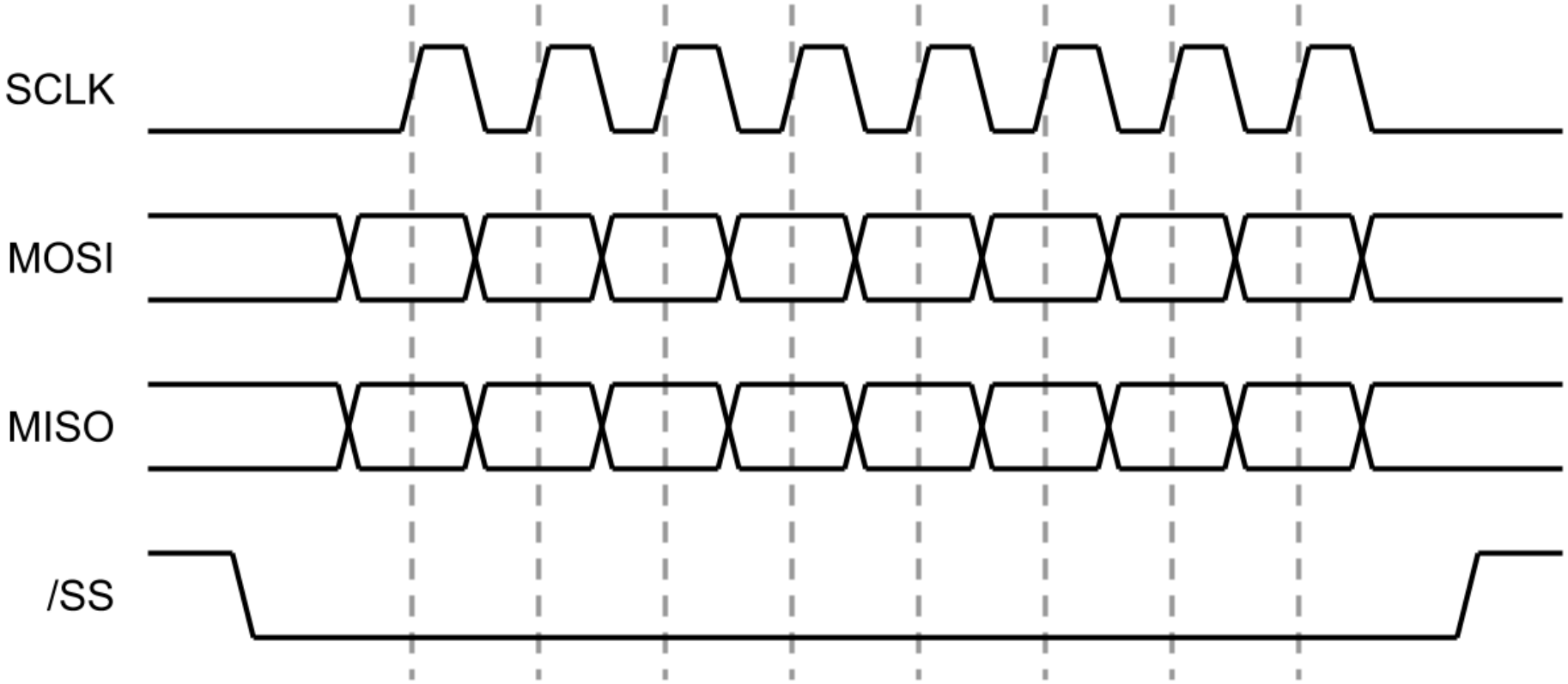
Herramientas

Quartus Prime Lite Edition - Intel

GHDL + GTKWake

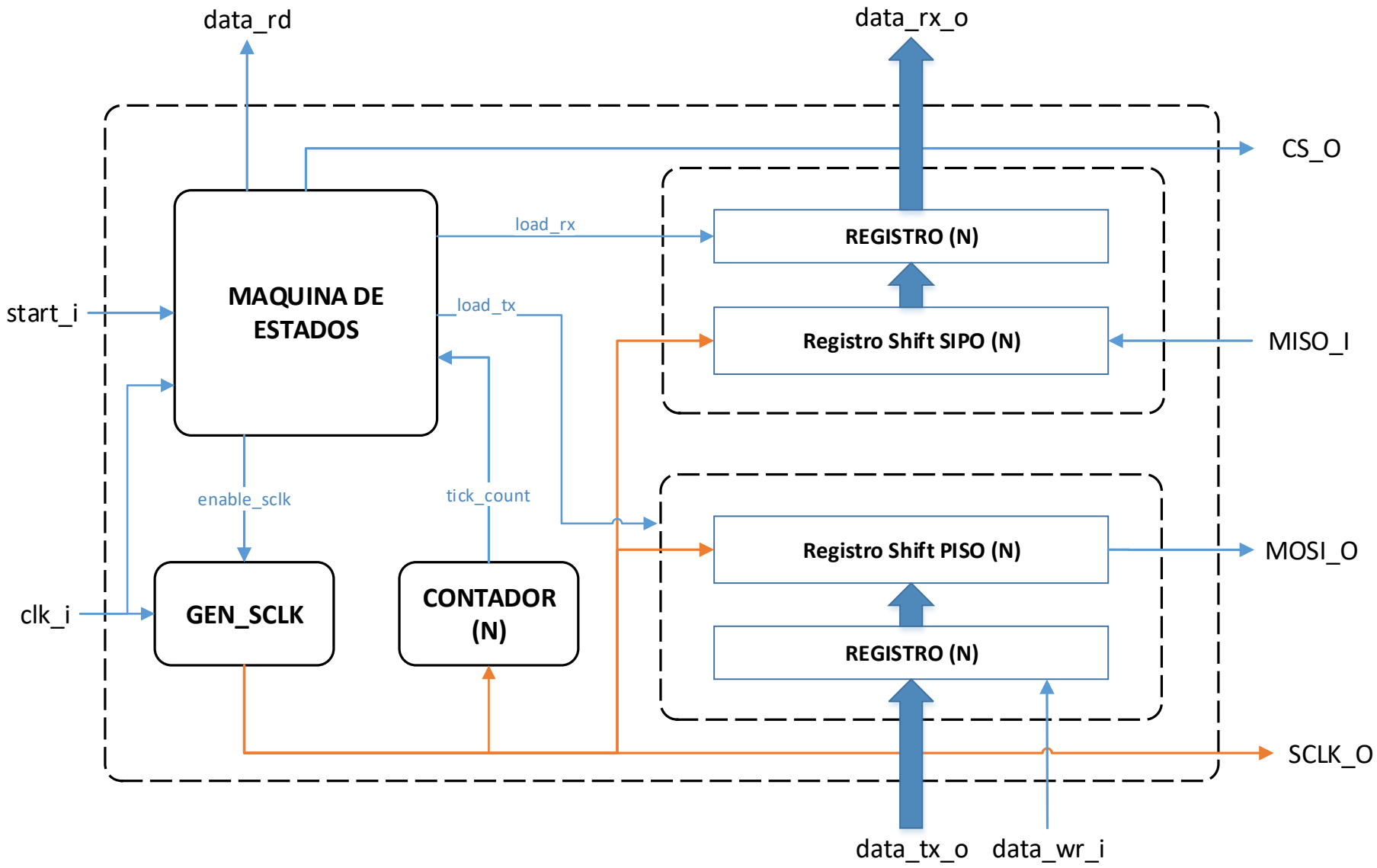
ISE 14.7 + ISIM

SPI – MODO 0



https://articles.saleae.com/logic-analyzers/spi-vs-i2c-protocol-differences-and-things-to-consider

