

Project Team Members

Abanoub Ashraf

Ramez Saber

Marian Bekhit

Marina El Kess Barsoum

Mina Essam

Mina Wahba

ACKNOWLEDGMENTS

We are fortunate to have **Dr.Hassan Mostafa** as our advisor, we would like to express our deepest appreciation to him; he continually and convincingly conveyed a spirit of hard working in regard to research, also for his great effort, without his guidance and persistent help this project would not have been possible.

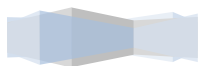
We would like to thank our department staff, who worked with us through the past five years demonstrated the value of knowledge and hard-working to be the way to achieve our targets.

It is an honor for us to be sponsored by “**National Telecommunications Regulatory Authority -NTRA**”.

We would like to thank NTRA staff for the financial support and their efforts with us through this full year.

Our special thanks are extended to the staff of “**OG-TECH**” company for their technical support .We owe our deepest gratitude to them for allowing access and use of their equipment, especially Eng.Magued Morad ,Eng.Abdel Rahman Ahmed and Eng.Ahmed Kandil.

We can never forget to mention that we are very much grateful to our parents. Any words would never express this immense gratitude for their long life helping, supporting, motivating and encouraging us till this moment had come.



ABSTRACT

Problem statement

Traffic congestion and tidal flow management were recognized as major problems in modern urban areas, they have negative impacts on the economy, the environment and the overall quality of life. Also, they have caused much frustration and loss of man hours. The use of existing popular technologies for traffic management has been explored by several researchers in recent times.

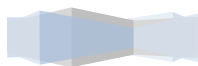
However, most of the works either deal with congestion detection strategies or address the issue of congestion control through **signal manipulation** or other aids. To the best of our knowledge, no effort has been made so far to develop a comprehensive system that automatically detects as well as controls traffic **-congestion based-** on detected level of congestion.

Approach

In order to solve the problem, an **intelligent RFID** traffic control system has been developed. It has circumvented or avoided the problems that usually arise with systems such as those, which use image processing and beam interruption techniques. **The aim of this project** is to design a complete strategy for automatic road traffic **congestion detection** in real-time and to devise an effective scheme for control and management of congestion using Passive RFID.

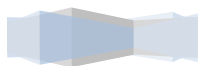
One of the **most important advantages** of our proposed system is the ability of **identification**. So our system provides both practically important traffic data collection and control information and **can trace criminal or illegal vehicles such as stolen cars**.

The system also has the ability of **detecting important vehicles** like **Police cars** and **ambulances**. So that, the road containing such car will be opened at the moment of detection regardless any other congestion condition. Thus, we **can overcome the problem of time delay** concerning these important vehicles.



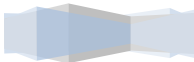
Contents

LIST OF TABLES	5
LIST OF FIGURES	6
Chapter 1: Introduction	8
Chapter 2: Radio Frequency Identification (RFID).....	11
Chapter 3: Traffic Problem & Our Motivation.....	28
Chapter 4: Large Scale.....	40
Chapter 5: System Implementation.....	57
Chapter 6: Conclusion and future work	90
References	92
Appendix	94



List of tables

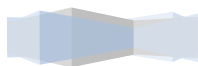
TABLE 2.1
TABLE 2.2
TABLE 2.3
TABLE 3.1
TABLE 4.1.....
TABLE 4.2



List of figures

Figure 1.1	8
Figure 1.2	9
Figure 2.1.....	10
Figure 2.2	13
Figure 2.3	14
Figure 2.4	17
Figure 2.5	18
Figure 2.6	20
Figure 2.7	22
Figure 2.8	23
Figure 2.9	24
Figure 2.10.....	25
Figure 3.1.....	27
Figure 3.2	31
Figure 3.3	32
Figure 3.4	31
Figure 3.5.....	
Figure 4.1	31
Figure 4.2	
Figure 4.3	31
Figure 4.4	
Figure 4.5	31
Figure 4.6	
Figure 4.7	31
Figure 4.8.....	3.3
Figure 4.9	31
Figure 4.10	3.3
Figure 4.11	31
Figure 4.12	3.3
Figure 4.13	31
Figure 4.14	3.3
Figure 5.1	31
Figure 5.2	
Figure 5.3	31
Figure 5.4	
Figure 5.5	31
Figure 5.6.....	3.3
Figure 5.7	31
Figure 5.8	3.3
Figure 5.9	31
Figure 5.10	3.3

Figure 5.11	31
Figure 5.12	
Figure 5.13	31
Figure 5.14	
Figure 5.15	31
Figure 5.16	
Figure 5.17	31
Figure 5.18	
Figure 5.19	31
Figure 5.20	
Figure 5.21	
Figure 5.22	
Figure 5.23	31
Figure 5.24	
Figure 5.25	31
Figure 5.26	
Figure 5.27	31
Figure 5.28	



Chapter 1: INTRODUCTION

Problem definition

Road traffic congestion poses a challenge for all large and growing urban areas. Traffic congestion is a condition on road networks that is characterized by slower speeds, longer trip times, and increased vehicular queuing. Today, the number of vehicles is increasing exponentially. However, road infrastructure cannot be increased in the same ratio. This leads to increasing traffic congestion.

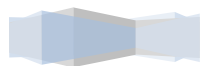
Most of the present intelligent traffic lights are sensor based with a certain algorithm that controls the switching operation of the system. This approach considers the traffic to be moving smoothly and hence does not require any management or monitoring of traffic conditions. When some unpredictable situation develops, or when congestion occurs, there is no proper way of dealing with such development.

A more elaborate approach has been introduced to overcome these problems. It employs real-time traffic flow monitoring with image tracking systems. Although this method can give a quantitative description of traffic flow, it involves several limitations. Some common problems involved in image processing system include False Acceptance Rate (FAR) and False Rejection Rate (FRR). Normally, in case of jam-packed traffic, the computer vision results in erroneous detection. The sensor based traffic light control on the other hand may require sensors that operate with a line of sight detection, which may present difficulty in detecting vehicles that pass through blind spots detection range.

Radio Frequency Identification (RFID) is an emerging technology that is still remains largely unexplored in the area of automatic congestion detection. Vehicle detection and counting can be done effectively using RFID technology.

Goal and Objectives

In this project, we propose to design a smart and fully automatic system to control the traffic that will detect the congestion in real time, and subsequently manage it efficiently to ensure smooth traffic flow with the use of Passive RFID devices.



Project Overview

The following is a simple **State diagram** of our proposed system to **show how our system works** and **deals with different components** in the system as shown in Fig.1.1.

State diagram

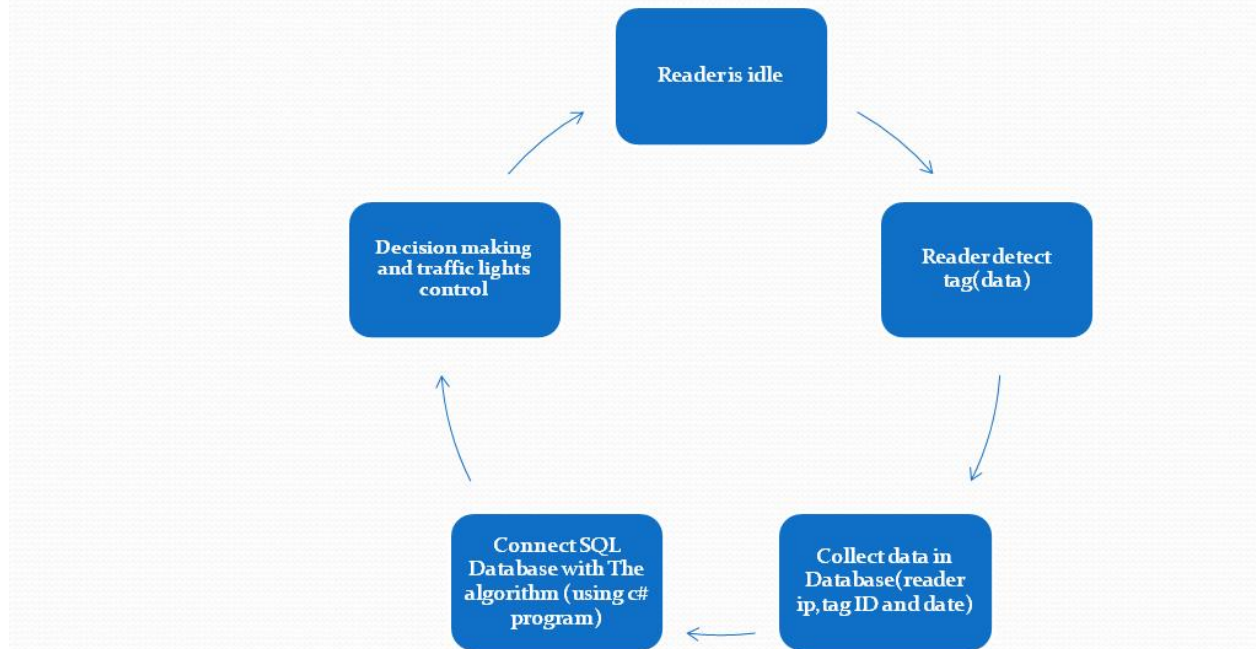
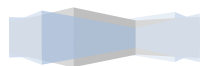


Figure 1.1 State Diagram

As shown above, a number of readers are deployed to detect and count the vehicles at each junction. The reader captures the Tag ID, the in-field time, and out-field time for each vehicle passing within its range, then, all information are collected in a database. This database is connected to our algorithm using C# program which, by its turn is responsible about decision making according to congestion conditions. Also we are using an Arduino kit with Bread-boards and LEDs to implement the traffic lights.



