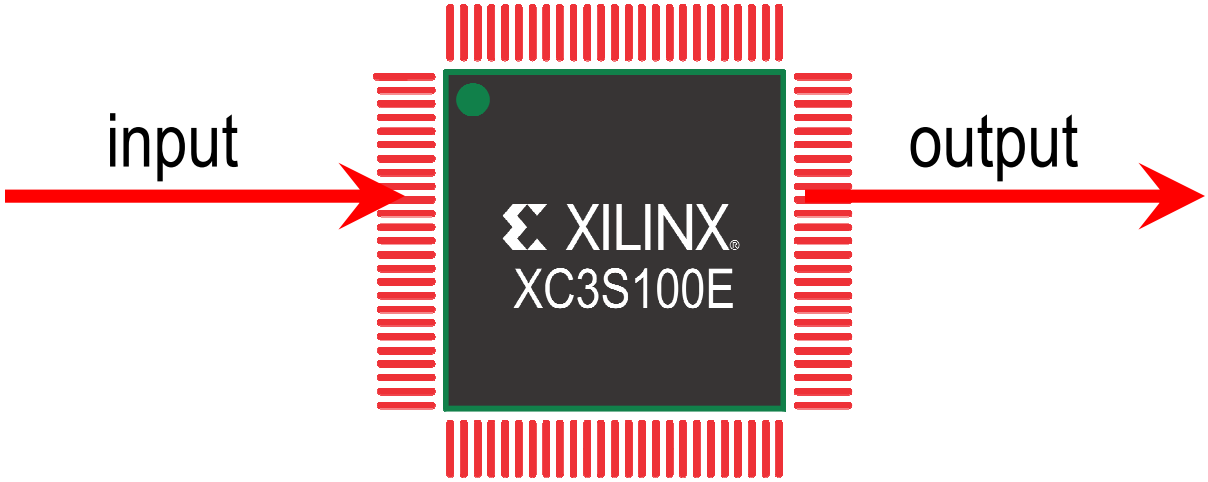
State Based Designs in VHDL

Dr.R.Sundaramurthy

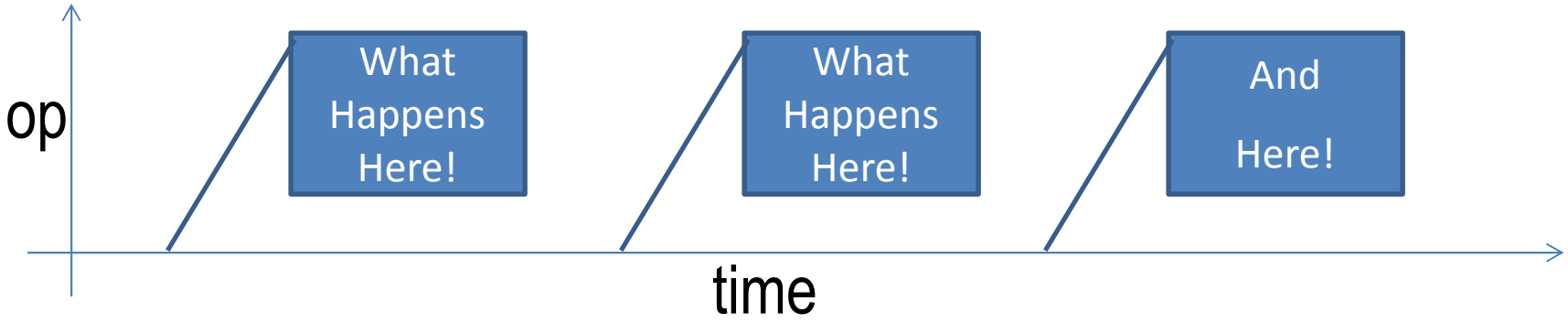
Department of EIE
Pondicherry Engineering College

sundar@pec.edu

What is a time Line ?

