

# Clock

---

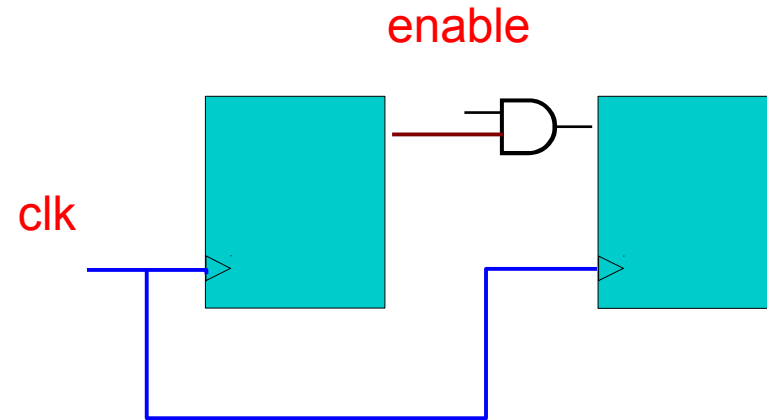
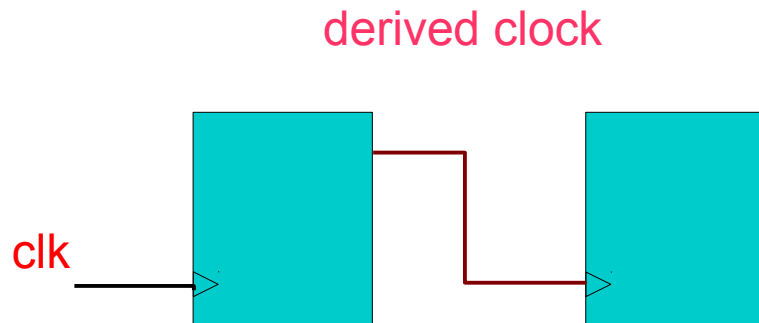
Copyright (c) 2011 Young W. Lim.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Please send corrections (or suggestions) to [youngwlim@hotmail.com](mailto:youngwlim@hotmail.com).

This document was produced by using OpenOffice and Octave.

# Derived Clock



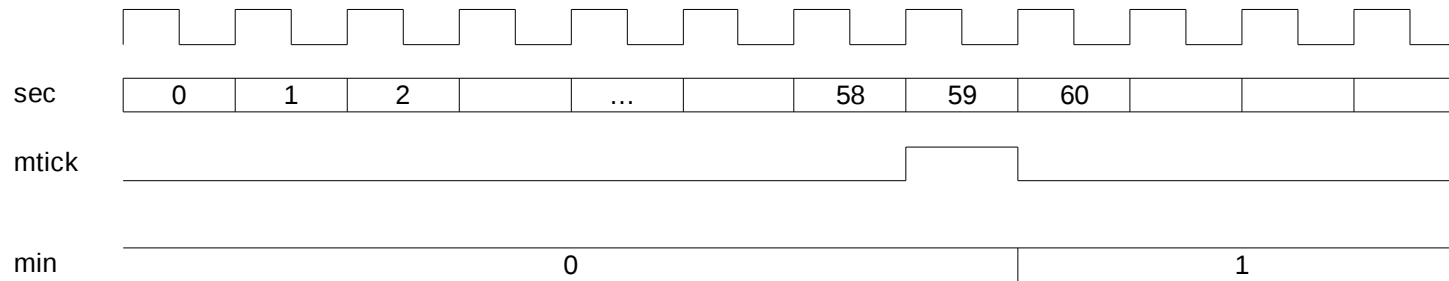
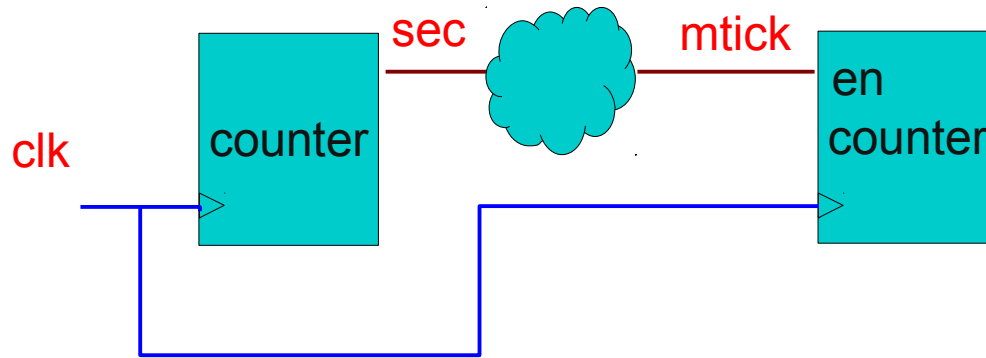
Clock signal requires special care  
- special driver and distribution network

