CS4362 - Hardware Description Languages

Traffic Light Controller

Assignment 2 Report

A.S.K.Jayasena
150256K
24/06/2019

TABLE OF CONTENTS

TASK **3**

DESIGN **4**

STATE CHART DIAGRAM 4
PROJECT LINK 4
TOP MODULE (TRAFFICCONTROLLERMAIN) 5
Integrated system Simulation *5*
SYNCHRONIZER 6
Module description *6*
Module overview *7*
Test bench output *7*
WALK REGISTER 7
Module description *7*
Module overview *7*
Test bench output *8*
TIME PARAMETERS 8
Module description *8*
Module overview *8*
Test bench output *9*
DIVIDER 9
Module description *9*
Module overview *9*
Test bench output *9*
TIMER 10
Module description *10*
Module overview *10*
Test bench output *10*
FINITE STATE MACHINE 10
Module description *10*
Module overview *10*
Test bench output *11*

FIGURES

FIGURE 1 DIAGRAM FOR INTERSECTION WITH CORRESPONDING LIGHTS AND SENSORS 3
FIGURE 2 FINITE STATE MACHINE 4
FIGURE 3 TOP MODULE BLOCK DIAGRAM 5
FIGURE 4 INTEGRATED SYSTEM SIMULATION WITH INTERNAL WIRES (NORMAL ROUTINE) 10MHZ TIMER 5
FIGURE 5 FULL SYSTEM SIMULATION (NORMAL ROUTINE | TIMER AT 10MHZ) 6
FIGURE 6 FULL SYSTEM SIMULATION (WALK REQUEST | TIMER AT 10MHZ) 6
FIGURE 7 FULL SYSTEM SIMULATION (SENSOR ACTIVATED | TIMER AT 10MHZ) 6
FIGURE 8 SYNCHRONIZER BLOCK DIAGRAM 7
FIGURE 9 SYNCHRONIZER SIMULATION 7
FIGURE 10 WALK REGISTER BLOCK DIAGRAM 7
FIGURE 11 WALK REGISTER SIMULATION 8
FIGURE 12 TIMEPARAMETERS BLOCK DIAGRAM 8
FIGURE 13 TIME PARAMETER SIMULATION 9
FIGURE 14 CHANGING T_{BASE} TO 15 SECONDS 9
FIGURE 15 DIVIDER BLOCK DIAGRAM 9
FIGURE 16 DIVIDER AT 10MHZ 9
FIGURE 17 TIMER BLOCK DIAGRAM 10
FIGURE 18 TIMER SIMULATION OUTPUT 10
FIGURE 19 FSM BLOCK DIAGRAM 10
FIGURE 20 NORMAL OPERATION 11
FIGURE 21 WALK REQUEST BY A PEDESTRIAN 11
FIGURE 22 VEHICLE SENSOR ACTIVATED 11

TABLES

TABLE 1 DEFAULT TIMING PARAMETERS 3
TABLE 2 STATES REPRESENTED BY DECIMALS 6
TABLE 3 INTERVAL VALUES 8

Task

Task is to develop a traffic light controller system to a intersection where a side street is crossing a main street. Both streets have usual traffic lights for vehicles and pedestrians.

