INTELLIGENT TRAFFIC LIGHT CONTROL AND MONITORING USING STATE TRANSITION TABLE (STT)

Ihedioha Ahmed C. and Eneh Ifeanyichukwu I.

Enugu State University of Science and Technology Enugu, Nigeria

---

ABSTRACT

Vehicular traffic is continuously increasing around the world, especially in large urban areas. The resulting congestion has become a major concern to transportation specialists and decision makers. The existing methods for traffic management, surveillance and control are not adequately efficient in terms of performance, cost, maintenance, and support. In this paper, the design of a system that utilizes and efficiently manages traffic light controllers is presented. A performance evaluation of the Algorithmic State Machine (ASM) chart developed and digital logic architectures were done using parameters such as control pattern memory demand. I/O line demand for complex systems, component count, complexity reduction, generic nature, flexibility, scalability, fault tolerance and code reusability was realized. The system designed featured a queue detecting device. The advantage of this is that unlike the conventional traffic light system, it does not necessary have to give right of way to a route when there is no queue within a given range rather it apportions right of way to the next route if there is a queue. It also combines a thorough initial hardware design approach leading to the State Transition Table (STT) before generating the corresponding software that has just one statement per link path in the STT.

Keywords: Traffic, State Transition Table (STT), Algorithmic State Machine (ASM), Digital logic.

1.0 INTRODUCTION

Traffic light control is one of the main means of controlling road traffic. Improving traffic control is important because it can lead to higher traffic throughput and reduced congestion.

At the same time, improving traffic control is difficult because the traffic system, when it is modeled with some degree of realism, is a complex, nonlinear system with large state and action spaces, in which suboptimal control actions can easily lead to congestions that spread quickly and that are hard to dissolve.

In practice, most traffic lights use very simple protocols that merely alternate red and green lights for fixed intervals. The interval lengths may change during peak hours but are not otherwise optimized [1].

A traffic signal is typically controlled by a controller inside a cabinet mounted on a concrete pad. Some electro-mechanical controllers are still in use. However, modern traffic controllers are solid state. The

cabinet typically contains a power panel, to distribute electrical power in the cabinet; a detector interface panel, to connect to loop detectors and other detectors; detector amplifiers; the controller itself; a conflict monitors unit; flash transfer relays; a police panel, to allow the police to disable the signal; and other components [2].

Such techniques are attractive because they can automatically discover efficient control strategies for complex tasks, such as traffic control, for which it is hard or impossible to compute optimal solutions directly and hard to develop hand-coded solutions. First the State Transition Table (STT) is described that is used to control traffic lights in this work. In this framework, multiple local controllers (agents) are each responsible for the optimization of traffic lights around a single traffic junction, making use of locally perceived traffic state information (sensed cars on the road), a learned probabilistic model of car behavior, and a learned value function which indicates how traffic light decisions affect long-term utility, in terms of the average waiting time of cars.