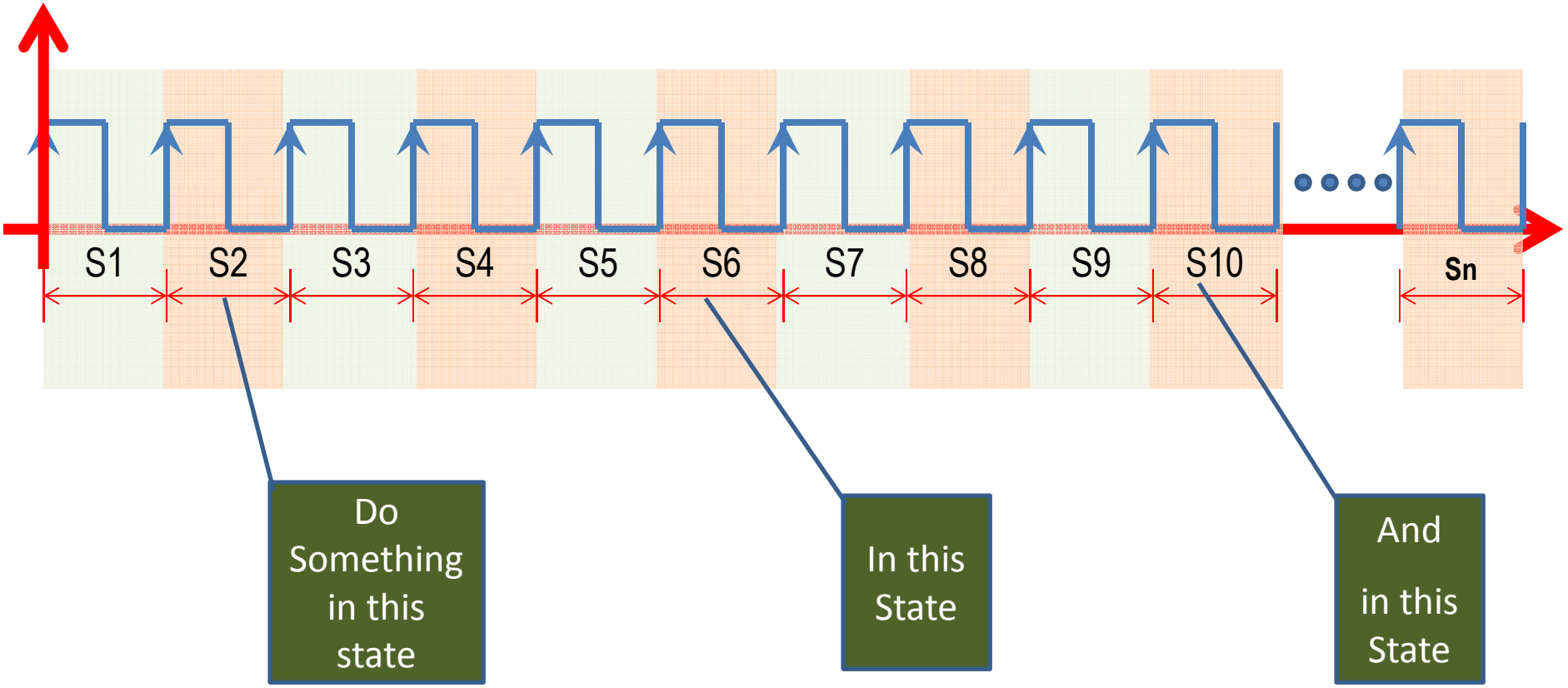


A time line is a graphical representation of output (Vs) time, on which important events are marked.



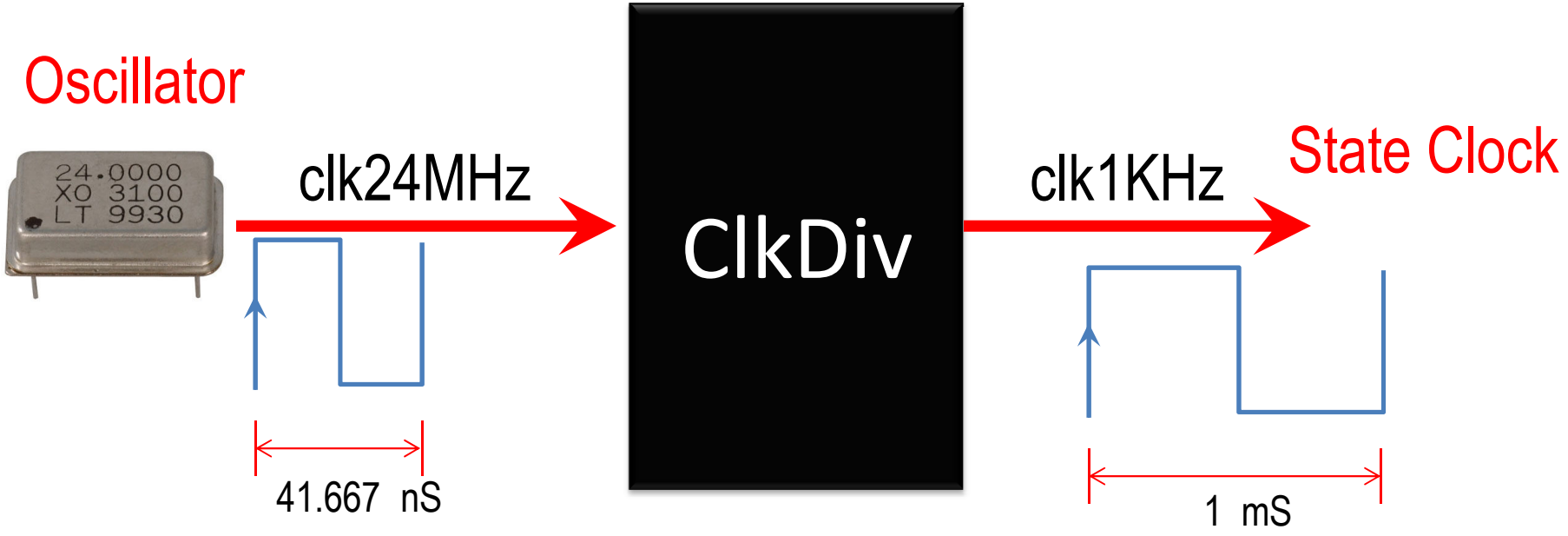
What is a state ?