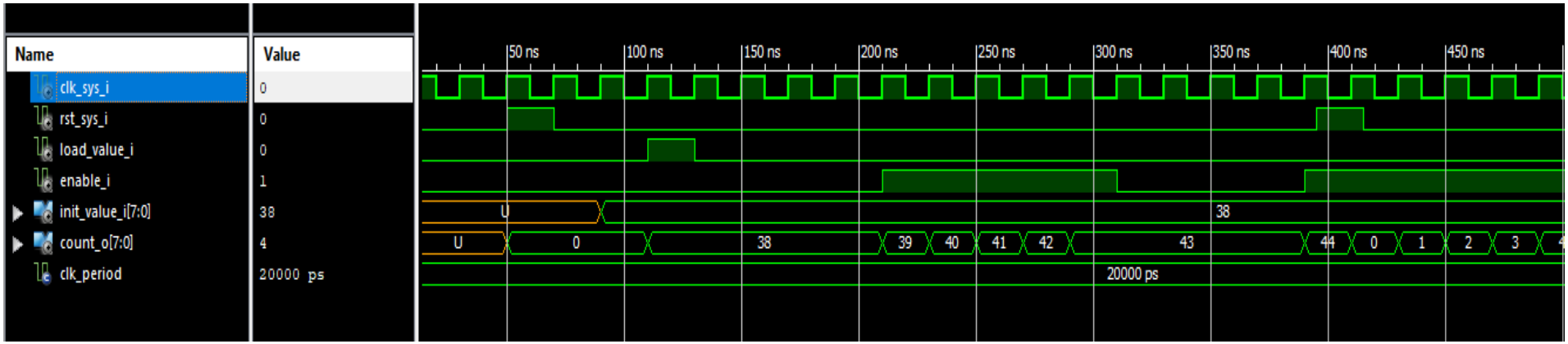
Arquitectura módulo Master SPI



Contador configurable

```vhdl
entity counter_N is
    generic(
            N: natural := 8
        );
    port (
            clk_sys_i    : in std_logic;
            rst_sys_i    : in std_logic;
            load_value_i : in std_logic;
            init_value_i : in std_logic_vector(N-1 downto 0);
            enable_i     : in std_logic;
            count_o      : out std_logic_vector(N-1 downto 0)
        );
end counter_N;
```

```vhdl
architecture behavioral of counter_N is
begin
    process(clk_sys_i, rst_sys_i)
        variable count_i : unsigned(N-1 downto 0);
    begin
        if (rising_edge(clk_sys_i)) then
            if (rst_sys_i ='1') then
                count_i := (others => '0');
            elsif (load_value_i = '1') then
                count_i := unsigned(init_value_i);
            elsif enable_i = '1' then
                count_i := count_i + 1;
            end if;
        end if;
        count_o <= std_logic_vector(count_i);
    end process;
end architecture;
```



Registro de desplazamiento PISO

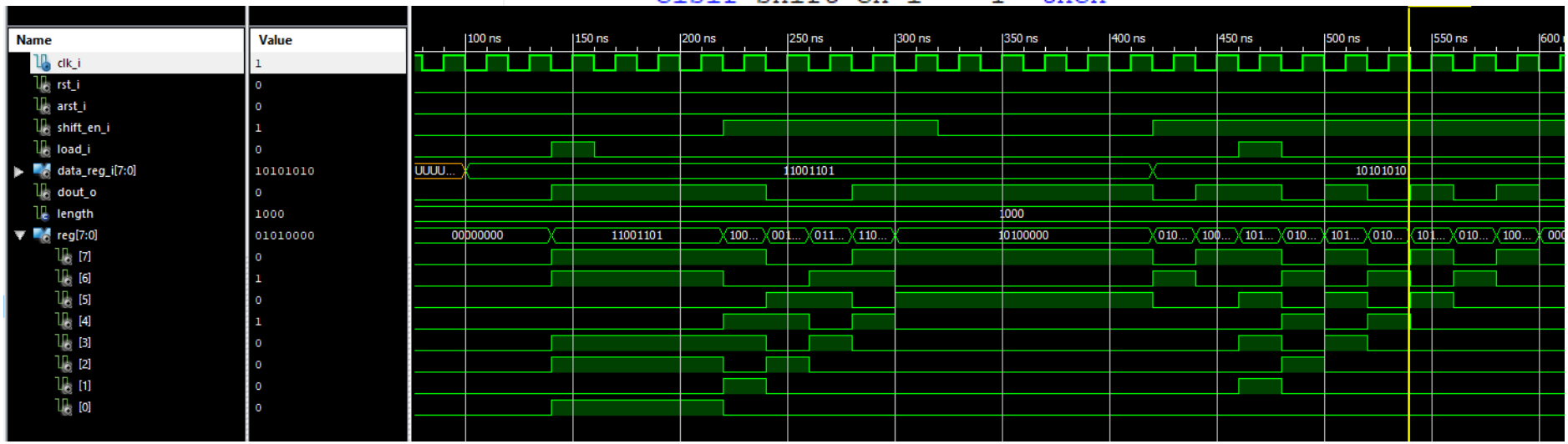
```
entity shift_reg_piso is
    generic(
        N: integer := 8
    );
    port(
        clk_i       : in  std_l
        rst_i       : in  std_l
        arst_i      : in  std_l
        shift_en_i  : in  std_l
        load_i      : in  std_l
        dout_o      : out std_l
        data_reg_i  : in  std_l
    );
end entity shift_reg_piso;
```

```
architecture behavioral of shift_reg_piso is

signal reg : std_logic_vector(N-1 downto 0) := (others => '0');

begin

    reg_process : process(clk_i , rst_i , arst_i) is
    begin
        if arst_i = '1' then
            reg <= (others => '0');
        elsif falling_edge(clk_i) then
            if rst_i = '1' then
                reg <= (others => '0');
            elsif load_i = '1' then
                reg <= data_reg_i;
            elsif shift_en_i = '1' then
```