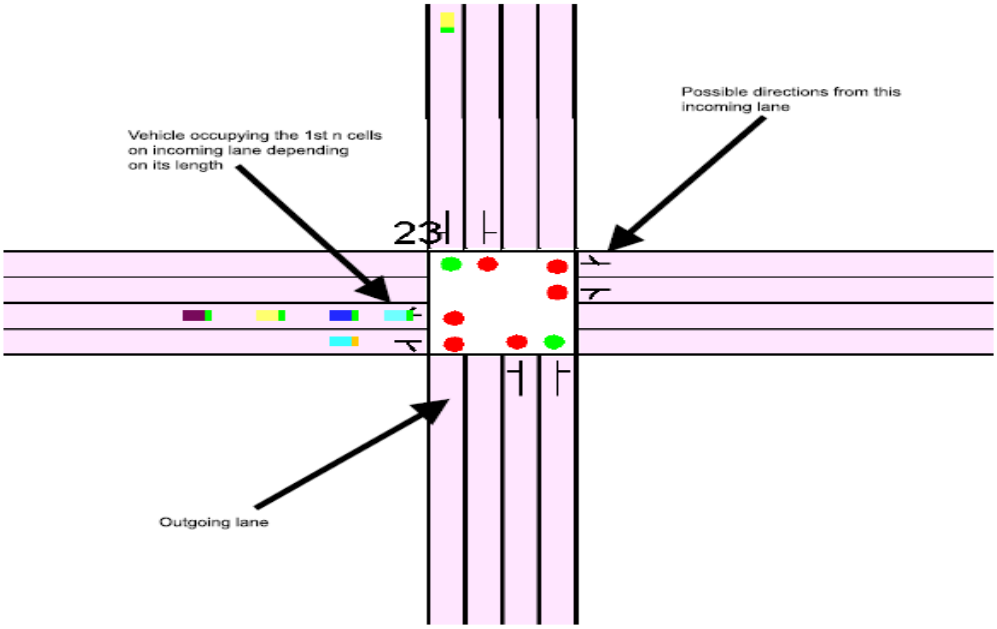


Fig. 1: Example traffic light intersection [3].

2.0 TRAFFIC CONTROLLER CONCEPTS

Traffic controllers use the concept of phases, which are directions of movement grouped together. For instance, a simple intersection may have two phases: North/South, and East/West. A 4-way intersection with independent control for each direction and each left-turn, will have eight phases. Controllers also use rings; each ring is an array of independent timing sequences. For example, with a dual-ring controller, opposing left-turn arrows may turn red independently, depending on the amount of traffic. Thus, a typical controller is an 8-phase, dual ring control.

Solid state controllers are required to have an independent conflict monitor unit (CMU), which ensures fail-safe operation. The CMU monitors the outputs of the controller, and if a fault is detected,

the CMU uses the flash transfer relays to put the intersection to FLASH, with all red lights flashing, rather than displaying a potentially hazardous combination of signals. The CMU is programmed with the allowable combinations of lights, and will detect if the controller gives conflicting directions a green signal, for instance [4].

In this research, a traffic light control system with queue detecting ability featured as one of the Algorithmic State Machine (ASM) chart based design was designed and implemented. A performance evaluation of the Algorithmic State Machine (ASM) chart developed and digital logic architectures were done using parameters such as control pattern memory demand. I/O line demand for complex systems, component count, complexity reduction, generic nature, flexibility, scalability, fault tolerance and code reusability was realized.

Fig. 2: A typical traffic light control system [5]

3.0 DESIGN AND IMPLEMENTATION OF THE SYSTEM

3.1 DESIGN

State Transition Table (STT) based process control software design is universal and can be applied to any system that can be represented with and Algorithmic State Machine (ASM) chart. In this research, a traffic light control system is designed for a T-junction. The sequence of movement for vehicles desired is in three phases as shown in Fig. 3a through 3c. The arrows without a bar (-) in front in each case indicates the lanes that are passed while the arrows with a bar (-) in front are those on hold. A queue detector is featured in this design such that when there are no more vehicles on a direction that has right of access, the next direction is triggered.

