Traffic Light Controller

In this lab you will implement a traffic light controller that controls a main street, a side street and walk lamps. You will be using a finite state machine to implement this controller. This lab provides you with a design methodology that will be useful in future labs and final projects. This involves planning your design, coding and debugging your design.

Procedure

There are two major phases. The first is the design phase, which consists of reading through the lab, planning, and coming up with a design. Although not required, it is suggested that you schedule a conference with a member of the course staff to review your design. This will help catch any major mistakes early in the process.
The next phase is to implement the first part of the lab using the FPGA. After you verify the traffic light controller's functionality, you can get checked off.

Traffic Light Controller Description

The traffic light controller is for an intersection between a Main Street and a Side Street. Both streets have a red, yellow, and green signal light. Pedestrians have the option of pressing a walk button to turn all the traffic lights red and cause a single walk light to illuminate. Lastly, there is a sensor on the Side Street which tells the controller if there are cars still on the Side Street. This is summarized in Figure 1.

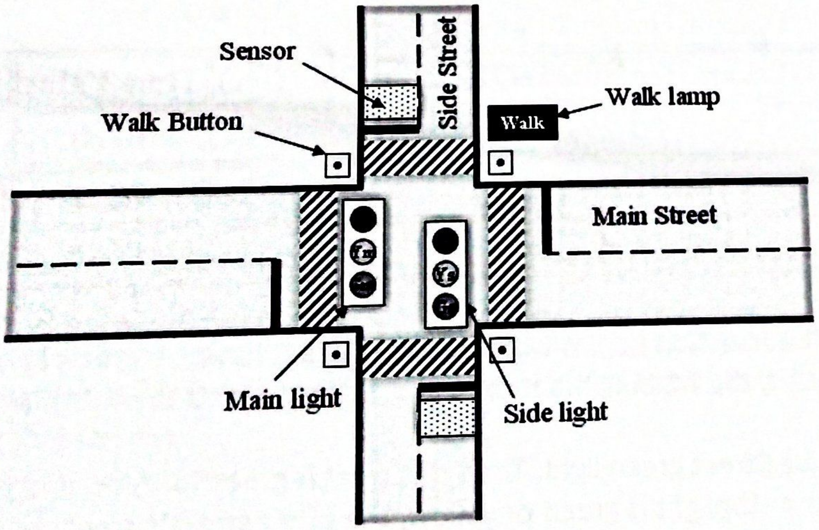

Figure 1: Diagram for intersection with corresponding lights.

Divider

The divider is necessary for the timer to properly time the number of seconds for any particular traffic light state. Using only the clock as input, this module generates a 1 Hz enable, which is sent to the timer. The signal generated is a pulse that is high for one clock cycle every 1sec.

Timer

The timer is responsible for taking the start_timer, 1Hz enable, and Time Parameter value to properly time the traffic light controller. When done counting a particular state, the expired signal will go high for one clock period to signal to the FSM that it should change states.

Finite State Machine

The finite state machine controls the sequencing for the traffic light. As previously described, it changes states based on the Walk Register and sensor signals, and with the expired signal.

You may assume that the 4 walk buttons placed at each street corner are hooked into the traffic light controller using a wired-OR. For this reason, you may assume that the controller only needs a single input called Walk-Request.

The side street sensor is placed near the intersection to tell the controller when there are cars passing over the sensor. You may assume the sensor remains constantly high if several cars pass over the sensor, rather than quick pulses, provided the cars are close enough together. You do not need to implement this specific functionality. This input is named Sensor.

The traffic lights are timed on three parameters (in seconds): the base interval (t_{BASE}), the extended interval (t_{EXT}), and the yellow light interval (t_{YEL}). The default values listed in the table below are to be loaded into the FPGA on reset, and may be reprogrammed on demand using switches and buttons on your kit with the Time_Parameter_Selector, Time_Value, and Reprogram signals. Time_Parameter_Selector uses the Parameter Number code to select the interval during programming. Time_Value is a 4-bit value representing the value to be programmed; therefore, it has a duration of seconds between 0 and 15. The Reprogram button tells the system to set the currently selected interval to Time_Value.

Default Timing Parameters

Interval Name	Symbol	Parameter Number	Default Time (sec)	Time Value
Base Interval	t_{BASE}	00	6	0110
Extended Interval	t_{EXT}	01	3	0011
Yellow Interval	t_{YEL}	10	2	0010

The operating sequence of this intersection begins with the Main Street having a green light for 2 lengths of t_{BASE} seconds. Next, the Main lights turn to yellow for t_{YEL} and then turn red while simultaneously turning on the Side Street green light. The Side Street is green for t_{BASE}, and its yellow is held for t_{YEL}. Whenever a stoplight is green or yellow, the other street's stoplight is red.

Under normal circumstances, this cycle repeats continuously.

There are two ways the controller can deviate from the typical loop. First, a walk button allows pedestrians to submit a walk request. The internal Walk Register should be set on a button press and the controller should service the request after the Main Street yellow light by turning all street lights to red and the walk light to on. After a walk of t_{EXT} seconds, the traffic lights should return to their usual routine by turning the Side Street green. The Walk Register should be cleared at the end of a walk cycle.

The second deviation is the traffic sensor. If the traffic sensor is high at the end of the first tBASE
length of the Main street green, the light should remain green only for an additional tEXT seconds, rather than the full tBASE. Additionally, if the traffic sensor is high during the end of the Side Street green, it should remain green for an additional tEXT seconds.