Project Outcome

We are implementing the system on small scale (wooden Maquette 1.20*1.20 m²). The system architecture consists of 8 RFID passive readers, passive tags and a personal laptop which is our processing unit. We are using C# program connected to database to implement our smart algorithm and is responsible about decision making according to congestion conditions. Finally, an Arduino Kit connected to Bread-boards and LEDs to implement the traffic lights...

Project Benefits

One of the most important benefits and advantages of our proposed system is the ability of identification. So our system provides both practically important traffic data collection and control information and can trace criminal or illegal vehicles such as stolen cars.

The system also has the ability of detecting important vehicles like Police cars and ambulances. So that, the road containing such car will be opened at the moment of detection regardless any other congestion condition. Thus, we can overcome the problem of time delay concerning these important vehicles.

Progress Timeline

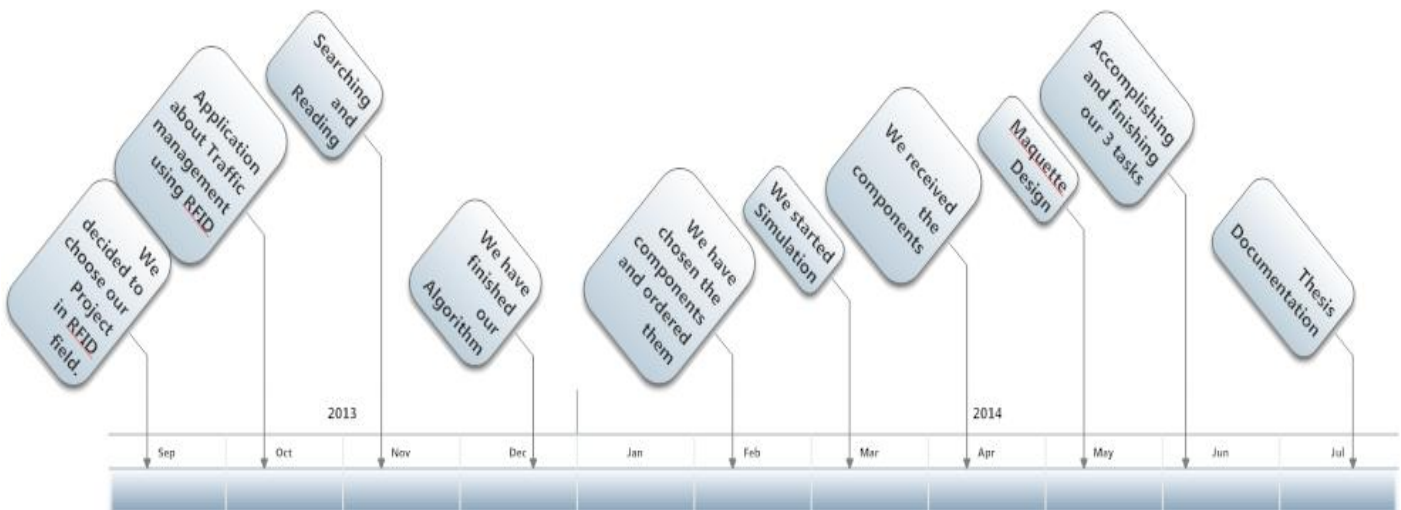


Figure 1.2 Progress Timeline



Chapter 2: RFID

2.1 What is RFID?

Radio Frequency Identification (RFID) is a flexible, wireless, automatic identification technology that transmits the identity (in the form of a unique serial number) of an object or person wirelessly, using radio waves. It comes under the category of automatic identification technologies.

Auto-ID technologies include optical character readers, bar codes and some biometric technologies, such as retinal scans. These technologies are mainly used to reduce time and labor needed for manually data entry and to enhance data accuracy. Some auto-ID technologies, like bar code systems, often require a person to manually scan a label or tag to capture the data. While bar code tags and bar code systems are much less expensive than RFID at present, RFID provides many benefits than barcode system, which is listed below.

First, the reader sending RF wave at certain frequency to awaken the RFID tags place on things in the specific effective area. Then, the awoken RFID tags will in response with the RFID reader by sending back the information in it through near field inductive coupling or far field back scattering method. After the information is retrieve at the reader, it will be send to microprocessor or computer database for data processing and update. The RFID technology is widely apply at various applications such as animal tracking, human identification, ware housing stock management, and traffic toll collection.

The basic working principle of the RFID system is illustrated in Figure 2.1,

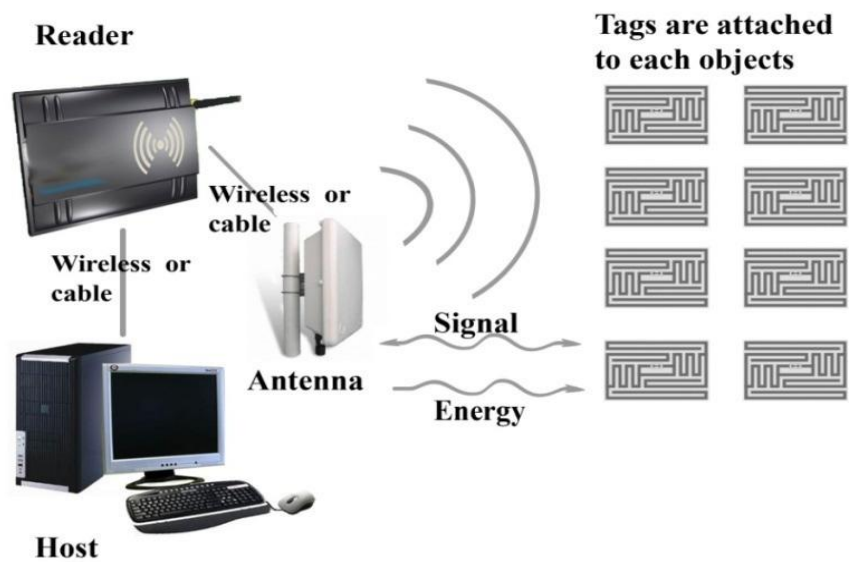


Figure 2.1 Basic RFID Working Principle



Linear barcodes use black-and-white stripes of different width to encode a number, while matrix barcodes use two-dimensional arrays of black-and-white patches to encode information. A scanner using a LASER reads the information encoded in barcodes and provides it to the information system.

RFID is not necessarily “better” than barcode. The two are different technologies that have different, yet sometimes overlapping, applications. The big difference between the two is that barcodes require a line-of-sight, while RFID does not. A barcode scanner has to “see” the barcode to read it, but the RFID reader can read the tag while it is optically hidden. The barcode also needs to be properly oriented toward a scanner for it to be read. RFID tags can be read as long as they are within the range of a reader, regardless of their orientation.

Barcodes have other shortcomings. If a label is ripped or soiled, the item cannot be scanned. Standard barcodes identify only the manufacturer and product, not the unique item. So, for example, the barcode on one juice carton is the same as every other, making it impossible to identify which one might pass its expiration date first. Many people think that RFID will replace barcodes, but if this happens, it will take several years since barcodes is a mature technology and currently has a very low cost of implementation compared to RFID.

Table 2.1 lists advantages and disadvantages of RFID and barcode.

Table2.1 Advantages and Disadvantages of RFID and Barcode

RFID	Barcode
No line of sight required	Line of sight required
Uniquely identifies items, cases, pallets	Identifies only item category
Item orientation to reader not important	Requires proper orientation
Simultaneous identification	Scans only single item at a time
Dynamic read/write capability	No write capability, static information
Can be used in harsher environment	Soiled labels are difficult to read
More data storage capacity	Limited data storage capacity
Worldwide standards still in process	Worldwide standards in place
More expensive: \$0.10-plus cost to attach	Cheaper to produce: \$0.001
Currently requires two steps: tag creation and tag attachment	Single step: can be easily printed on boxes during manufacturing



2.2 Brief History of RFID

RFID is called a new technology, but it is actually older than barcodes. The technology that forms the basis for RFID was first developed during World War II to identify airplanes. At that time it was called friend-or-foe technology, and a greatly modernized version is still used in aircraft identification today.

Barcodes were invented in the late 1940s but did not see substantial use until the late 1960s and early 1970s. However, barcode's low price compared to RFID made it a much more attractive option for Auto-ID usage at that time. Well, at least if you were not trying to identify an airplane! But, as the cost of RFID slowly dropped, industry began to use it for more applications. Since 1979, RFID has been in use to identify and track animals. In 1994, all rail cars in the

United States used RFID tags for identification. A recent surge has occurred in RFID technology research, manufacturing, and usage due to the advances in semiconductor manufacturing, which has reduced the cost of RFID, making their use economically feasible in supply-chain and other applications where tags are attached to objects not normally returned to the manufacturer.

