DEVELOPMENT OF A MICROCONTROLLER-BASED, DISTRIBUTED, ADAPTIVE TRAFFIC CONTROL SYSTEM

K. Tavladakis, K. Kostalias, A. Dollas, K. Kalaitzakis and N. C. Voulgaris

Technical University of Crete,
Dept. of Electronic and Computer Engineering
GR-73100 Chania, Greece

Abstract: Traffic flow is by nature a dynamic process, rendering static, fixed-time signaling systems inadequate. A new, adaptive traffic control system is presented in this paper. Traffic load is continuously monitored by loop detectors connected to a microcontroller-based system, which also performs all intersection control functions. Intersection controllers of an area are interconnected with a communication network, passing to each-other traffic load information, and getting synchronized. As a result, the duration and relative phases of each traffic light cycle dynamically change. The new architecture is completely distributed and eliminates the need for a region computer to individually control traffic lights.

Keywords: Traffic control, Microprocessors, Adaptive algorithm, Detectors, Loops, Controllers, Phase-locked-loops.

1. INTRODUCTION

Traffic control systems in urban areas are usually of static nature and their design is based on measurements and statistical analysis performed prior to the installation of these systems. However, traffic load is highly dependent on parameters such as time, day, season, weather and unpredictable situations such as accidents, special events or construction activities. If these parameters are not taken into account, the traffic control system creates delays and bottlenecks. A dynamic traffic control system solves these problems by continuously sensing and monitoring traffic conditions and adjusting the timing of traffic lights according to the actual traffic load. Furthermore, a control system can be synchronized along a route so that the resulting traffic waves will move with optimal speed, thus minimizing waiting intervals.

The main purpose of any traffic control system is to reduce the mean overall travel time of vehicles moving in the controlled area, without compromising vehicle and pedestrian safety. The mean reduction time for a dynamic traffic control system ranges from 15% to 40% as compared to a static system, resulting in 10% fuel consumption reduction and 15% air pollution reduction. Other benefits include reduction of accidents, noise and workers' man-hours. Total benefits have been demonstrated to exceed the implementation cost of a dynamic traffic control system by a ratio of 10:1 and the system deployment cost may be recovered in one to two years of operation (Rowe 1991; Sims and Dobinson 1980).

Most traffic control systems implement a three-level architecture, as shown in Fig. 1, comprising of the central computer, the regional computers and the intersection controllers. The computer, or local area computer network (Haver and Tarnoff 1991; Hanchey and Smith 1995), in the top level of the architecture, is located in a control building. This computer, besides controlling the system, is used for surveillance, for data recording and database update, and for report generation. One level down in the

architecture, there are the regional computers. Their purpose is to control and synchronize a number of intersection controllers. Typically, a regional computer controls 200 to 400 intersection controllers, depending on the size of the system and geographical considerations. A star topology is used to connect the intersection controllers to the regional computer, using private or leased lines. An intersection controller, which is implemented using a microcontroller, takes commands from the regional computer, changes phases, checks and monitors the system for malfunctions, collects data from loop detectors (if they exist) and transmits them via the regional computer to the control center.

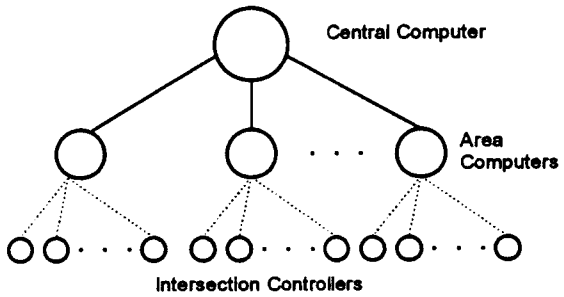

Fig. 1 Typical Traffic Control System Architecture

The intersection controllers in existing systems use either relay logic, or, microcontrollers replacing relay logic. When microcontrollers are used, their capabilities are not fully exploited toward maximum system effectiveness and efficiency. The microcontrollers are used to simply replace the relay logic in previous generation systems. However, a microcontroller has greater capabilities: it can be used to process data from loop detectors, to take decisions based on these data and to perform sophisticated communications functions.

The system described in this paper is an autonomous area-wide traffic control system. In the context of the architecture described above, it performs the functions of an area computer and its intersection controllers. The new system combines the data processing with the communication capabilities of microcontrollers to achieve traffic light control in a distributed fashion: the reasonably powerful regional computer controlling "dumb" intersection controllers is replaced with a network of low computational power microcontrollers, in which every microcontroller collects data locally, exchanges information with its neighbors, controls its local traffic light, and relays data for data logging purposes. In previous systems there has been a need to synchronize all intersection controllers so that they will produce correct phases of the traffic light cycles; by comparison, the new system achieves synchronization by passing time information from a master controller to the remaining ones, with time skewing no worse than a few milliseconds.