- A state is a specific Time Instant.
- We define the events of the FPGA using states.
- We use state clocks to represent states.



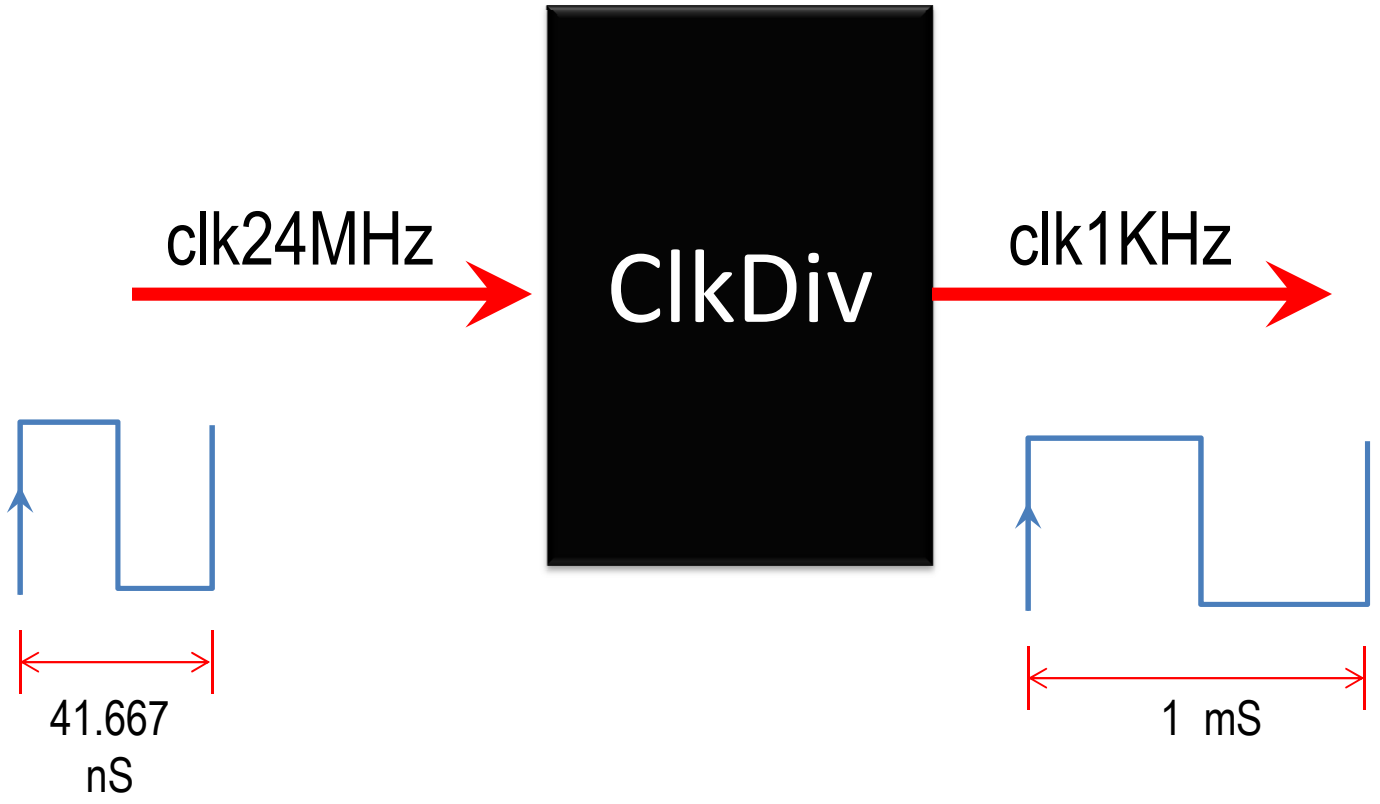
What is a state clock ?

- State clocks are used to represent the states.
- Time period of the state is determined by the application.
- State clocks are normally derived by using a clock divider logic from the oscillator clock.