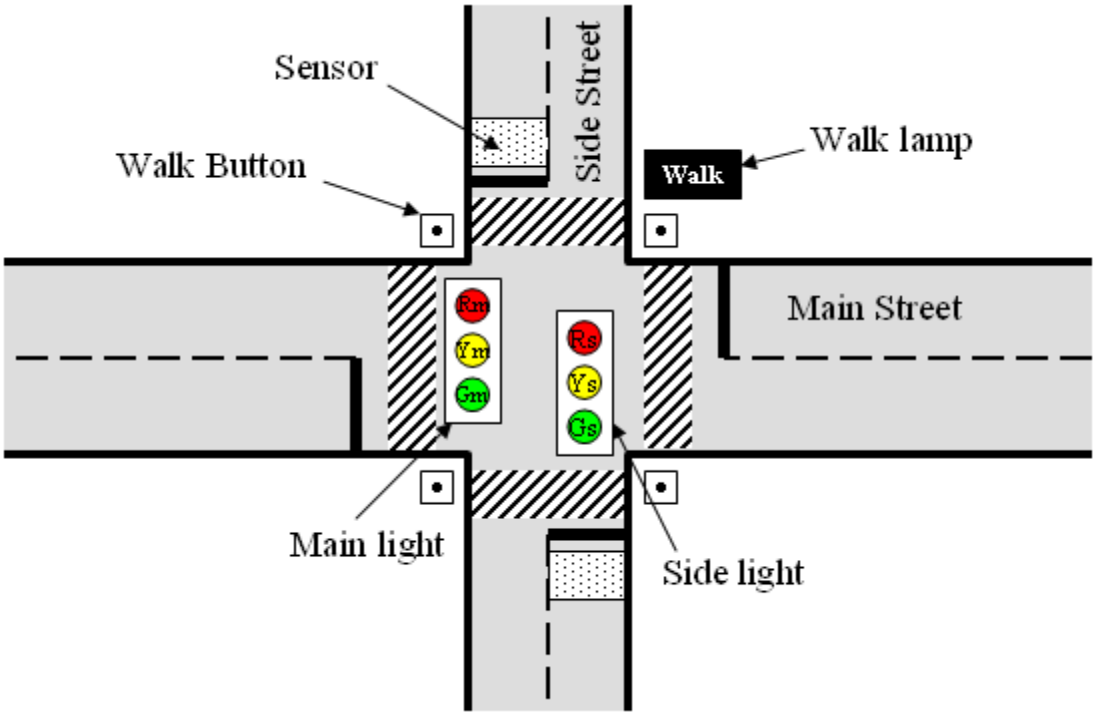

Figure 1 Diagram for intersection with corresponding lights and sensors

All the walk request buttons are attached to the controller using a wired OR. There are two sensors on the side street to detect vehicles that are passing over them. (Assumption: Sensor remains high constantly when several cars pass over it).
The Traffic light controller is timed based on three parameters (in seconds) t_{BASE}, t_{EXT}, t_{YEL}. These parameters can be changed using operation. Default values are as follows.

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

Table 1 Default timing parameters

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 will set on a button press and the controller will service

the request after the Main Street yellow light by turning all streetlights to red and the walk light to on. After a walk of t_{EXT} seconds, the traffic lights will return to their usual routine by turning the Side Street green. The Walk Register will 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 t_{BASE} length of the Main street green, the light will remain green only for an additional t_{EXT} second, rather than the full t_{BASE}. Additionally, if the traffic sensor is high during the end of the Side Street green, it will remain green for an additional t_{EXT} second.

