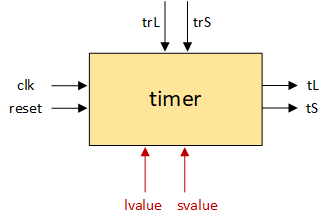
Assignment

In this assignment, you will create a verilog implementation of a **timer**. The timer is able to count two different timing intervals based on the trigger inputs. You can count for a shorter timing interval and a longer timing interval. These timing intervals are parameterized. The top level diagram of the timer is shown below:



As shown in the diagram above, the timer module has the following signals:

* **clk (1 bit)**
* **reset(1 bit)**
* **trS (1 bit) -**trigger signal to start counting for **svalue** cycles
* **trL (1 bit) -**trigger signal to start counting for **lvalue** cycles
* **tL (1 bit) -**signal indicating expiration of a time interval of **lvalue** cycles
* **tS (1 bit) -** signal indicating expiration of a time interval of **svalue** cycles

**Important Points:**

**1) "svalue" and "lvalue" are input parameters;**

**They determine the shorter and the longer timing intervals respectively.**

**2) Input signals "trS" and "trL" cannot be high at the same time.**

**3) Output signals "tS" and "tL" cannot be active at the same time.**

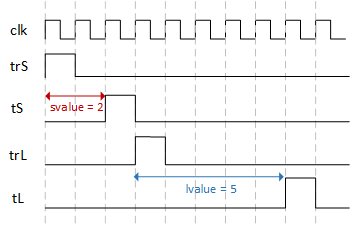
**Timing behaviour of Timer:**

**1) trL = 1, trS = 0: Make tL = 1 after lvalue cycles.**

**2) trL = 0, trS = 1: Make tS = 1 after svalue cycles.**

**3) trL = 0 , trS = 0: No change**

Timing behaviour of the timer for **svalue = 2 cycles**and **lvalue = 5 cycles** is shown below:



**Note:** The **reset**is a **synchronous** signal (i.e. it goes high only at posedge of clock). Also, make all transitions at the posedge of **clk**. **Make sure the maximum possible time interval for which the timer can count is greater than 500 cycles.**

Use the template given below for desiging the timer:

`timescale 1ns / 1ps

module timer(clk,reset, trL, trS, tL, tS);

input clk, reset, trL, trS;

output tL, tS;

parameter svalue = <<svalue>>;

parameter lvalue = <<lvalue>>;

<<Logic for timer>>

endmodule