

EMMAN ACE G MENION
BSCPE 3A

ASYNCHRONOUS BINARY UP COUNTER

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity ASYNCHRONOUS_BINARY_UP_COUNTER    is
Port (
    clk    : in  STD_LOGIC; -- Connect to PIN_23 (50MHz clock)
    reset_n : in  STD_LOGIC; -- Connect to PIN_25 (Active-low reset button)
    leds    : out STD_LOGIC_VECTOR(2 downto 0) -- Connect to Pins 87,86,85
);
end ASYNCHRONOUS_BINARY_UP_COUNTER      ;

architecture Behavioral of ASYNCHRONOUS_BINARY_UP_COUNTER    is
    signal counter    : STD_LOGIC_VECTOR(2 downto 0) := "000";
    signal slow_clk    : STD_LOGIC := '0';
    signal clk_divider : integer range 0 to 25000000 := 0; -- For 1Hz @ 50MHz
begin

    -- Clock divider process (makes counting visible on LEDs)
    process(clk)
    begin
        if rising_edge(clk) then
            if clk_divider = 25000000 then -- Adjust this value for speed
                slow_clk <= not slow_clk;
                clk_divider <= 0;
            else
                clk_divider <= clk_divider + 1;
            end if;
        end if;
    end process;

    -- Asynchronous counter process
    process(reset_n, slow_clk)
    begin
        if reset_n = '0' then -- Active-low reset
            counter <= "000";
        else
            -- First stage (LSB)
            if rising_edge(slow_clk) then
                counter(0) <= not counter(0);
            end if;
        end if;
    end process;
end;
```

end if;

-- Second stage

if falling_edge(counter(0)) then

 counter(1) <= not counter(1);

end if;

-- Third stage (MSB)

if falling_edge(counter(1)) then

 counter(2) <= not counter(2);

end if;

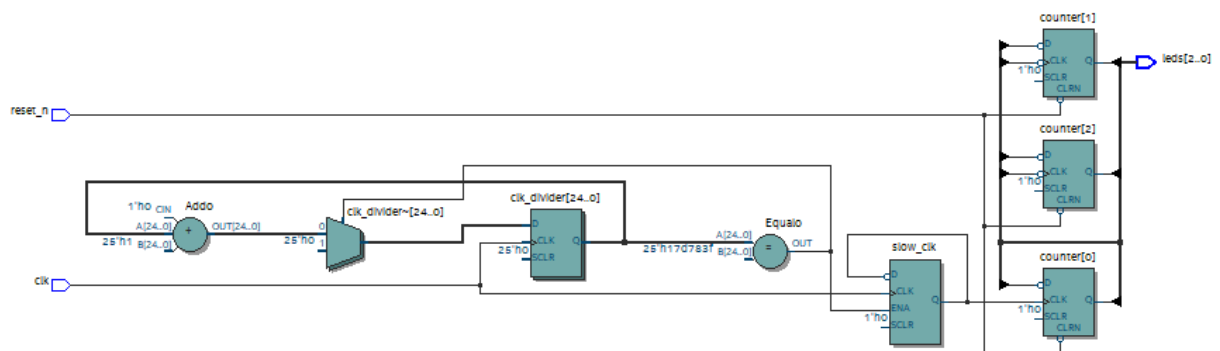
end if;

end process;

-- Active-high LED outputs (invert if your board needs active-low)

leds <= counter;

end Behavioral;



Project Navigator | Hierarchy | practice.vhd | Compilation Report - practice | IP Catalog

Entity: Instance

Cyclone IV E: EP4CE6E22C8

Tasks

Compilation

Task

Compile Design 00:00

Analysis & Synthesis 00:00

Fitter (Place & Route) 00:00

Assembler (Generate programming files) 00:00

Timing Analysis 00:00

EDA Netlist Writer 00:00

Edit Settings

Program Device (Open Programmer)

```

1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  use IEEE.NUMERIC_STD.ALL;
4
5  entity practice is
6  port
7  (
8      clk      : in  STD_LOGIC;           -- 50MHz clock input
9      reset_n  : in  STD_LOGIC;           -- Active-low reset
10     leds     : out STD_LOGIC_VECTOR(2 downto 0) -- 3 LEDs output
11 );
12 end practice;
13
14 architecture Behavioral of practice is
15     signal counter      : unsigned(2 downto 0) := (others => '0');
16     signal slow_clk      : STD_LOGIC := '0';
17     signal clk_divider   : unsigned(24 downto 0) := (others => '0'); -- 25 bits to c
18
19     -- clock divider (50MHz -> 1Hz slow clock)
20     process(clk)
21     begin
22         if rising_edge(clk) then
23             if clk_divider = 24999999 then
24                 clk_divider <= (others => '0');
25                 slow_clk <= not slow_clk;
26             else
27                 clk_divider <= clk_divider + 1;
28             end if;
29         end if;
30     end process;
31
32     -- Asynchronous counter (ripple counter)
33     -- Note: Ripple counters naturally use flip-flops triggered on edges of the pre
34     process(reset_n, slow_clk)
35     begin
36         if reset_n = '0' then
37             counter <= (others => '0');
38         else
39             if rising_edge(slow_clk) then
40                 counter(0) <= not counter(0);

```

Messages

System (1) | Processing (124)

293000 Quartus Prime Full Compilation was successful. 0 errors, 18 warnings

Running Quartus Prime Netlist Viewers Preprocess

Command: quartus_npp practice -c practice --netlist_type=sgate

18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PAR

Quartus Prime Netlist Viewers Preprocess was successful. 0 errors, 1 warning

14. ASYNCHRONOUS BINARY UP COUNTER:

3-bit Asynchronous up counter			
Clock	QC	QB	QA
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8	0	0	0
9	0	0	1