RFID systems in the past used proprietary technologies; no worldwide open standards existed. There was little or no interconnectivity between different RFID vendor's products. Every vendor had their own readers, tags, signals, and equipment—and nothing worked together. This lack of interoperability made it challenging for companies to adopt RFID and constrained the deployment of RFID technology in the global supply chain. However, that problem is quickly disappearing due to new standards. Since 2006, many international and industry organizations have created open standards. RFID equipment manufacturers have started selling tags and readers that conform to these new standards, resulting in an increase in the supply and a reduction in the price of RFID devices, which in turn has accelerated RFID technology adoption.

2.3 Technical characteristics of RFID

- Data read and write – RFID reader can read the data to the database without contact, and process multiple tags once, and write the logistic processing state into the tag for the logistic processing in the next stage.
- Miniaturized and diverse form – RFID will not be limited by the size or form when it reads data, so it needs not to use the paper with fixed size or print quality to fit for the precision. In addition, E-tag of RFID can be applied in different products by small size, so we can more flexibly control the production of the products, especially the application on the production line.
- Anti-pollution – RFID possesses strong anti-pollution nature for water, oil or drugs. And in the dark or polluted environment, RFID also can read data.
- Repetitive use – Because RFID is electric data which can be written repetitively, so the tag can be used repetitively.
- Penetrability – If RFID is covered by the paper, wood, plastics or non-metal or non-transparent materials, it can communicate through these materials except for the irons or other metals.

2.4 Architecture & Operation of a RFID System

A RFID system is composed of three basic components: a tag, a reader, and a host computer.

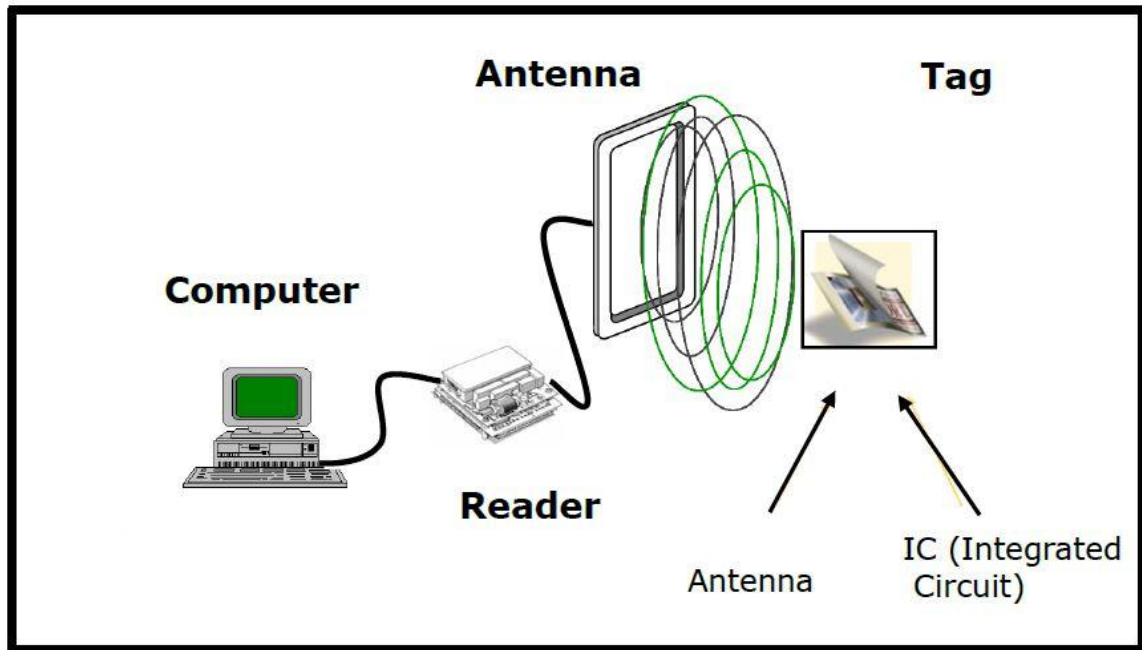


Figure 2.2 Components of RFID system

2.4.1 RFID tags Contain tiny semiconductor chips and miniaturized antennas inside some form of packaging. They can be uniquely identified by the reader/host pair and, when applied or fastened to an object or a person, that object or person can be tracked and identified wirelessly and on the move. RFID tags come in many forms. For example, some look like paper labels and are applied to boxes and packaging; others are incorporated into the walls of injection molded plastic containers; and still others are built into wristbands and worn by people.

A-Types of RFID tags

I. Active RFID tags include on-board power source (miniature batteries) that are used to power the tag, and can transmit signals autonomously.

II. Passive RFID tags don't include an on-board power source and have power beamed to them by the reader.

III. Battery Assisted Passive (BAP) or Semi-passive RFID tags require an external source to wake up but have significant higher forward link capability providing greater range.

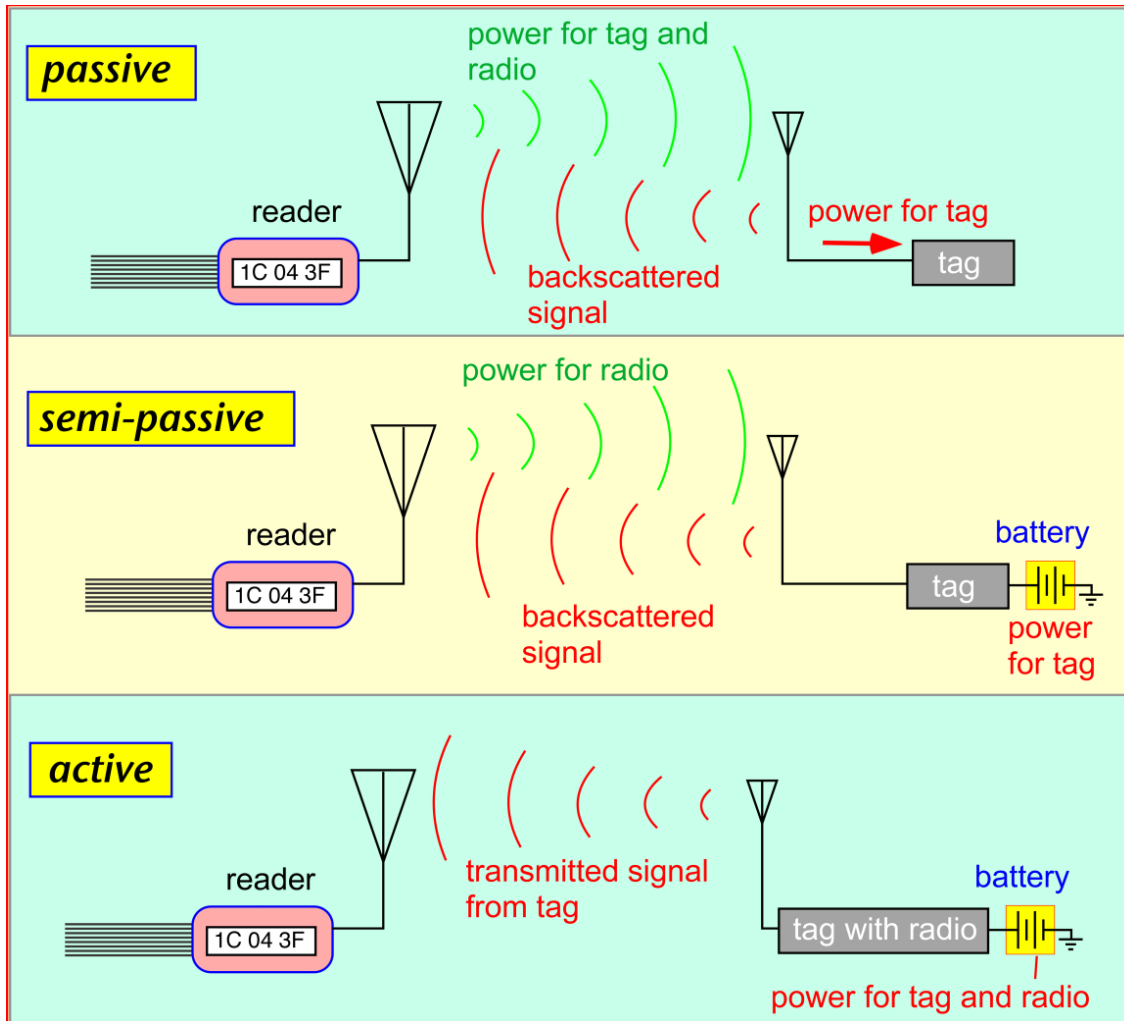


Figure 2.3 Types of RFID Tag and Their Characteristic

B-Smart Tags

I. Read only tags Information is programmed onto chip during manufacturing, no overwriting, and information constant, least expensive.

II. Write Once Read Many (WORM) tags Information added only once along with unique identifier but can be read many times.

III. Read-Write tags open to data manipulation by user's system without restrictions. It contains a unique identifier but carry an updateable memory for that to be added. It is expensive also.