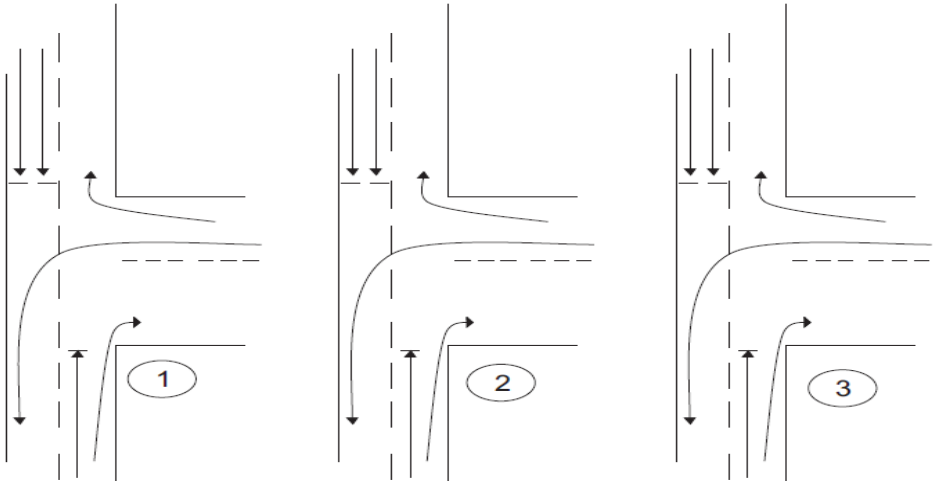


Figure 3. Traffic flow sequence in a T- junction

3.2 IMPLEMENTATION

The Algorithmic State Machine (ASM) chart of figure 4 shows the control flow for sequence shown in fig 3a through 3c. The conditional output HCLRT is used to clear the timing for the present sequence and move to the next when no queue is detected.

In the ASM chart (figure 4), N, NE and LF are used in differentiate the traffic light line in each of the three directions. HGRNLE, HAMBLE HREDLE represents the Green number and Red lights of one direction. HGRNN, HAMBN, HREDN are for the next direction while HGRNL, HAMBE, HREDE are for the third direction.

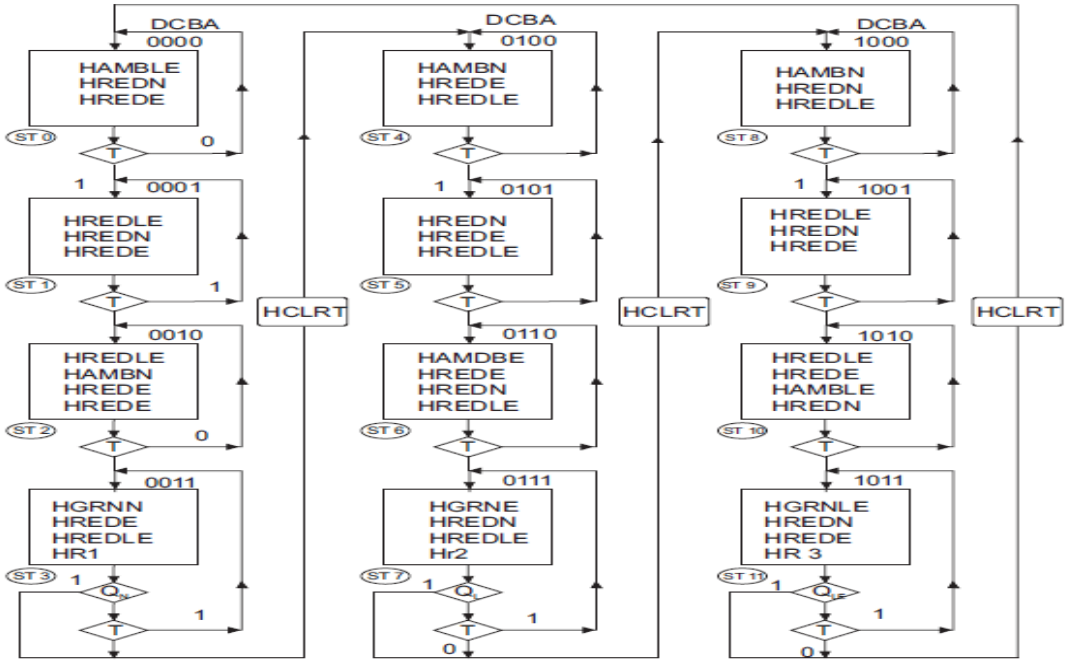
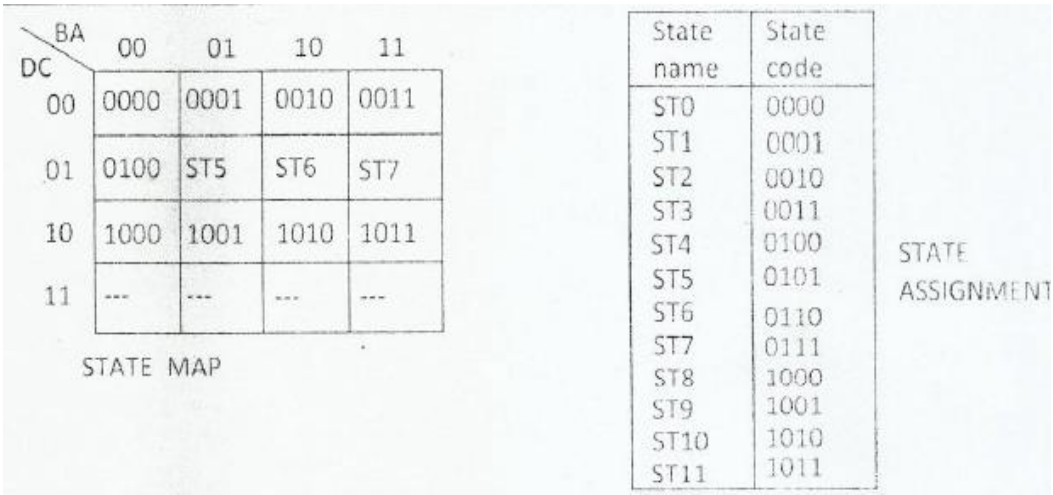