Generation of State Clocks

Generation of State Clock @ 1KHz



Count Calculations

Input Oscillator Frequency = 24 MHz

Desired Output Frequency = 1 KHz

Number of Input Clock cycles
For 1 Time Period of o/p Clock = 24 MHz / 1 KHz

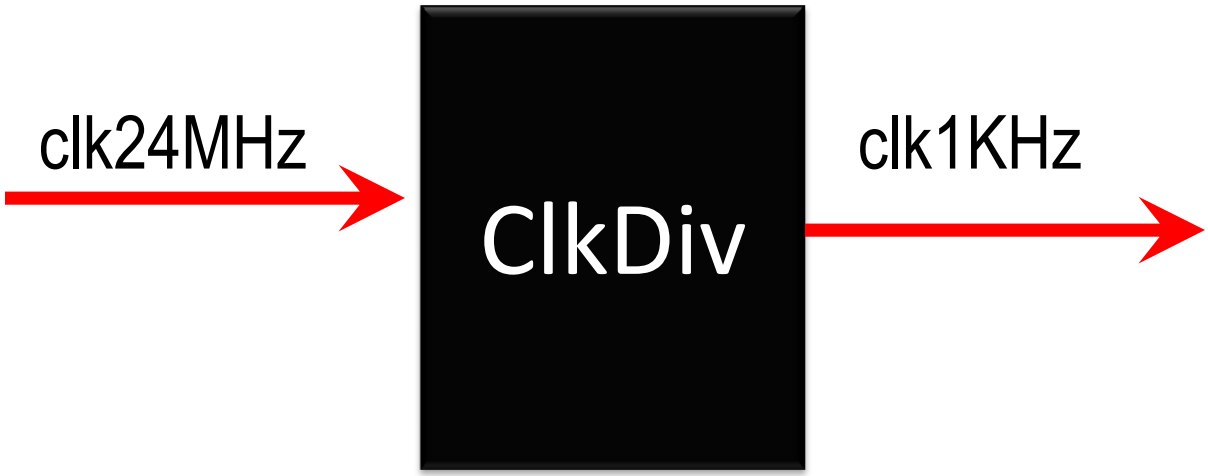
= 24000000/1000

= 24000 clocks

Number of Input Clock cycles
For 1 Time Period of o/p Clock = 24000 / 2

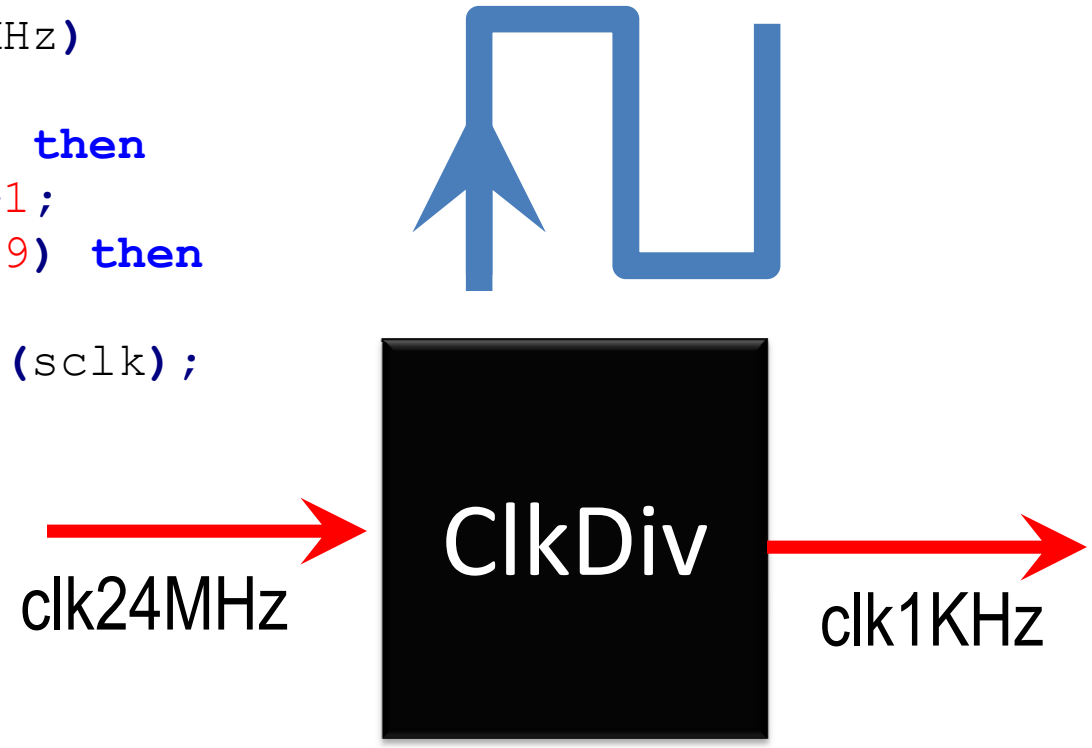
= 12000 clocks

Toggle a 'Signal' every '12000' counts

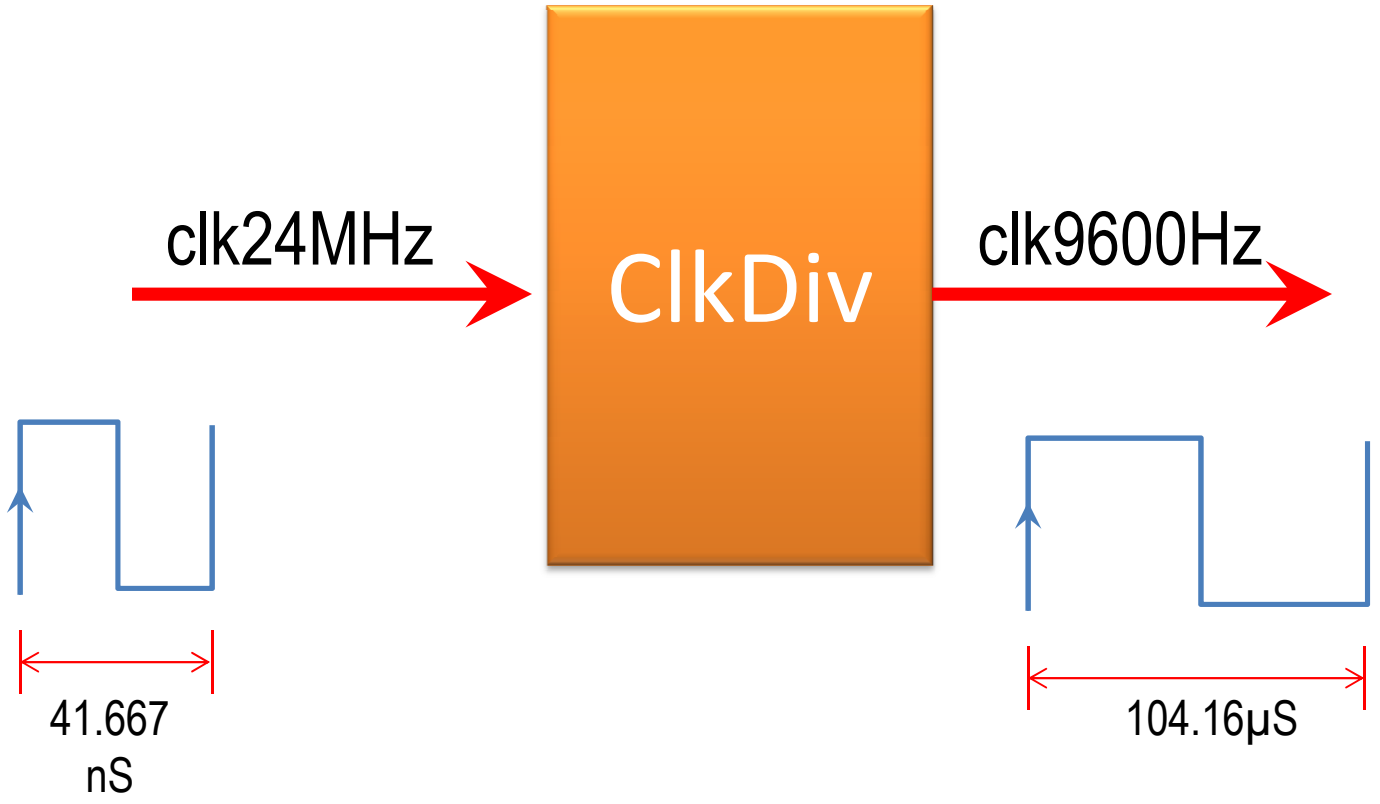



```
library IEEE;

use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_unsigned.ALL;
use IEEE.STD_LOGIC_arith.ALL;

entity clkdiv is
    Port ( clk24MHz : in  STD_LOGIC;
           clk1KHz : out  STD_LOGIC);
end clkdiv;
```

```
architecture Behavioral of clkdiv is

signal count : integer:=0;
signal sclk : std_logic:='0';

begin

stateclk: process(clk24MHz)
begin
if rising_edge(clk24MHz) then
        count <= count +1;
        if (count = 11999) then
            count <=0;
            sclk <= not (sclk);
        end if;
end if;
end process;

clk1KHz <= sclk;
end Behavioral;
```