Design

State chart diagram

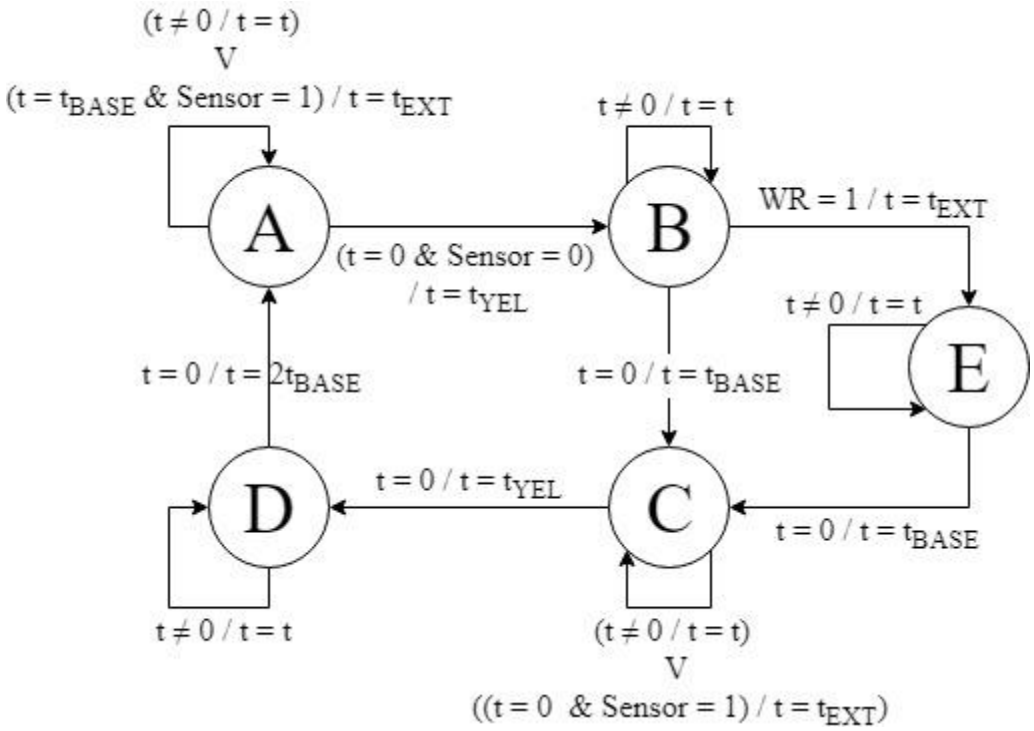

Figure 2 Finite state machine

For the convenience of the design for the state chart time is used as the parameter for the state chart. In the implementation this is replaced with expired from the timer module.

Project Link

https://github.com/Archfx/TrafficLightController

Top Module (TrafficControllerMain)