Figure 4:ASM chart for traffic light control at a T- junction with queue detectors Q , Q , Q NELE



BA / DC	00	01	10	11
00	0000	0001	0010	0011
01	0100	ST5	ST6	ST7
10	1000	1001	1010	1011
11	---	---	---	---

STATE MAP

State name	State code
ST0	0000
ST1	0001
ST2	0010
ST3	0011
ST4	0100
ST5	0101
ST6	0110
ST7	0111
ST8	1000
ST9	1001
ST10	1010
ST11	1011

STATE ASSIGNMENT

Figure 5: State Map and State assignment

In an ASM chart, a regular box is called a state [6]. A state has a state name in the lower left hand corner and state code at the top right hand corner. Thus the ASM chart of fig 4 has twelve states. Since no two states may have the same state code, a state map followed by a state assignment is used to assign state code to ensure that this is achieved (fig 5). Typically, when the ASM chart is going to be used for hardware design, the state codes are assigned such that only one bit changes as one moves from one state code to the next either above or below it. However in software based ST1 for micros, it is more convenient to assign the state codes materially in order that the corresponding ease constructs based on state code might be easier to follow. In the ASM chart of fig 4, HCLRT is a conditional output.

3.2.1 State Transition Table (STT)

The information contained in an ASM chart (fig 4) can be represented as a flat table (table 1) called State Transition Table (STT). The reason for the transition from ASM chart to STT is that the remaining software or logic design steps are easier to visualize from a table than from the corresponding ASM chart. The qualifiers T, QE, QLE QN, and the present states D, C, B, A would constitute the input needed for this operation. Similarly, there are 15 output lines, D', C', B', A', HAMBLE, HREDN, HREDE, HREDLE, HAMBN, HGRNN, HR2, HAMBE, HGRNE, HR1, HGRNLE.

