Traffic Light Controller

Problem Statement/Task:

Design, implement , Test and Demonstrate a simplified traffic light controller using verilog on Basys3 FPGA board.

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 SideStreet which tells the controller if there are cars still on the Side Street. This is summarised in Figure 1.

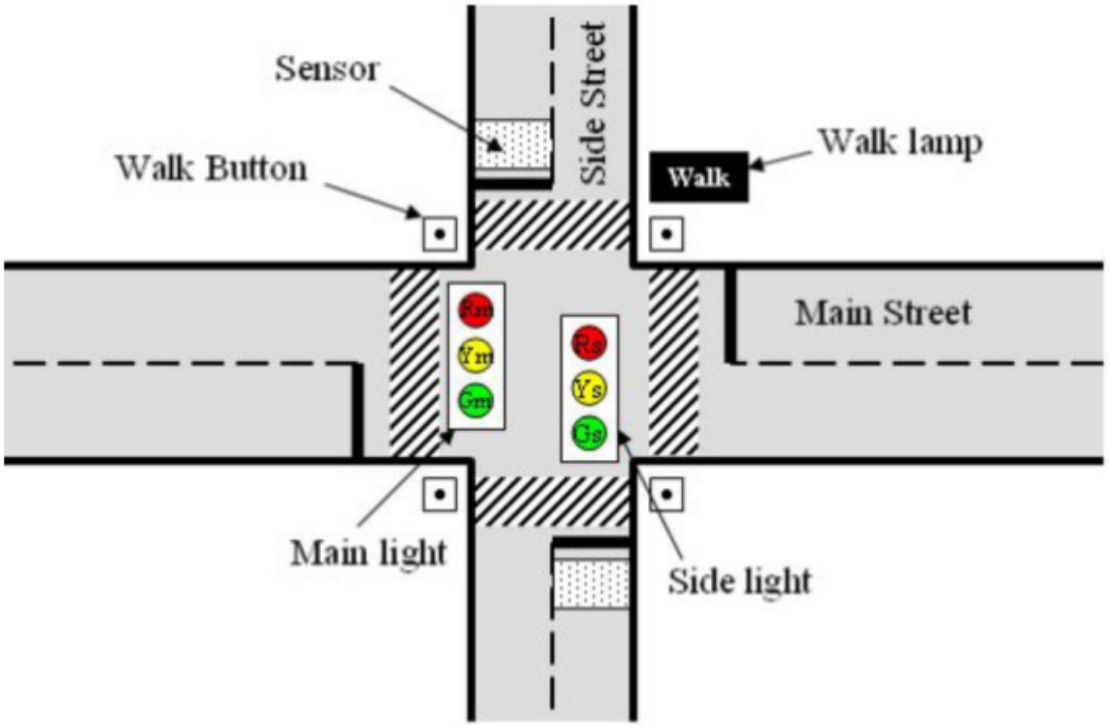


Figure 1

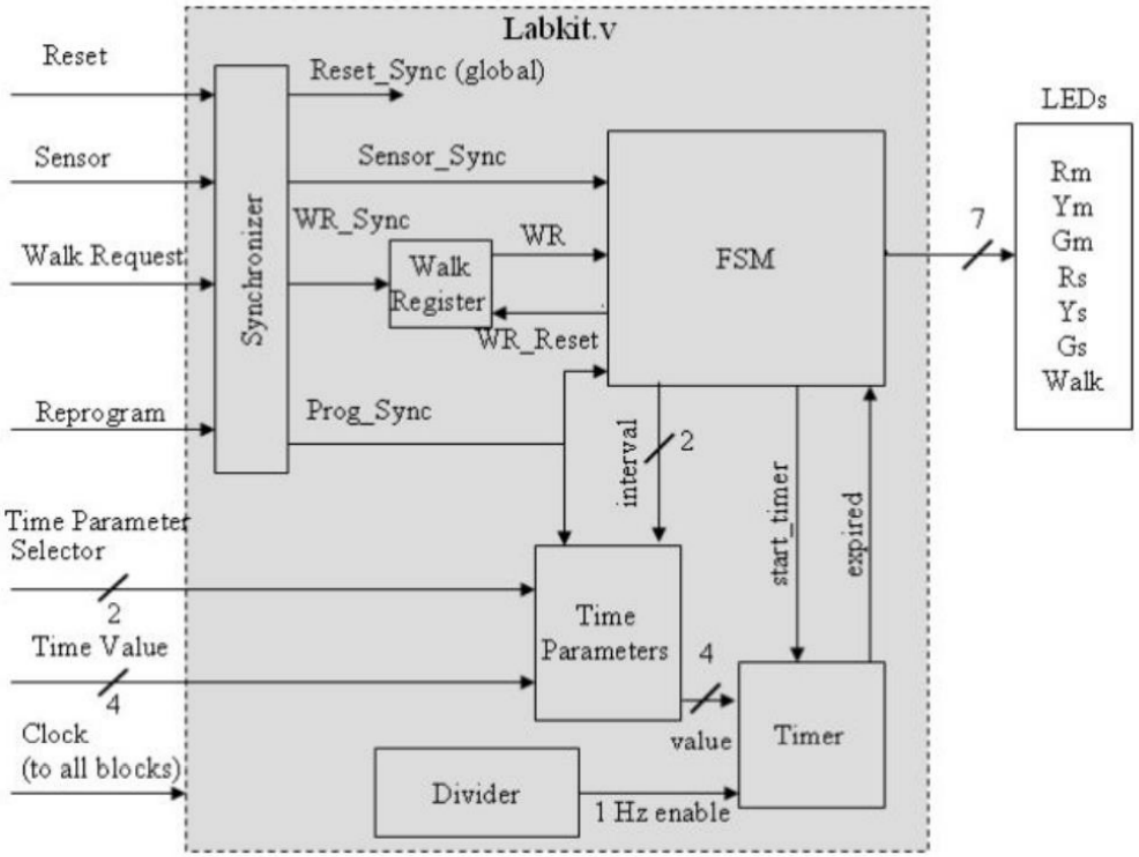
You may assume that the 4 walk buttons placed at each street corner are hooked into the traffic light
controller using a wired OR. For this reason, you may assume that the controller only needs a single