Table 2.2 briefly describe the different characteristics of these three categories of RFID tag,



Table 2.2 Type of RFID Tags and Their Characteristics

Tag Type	Passive	Semi-passive	Active
Communication Model	Reader talk first(RTF)	RTF	Tag talk first (TTF)
Communication principle	Inductive coupling/ Backscatter	Backscatter	Self generate EM wave
Operating Frequency	LF/HF/UHF/Microwave	UHF	UHF/Microwave
Tag characteristic	Thin and flexible	Thin and flexible	Large and bulky
Common read range	0.1m~7m	60m~80m	>100m

C-Standard RFID Operating Frequency

I. Microwave works on 2.45 GHz, it has good reader rate even faster than UHF tags. Although at this frequency the reading rate results are not the same on wet surfaces and near metals, the frequency produce better results in applications such as vehicle tracking (in and out with barriers), with approximately 1 meter of tags read range.

II. Ultra High Frequency works within a range of 860-930 MHz, it can identify large numbers of tags at one time with quick multiple read rate at a given time. So, it has a considerable good reading speed. It has the same limitation as Microwave when is applied on wet surface and near metal. However, it is faster than high frequency data transfer with a reading range of 3 meters.

III. High Frequency works on 13.56MHz and has less than one meter reading range but is inexpensive and useful for access control, items identifications on sales points etc. as it can implanted inside thin things such as paper.

IV. Low Frequency works on 125 kHz, it has approximately half a meter reading range and mostly used for short reading range applications such as shops, manufacturing factories, inventory control through in and out counts, access control through showing a card to the reader. These low frequency tags are mostly not affected when applied on wet and near metal surfaces.

All four kind of operating frequency range of RFID system is summarized in Table 2.3.



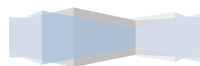
Table 2.3: Summary of RFID Operating Frequency

Frequency Type	LF	HF	UHF	Microwave
Frequency range	<135kHz	13.56MHz	860 - 930 MHz	2.45 GHz
Physical coupling	Inductively-coupled systems		Backscatter systems	
Tag use	Passive	Passive	Passive/Semi-passive/Active	Passive /Active
Communication Boundary	Near Field	Near Field	Far-Field	Far-Field
Approximate read range	< 0.5m	< 1m	< 7m (passive) < 80m (semi) > 100m (active)	> 100m (active) < 3m (passive)
Antenna	Coil	Coil	Dipole	Dipole
Effect of liquid	None	Low attenuation	High attenuation	High attenuation
Effect of metal	Disturbance	Disturbance	Attenuation and reflection	Attenuation and reflection
Data rate	< 10 kbit/s	< 100 kbit/s	< 100 kbit/s	< 200 kbit/s

D-RFID Tag Signal Transmission

There are two kind of response signal transmission method for the RFID tag, which is inductive coupling and backscattering propagation. Induction coupling signal transmission method is near field communication mechanism that work base on the Faraday’s principle of magnetic induction. It commonly used for RFID system with frequency range less than 100MHz that held in the HF and LF frequency range. In this transmission method, small coil antenna incorporate in the tag induce alternating voltage when pass through the alternating magnetic field around the RFID reader.

The small magnetic field induce at the tag will oppose the reader localize magnetic field and produce a small increase in current flowing through the reader antenna coil that was proportional to the load applied to the tag coil. Figures 2.4 illustrate the concept of induction coupling transmission of RFID tag.



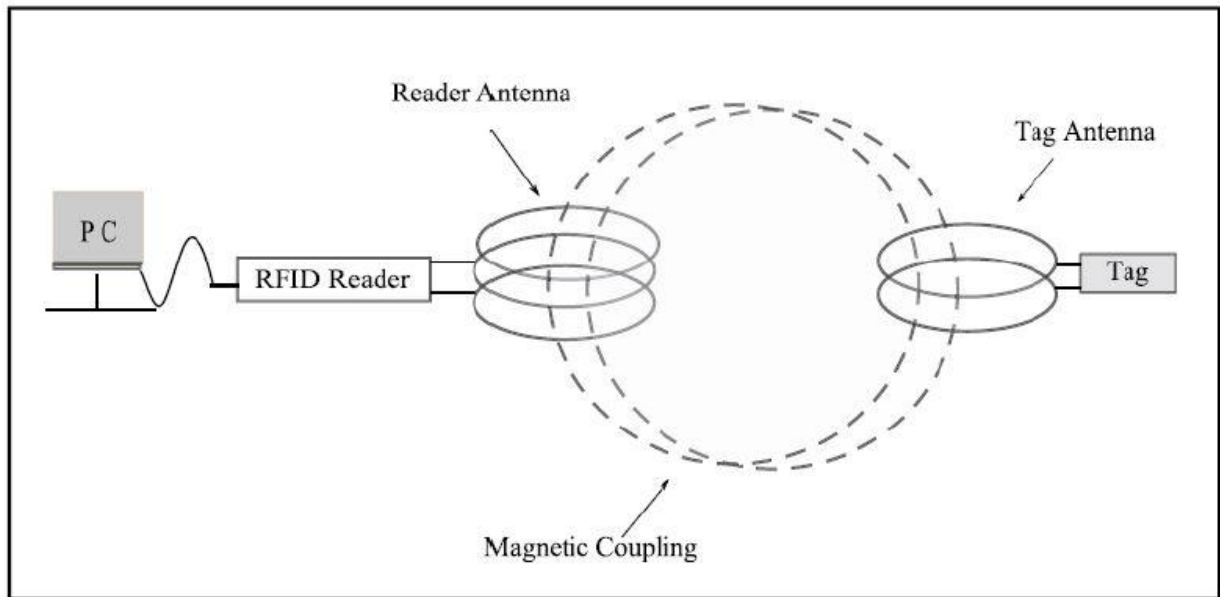
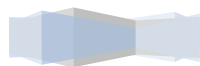


Figure 2.4 Induction Coupling Transmission (Klair, D. K., K.-W. Chin)

However, it works on near field because the range for which the magnetic induction approximates to $c/2 \cdot f$, where c is light speed and f is the operating frequency. Thus, as the frequency of operation increases, the distance over which near-field coupling can operate decrease as well. The induction energy of the antenna coil also limited by distance because magnetic field drops off at a factor of $1/r^3$, where r is the separation of the tag and reader, along a centered line perpendicular to the coil antenna. Next, back scattering and EM wave propagation signal transmission method for the RFID tag is far field communication mechanism that work well for RFID system with UHF and microwave frequency range. Instead of using coil antenna, dipole antenna is used at the reader and tag to receive and transmit signal. The back scattering signal transmission method is work by changing the impedance of the tag antenna overtime that create impedance mismatch between the reader and tag antenna to reflect back certain incoming signal in a pattern that encodes the tag response. Figures 2.5 illustrate the concept of back scattering and EM wave propagation transmission of RFID tag.

Hence, it can achieve further transmission distance than induction coupling method. However, it also limited by the amount of energy that reaches the tag from the reader and sensitivity of reader to the reflected signal. The backscatter transmission introduce to two attenuations that based on the inverse square law. The first attenuation when signal transmit from reader to tag and second attenuation is occurs when the reflected signal is travel back to the reader. The return energy of the RFID tag is approximately $1/r^4$, where r is the distance between the reader and tag.



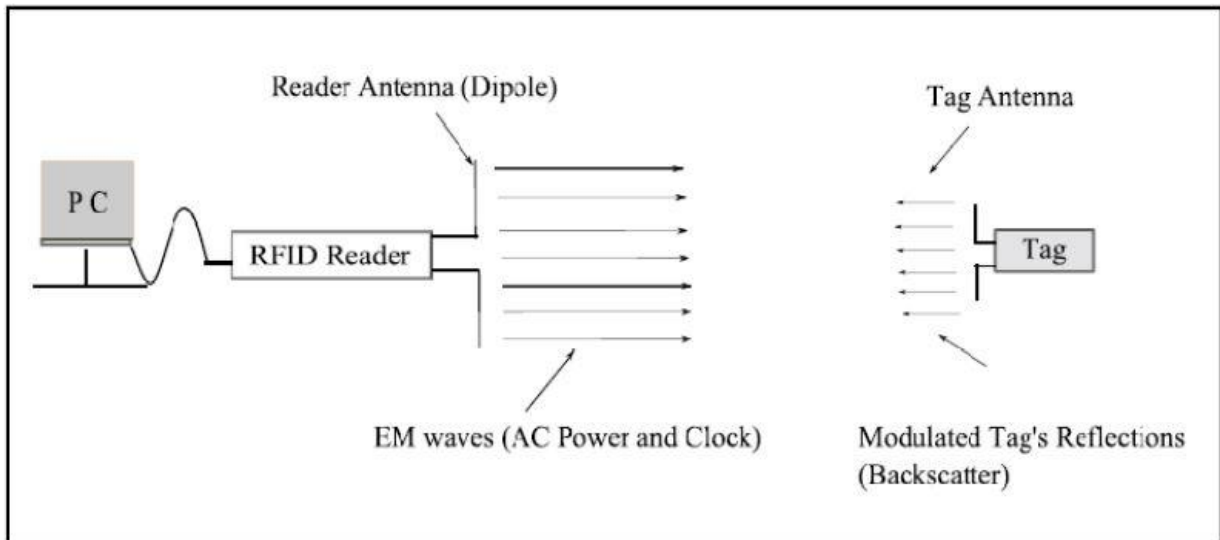
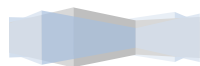


Figure 2.5 Backscatter Transmission (Klair, D. K., K.-W. Chin)

2.4.2 RFID Readers are composed of an antenna and an electronic module. The antenna is used for communicating with RFID tags wirelessly. The electronic module is most often networked to the host computer through cables and relay message between the host computer and all the tags within the antenna's range. The electronic module also perform a number of security functions such as encryption/decryption and user authentication, and another critical function called anti-collision, which enables a reader to communicate with multiple tags simultaneously.