Table 1: State transition table for the traffic light control

Link Path	Present state name	Present state code	Qualifiers	Next State name	Next State code	State Output	Conditional Output	Output in **HEX**
		DCBA	T Q_N Q_E Q_{LE}		D'C'B'A'	HAMBLE HREDN HREDE HREDLE HAMBN HGRNN HR1 HAMBE HGRNE HR2 HGRNLE HR3	HCRT	
L1	ST0	0000	0 - - -	ST0	0000	1 1 1 0 0 0 0 0 0 0 0 0	0	00
L2	ST0	0000	1 - - -	ST1	0001	1 1 1 0 0 0 0 0 0 0 0 0	0	02
L3	ST1	0001	1 - - -	ST1	0001	0 1 1 1 0 0 0 0 0 0 0 0	0	02
L4	ST1	0001	0 - - -	ST2	0010	0 1 1 1 0 0 0 0 0 0 0 0	0	04
L5	ST2	0010	0 - - -	ST2	0010	0 1 1 1 1 0 0 0 0 0 0 0	0	04
L6	ST2	0010	1 - - -	ST3	0011	0 1 1 1 1 0 0 0 0 0 0 0	0	06
L7	ST3	0011	1 0 - -	ST3	0011	0 1 1 0 1 1 0 0 0 0 0 0	0	06
L8	ST3	0011	0 0 - -	ST4	0100	0 1 1 0 1 1 0 0 0 0 0 0	1	09
L9	ST3	0011	- 1 - -	ST4	0100	0 1 1 0 1 1 0 0 0 0 0 0	1	09
L10	ST4	0100	0 - - -	ST4	0100	0 0 1 1 1 0 0 0 0 0 0 0	0	08
L11	ST4	0101	1 - - -	ST5	0101	0 0 1 1 1 0 0 0 0 0 0 0	0	0A
L12	ST5	0101	1 - - -	ST5	0101	0 1 1 1 0 0 0 0 0 0 0 0	0	0A
L13	ST5	0110	0 - - -	ST6	0110	0 1 1 1 0 0 0 0 0 0 0 0	0	0C
L14	ST6	0110	0 - - -	ST6	0110	0 1 1 1 0 0 0 1 0 0 0 0	0	0C
L15	ST6	0111	1 - - -	ST7	0111	0 1 1 1 0 0 0 1 0 0 0 0	0	0E
L16	ST7	0111	1 – 0 –	ST7	0111	0 1 0 1 0 0 0 0 1 1 0 0	0	0E
L17	ST7	0111	0 – 0 –	ST8	1000	0 1 0 1 0 0 0 0 1 1 0 0	1	11
L18	ST7	0111	- - 1 -	ST8	1000	0 1 0 1 0 0 0 0 1 1 0 0	1	11
L19	ST8	1000	0 - - -	ST8	1000	0 1 0 1 0 0 0 1 0 0 0 0	0	10
L20	ST8	1000	1 - - -	ST9	1001	0 1 0 1 0 0 0 1 0 0 0 0	0	12
L21	ST9	1001	1 - - -	ST9	1001	0 1 1 1 0 0 0 0 0 0 0 0	0	12
L22	ST9	1001	0 - - -	ST10	1010	0 1 1 1 0 0 0 0 0 0 0 0	0	14
L23	ST10	1010	0 - - -	ST10	1010	1 1 1 1 0 0 0 0 0 0 0 0	0	14
L24	ST10	1010	1 - - -	ST11	1011	1 1 1 1 0 0 0 0 0 0 0 0	0	16
L25	ST11	1011	1 - - 0	ST11	1011	0 1 1 0 0 0 0 0 0 0 1 1	0	16
L26	ST11	1011	0 - - 0	ST0	0000	0 1 1 0 0 0 0 0 0 0 1 1	1	01
L27	ST11	1011	- - - 1	ST0	0000	0 1 1 0 0 0 0 0 0 0 1 1	1	01

To limit the number of direct output lines from the processor to 8 or less, the state output are realized by decoding each present state. The next state codes (4 bits) and the conditional output HCLRT (1-bit) are generated directly from the output port. This approach reduced the number of direct output lines needed to just 5. A 4-to-16 line decoder can be used to generate all of the state outputs using the state codes C, B, B, A as the control input to the 4-to-16 line decoder.

However, each output exists in more than one state of the ASM chart and this necessitates ORing of a number of decoded states for each output signal. Since an exhaustive decoder followed by OR-tie constitutes a ROM, it is better to use a ROM to achieve the dual purpose of decoding states and ORing different signals. This leads to the optimized design shown in fig 6 at the output side. The signals from the ROM on the output side are used to turn the traffic lights ON or OFF via the traffic light interface. The interface could be implemented using Solid State Switches (SSS) such that when a signal from the ROM is logic 1, the corresponding light turns ON and when it is logic 0, the light turns OFF. Thus a SSS receives a binary control input and connects AC power to the corresponding light if the binary control input is a '1'. The AC power is cut OFF from the corresponding light if otherwise as shown in fig 7.