input called WalkRequest. The side street sensor is placed near the intersection to tell the controller

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. This input is named Sensor. The traffic lights are timed on three parameters (in seconds): the base interval (tBASE), the extended interval (tEXT), and the yellow light interval (tYEL). 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 Basys3 board 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

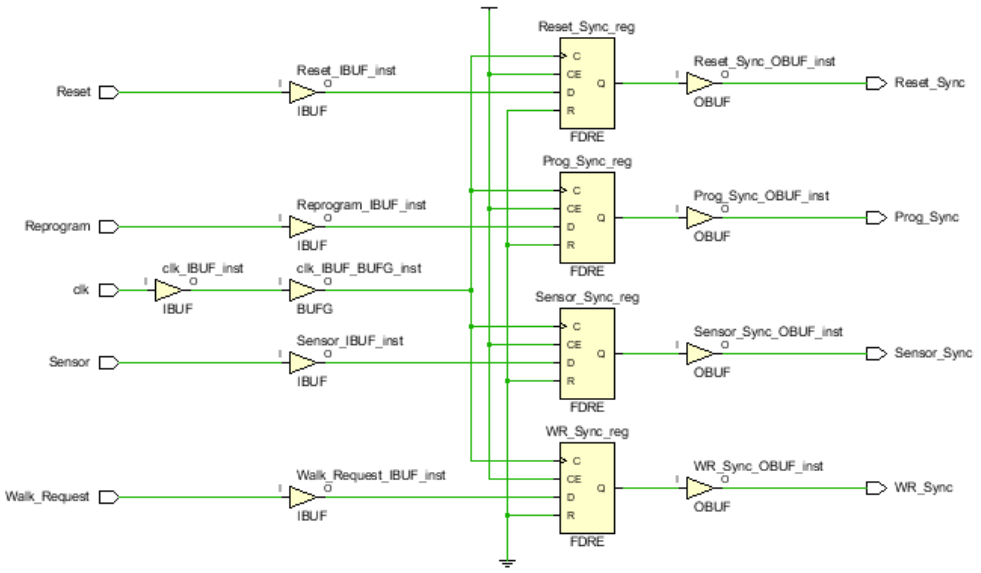
Interval Name	Symbol	Parameter Number	Default Time (sec)	Time Value
Base Interval	tBASE	00	6	0110
Extended Interval	tEXT	01	3	0011
Yellow Interval	tYEL	10	2	0010