Derived clocks make things complicated

Use a synchronous enable signal

# Second and Minute Counter

`mtick <= '1' when sec = "111011" else '0';`



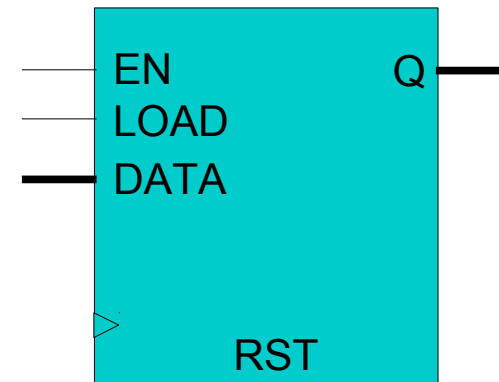
# Counter Template

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all; -- for the unsigned type

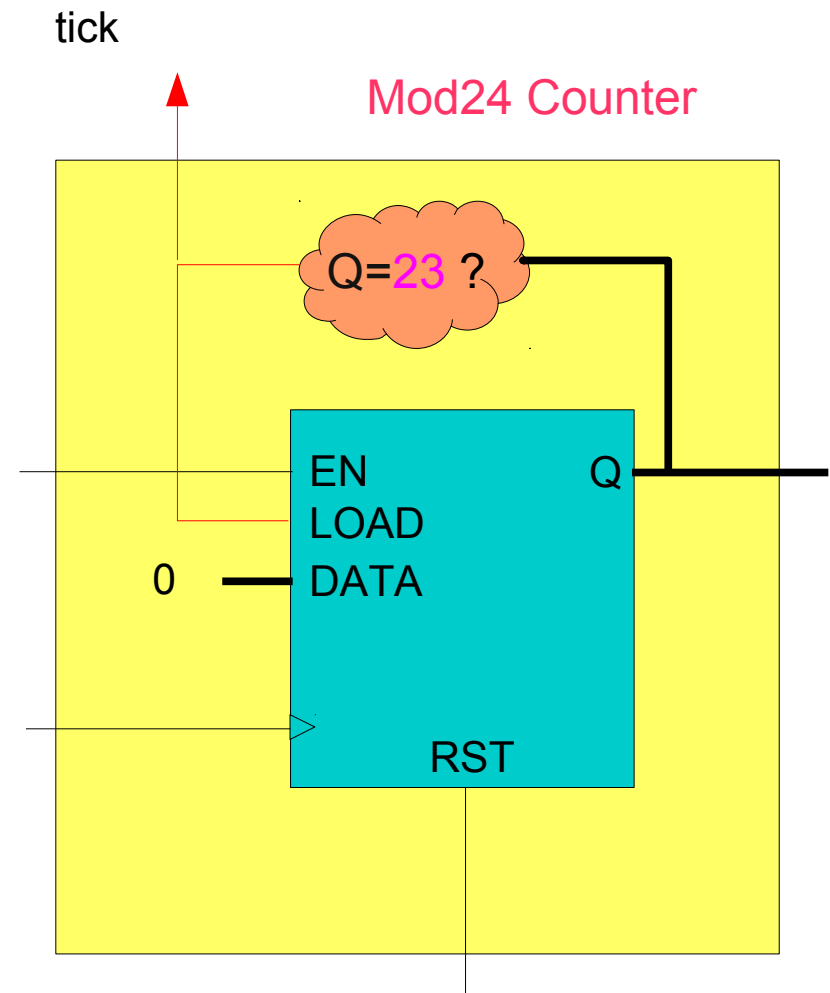
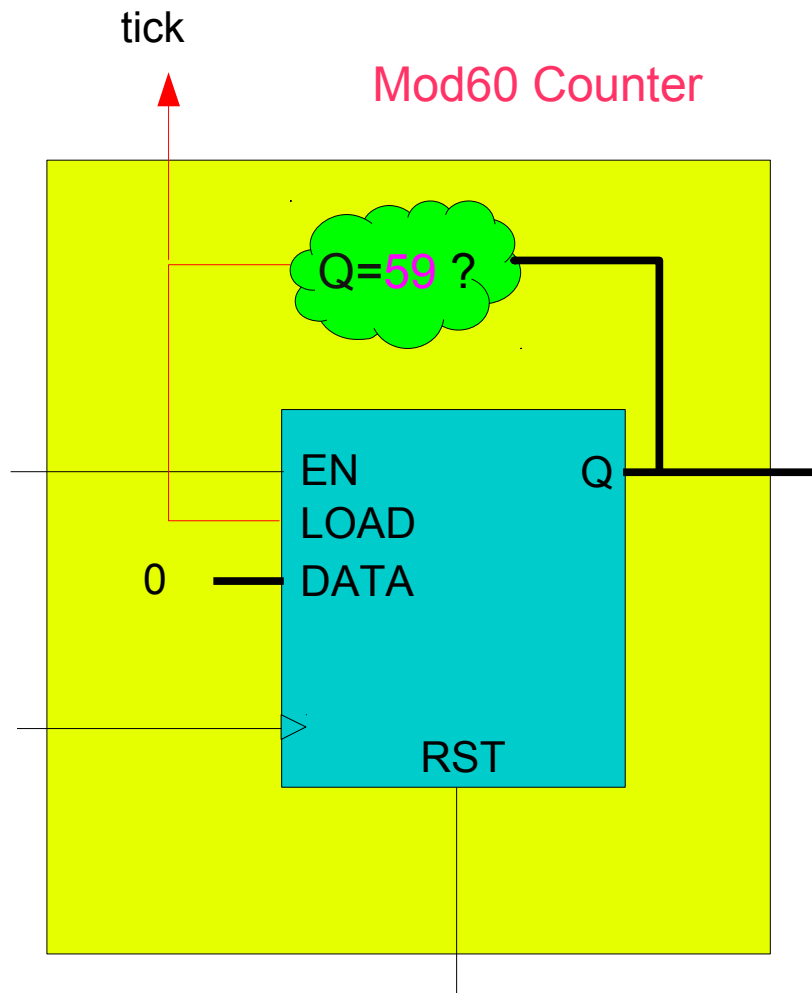
entity COUNTER is
  generic (
    WIDTH : in natural := 32);
  port (
    RST   : in std_logic;
    CLK   : in std_logic;
    EN    : in std_logic;
    LOAD  : in std_logic;
    DATA : in std_logic_vector(WIDTH-1 downto 0);
    Q     : out std_logic_vector(WIDTH-1 downto 0));
end entity COUNTER;

architecture RTL of COUNTER is
  signal CNT : unsigned(WIDTH-1 downto 0);
begin
  process(RST, CLK) is
  begin
    if RST = '1' then
      CNT <= (others => '0');
    elsif rising_edge(CLK) then
      if LOAD = '1' then
        CNT <= unsigned(DATA); -- type is converted to unsigned
      elsif EN = '1' then
        CNT <= CNT + 1;
      end if;
    end if;
  end process;