A characteristic feature of the new system is that each traffic light cycle can potentially be of different duration than the previous one. This is achieved by each controller sending a request for the desired next cycle duration, whereas the master controller computes and broadcasts the common cycle time to be used by all slave controllers. This process is repeated during every traffic light cycle. Thus, the cycle itself can change, whereas all microcontrollers remain synchronized.

2. TRAFFIC DETECTORS

An essential feature of any dynamic traffic control system is its capability to compute traffic flow rates of the controlled intersection. There are several methods to detect a passing vehicle (Klein L. A., et al, 1995), the most common of which being the loop detector. The main advantages of the loop detector are that it is robust and reliable, because it does not come in contact with vehicles, and that it is easily installed. Its principle of operation is based on the detection of a vehicle as a metallic mass entering the magnetic field of an inductor, and thus changing its inductance. This section describes a loop detector, which was designed and implemented for the new system presented in this paper.

The sensing element is a coil with dimensions two by three meters, placed just below the road surface and characterized by an inductance variation proportional to the proximity of a vehicle. The detector must be positioned at a distance of about 50 m before the stop line at a traffic node. If it is expected that during the interval of the red light period, the waiting queue will grow longer than 50 m, then the detector must be placed closer to the end of the predicted queue, in order to detect passing vehicles and not those waiting for a green light close to the node. For a road with more than one traffic lane, one detector must be placed at each lane.

The detector circuit senses loop inductance variations and produces a digital output. The output is a logic "1", if there is a vehicle over the coil and a logic "0", otherwise. A printed circuit input card was designed to group loop detectors, so that the microcontroller can read a byte for each group of eight detectors. There is no restriction as to the number of input cards that can be used in a system and, as a result, even the largest intersections can be handled at a loop detection cost directly proportional to the monitored lanes. As an example, the largest intersection of the city of Chania, Crete, with a population of 60,000, has a total of ten lanes, which would require two cards for monitoring, whereas a typical

intersection has a total of four to six lanes and could be monitored with one card. The detector circuit which converts the loop inductance variation to a digital signal is described in more detail below.

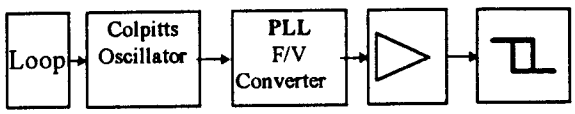

Fig. 2. Loop detector block diagram

Since the sensing coil inductance variations are very low (typically <5%), a sensitive circuit is used to ensure a reliable vehicle detection. An oscillator, a phase-locked-loop (PLL), an amplifier and a comparator stage are cascaded to form the conditioning circuit. The block diagram of that circuit is shown in Fig. 2.

The loop coil is part of a Colpitts oscillator and thus any inductance variation will result in an oscillator frequency shift. The PLL stage, acting as a FM demodulator, converts the frequency shift into a voltage variation. This voltage indicates whether or not there is a vehicle passing over the coil. Because initial inductance variations are very small, corresponding frequencies are very close to each other. Although the use of a filter to separate frequencies would cost less than a PLL stage, this approach was rejected, as temperature and other variations would potentially lead to incorrect output.. After the PLL stage, the signal is characterized by a small voltage difference among the two states (i.e. 6.2 Volts when no car is present and 6.6 Volts, when a car is present). The next stage subtracts the DC component, performs signal amplification and then passes it through a low-pass filter to cut off high noise frequencies that may be contained in the signal. This stage is implemented with two cascaded operational amplifiers, the first of which is a difference amplifier, while the second acts as a low-pass filter. The last stage is a comparator which converts the amplified signal to TTL levels. The comparator output is suitable to drive the input card pins. A hysteresis loop around the comparator ensures further noise rejection.

3. THE CONTROLLER

An intersection controller of the system described in this paper performs the following functions: it reads the detectors' state and hence it computes traffic flow information (car counts), it computes the timing schedule for the intersection based on the traffic flow, it communicates with the adjacent node controllers to share data and synchronization information, and finally it switches traffic lights on and off.

All integrated circuits used in the controller implementation are of CMOS technology. CMOS gates have two significant advantages. The first is that at the 12-MHz system clock frequency, the power consumption is 100 times lower than a TTL LS gate. This ensures low power consumption, necessary for system operation on a 24-hour basis. The second is that the noise margin of CMOS gates is twice as high as that of the TTL LS series. The heart of the system is Intel's 8051 microcontroller, capable of executing 1 million instructions per second and addressing a total of 128 Kbytes of program and data memory.