The operating sequence of this intersection begins with Main Street having a green light for 2 lengths of tBASE seconds. Next, the Main lights turn to yellow for tYEL and then turn red while simultaneously turning on the Side Street green light. The Side Street is green for tBASE, and its yellow is held for tYEL. 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 streetlights to red and the walk light to on. After a walk of tEXT 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.

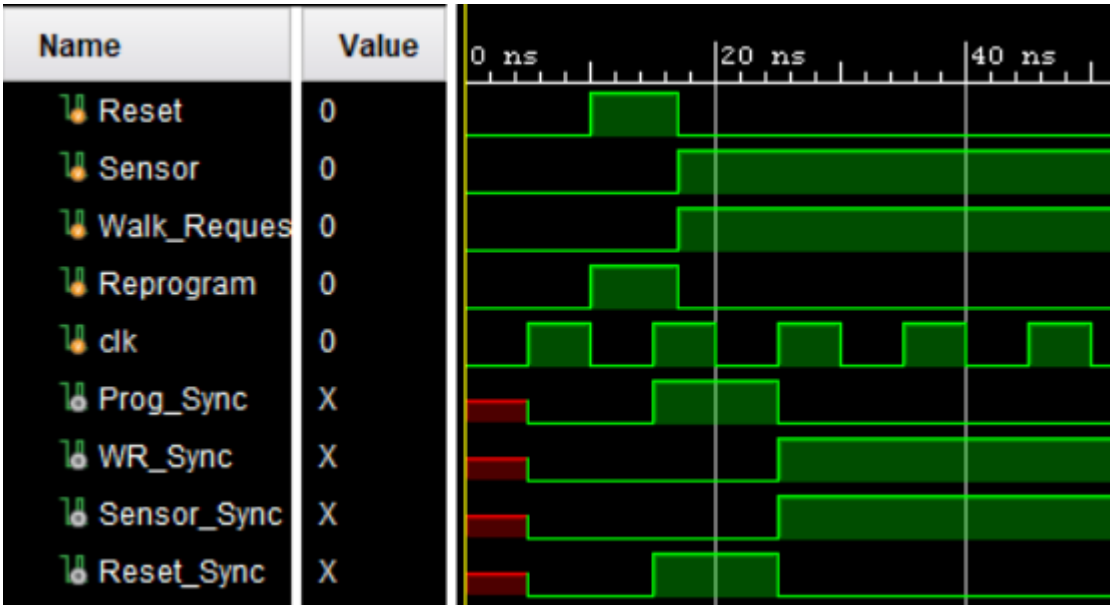


Synchronizer

The purpose of the synchronizer is to ensure that the inputs are synchronised to the system clock.