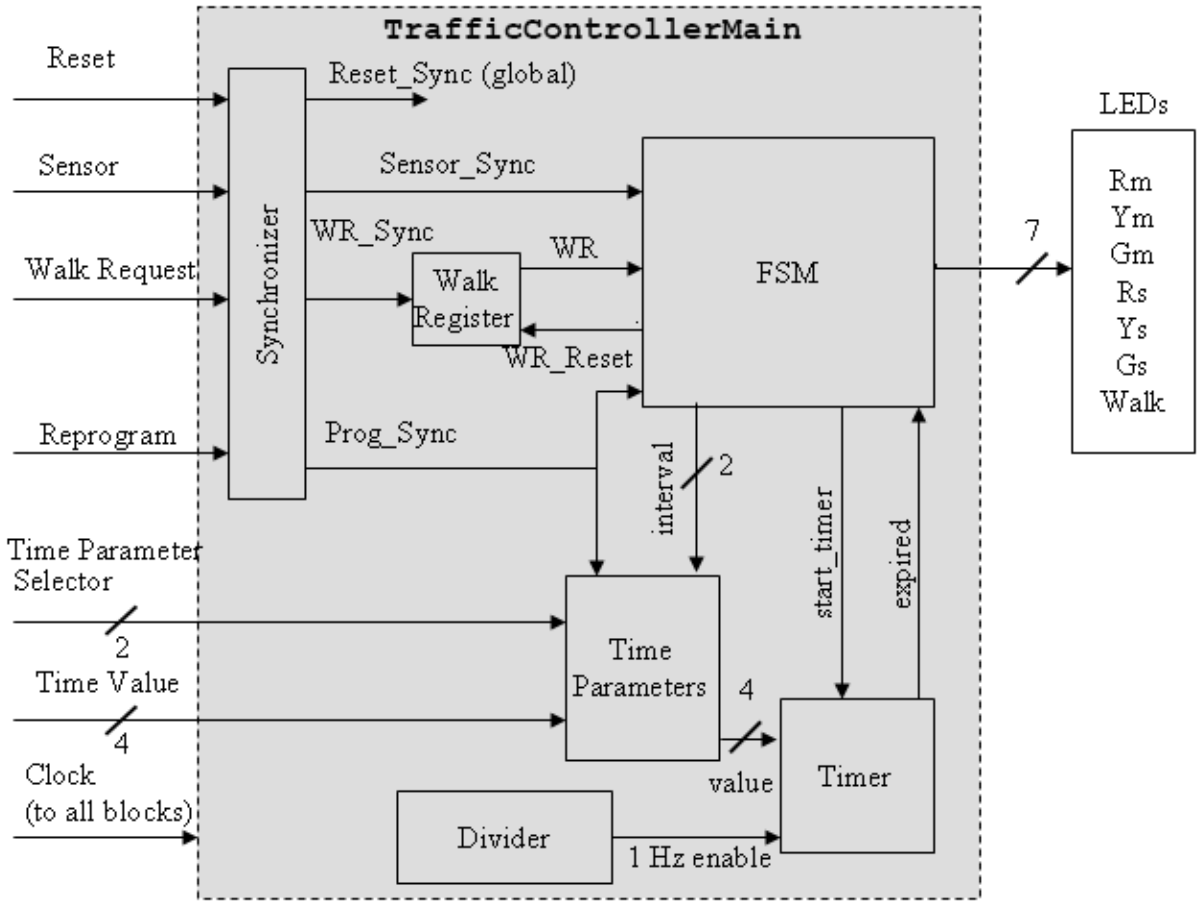

Figure 3 Top module block diagram

All the variable names are used as in the above diagram. For the lights 7bit vector array is used as follows. LED sequence: [Red_{main}, $Yellow_{main}$, $Green_{main}$, Red_{side}, $Yellow_{side}$, $Green_{side}$, Walk]

Integrated system Simulation

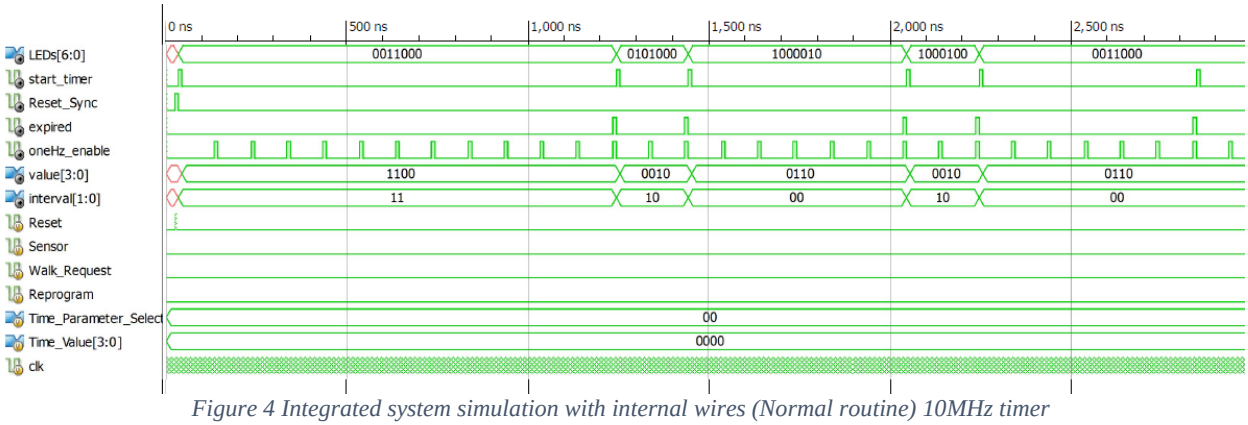

Figure 4 Integrated system simulation with internal wires (Normal routine) 10MHz timer

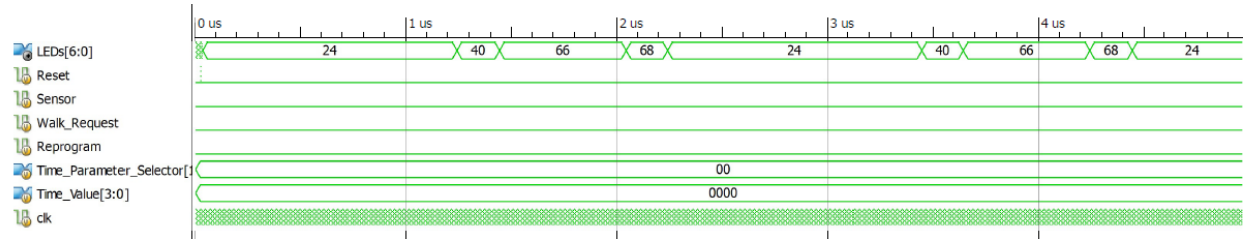

Figure 5 Full system simulation (Normal routine | timer at 10Mhz)