Generation of State Clock @ 9600Hz



Count Calculations

Input Oscillator Frequency = 24 MHz

Desired Output Frequency = 9600Hz

Number of Input Clock cycles
For 1 Time Period of o/p Clock = 24 MHz / 9600Hz

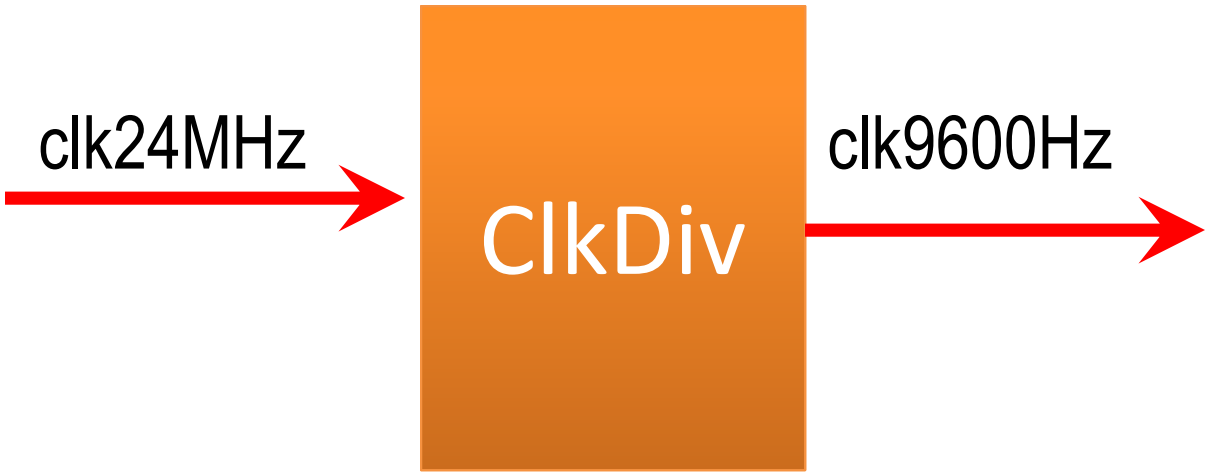
= 24000000/9600

= 2500 clocks

Number of Input Clock cycles
For 1 Time Period of o/p Clock = 2500 / 2

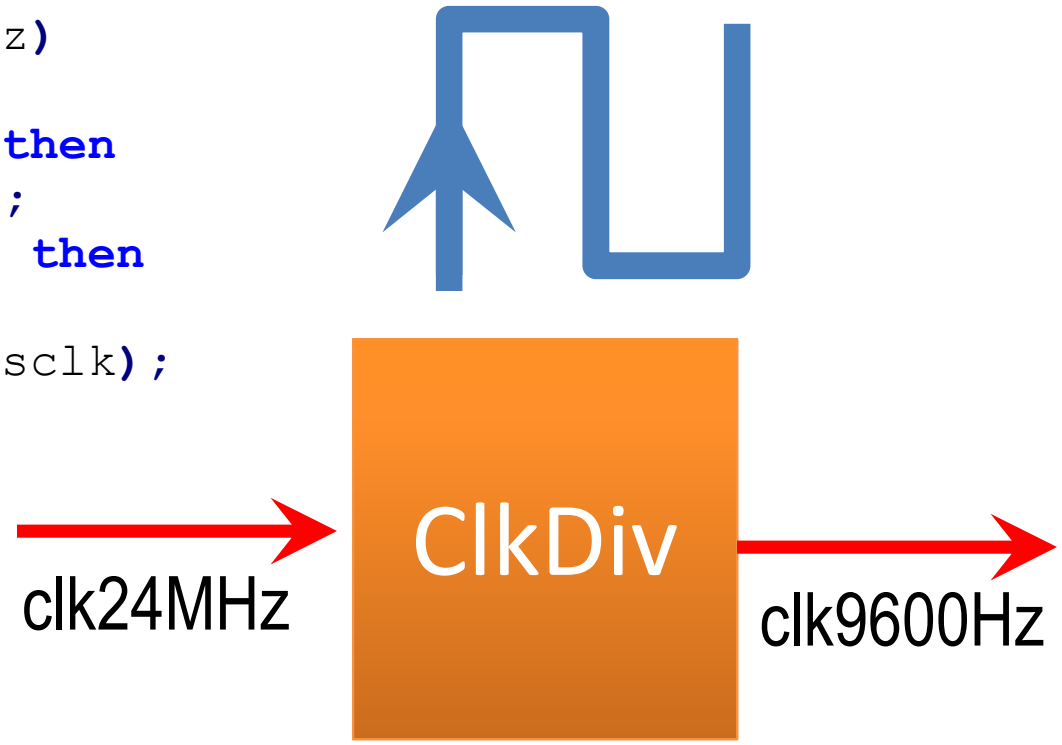
= 1250 clocks

Toggle a ‘Signal’ every ‘1250‘ counts

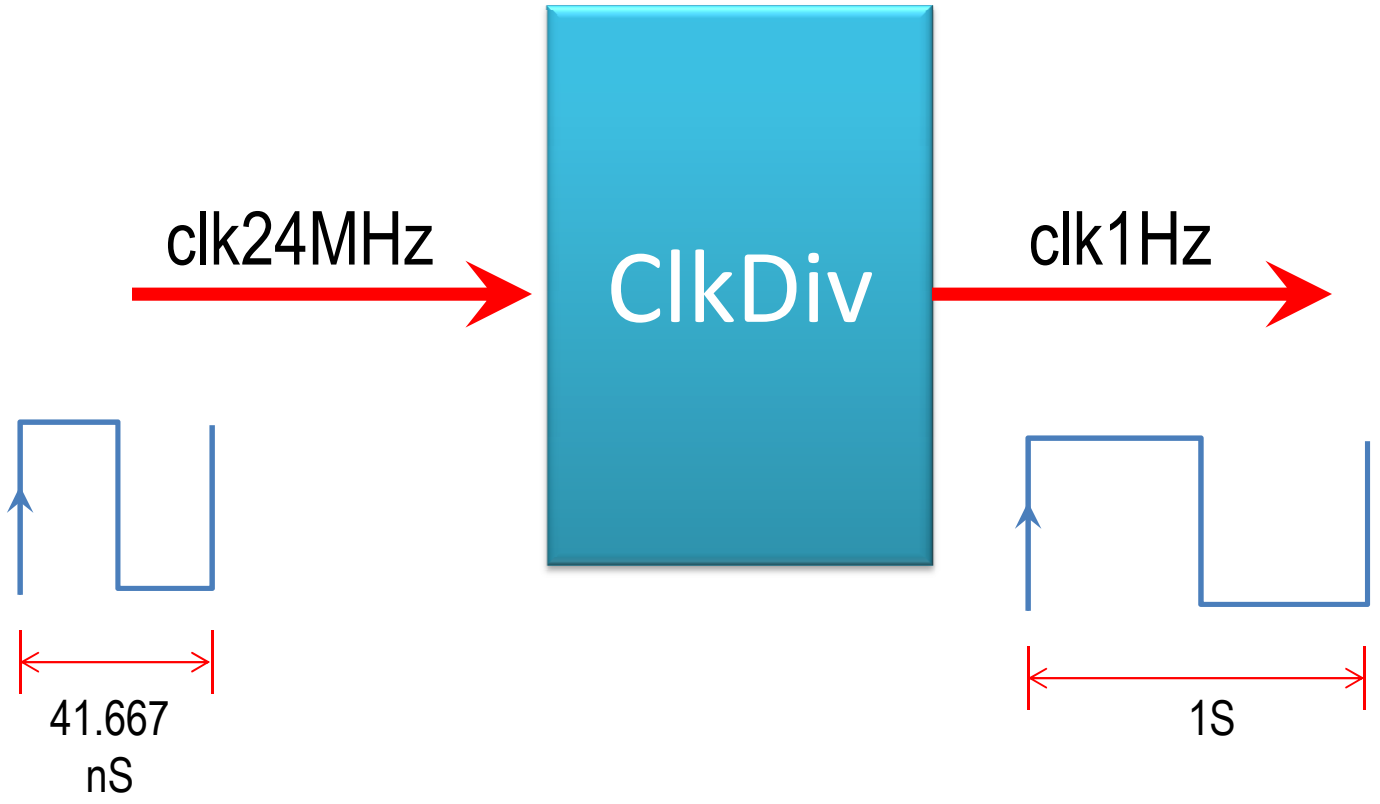



```
library IEEE;

use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_unsigned.ALL;
use IEEE.STD_LOGIC_arith.ALL;

entity clkdiv is
    Port ( clk24MHz : in  STD_LOGIC;
           clk9600Hz : out  STD_LOGIC);
end clkdiv;
```

```
architecture Behavioral of clkdiv is

signal count : integer:=0;
signal sclk : std_logic:='0';

begin

stateclk: process(clk24MHz)
begin
if rising_edge(clk24MHz) then
        count <= count +1;
        if (count = 1249) then
            count <=0;
            sclk <= not (sclk);
        end if;
end if;
end process;

clk9600Hz <= sclk;
end Behavioral;
```