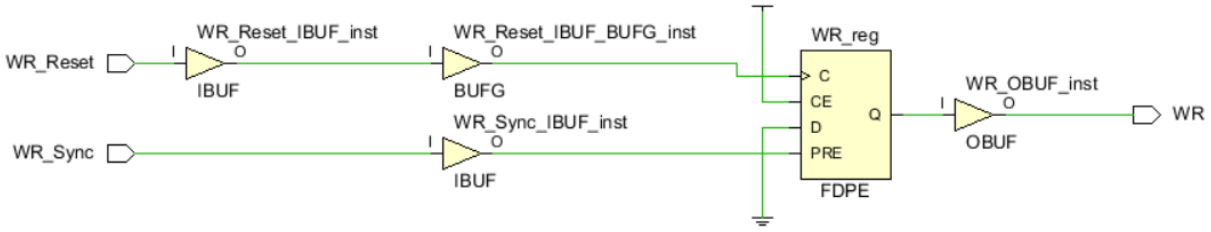
Synchronizer simulation



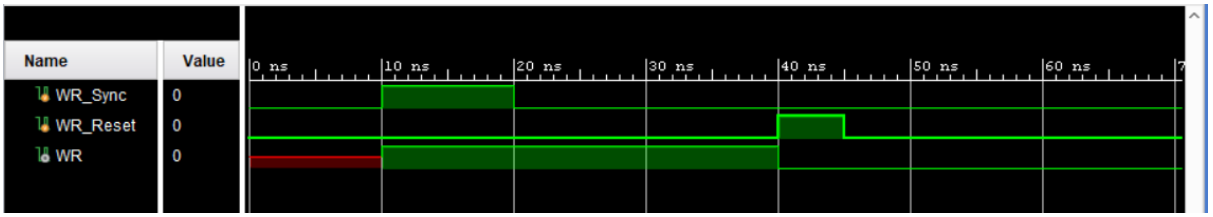
This simulation displays how the signals of reset, reprogram, sensor and walk request are synchronised to the clock.

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.



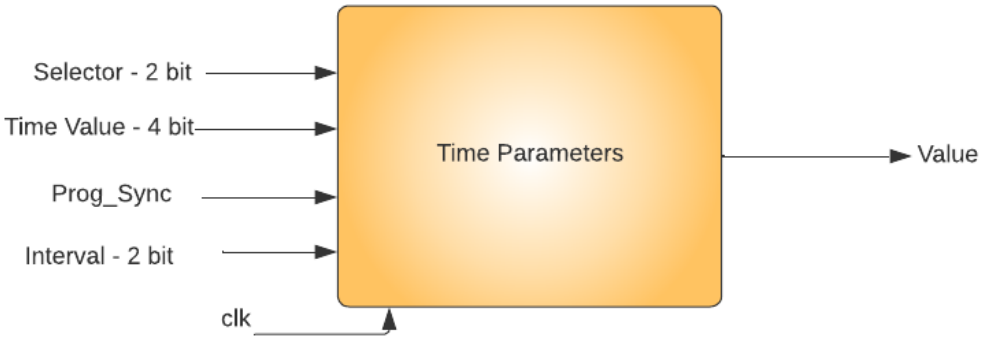
Walk Register simulation



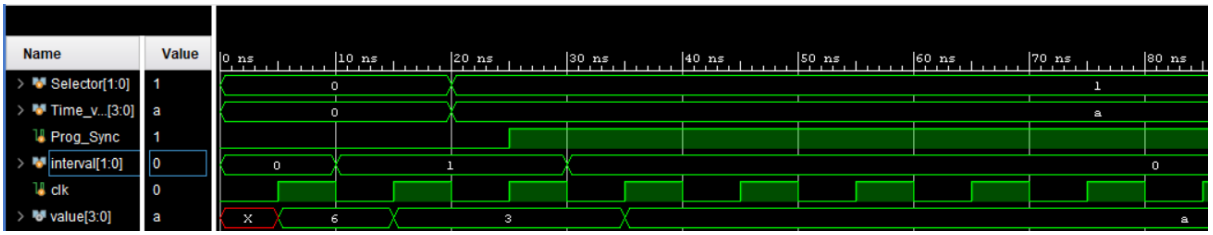
This simulation displays how a walk register picks up a walk request signal from the synchronizer and sends a walk request to fsm. Also the walk request reset function can be seen here.

Time Parameters

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.