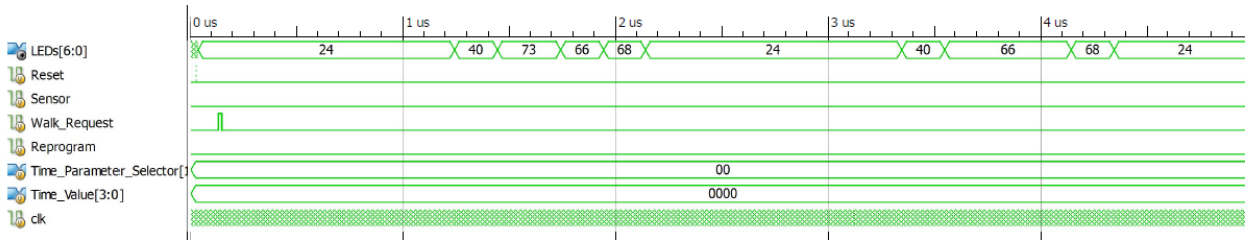

Figure 6 Full system simulation (Walk request | timer at 10Mhz)

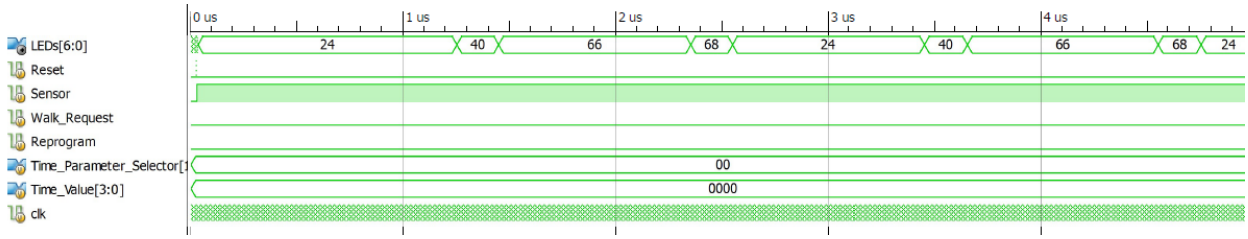

Figure 7 Full system simulation (Sensor activated | timer at 10Mhz)

States represented by decimal values

Decimal value	State	Binary value
24	A	0011000
40	B	0101000
66	C	1000010
68	D	1000100
73	E	1001001

Table 2 States represented by Decimals

Synchronizer

Module description

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.

Module overview

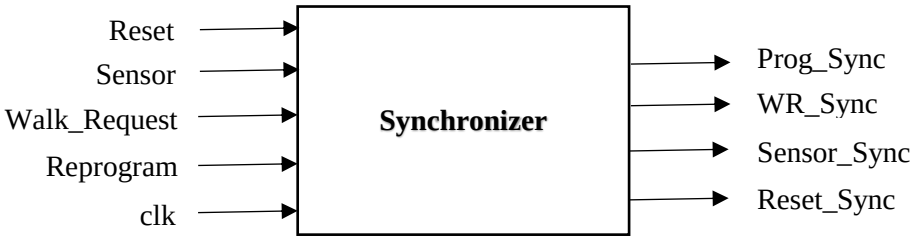

Figure 8 Synchronizer block diagram

Test bench output

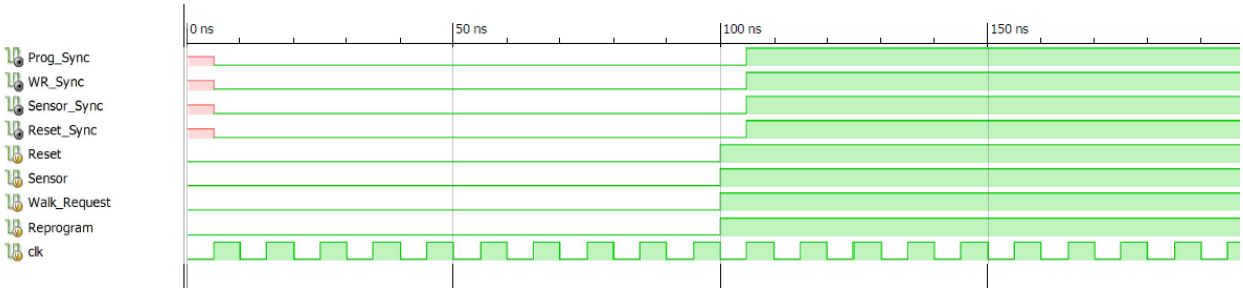

Figure 9 Synchronizer simulation

Walk Register

Module description

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.