The reader is also called the coupler. The coupler can send information in two directions: it can read information from a tag and send it to the PC (read mode), or it can read information from the PC and to an RFID tag (write mode).

2.4.3 Host Computer or PC provides an interface between the RFID hardware and application based system, which is the "brain" of any RFID system. They are used to network multiple RFID interrogators together and to centrally process information. The controller in any network is most often a PC or a workstation running database or application software, or a network of these machines.



2.5 Improvement of RFID System against Metallic and Liquid Object

In RFID scanning system, the effect for the present of metallic and high humidity object must take into serious consideration because it will highly affect the overall performance of the system. When metallic platform is placed too near to the RFID tag, it can appear as an antenna for the RFID tag and it also will act as a reflector that reflect most of the incoming electromagnetic wave signal. Therefore, the present of metallic platform near the RFID tag will decrease the signal reception of the RFID tag in some case and dramatically varied the parameters of the RFID tag build in antenna. Some common parameters of the RFID tag antenna that will affect by nearby metal platform including the impedance properties, resonant frequency, and antenna gain and radiation pattern. The variation of RFID tag property due to present of metal object can be positive or negative, but most of the time it gives negative effect if proper adjustment to the placement and design of the RFID tag is not applied.

Besides that, present of high liquid contain object around the RFID tag will also cause degradation of performance for the RFID system .The degradation level of the RFID system due to the present of liquid is highly depending on the properties of the liquid. Normally oil based liquid will tend to give less impact to the RFID system compare to water base liquid due to the electromagnetic wave absorption properties of it. The absorption of signal from nearby liquid contain cause the RFID tag unable to receive sufficient signal from RFID reader to power up and transmit feedback signal. The liquid may also absorb part of the transmitting signal from the RFID tag that cause the reader having difficulty in detect the weak signal from RFID tag.

In current technology, there are many ways to increase the reliability of RFID system against the present of metal and liquid object through modification of RFID tag design, but most of the method will increase the production price of the RFID tag which is not desirable. The influence of the nearby metal platform against the RFID tag can be reduced with proper separation between them with Styrofoam and the distance is around 0.5 cm from the research outcome. Figure 1.4 will briefly descript the concept behind the design. Next, it is also suggested that multiple RFID tag placement on the items can increase the scanning outcome of RFID system in metallic and liquid contain object. Lastly, increase the coverage scanning direction can also increase line of sign scanning of the RFID tag on both metal and liquid contain object which lead to higher detection rate.



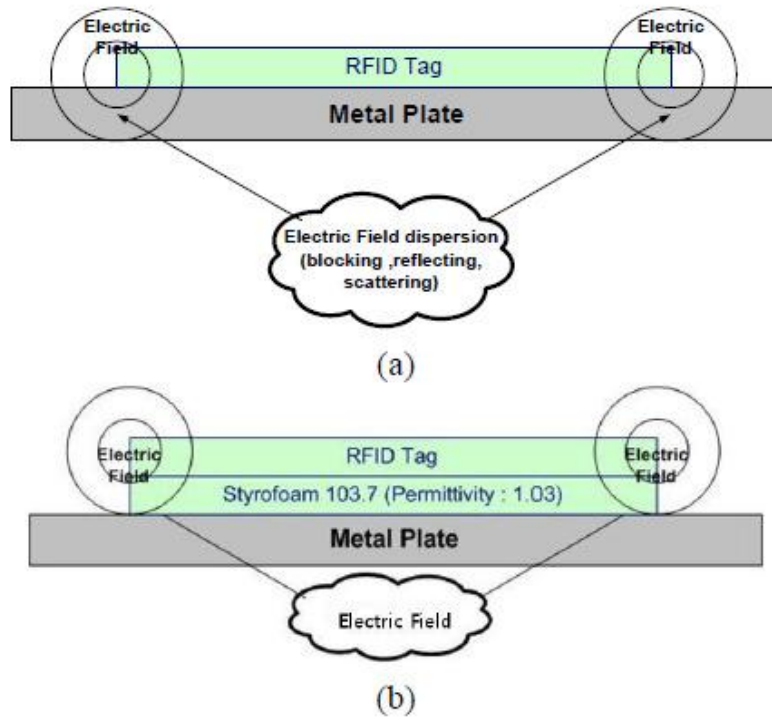


Figure 2.6 Simple RFID Tag Improvement against Metal platform

2.6 Applications of RFID

The RFID concept is not new but has been around for decades; in fact, it was introduced to the world for the first time during World War II by the British Air Force to distinguish Allied aircraft from enemy aircraft using radar (table 2.6 provides a brief overview of the history of RFID technology).

Since then, this technology has been used for various applications. RFID technology has been used by thousands of companies in many different ways for a decade or more to create value. Here are some of the business applications where this technology is used

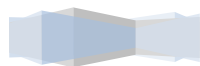


Table 2.4: RFID's History

Date	Event
1930 – 1940	American navy research laboratories developed a system known as IFF (Identify Friend or Foe).
1940 – 1950	The first application of RFID consisted of identifying Allied or enemy planes during WWII through the use of the IFF system.
1950 – 1960	IFF technology was used to develop the modern air traffic control system. First RFID applications in the military sector, in research laboratories and in major commercial enterprises.
1960-1970	Sensormatic and Checkpoint Systems introduced new applications for RFID, such as electronic article surveillance (EAS) equipment.
1970 – 1980	Technological advancements led to the creation of the passive tag, and the first initiatives for animal tracking and factory automation took place.
1980 – 1990	Many American and European companies started to manufacture RFID tags. First RFID application for automatic toll payment.
1990 – 2000	Standards for RFID equipment interoperability were developed.
2003	The Auto-ID Centre from MIT became EPCglobal, an organization whose objective is to promote the use and adoption of EPC technology.
2005	Wal-Mart launched an EPC pilot.

(Source: AIM Publication (2001), Manish (2005), EPCglobalinc.org)

- Asset Tracking – It is one of the most common uses of RFID. RFID tags can be put on assets that are lost or stolen.
- Supply Chain Management – It is used in closed loop supply chains or to automate parts of the supply chain within a company.
- Retailing – It is used by retailers to improve supply chain efficiency and making sure product is on the shelf when customers want to buy it.
- Payment Systems – One of the most popular uses of RFID today is to pay for road tolls without stopping. It can also be used in a convenient way to pay for bus, subway and train ticket.
- Security and Access Control – It can be used as an electronic key to control who has access to office buildings or areas within office building.
- RFID is produced to maximize company benefits and to help consumers with everyday work. In the late 1990s, Exxon Mobil introduced Speed pass, a RFID system that made it possible for costumers to pay for gas automatically. They received a small, passive transponder that they could put on their key chain. To pay for the gas, the customer needed to do, was to wave the key tag close to the

reader built into the gas pump. With over 5 million users, it became one of the most popular consumer RFID devices in the world.



Figure 2.7 Paying gas with Exxon Mobil's Speed pass

- MasterCard and Visa are experimenting with the RFID technique to give the costumers a smart way to handle small payments with a key card. There is an international standard for paying with MasterCard and Visa, but there are no stores that will allow customers to pay using RFID yet.
- Mobile phone payment has been tested on various locations. A vending machine where payment could be done by calling a number were equipped with tags containing the same number. Using the RFID reader in the mobile phone removed the effort of examining and entering the number and was perceived to be easy to use.
- RFID is not only made for consumers so they can use a convenient payment system. Merloni Elettrodomestici is an Italian company that has created a smart washing machine (RFID Journal – Merloni Unveils RFID Appliances). Their idea is to put tags on the clothes and a reader in the washing machine. The reader scans the clothes and tells the machine in what temperature to wash the clothes, based on the information from the tags. The more cloth manufacturers that would weaving tags into their clothes, the better the machine would work (Boutin 2003) Snagg in Palo Alto, California, has created an electronic registry for musical instruments. They put tags with information about the instruments, such as the license number, on valuable instruments. If an instrument is recovered after being stolen, the police can contact Snagg to help find the owner.
- Another example, originally developed for disabled people, is the using of business card or photos to make phone calls. A tag containing the number of the person on the business card or photo initiates an application in a reader-enabled mobile phone that calls the person.

- RFID technology is cheap and many new applications are being developed to solve common and unique business problems.

We studied some applications from RFID applications as follows:

2.6.1 RFID Checkout System:

Shopping in the present day usually involves waiting in line to get your items scanned for checkout. This can result in a great deal of wasted time for customers. Furthermore, the technology currently used in checkouts' barcodes - is from another era, developed in the 1970s, before the age of the affordable computing power.