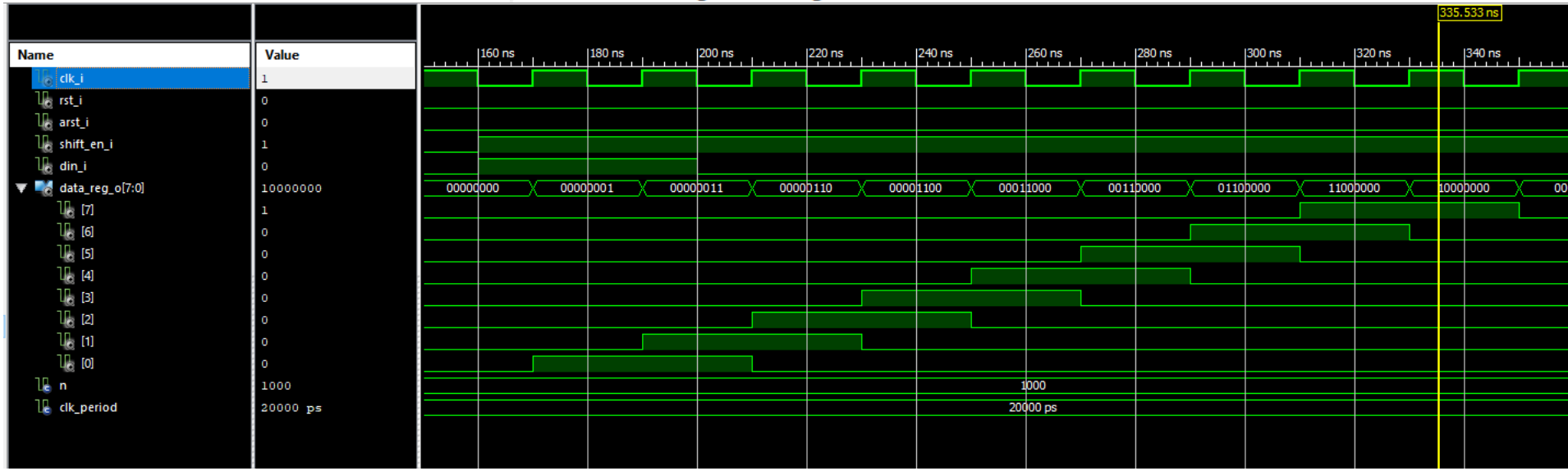
Registro de desplazamiento SIPO

```vhdl
entity shift_reg_sipo is
    generic(
        N: integer := 8
    );
    port(
        clk_i       : in  std_log
        rst_i       : in  std_log
        arst_i      : in  std_log
        shift_en_i  : in  std_log
        din_i       : in  std_log
        data_reg_o  : out std_log
    );
end entity shift_reg_sipo;
```