Module overview

Figure 10 Walk Register block diagram

Test bench output

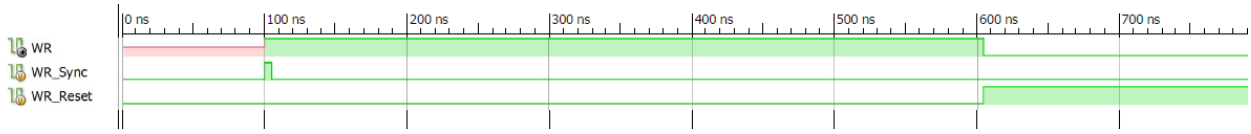

Figure 11 Walk register simulation

Time Parameters

Module description

The time parameters module stores the three different time parameter values, namely tBASE, tEXT, and tYEL 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.

Module overview

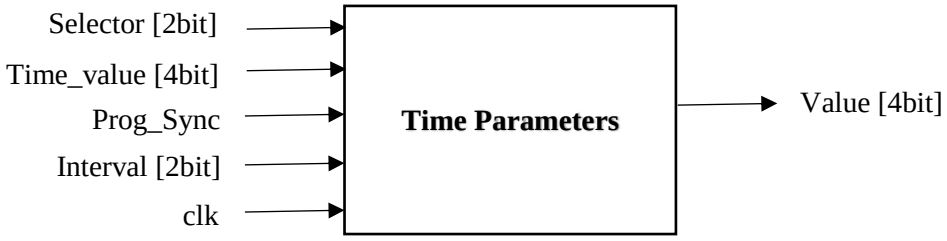

Figure 12 TimeParameters block diagram

Interval value is assigned for t_{BASE}, t_{EXT} and t_{YEL} as follows.

Interval	Symbol	Default Time (sec)
00	t_{BASE}	6
01	t_{EXT}	3
10	t_{YEL}	2
11	$2*t_{BASE}$	$2*t_{BASE}$

Table 3 Interval values

Times resets when the selector input is 00. Otherwise the values on the time_value input are assigned to the t_{BASE}, $t_{EXT}$and t_{YEL} respectively. Since register stores the values, selected values will be saved untill selector becomes 00 (Reset).

Test bench output

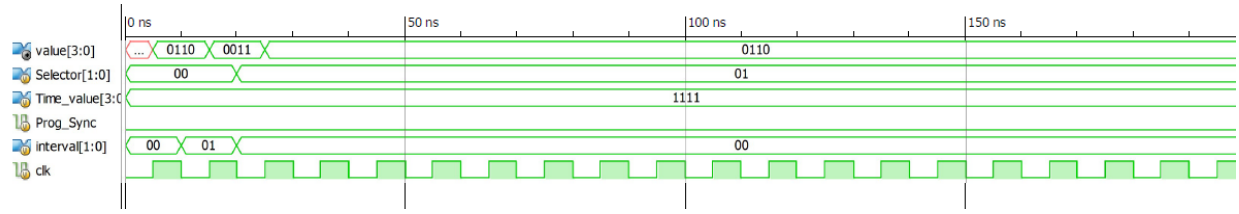

Figure 13 Time parameter simulation

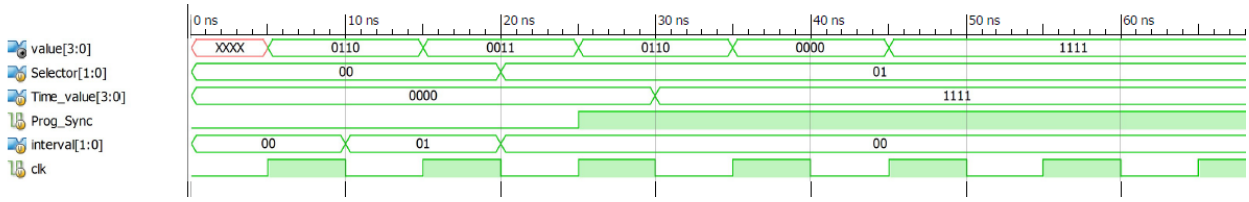

Figure 14 Changing t_{BASE} to 15 seconds

Divider

Module description

The divider is necessary for the timer to properly time the number of seconds for any 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.