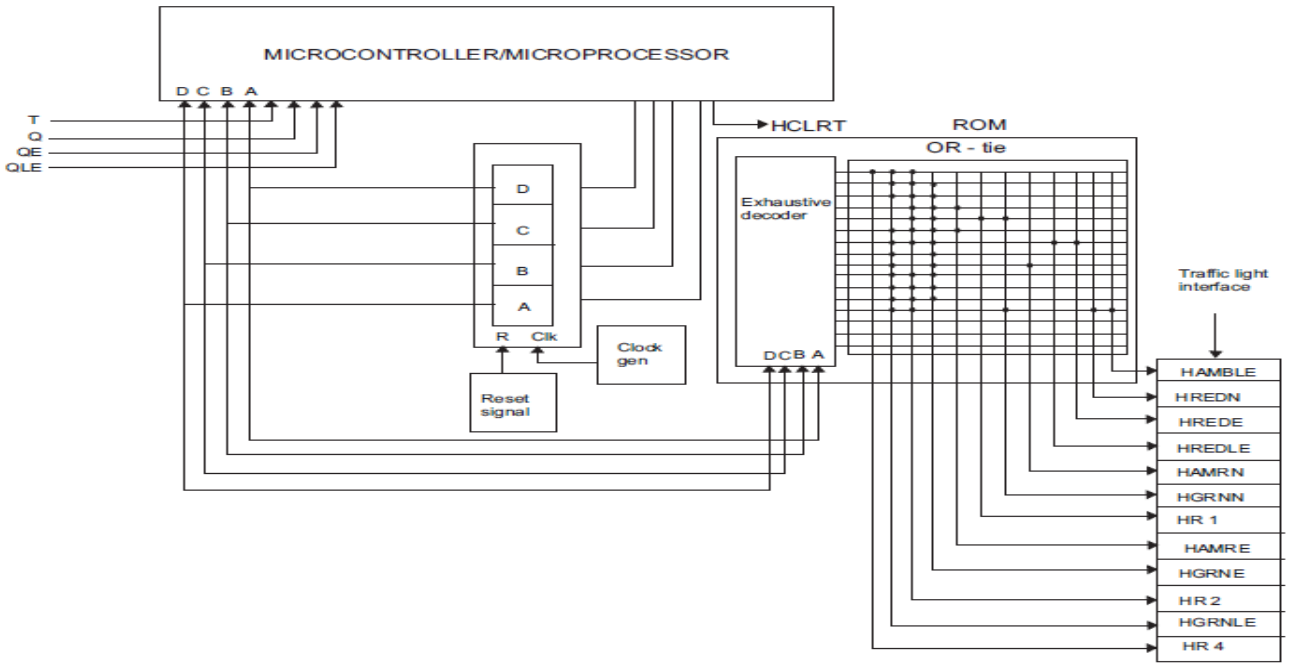


Figure 6: Microprocessor based implementation of T-junction traffic light control system.

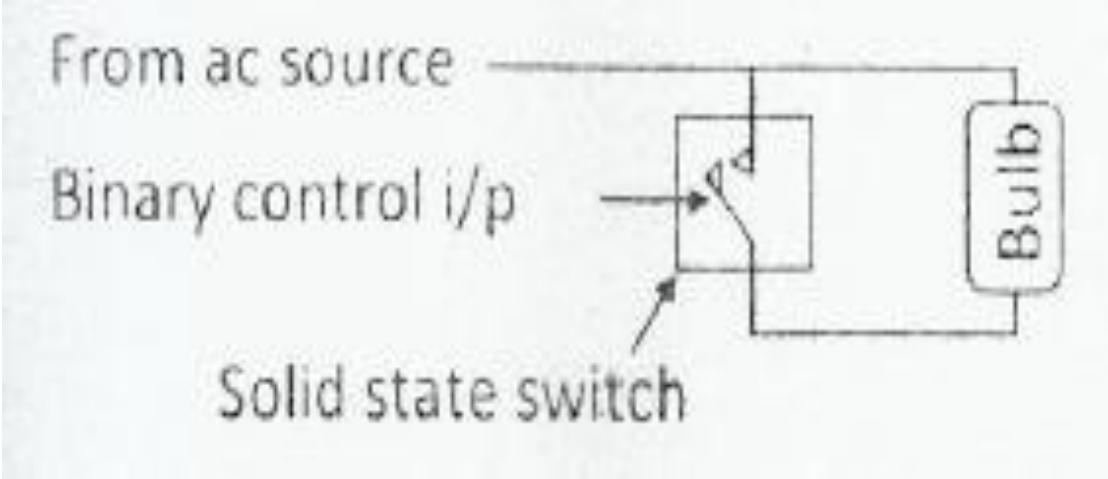


Figure 7: The AC power

3.2.2 The pseudo code of the STT-based software

Since the STT (table 1) already has the present code before the qualifier bits as desired, one can now proceed to generate STT- based software, using state code in a case construct. When each state code is selected, the output options are determined by the qualifier bits attendant in that state.