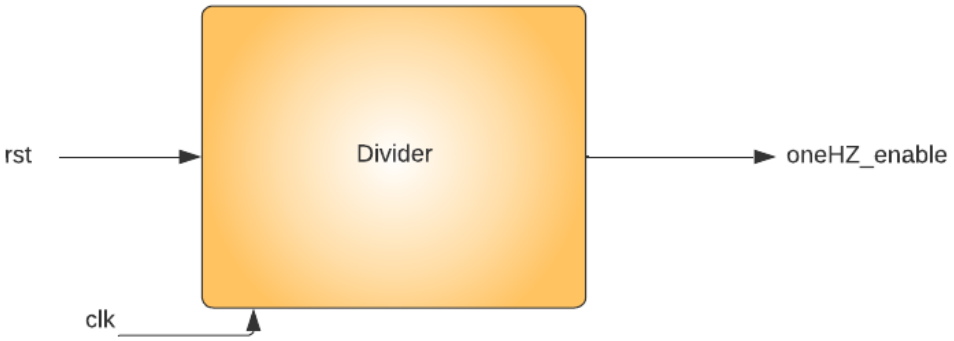
Time Parameters Simulation



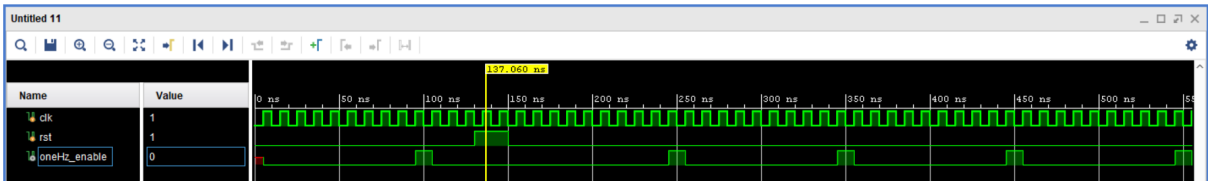
This simulation displays a scenario where 10 (a) is assigned to tBASE. By setting selector to tBASE parameter and setting the Time_value to 10 and pressing Reprogram.

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 1 sec.



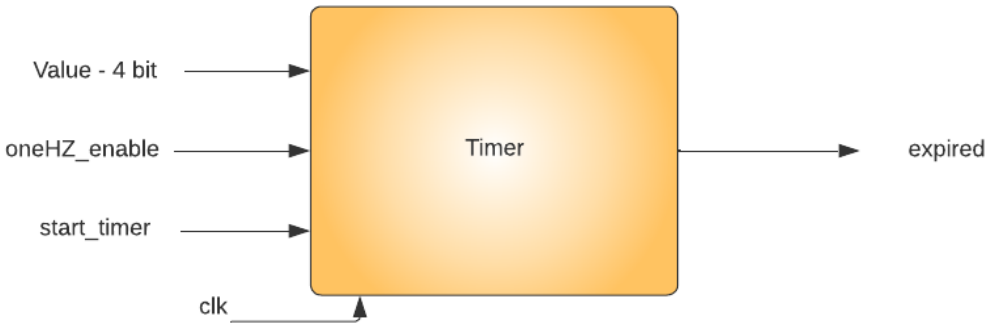
Divider simulation



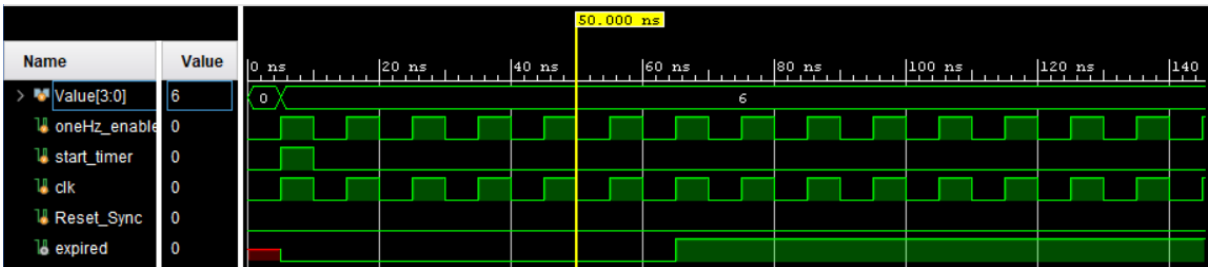
This simulation displays how the divider generates a oneHz enable and how the reset function works.

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.