With the increasing prevalence and affordability of radio frequency identification (RFID) tags in everyday authentication systems, RFID holds great promise in the retail world for both customers and stores in inventory control, convenience, and cost savings. This project utilized these RFID tags to automate the checkout process by building a system that could read the RFID signals of all the objects that were placed in proximity to an antenna platform. This eliminated the need for barcode scanning of each individual item, making checkout a significantly faster experience. Furthermore, as each item has a unique tag, even copies of the same product contrasted to the current UPC model, much better inventory control, recall ability, and monitoring of consumer behavior trends are possible, with privacy concerns considered of course.

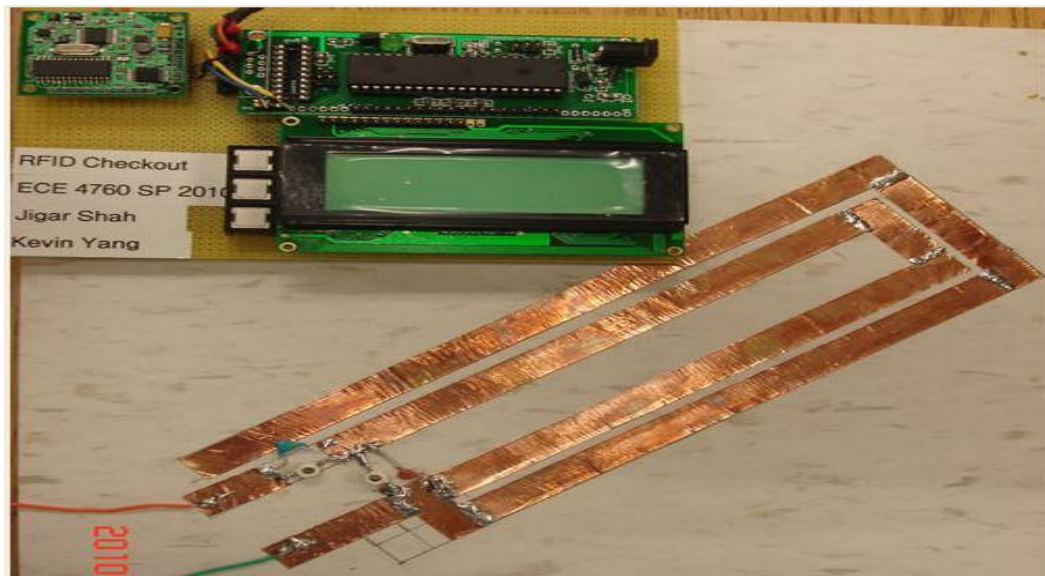


Figure 2.8 the completed RFID Checkout system

A-Program Overview

This program for the RFID Checkout system contains three major sections. In the beginning of this program, they declare multiple arrays that correspond to known RFID tags and their associated information. They implemented a state machine that defined an

intuitive user interface for a checkout lane. Finally, they also defined many methods that were used to search or particular RFIDs or recognize certain RFID as a particular product.

B-RFID Tag Arrays and Associated Information

A list of the RFID tag arrays and its associated information is declared at the beginning of the program. The associated information is in the same order as the RFID tag codes. This would allow the later find functions determine if a particular RFID tag was valid and what information is associated with it (i.e. price, product name).

RFID Find Functions

The RFID find functions utilize a linear search in determining if a particular RFID is in the array. From these functions, they can determine the index of the scanned RFID in the array and be able to extract information such as price and name to display.

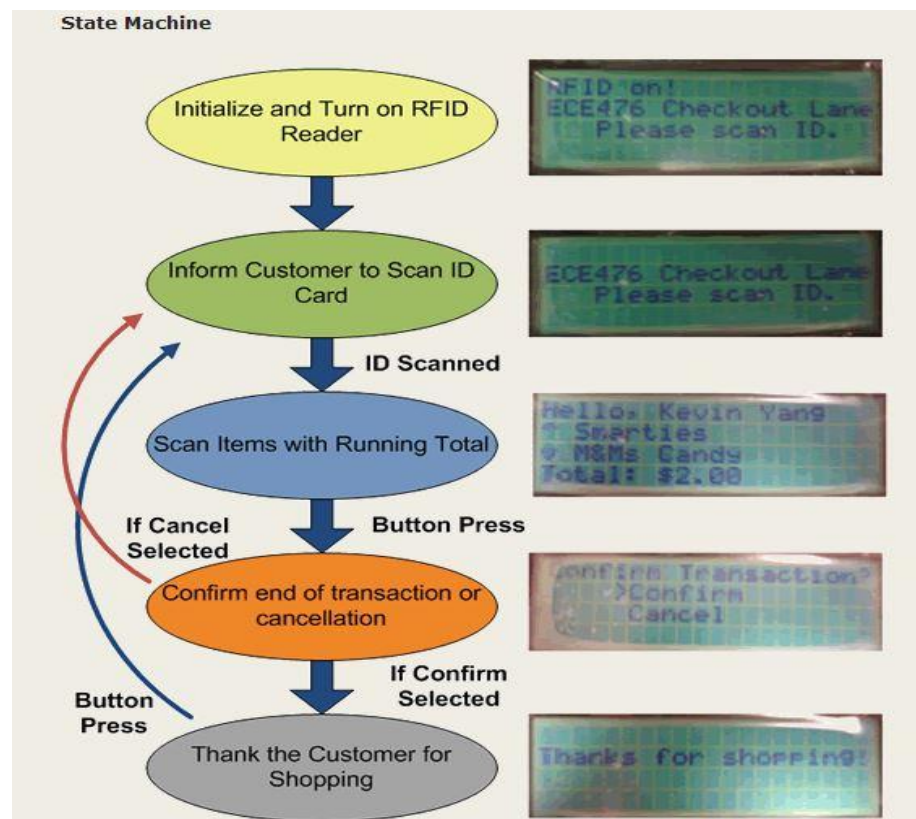


Figure 2.9 RFID Checkout software state Machine operation



2.6.2 Intelligent Dynamic Traffic Light Sequence Using RFID

The proposed RFID traffic control avoids problems that usually arise with standard traffic control systems, especially those related to image processing and beam interruption techniques. This RFID technique deals with a multi-vehicle, multilane, multi road junction area. It provides an efficient time management scheme, in which a dynamic time schedule is worked out in real time for the passage of each traffic column. The real time operation of the system emulates the judgment of a traffic policeman on duty. The number of vehicles in each column and the routing are proprieties, upon which the calculations and the judgments are based.

The operation of standard traffic lights which are currently deployed in many junctions, are based on predetermined timing schemes, which are fixed during the installation, and remain until further resetting. The timing is no more than a default setup to control what may be considered as normal traffic. Although every road junction by necessity requires different traffic light timing setup, many existing systems operate with an over simplified sequence. This has instigated various ideas and scenarios to solve the traffic problem. To design an intelligent and efficient traffic control system, a number of parameters that represent the status of the road conditions must be identified and taken into consideration. Most of the present intelligent traffic lights are sensor based with a certain algorithm that controls the switching operation of the system.

This approach considers the traffic to be moving smoothly, and hence does not require any management or monitoring of traffic conditions. When some unpredictable situation developed, or when congestion occurs, there is no proper way of dealing with such development. The multilane traffic sequence flow is presented in figure 2.10

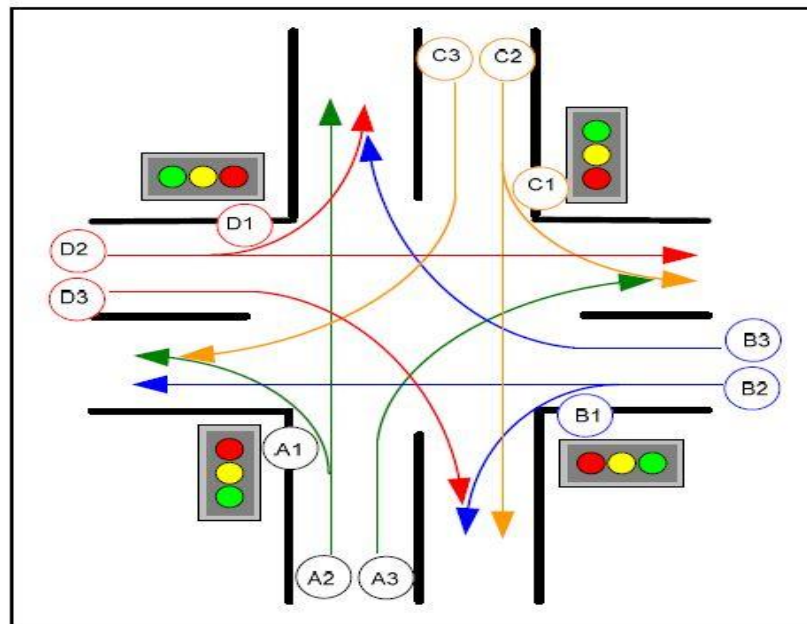
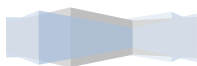


Figure2.10 Multilane traffic sequence flow

The sensor based traffic light control on the other hand may require sensors that operate with a line of sight detection, which may present difficulty in detecting vehicles that pass

through blind spots detection range. The RFID technology offers an advanced object tagged recognition which supports the intelligent dynamic traffic sequence algorithm.