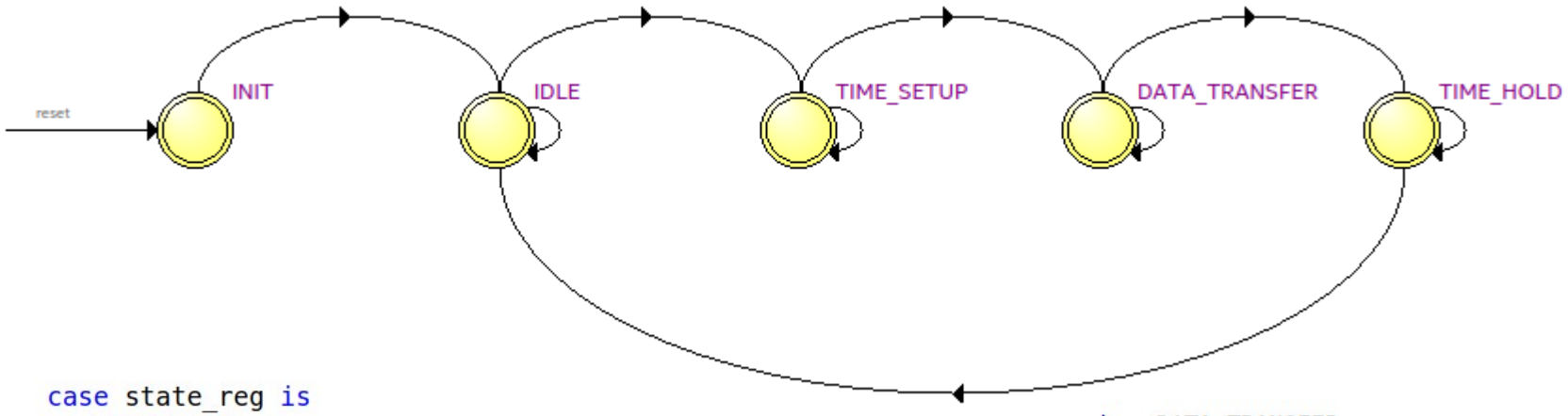
```vhdl
architecture behavioral of shift_reg_sipo is

signal reg : std_logic_vector(N-1 downto 0) := (others => '0');

begin
    reg_process : process(clk_i , rst_i , arst_i) is
    begin
        if arst_i = '1' then
            reg <= (others => '0');
        elsif rising_edge(clk_i) then
            if rst_i = '1' then
                reg <= (others => '0');
            elsif shift_en_i = '1' then
                reg <= reg(N-2 downto 0) & Din_i;
            else
                reg <= reg;
```



Máquina de estado




```
case state_reg is
   when INIT =>
      load_rx_s <= '1';
      rst_timer_s <= '1';
      load_count_s <= '1';
      state_next <= IDLE;

   when IDLE =>
      if start_i = '1' then
         state_next <= TIME_SETUP;
         load_tx_s <= '1';
         cs_next <= '0';
         rst_timer_s <= '1';
      end if;

   when TIME_SETUP =>
      enable_setup_s <= '1';
      if timeout_setup_s = '1' then
         state_next <= DATA_TRANSFER;
         rst_count_s <= '1';
      end if;
```

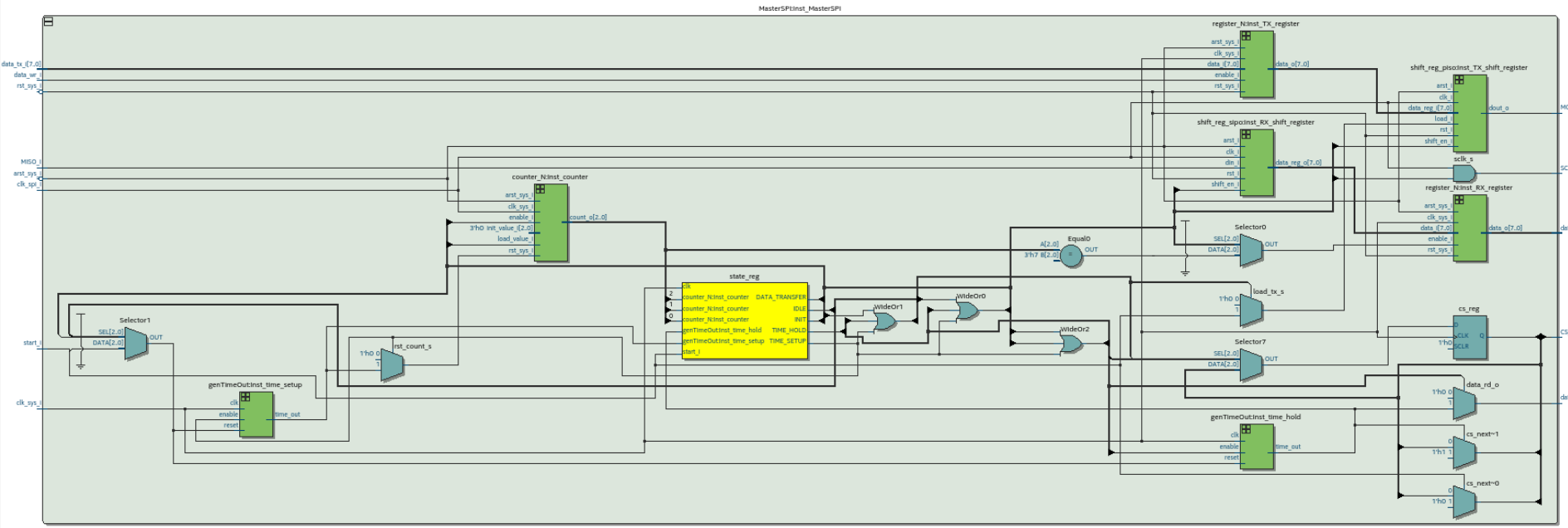
```
   when DATA_TRANSFER =>
      sclk_enable_s <= '1';
      shift_en_s <= '1';
      enable_count_s <= '1';
      if counter_value_s = MOD_EDGE_COUNT then
         load_rx_s <= '1';
         state_next <= TIME_HOLD;
      end if;

   when TIME_HOLD =>
      enable_hold_s <= '1';
      if timeout_hold_s = '1' then
         cs_next <= '1';
         data_rd_o <= '1';
         state_next <= IDLE;
      end if;

   when others =>
      state_next <= IDLE;

end case;
end process control;
```