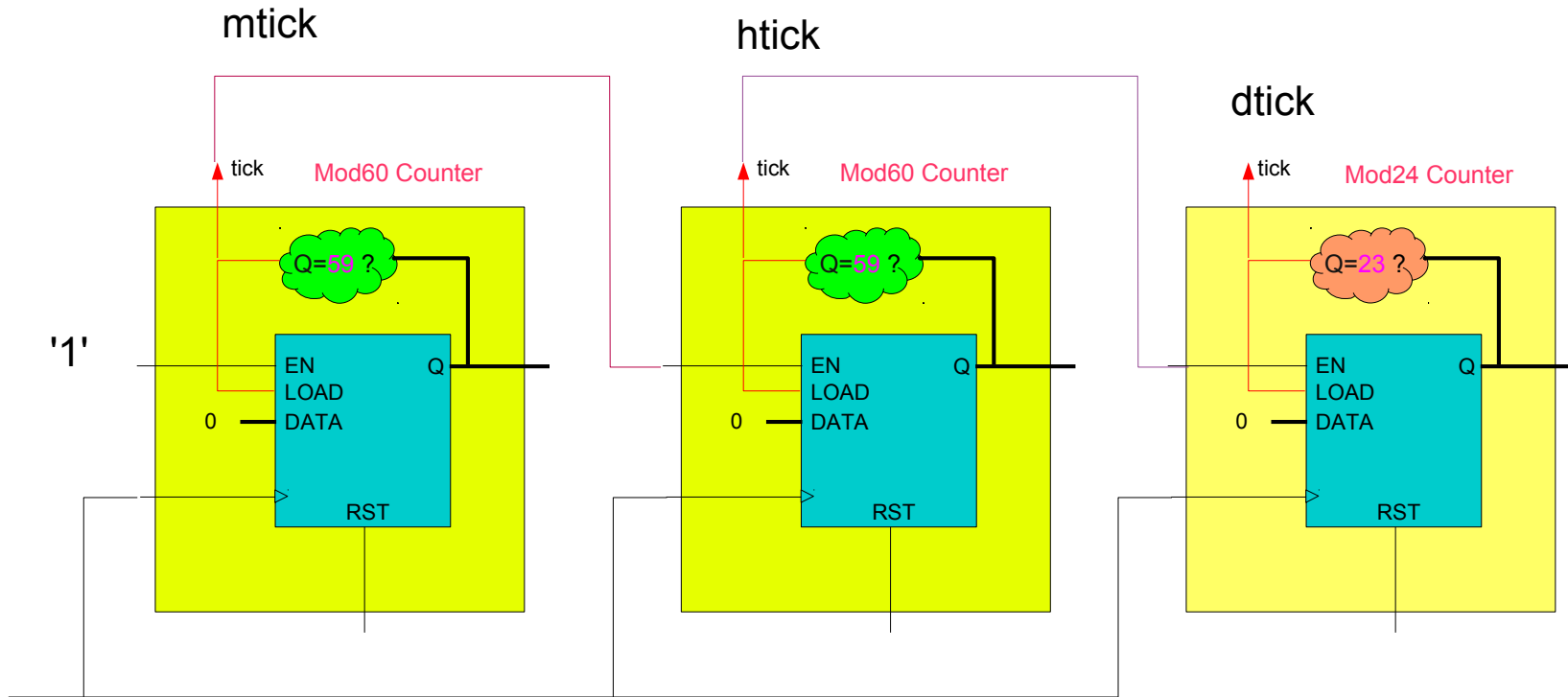
  Q <= std_logic_vector(CNT); -- type is converted back to std_logic_vector
end architecture RTL;
```



# Mod-60 and Mod-24 counter



# Sec, Min, Hr Counters



## References

[1] <http://en.wikipedia.org/>

[2]