In the following section we are going to discuss our project which is “Traffic Control Management Using RFID”



Chapter 3: Traffic Problem & Our Motivation

“A drive-by shooting in a traffic jam is not a good idea, O’Flanagan.”

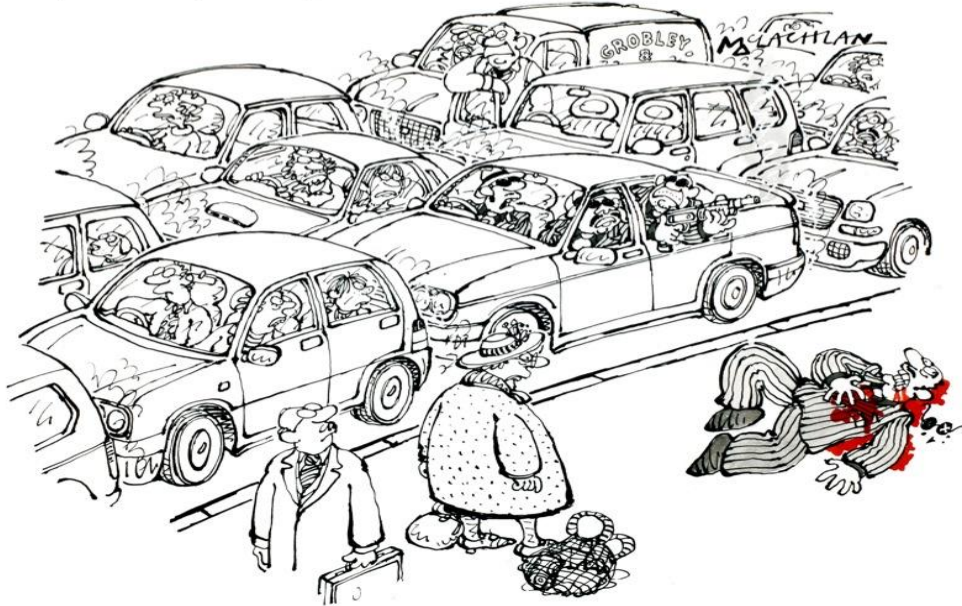


Figure3.1 Congestion: A National Issue

- Our ability to move about our neighborhood, our city and between cities has been taken for granted for years. Transportation mobility affects our ability to do our jobs, our quality of life and the economic productivity of our country. In today’s environment, our mobility is also important to our safety and security. Increasingly, however, our mobility is jeopardized by congestion. The Texas Transportation Institute estimates that, in 2000, the 75 largest metropolitan areas experienced 3.6 billion vehicle-hours of delay, resulting in 5.7 billion gallons in wasted fuel and \$67.5 billion in lost productivity. Congestion impacts everything we do – going to work, picking up the kids at school, getting them to the soccer game, doing grocery shopping, or delivering

products to stores for our consumption. Congestion is a part of daily life for millions of people.

3.1.1 What is Congestion?

- Highway congestion is caused when there are more vehicles than available space on the road, or, stated differently, when traffic demand approaches or exceeds the available capacity of the highway system. Traffic demands vary significantly depending on the season of the year, the day of the week, and even the time of day. Also, the capacity, often mistaken as constant, can change because of weather, work zones, or traffic incidents.
- Roughly half of the congestion experienced by Americans happens virtually every day – it is "recurring". This is the type of congestion where there are simply more vehicles than roadway. The other half of congestion is caused by temporary disruptions that take away part of the roadway from use – or "nonrecurring" congestion. The three main causes of nonrecurring congestion are: incidents ranging from a flat tire to an overturned hazardous material truck (25% of congestion), work zones (10% of congestion), and weather (15% of congestion). Nonrecurring events dramatically reduce the available capacity and reliability of the entire transportation system. This is the type of congestion that surprises us. We plan for a trip of 20 minutes and we experience a trip of 40 minutes. Travelers and shippers are especially sensitive to the unanticipated disruptions to tightly scheduled personal activities and manufacturing distribution procedures.
- The effects of congestion are growing. Rush "hour" is no longer an hour. In fact, in 1982, rush "hour" averaged 2-3 hours, and, in 1999, rush "hour" had increased to 5-6 hours. And, there is no end in sight. Freight transportation is expected to almost double in the next twenty years. The number of miles that we travel continues to increase. Between 1980 and 1999, vehicle miles of travel grew by 76 percent while the amount of new roads or lanes increased 1.5 percent. Without aggressive and effective new strategies to mitigate congestion, congestion will continue to grow with crippling impacts on our lives.
- As demand approaches the capacity of a road (or of the intersections along the road), extreme traffic congestion sets in. When vehicles are fully stopped for

periods of time, this is colloquially known as a traffic jam or traffic snarl-up. Traffic congestion can lead to drivers becoming frustrated and engaging in road rage.

3.1.2 Causes

Traffic congestion occurs when a volume of traffic or modal split generates demand for space greater than the available road capacity; this point is commonly termed saturation. There are a number of specific circumstances which cause or aggravate congestion; most of them reduce the capacity of a road at a given point or over a certain length, or increase the number of vehicles required for a given volume of people or goods. About half of traffic congestion is recurring, and is attributed to sheer weight of traffic; most of the rest is attributed to traffic incidents, road work and weather events.

Traffic research still cannot fully predict under which conditions a "traffic jam" (as opposed to heavy, but smoothly flowing traffic) may suddenly occur. It has been found that individual incidents (such as accidents or even a single car braking heavily in a previously smooth flow) may cause ripple effects (a cascading failure) which then spread out and create a sustained traffic jam when, otherwise, normal flow might have continued for some time longer.

3.1.3 Negative impacts

Traffic congestion has a number of negative effects:

- **Delays** The first thing many people think of when it comes to congested roadways is the delay. During the morning commute there is additional stress because delays caused by traffic can make people late for work. And at the end of the day, the afternoon rush hour is again a frustrating time because the workday is done and people want to get home to relax, and traffic is preventing it. These delays are the effects most people feel because they are universal to everyone who has to maneuver through congested roads.
- **Wasting time of motorists and passengers ("opportunity cost")** as a non-productive activity for most people, congestion reduces regional economic health.

- ***Inability to forecast travel time*** accurately, leading to drivers allocating more time to travel "just in case", and less time on productive activities.
- ***Wasted fuel*** increasing air pollution and carbon dioxide emissions owing to increased idling, acceleration and braking as will be discussed in details later.
- ***Wear and tear*** on vehicles as a result of idling in traffic and frequent acceleration and braking, leading to more frequent repairs and replacements.
- ***Emergency Vehicles*** When you request a police officer, an ambulance or a fire truck and the emergency vehicle is unable to respond in an appropriate amount of time because of traffic congestion it can be a danger to you and your property. Systems are available that help alleviate the problem by allowing the emergency crews to automatically change the traffic lights to keep the line moving.
- ***Spillover effect*** from congested main arteries to secondary roads and side streets as alternative routes are attempted ('rat running'), which may affect neighborhood amenity and real estate prices.
- ***Higher chance of collisions*** due to tight spacing and constant stopping-and-going.
- ***Motor vehicle emissions*** contain pollutants that contribute to outdoor air pollution. One in particular, fine particulate matter (referred to as PM2.5) is strongly influenced by motor vehicle emissions. Studies that evaluate the sources of PM2.5 in our environment find that vehicles contribute up to one-third of observed PM2.5 in urban areas. PM2.5 has been associated with premature deaths in many studies, and health impact assessments have shown PM2.5-related damages on the order of hundreds of billions of dollars per year. Recently, an expert committee convened by the Health Effects Institute in Boston, Massachusetts, summarized the available evidence on exposure to traffic-generated air pollution and negative health effects. They find strong evidence for a causative role for traffic related air pollution and premature death, particularly from heart attacks and strokes. PM2.5 is emitted directly, and it is also produced by secondary formation, as sulfur dioxide (SO₂) and nitrogen oxide (NO_x) emissions contribute to the formation of sulfate and nitrate particles. Exposure to PM2.5 also causes other health effects such as asthma attacks, and other respiratory illnesses.
- Stressed and frustrated motorists, encouraging road rage and reduced health of motorists.



- **Road rage** Road rage is aggressive or angry behavior by a driver of an automobile or other motor vehicle. Such behavior might include rude gestures, verbal insults, deliberately driving in an unsafe or threatening manner, or making threats. Road rage can lead to altercations, assaults, and collisions which result in injuries and even deaths. It can be thought of as an extreme case of aggressive driving.