Generation of State Clock @ 1Hz



Count Calculations

Input Oscillator Frequency = 24 MHz

Desired Output Frequency = 9600Hz

Number of Input Clock cycles
For 1 Time Period of o/p Clock = 24 MHz / 1Hz

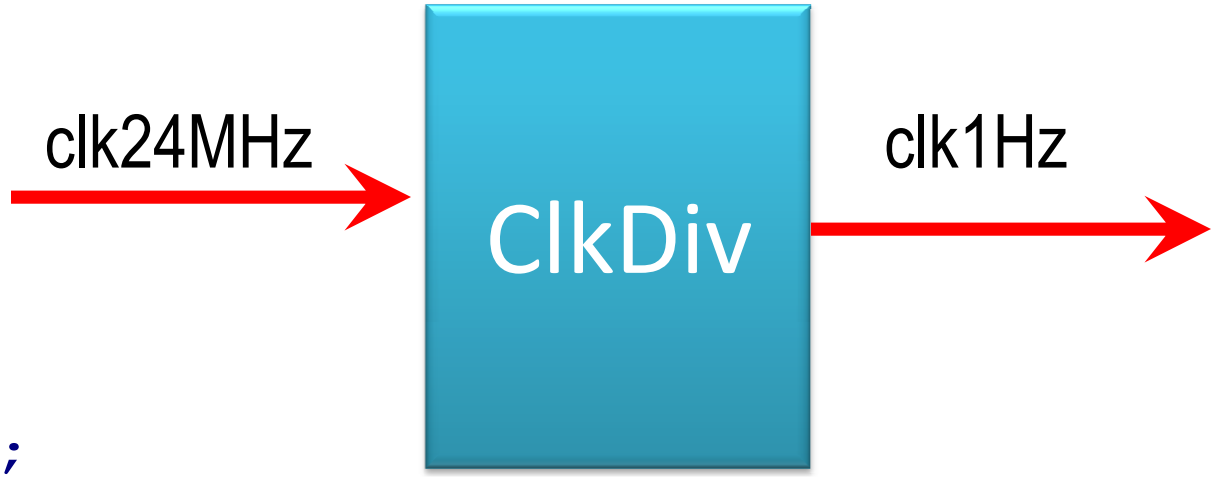
= 24000000/1

= 24000000 clocks

Number of Input Clock cycles
For 1 Time Period of o/p Clock = 24000000 / 2

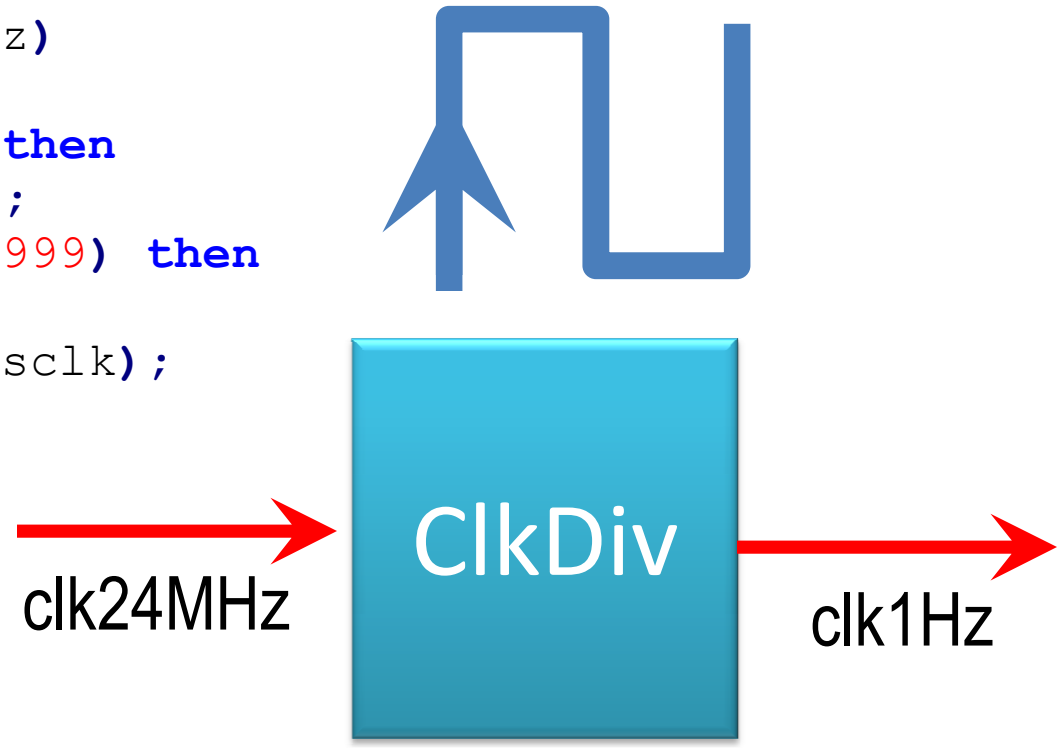
= 12000000 clocks

Toggle a 'Signal' every '12000000 ' counts




```
library IEEE;

use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_unsigned.ALL;
use IEEE.STD_LOGIC_arith.ALL;

entity clkdiv is
    Port ( clk24MHz : in  STD_LOGIC;
           clk1Hz : out  STD_LOGIC);
end clkdiv;
```

```
architecture Behavioral of clkdiv is

signal count : integer:=0;
signal sclk : std_logic:='0';

begin

stateclk: process(clk24MHz)
begin
if rising_edge(clk24MHz) then
        count <= count +1;
        if (count = 11999999) then
            count <=0;
            sclk <= not (sclk);
        end if;
end if;
end process;

clk1Hz <= sclk;
end Behavioral;
```



End of Session

sundar@pec.edu