Module overview

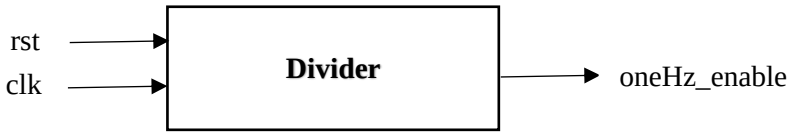

Figure 15 Divider block diagram

Test bench output

1Hz enable requires 10^6 clock cycles at 100MHz. Since it is output cannot be visualized clearly 10MHz pulses were generated instead of 1Hz pulses.

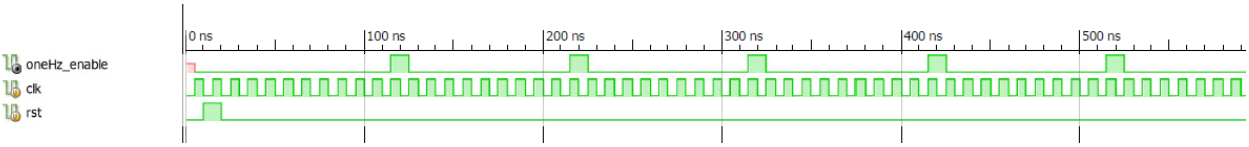

Figure 16 Divider at 10MHz

Timer

Module description

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.

Module overview

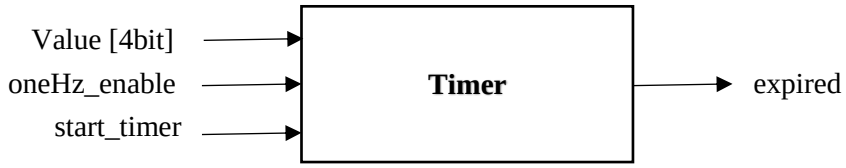

Figure 17 Timer block diagram

Since the value change is behind the start_timer by a one clock cycle this is solved by assigning value to the time_left after a one clock cycle. This way it does not change the time values.

Test bench output

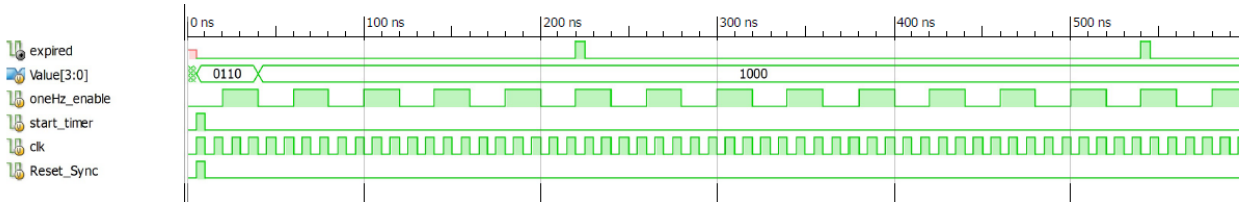

Figure 18 Timer simulation output

Finite State Machine

Module description

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.