The pseudo code of the STT-based software now follows:

Do case

0: If T=0 then output=0//link path 1

Else output=2 End if //link path 2

1: If T=0 then output=2 //link path 3

```
        Else output=4 End if //link path 4
2: If T=0 then output =4 //link path 5
        Else output=6 End if // link path 6
3: If T=1 And Qn=0 then output=6 //link path 7
        Else if T=0 And Qn=0 then output=9 // link path 8
                Else if Qn=1 then output = 9 End if // link path 9
        End if
    End if
4: If T=0 then output = 8//link path 10
        Else output= 0Ah End if // link path 11
5: If T=1 then output = 0Ah //link path 12
        Else output =0Ch End if // link path 13
6: If T=0 then output = 0Ch //link path 14
        Else output = 0Eh End if // link path 15
7: If T=1 And Qe=0 then output =0Eh // link path 16
        Else if T=0 And Qe=0 then output =11h//link path 17
                Else if Qe=1 then output =11 End if// link path 18
        End if
   End if
8: If T=0 then output = 10h //link path 19
        Else output =12h End if// link path 20
9: If T=1 then output = 12h // link path 21
        Else output =14h End if //link path 22
10: If T=0 then output = 14h//link path 23
        Else output =16h End if//link path 24
11: If T=1 And Qle=0 then output =16h//link path 25
        Else if T=0 And Qle=0 then output = 01h//link path 26
                Else if Qle=1 then output =01 End if // link path 27
        End if
     End if
End case.
```

1) 3.2.3 Evaluation of the STT based design for micros with respect to ROM

If a fully expanded ROM-based design were to be implemented for the STT of Table 1, it would have been necessary to first expand the Table so as to eliminate the dashes (-), replacing them with binary combinations. A dash entry in Table 1 implies that the qualifier at the top of that column does not feature in the states where it is represented as a dash. Because a dash cannot be programmed into a ROM, each combination of dashes would have to be replaced by the full binary combinations that are possible. Thus, each row in the STT with two dashes (- -) need to be replicated four times in each of where the two dashes are represented either as 00 or 01 or 10 or 11. Similarly any row with 3 dashes

leads to 8 replications in each of which the 3 dashes are represented as either 000 or 001 or 010 or 011 or 100 or 101 or 111 and so on. Thus for a fully expanded STT needed for a ROM based design. Table 1 would have 192 rows instead of 27 rows in the original STT. This is called combinatorial explosion, which is a drawback for ROM-based designs, where unwanted combinations that contribute nothing to the logic would have to be represented in an exhaustive decoding process associated with ROMs.

Contrast this with Programmable Logic Array (PLA)-based design where a selective address decoder is used to represent only the rows shown in Table 1 ignoring dashes. Thus the PLA height would be much shorter than the corresponding ROM height.

The STT based software engineering shown in the foregoing used just one program statement to represent each row (or link path) of the STT. Thus it closely approximates a PLA based design by not insisting on exhaustive decoding of unused combinations represented as dashes. Therefore for microprocessor/microcontroller based design, the STT based software engineering approach combines the selective decoding feature of PLAs with the flexibility of software associated with micros but not possible with PLAs.

2) 3.2.4 Traffic Timing Signal t

Each set of traffic control lights has duration as follows

6 seconds of amber for the direction about to hand-over right of way

2 seconds of red in the direction about to handover right of way

6 seconds of amber in the direction about to receive right of way

18 seconds of green to the direction that has the right of way.