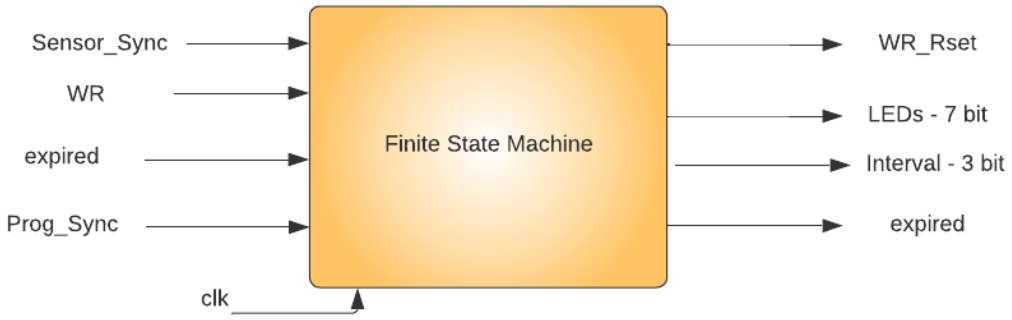
Timer simulation



This simulation shows how the timer handles a request of 6 seconds

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.



Finite state machine simulation

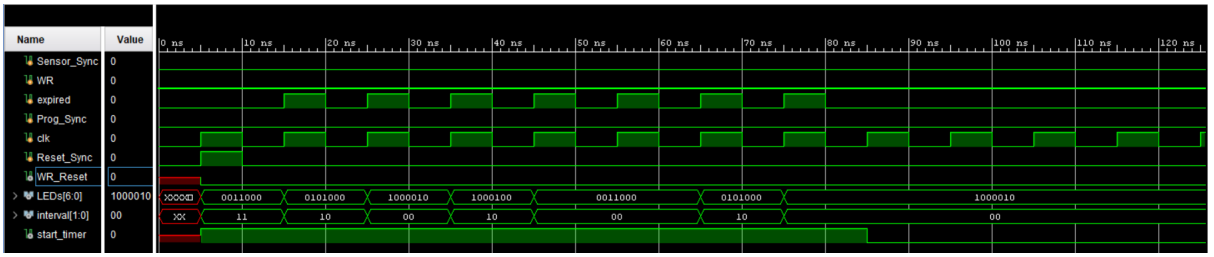
LEDs configurations:

0011000 - Main green
0101000 - Main yellow
1000010 - Side green
1000100 - Side yellow
1001001 - walk

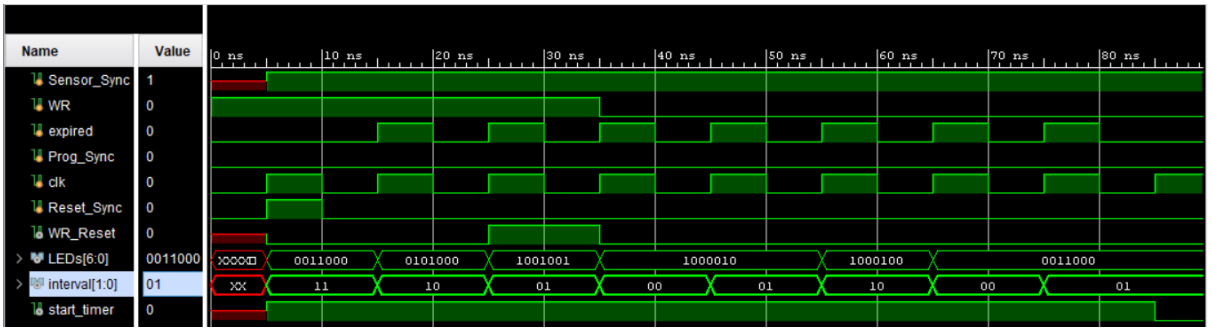
Interval configurations:

00 - tBASE
01 - tEXT
10 - tYEL
11 - 2*tBASE

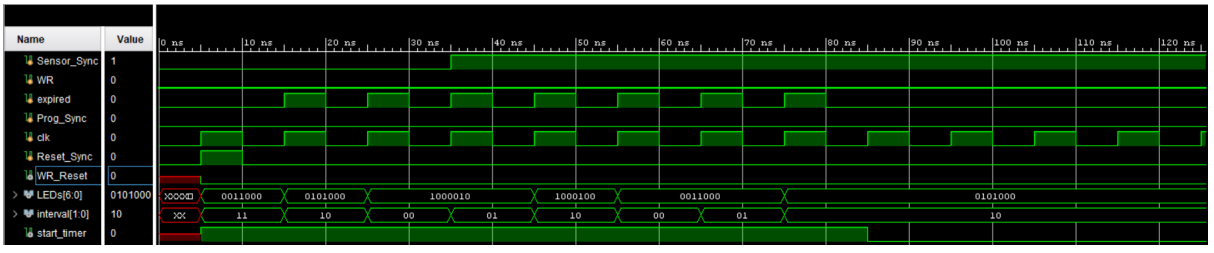
- **Normal**



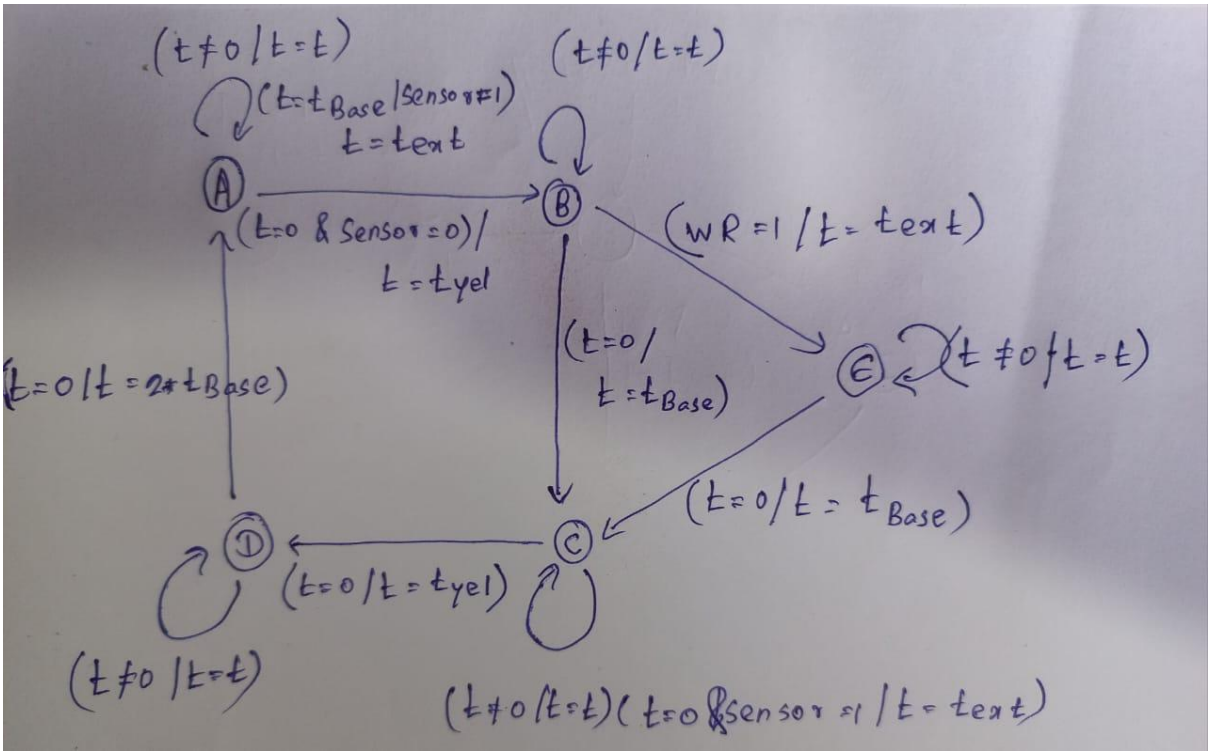
- **Walk Request + Sensor**



- **Sensor**



Finite State Machine state diagram



State	Main Street	Side Street	Walk lamp
A	Green	Red	Red
B	Yellow	Red	Red
C	Red	Green	Red
D	Red	Yellow	Red
E	Red	Red	Green

Git repo

https://github.com/narada98/trafficlight-controller-HDL.git