The controller consists of a motherboard and four types of extension cards. In addition to the microcontroller, the motherboard has a power up reset circuit, a crystal oscillator, bus control circuits, bus driver circuits, some LED's and a timer to interrupt the microcontroller at a 10-Hz rate for detector sampling. During each interrupt, the controller reads the state of a detector and compares it with the corresponding state of the previous interrupt. If a detector state has changed from 0 to 1, the corresponding variable is updated. Counting vehicles (traveling up to 100 Km/hr) with this approach requires only a 10-Hz external timer. If, on the other hand, the processor was interrupted every time a detector changes state, an extended external circuit would have been necessary.

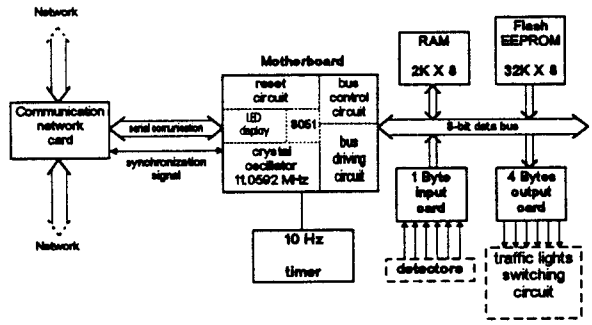

Fig. 3. Controller architecture

Another function that the controller performs is to compute optimum cycle times and to distribute time over the intersection phases. These computations are based on flow rates and are computed from detector sampling. The algorithm that performs the above computations makes the system distributed and adaptive. The algorithm is described in the section "The Algorithm", below.

As previously noted, the controller consists of a motherboard and a bus connecting the extension cards. This architecture gives versatility to the system. Thus, for each particular traffic node, only the necessary cards need to be added to the bus. Cards are of four kinds: RAM, ROM, input and output.

The RAM card consists of a 2-KB RAM unit and necessary circuits to read and write it. The microcontroller has the ability to address 64 KB of RAM memory, but 2 KB are enough for all intersections. The ROM card is similar to the RAM card. It consists of a integrated circuit of a 32-KB flash EPROM and the necessary circuits to read and write it. Input cards and output cards implement a memory mapped input-output. The controller has the ability to read 32 KB of data from input cards and to write 32 KB to output cards. Loop detectors are connected to input cards, whereas power circuits switching traffic lights on and off are connected to output cards. Cards may be used for other purposes too. For example, output cards may be used to pass data to displays informing drivers about road conditions, hazards, construction ahead, etc., but such features have not been implemented yet.

There is one more type of card, in addition to the above. This is the communication card which is connected directly to the microcontroller, rather than the bus. As described in the next section, all intersection controllers of an area are interconnected in a network, in which data are transmitted in an asynchronous serial mode. The communication card consists of line driving circuits, signal regeneration circuits and circuits to multiplex the controller to the network.

4. THE NETWORK

The network topology is shown in Fig. 4. It uses point-to-point connections having a serial protocol. Only three data lines connect each node with the next. Therefore, both the complexity and the cost of a star topology are avoided, as they grow to be quite high in traffic control systems with many nodes. The lines connecting the controllers consist of three twisted wire pairs. Data passed on these lines are transmitted in serial mode and are of TTL level. The serial communication speed is 9600 bps. The network card at every node regenerates the incoming signal, and adds packets of the local node to form the output. There are no collisions of packets, as will be described below. The functions are performed by the communication card, and no controller resources are needed for them.

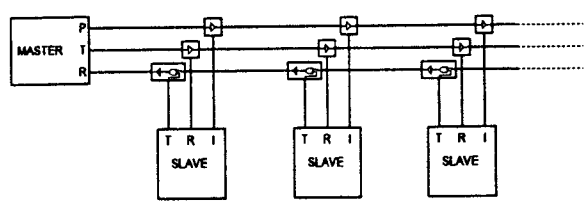

Fig. 4. Network topology

As Fig. 4 shows, there is a node characterized as the master node, the rest of the nodes being slaves. Any node of the network can be the master node, and a jumper setting on the communication card determines at power-up time the master or slave status of a communication node. All the slaves have the same access rights within the network. The master is responsible for the network operation, in order to avoid collisions. Of the three network lines, two are used for data transmission and the third is used by the master to send a synchronization interrupt to all slaves. The first data line transmits data from the master to the slaves and the second transmits data from the slaves to the master. On the second line, more than one node can pass data. This is achieved with the AND gate shown in Fig. 4. A token-based protocol does not allow two slaves to transmit data on the same line simultaneously.

