Adam Smith n3276931

Embedded Systems – Software

Timing Requirements Justification

The Following is my interpretation of the timing requirement of the assessment. I have named then in numerical order. My states and times are as follows are as follows:

-- 1 ALL\_RED (0.2)

-- 2 EV\_NS\_RA (0.2)

-- 3 EV\_NS\_G (5.0)

-- 4 EV\_NS\_A (3.0)

-- 5 EV\_EW\_RA (0.2)

-- 6 EV\_EW\_G (5.0)

-- 7 EV\_EW\_A (3.0)

-- 8 All-RED-BEFORE\_NS (0.2)

-- 9 NS\_RED\_AMBER (0.2)

-- 10 NS\_GREEN (5.0)

-- 11 NS\_AMBER (3.0)

-- 12 All-RED-BEFORE\_EW (0.2)

-- 13 EW\_RED\_AMBER (0.2)

-- 14 EW\_GREEN (5.0)

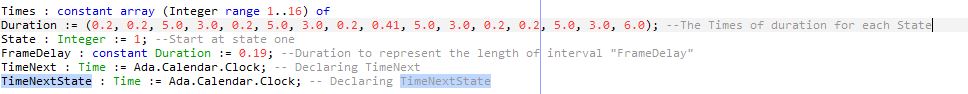
-- 15 EW\_AMBER (3.0)

-- 16 PEDESTRIAN\_GREEN (6.0)

**11. The delay from a pedestrian pressing a button until receiving a green light should be no more than 60 seconds:**

From one pedestrian green light to the next without any events triggered would be 16.6 seconds. For the Pedestrian green light to be more than 60 seconds the EV Sensor in either direction would have to come nearly 3 times. Since state 15 takes priority from EV over the Pedestrians. The states would reach 11 and check for N/S EV. When state 15 is checks for E/W EV and takes priority over Pedestrian green light, in this case State 16. I track the last direction so that is EV N/S is triggered, when normal states resume the direction will be E/W. Worst case scenario is that EV will get a green light in 8.8 seconds, with 5 seconds for green and ten seconds after the EV sensor have gone that is 22.8 seconds. This worst case scenario states that is EV Sensor appears 2.8 times then the Pedestrians will be waiting longer than the ICA specification.

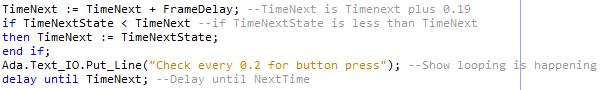
**12. The delay from a pedestrian pressing a button until the wait light is lit should be no more than 0.2 seconds:** In order to display the wait light in less than 0.2 seconds I made the following declarations at the start of my controller. I set the duration to represent the length of interval ‘FrameDelay’. I’ve already predefined the library ‘Calender’ and I have defined ‘TimeNext’ and ‘TimeNextState’ which are both of type Time.



Within my states manipulation if statement I have set ‘TimeNextState’ to Ada.Calendar.Clock + ‘Times’ which takes the parameter of ‘State’. ‘Times’ is the name of my constant array of integers which store the time for each State.



Right at the end of my loop I have code which first sets ‘TimeNext’ equal to ‘TimeNext’ + ‘FrameDelay’. There is then an if statement which says that if ‘TimeNextState’ is smaller than ‘TimeNext’ then ‘TimeNext’ is ‘TimeNextState’. The last line within this block of code is ‘delay until TimeNext’ which picks the smaller of ‘TimeNext’ and ‘TimeNextState’.



**13. The delay between a traffic green light turning off until it is next lit must not be more than 60 seconds:** With no other events triggered, from one traffic green turning off until it is next lit takes 11.6 seconds. I have programmed my code in such a way that after an EV green light it will track the direction it was prior. So if it was North/South Green and then North/South Green EV, the next direction will be East/West. The timing threshold would be more than 60 seconds as explained in 11, if the EV was re-triggering causing the time to be greater than 60 seconds.

**14. The delay between a priority sensor first indicating the presence of an emergency vehicle until that green traffic light is lit should be no more than 10 seconds:** In order to get the green light within 10 seconds of the EV sensor first indicating the EV green is as follows. If N/S or E/W EV sensor is triggered I have the same times and events in place for both. They are checked at their amber stage and if EV = 1 then state will go to 1. When state is 1 N/S and E/W will go to their EV states respectively.

Worst case scenario, in this instance North/South, will go through All\_Red, NS\_RED/Amber, NS\_Green, NS\_Amber, ALL\_RED, EV\_NS\_Red/Amber, EV\_ NS\_Green. That will produce the following time in seconds, 0.2, 0.2, 5.0, 3.0, 0.2, 0.2, GREEN. = 8.8 Seconds.