Table 2 shows the truth table for T in terms of the 4-bit counter output lines D, C, B, A. A power up one shot which sends a pulse when the power comes ON resets the 4-bit counter at start-up. Since 1 count is achieved every 2 seconds. T stays low for 3 counts (i.e. 6 seconds), then goes high for 1 count (i.e. 2 seconds) and goes low again for 3 counts (i.e. 6 seconds) and finally goes high for 9 counts (i.e. 18 seconds). When the queue in the direction that has right of way is finished before 18 seconds, HCLRT is generated to restart the timing cycle for another direction otherwise, the counter restarts by itself when it reaches 18 seconds in the direction of right of way.

Table 2: Truth table of the 4 bit counter

M	D C B A	T	T WAVEFORM
	0 0 0 0	0	
	0 0 0 1	0	
	0 0 1 0	0	
	0 0 1 1	1	
	0 1 0 0	0	
	0 1 0 1	0	
	0 1 1 0	0	
	0 1 1 1	1	
	1 0 0 0	1	
	1 0 0 1	1	
	1 0 1 0	1	
	1 0 1 1	1	
	1 1 0 0	1	
	1 1 0 1	1	
	1 1 1 0	1	
	1 1 1 1	1	

4.0 RESULT

In this research, an intelligent traffic light control and monitoring using State Transition Table (STT) was designed. The system featured a queue detecting device. The evaluation of the traffic light control system with queue detector was evaluated by placing objects (cars) away from the area marked as no queue zone when it was the right of way for that particular route. After some time the system automatically changed the right of way to the next route.

5.0 CONCLUSION

In this research, traffic light control system was designed. The system featured a queue detecting device. The advantage of this is that unlike the conventional traffic light system, it does not necessary have to give right of way to a route when there is no queue within a given range rather it apportions right of way to the next route if there is a queue. It also combines a thorough initial hardware design approach leading to the STT before generating the corresponding software that has just one statement

per link path in the STT. This structured software is therefore easy to debug because of the one-to-one mapping of each statement with the rows of the STT. It combines the disciplined approach to hardware design with a structured software engineering approach to realize a dependable end product.

REFERENCES

[1] M. Shoufeng, et al. Agent-based learning control method for urban traffic signal of single intersection. In *Journal of Systems Engineering*,17(6): pg. 526-530, 2010.

[2] Xiao-Feng Xie, et al. Real-time traffic control for sustainable urban living. IEEE International Conference on Intelligent Transportation Systems (ITSC), Qingdao, China, 2014: 1863-1868

[3] C. R. Clare. *Designing Logic Systems using State Machine.*Uk: McgrawHill. Pp 1-108, 2013.

[4] https://support.axeda.com/help/en/state rule definition.htm. State machine definition retrieved on 02/11/2015.

[5] The Vehicle Detector Clearinghouse, "A summary of vehicle detection and surveillance technologies used in intelligent transportation systems," Southwest Technology Development Institute, 2010.

[6] S. Y. Cheung, S. Coleri, B. Dundar, S. Ganesh, C. W. Tan, and P. Varaiya, "Traffic measurement and vehicle classification with a single magnetic sensor," University of California, Berkeley, *84th Annual Meeting Transportation Research Board*, January 2011, Washington, D.C.

AUTHORS

Ihedioha Chukwudi Ahmed is a PhD scholar in the department of Electrical and Electronics Engineering (Control option), Enugu State University of Technology, Enugu Nigeria. He holds a Master's degree (M.Eng) in Electrical/Electronics Engineering. Ihedioha Chukwudi Ahmed is also a member of the following; Council for the Regulation of Engineering in Nigeria, [COREN], Nigerian Society of Engineers, [NSE], Nigeria Association of Technologist in Engineering, [NATE], IEEE.

His research interests are in the fields of Power systems, high-tech engineering intelligence systems, systems stability control, etc.

Eneh Ifeanyichukwu Innocent is a Professor and Dean Faculty of Engineering, Enugu State University of Technology, Enugu Nigeria. He is a renowned scholar in the fields of Electrical and Electronics Engineering. Eneh Ifeanyichukwu Innocent is a member of the following; Council for the Regulation of Engineering in Nigeria, (COREN), Nigerian Society of Engineers (NSE), IEEE etc.