The microcontroller of every network node has a different, hardwired in ROM address, with the master having the address 0. A node receives only those packets having the node address and rejects those with a different address. Each information packet consists of four bytes and that is the minimum amount of data a node can transmit. The byte position in the packet has the following meaning: The first byte is always "FF" and protects from random byte interference in the channel. The second byte is the receiver address, the third is the type of packet and the fourth is a data byte. Five types of packets have been implemented. Of these, two have a particular importance for the network: the ACK and ERROR packets. The ACK packet is sent by a slave to the master, to confirm a packet reception. The ERROR packet is also sent by a slave to the master, whenever an error is detected, resulting in retransmission of the corrupted packet.

No slave can initiate packet transmission to the channel. There are two types of information transmission. The first is when the master must send information to the slaves. The master sends a packet containing the information, starting from the first slave. Then the slave sends an ACK packet. When communication with the first slave ends, the master goes on to the second slave, and so on. The second type of information transmission is when data must be sent by the slaves to the master. Again, the master initiates the process with the first slave and sends a packet which is a specific data request. Then the slave sends a packet containing that information. When the packet is received, the master continues with the next slave. According to the above protocol, no one slave puts data on the channel, except when the master has requested it. Since the master never requests a packet from a slave, if communication with the previous node has not been completed, there is no possibility for collision. The above token-based protocol, in which a transmission token is held by

the master and is passed to one slave at a time, in addition to collision avoidance, is very simple to implement in software and results in low-cost, robust communication. Actual measurements over extensive periods showed no transmission errors, and as a result, no error detection and correction codes were included in this protocol; however, it would be very easy to add such codes, albeit, at the cost of additional information transmission and hence slower network operation.

There are two cases that may create problems during system operation. The first one is when a slave receives a packet containing an error (e.g. an illegal packet type) and sends back an ERROR message requesting retransmission. The second occurs when a slave cannot send an answer because there is a problem in the network. If this is the case, the master waits during a time-out interval and then assumes that there is a problem, gives an alarm message and continues with the next slave. In actual tests, no ERROR messages have been logged, however, the second type of error has been detected, e.g. when the power of a slave was shut off.

With a speed of 9600 bps, the packet consisting of 40 bits (counting also start and stop bits) requires transmission time equal to 40 bits/9600 bps = 4.17 msec. Based on this time, the time-out interval was selected to be 15 msec and this period determines the number of nodes that can be connected to the network. A full conversation of the master with 100 slaves can take place in a 1-sec period. Thus, the 256 nodes that can be addressed with the existing protocol can communicate within a reasonable time. A fairly large region generally contains well below this number of traffic lights, e.g., the city of Chania has a total of roughly 40 traffic lights.

5. THE ALGORITHM

A fundamental property and the main contribution of the system described here is that it is adaptive to traffic requirements. At the beginning of each traffic light cycle, the period of the next cycle is computed. This time computation is based on intersection traffic flow, which was computed during the previous cycle. Because the system is synchronized, the cycle length is common for all nodes in the controlled area. The dynamic traffic flow data of all nodes are taken into account for the computation of the common cycle time, and as a result the system controls traffic in a distributed fashion, as desired (Hecht 1982; Lum, et al, 1980; Tarnoff 1980).

Before the beginning of a traffic light cycle, each controller computes the traffic flow rates according to the detector measurements during the previous cycle. From these flow rates, the optimal cycle time of the specific intersection is computed. The algorithm used is the well known Webster algorithm (Webster and Cobbe 1956) which minimizes the average delays, but any other method can be programmed as well.

At the end of a cycle, the master is informed, through the network, about the optimal times that every slave has computed. Then the master computes the common cycle time that will take effect in the next cycle. There are several methods to compute the common cycle time after the master node has collected all of the requested times from the slaves: The average time, the largest time, or, if there is a wide variation, it is possible to ignore the extremes and have some slaves handle the traffic load in multiple cycles whereas the rest complete in one cycle. In the current implementation, the average of all the slave times is computed and this is the cycle time of the entire system. When the cycle time is computed, the master transmits it to every slave separately.