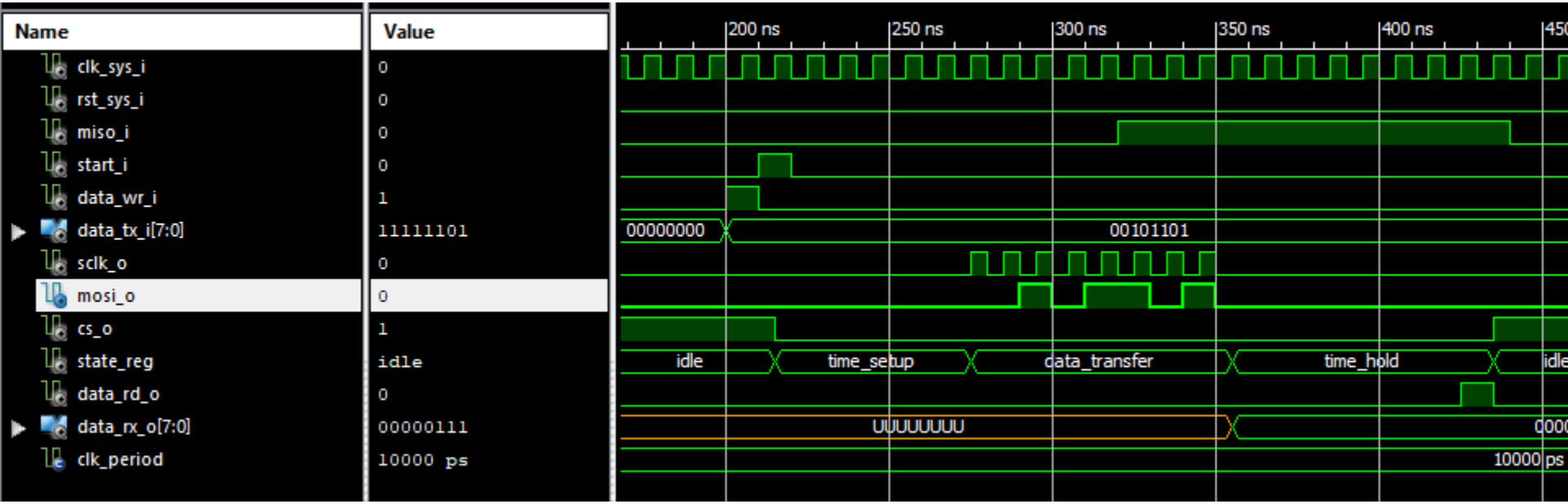
Diagrama RTL



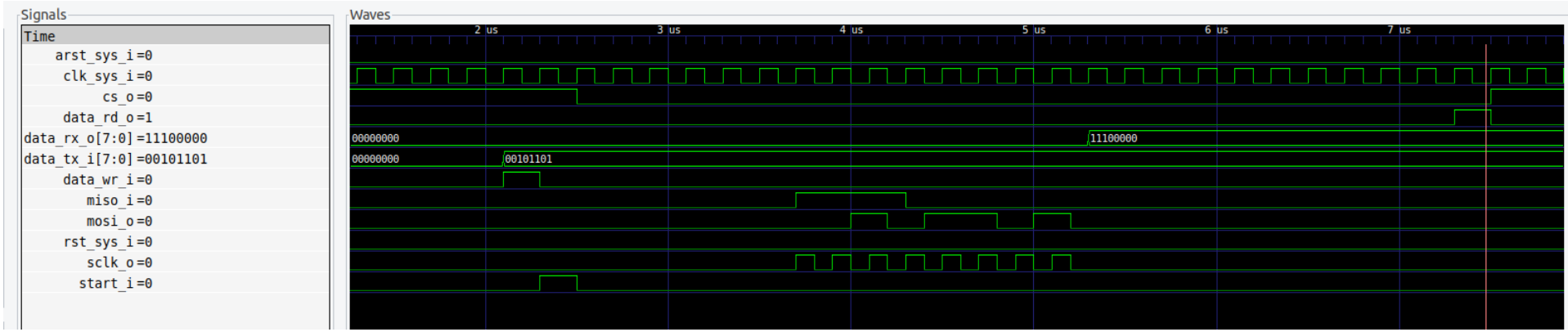
Quartus Prime Lite Edition - Intel

Simulación módulo completo