Module overview

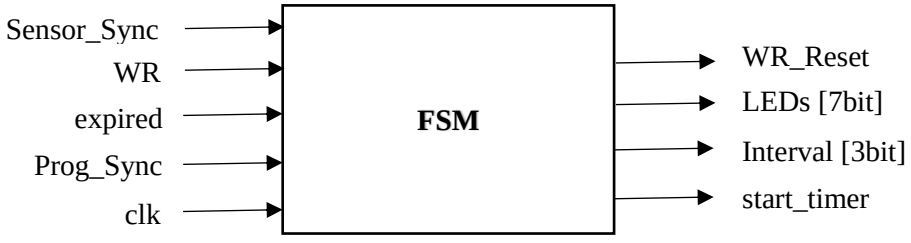

Figure 19 FSM block diagram

Test bench output

Normal routine

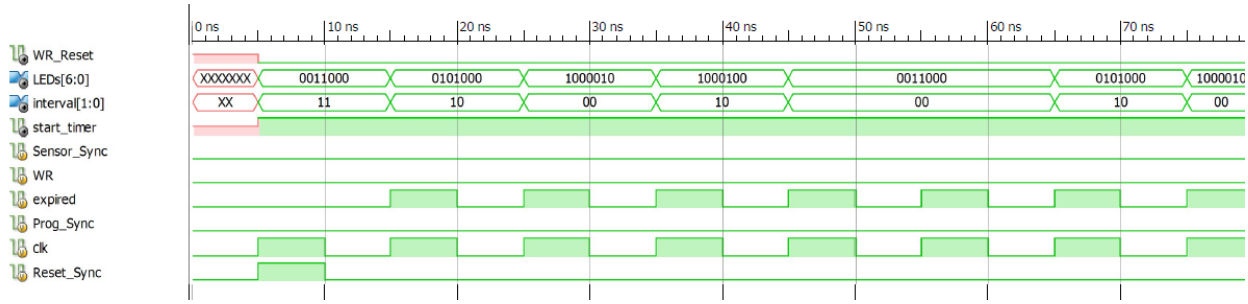

Figure 20 Normal operation

Walk request

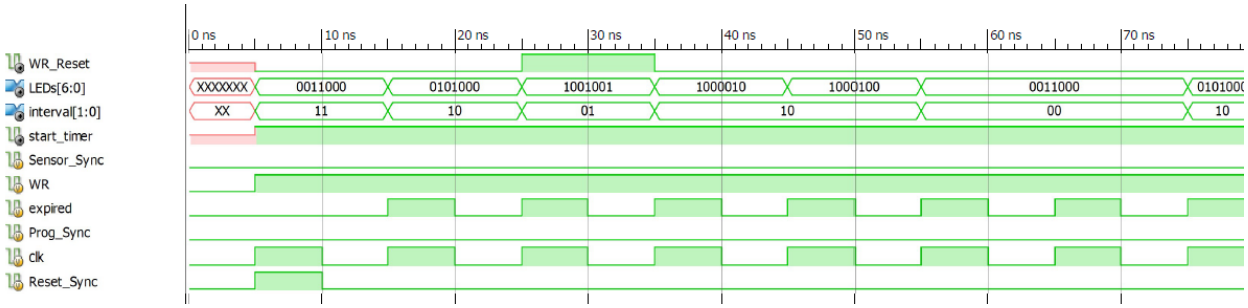

Figure 21 Walk request by a pedestrian

Vehicle sensor

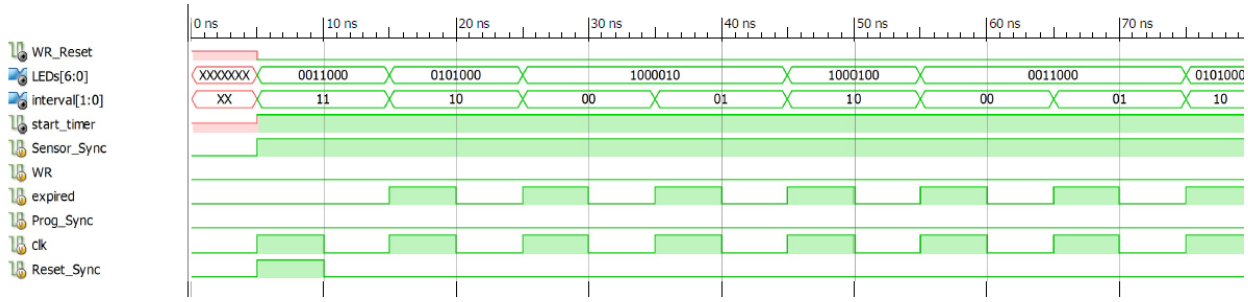

Figure 22 Vehicle sensor activated

At the start of the system, FSM directly enter the State A for a time of 2* t_{BASE} without considering about any other sensor inputs. This occurs when the system is reset.

LEDs values represent the Following states of the system

Bits of the LEDs represent lights in following manner

LED sequence: [Red_{main}, $Yellow_{main}$, $Green_{main}$, Red_{side}, $Yellow_{side}$, $Green_{side}$, Walk]

State	Representation	Meaning
A	0011000	Main green
B	0101000	Main yellow
C	1000010	Side green
D	1000100	Side yellow
E	1001001	walk

Interval values represent following timer values

Representation	Timer value
00	t_{BASE}
01	t_{EXT}
10	t_{YEL}
11	$2*t_{BASE}$