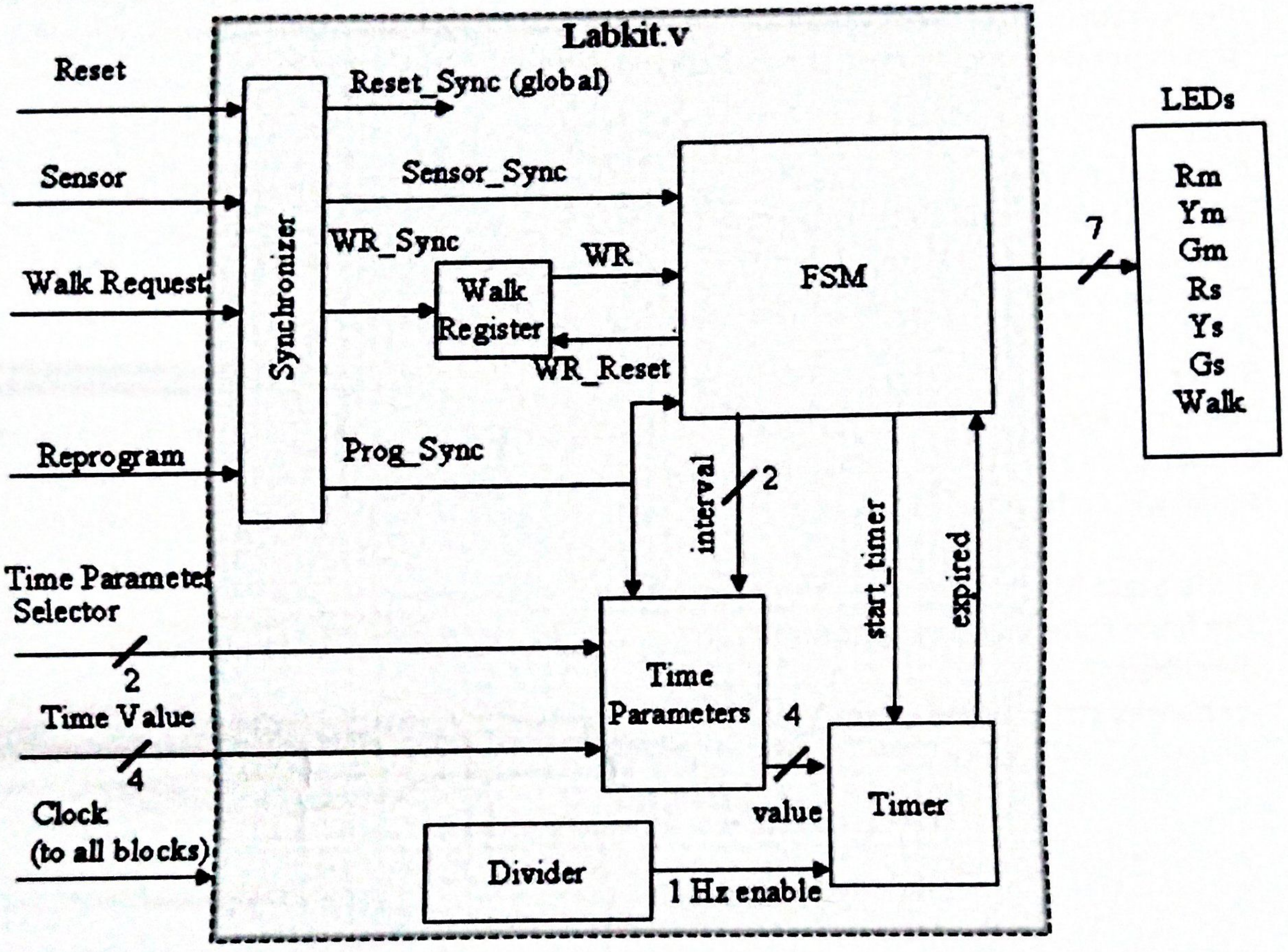

Figure 2: Block Diagram of Traffic Light Controller.

You should implement this lab by programming each block individually and then instantiating and connect the Verilog modules together in the toplevel module. Then compile your
Implementation.

Synchronizer
On the block diagram, you see that all input signals pass through the synchronizer before going to other blocks. The purpose of the synchronizer is to ensure that the inputs are synchronized to the system clock.

Walk Register
The Walk Register allows pedestrians to set a walk request at any time. There is also a signal controlled by the finite state machine that will be able to reset the register at the end of the actual walk cycle.

Time Parameters

The time parameters module stores the three different time parameter values, namely t_{BASE}, t_{EXT}, and t_{YEL} on the FPGA. The module acts like a (small) memory from the FSM and Timer blocks, where the FSM addresses the three parameters and the timer reads the data. From the user's perspective, the three time parameter values can be modified. On a reset, the three parameters should be respectively set to 6, 3, and 2 seconds. However, at any time, the user may modify any of the values by manipulating Time_Parameter_Selector, Time_Value, and Reprogram. Each of these values are 4 bits, and is selected using a 2 bit address. Whenever a parameter is reprogrammed, the FSM should be reset to its starting state.

Divider

The divider is necessary for the timer to properly time the number of seconds for any particular traffic light state. Using only the clock as input, this module generates a 1 Hz enable, which is sent to the timer. The signal generated is a pulse that is high for one clock cycle every 1sec.

Timer

The timer is responsible for taking the start_timer, 1Hz enable, and Time Parameter value to properly time the traffic light controller. When done counting a particular state, the expired signal will go high for one clock period to signal to the FSM that it should change states.

Finite State Machine

The finite state machine controls the sequencing for the traffic light. As previously described,
it changes states based on the Walk Register and sensor signals, and with the expired signal.