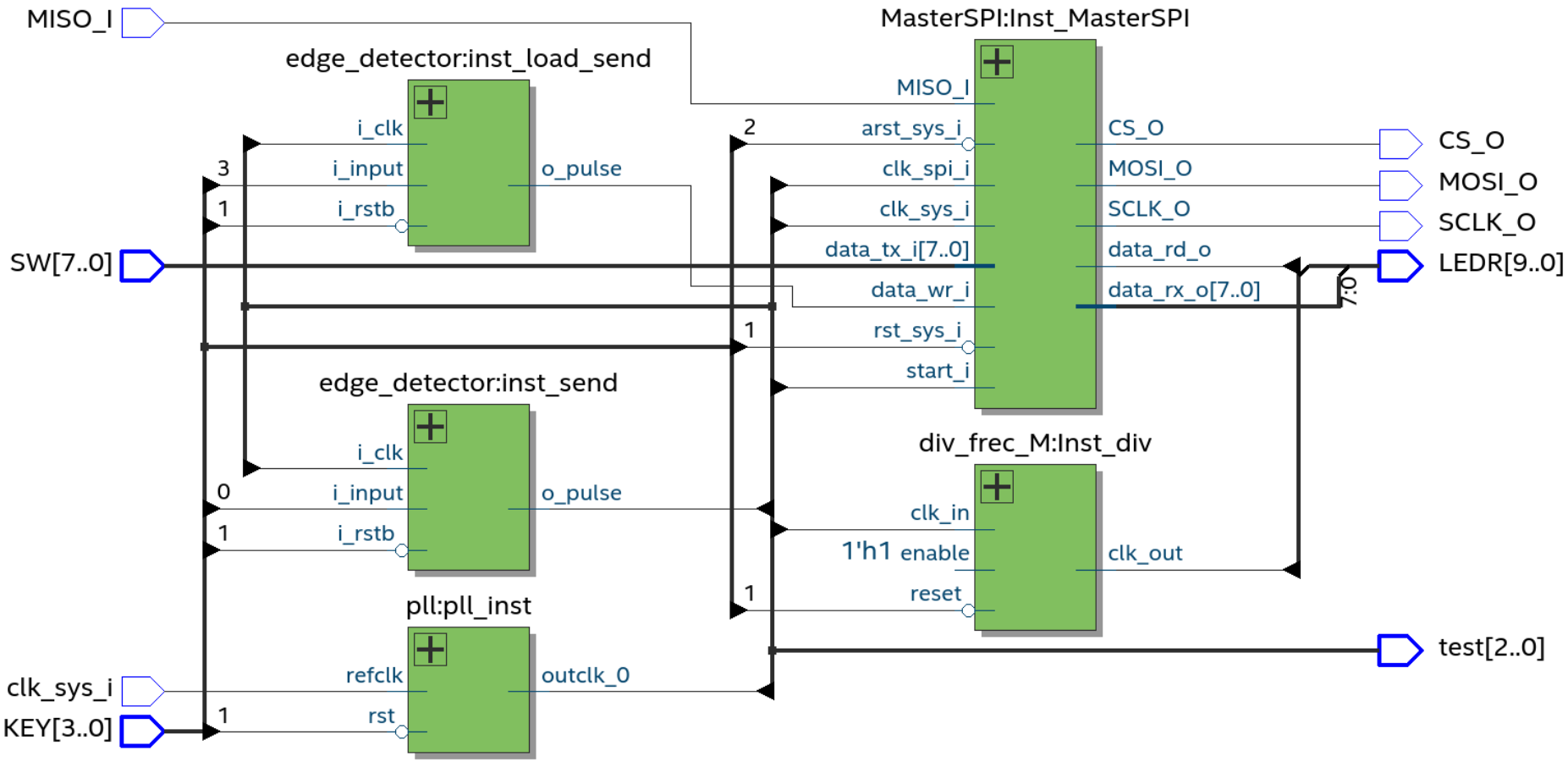
Isim - Xilinx



GTKWave



Prueba en placa DE1-SOC

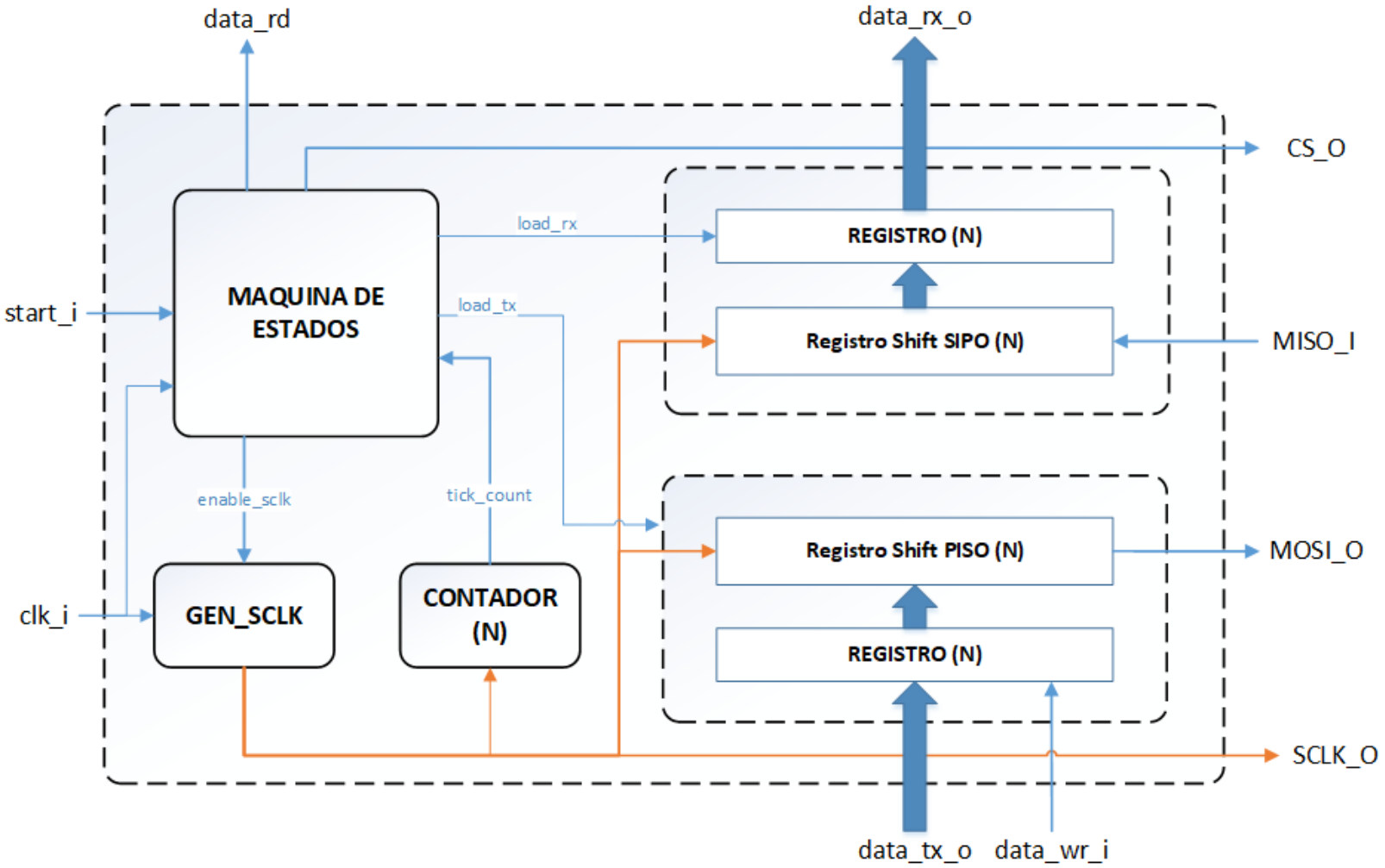


Reloj del sistema 5 MHz

Mejoras

Sincronismo pin MISO – Prevención de Metaestabilidad

Análisis de sincronismos al utilizar reloj diferente para SPI



Problemas de sincronización