The continuous cycle time change requires synchronization at every cycle. To start a cycle, the master sends a synchronization pulse through the network. Such a pulse is sent every time the master initiates a cycle. A slave does not start its cycle immediately with the synchronization pulse, but waits for a time called intersection delay, and then starts the new cycle. For every node there is a different time delay with respect to the master-slave distance. This delay causes a progressive movement on the street of vehicles having a desired speed. This process allows for the relative phase synchronization between the master and the slaves. Since the locations of traffic lights and the desired speed of traffic flow are known in advance, the delay time is pre-computed and is hardwired in each node. Nonetheless, the relative phase times of each node can vary, as described below, and thus adjustments to the optimum times can result from the dynamic variations at each node.

After a slave controller is informed of the next light cycle time from the master, it is responsible to manage it and to use it in an optimal way. Based on locally measured traffic flow, the intersection approach time distribution is computed from the Webster relation. A phase sequence table is subsequently constructed, to determine the phase code and the phase duration on every phase. For the remaining cycle duration, the controller switches traffic lights on and off according to this table, and gathers traffic flow data. After all the phases of the table are executed, a synchronization pulse is sent from the master to the slaves denoting the beginning of the next cycle.

6. EXPERIMENTAL RESULTS

A complete three-node traffic control prototype system has been developed and tested in the laboratory. One controller was programmed as master and two controllers were programmed as slaves, interconnected with 100-meter long lines to comprise the control network. The detector was tested under actual conditions and was adjusted to detect passenger cars or heavier vehicles, avoiding false operations caused by bicycles or small metallic objects. The network was thoroughly tested by transmitting packets over long distances without any error reports. Suitable diagnostic programs check the integrity of the system. For testing purposes, a three-node road miniature replica was also constructed with computer simulated traffic. The controllers were connected to the traffic lights and the synchronization between the nodes was never lost, even under simulated severe traffic conditions.

7. CONCLUSIONS

In this paper, the development of an area-wide traffic control system is presented. At each intersection there is an intersection controller implemented with a microcontroller. The controller collects data from loop detectors located in each intersection and manages traffic lights. All intersection controllers of an area are interconnected via a communication network. Traffic control is accomplished in a distributed way, where each node computes the relative phases of each traffic light cycle. A major advantage of the described system is the adaptation of the traffic light cycle period to the entire region's traffic profile. Benefits arising from this adaptive approach are the reduction of the mean travel time, of the fuel consumption, and of the air pollution.

Another benefit of the system is the network configuration. The network is based on a point-to-point architecture, thus avoiding the complexity and cost of a star connection. Furthermore, network lines are private, thus avoiding problems that usually arise from the telephone company, if leased lines are used (Cimento A. A. 1980).

The system has a high degree of versatility. Any intersection can be controlled by simply connecting input and output cards to a motherboard. Other advantages of the system are its low-implementation cost and low-power consumption. Finally, the hardware capabilities are not completely used by software. Thus, in the future, the system can be programmed to perform more functions that are required at an intersection (e.g. to display warning messages to drivers, to include error detection in the packet transmission, and to send local statistics to a centralized computer).

REFERENCES

Cimento A. A. (1980). Traffic control systems hardware. *IEEE Trans. on vehicular technology. VOL. VT-29 NO. 2 May 1980. Pp106-124*

Hanchey C. M., J. A. Smith (1995). A medium-size city's approach to IVHS-Jackson's traffic management system. *ITE JOURNAL. May 1995. Pp18-23.*

Haver D. A., P. J. Tarnoff (1991)., Future directions for traffic management systems. *IEEE Trans. on vehicular technology. VOL 40 NO. 1 February 1991 pp4-10.*

Hecht H. (1982). Distributed computing for traffic control. *Conf of the IEEE vehicular technology society. P 495-498.*

Klein L. A., M. R. Kelley, M. K. Mills (1995). Traffic detection technologies for a modern transportation infrastructure. *Proc of the intern soc for optical Eng. VOL 2592 pp 99-113.*

Lum M., L. L. Kinney, K. S. P. Kumar (1980). A distributed computer system for street traffic control. *IEEE conf on decision and control. Pp 1011-1019.*

Rowe E., (1991). The Los Angeles automated traffic surveillance and control (ATSAC) system. *IEEE trans on vehicular technology. VOL. 40 NO.1 February 1991. Pp16-20.*

Sims A. G., K. W. Dobinson(1980). The Sydney coordinated adaptive traffic (SCAT) system. Philosophy and benefits. *IEEE trans on vehicular technology. VOL. VT 29 NO. 2 May 1980 pp 130-136.*

Tarnoff P. J. (1980). Distributed processing applied to traffic control. 30th Annual *Conf of the IEEE vehicular technology soc pp D1.4-D1.5*

Webster F. V., B. M. Cobbe (1956). Traffic signals. *Ministry of transport, Road research technical paper, No 56*