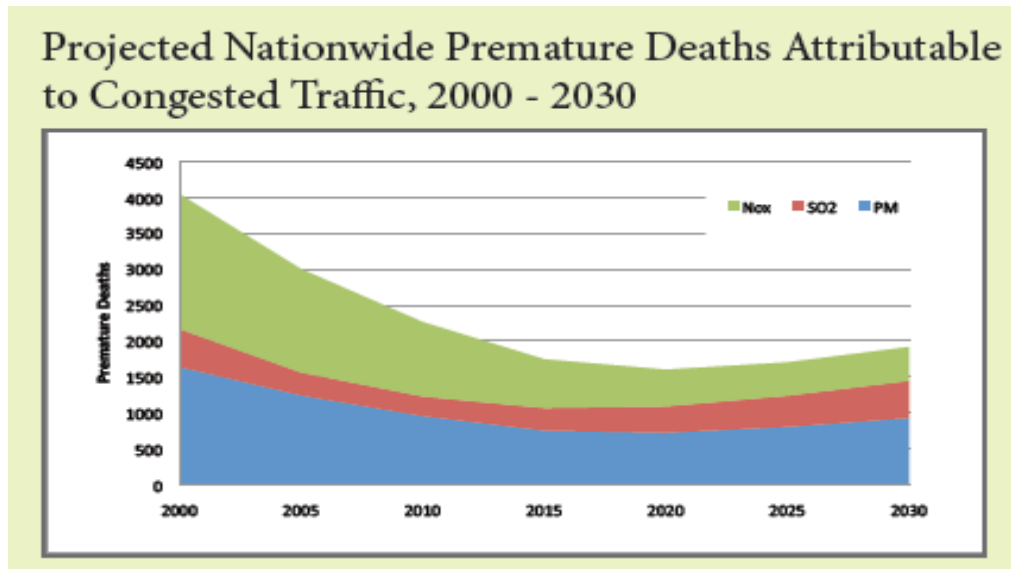


Figure 3.2 The nationwide estimates for premature deaths attributable to congested traffic for 2000-2030. The colored sections indicate the portion of these premature deaths attributable to Nox, primary PM25 and SO2

3.2 Some Solutions for Solving the Traffic problem

In cities, the ability to respond precisely and flexibly to changing traffic levels can reduce the impact on people and the environment. With roughly one-third of the world's population expected to be living in cities by 2030, investing in smart traffic management solutions makes sense. In a survey conducted with Siemens' support, 25 of the world's largest megacities stated that traffic was by far their most pressing infrastructure problem. The average speed on roads in Mexico City, for instance, has dropped to 11 kilometers per hour – around the same as in 1910, when horse-drawn carriages were still on the roads. Cutting down the amount of stop-and-go involved in urban driving could bring down fuel consumption by as much as 20 percent and cut nitrogen oxide and carbon dioxide emissions by up to 50 and 33 percent respectively. Managing urban traffic sustainably therefore means aiming to enable drivers to ride a wave of green lights. This is posing a sizeable mathematical challenge for Siemens researchers, but in Warsaw, Vilnius, Copenhagen and Münster, adaptive traffic signal control has been shown to successfully reduce traffic's environmental impact by several hundred tons of CO2 a year.

Use of existing popular technologies for traffic congestion detection and management has been explored by several researchers in recent times. However, most of the works

either deal with congestion detection strategies or address the issue of congestion control through signal manipulation or other aids. To the best of our knowledge, no effort has been made so far to develop a comprehensive system that automatically detects as well as controls traffic congestion based on detected level of congestion. The aim of this project is to design a complete strategy for automatic road traffic congestion detection in realtime and to devise an effective scheme for control management of congestion using Passive RFID.

In this section, we discussed some of the most widely adopted technologies for congestion detection and management and their limitations. Finally, we described the outline of our Passive RFID based congestion detection and management scheme to overcome the existing problems. Further, we have illustrated a simulation model that we have developed to generate different traffic congestion scenarios, to implement and test our scheme for congestion detection and management.



Figure 3.3 Traffic congestion detectors in Germany.

3.2.1 SMART TRAFFIC CONTROL SYSTEM USING PLC and SCADA

- The scope of this system is to present the implementation of a smart traffic light control system based on Programmable Logic Controller (PLC) technology. We, in this method, intend to measure the traffic density by counting the number of vehicles in each lane and their weight, then park in automated parking or diverge them accordingly. It is also difficult for a traffic police to monitor the whole scenario round the clock. So, this system can be implemented on highways and city traffic.

- In this method we are proposing to reduce the heavy traffic and congestion on the road by using PLC based traffic diversion system. This would work on weight sensing using sensors whose output will be fed to a PLC, which will control the traffic diversion. This method is in two parts:

A. Diversion:

Weight sensor is placed at toll booth. It senses the weight & sends signal to PLC. PLC will generate a slip having the info about the vehicle in the form of barcode. PLC will give the diversion according to the weight of the vehicle (Figure 3.4).

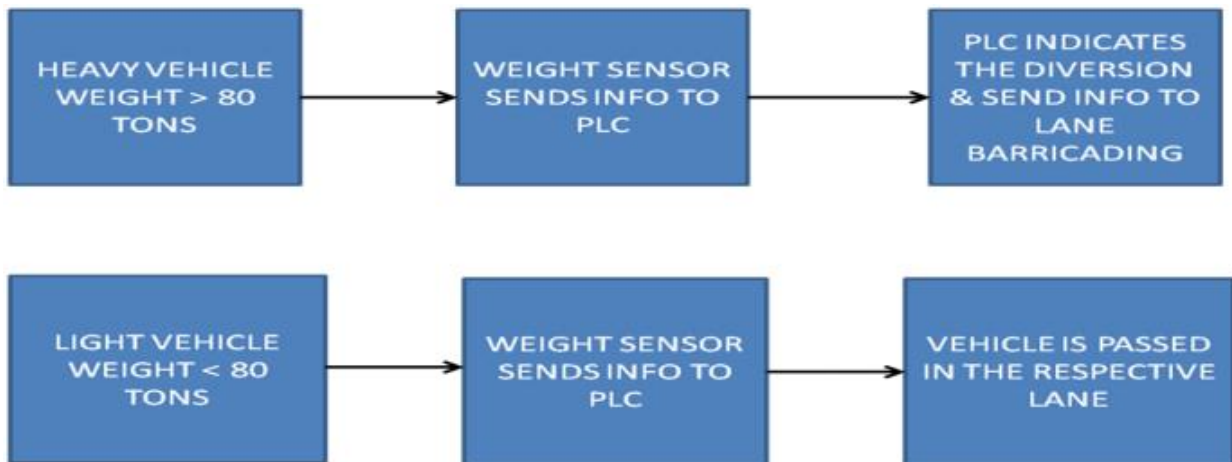


Figure 3.4 Flow chart of diversion of vehicles based on weight

B. Congestion Control

In this there are two counters– UP Counter (at the starting of the road) & DOWN Counter (at the end of the road) whose maximum value is 100. When a vehicle enters the road, UP Counter is set and vice versa. There are 3 conditions for allowing the vehicle in the area Figure 3.5



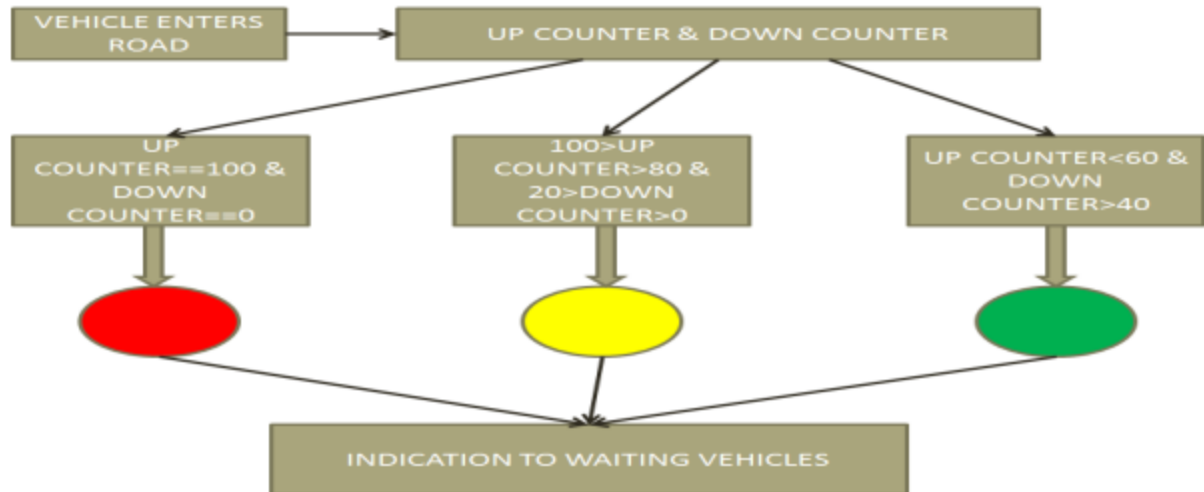


Figure 3.5 Flow chat of diversion of vehicles based on density

- If UP Counter==100 & DOWN Counter==0, then red light will be shown i.e. no vehicle will be allowed to enter the area.
- If $100 > \text{UP Counter} > 80$ & $20 > \text{DOWN Counter} > 0$, then yellow light will be shown i.e. vehicles will be told to be ready to enter the area.
- If $\text{UP Counter} < 60$ & $\text{DOWN Counter} > 40$, then green light will be shown i.e. vehicles will be allowed to enter the area.

3.2.2 Intelligent transportation system

- 21st century technologies, referred to as intelligent transportation system (ITS) technologies, have been researched, deployed, and tested to some degree for ten years or more. These technologies provide information about the transportation system and support development of tools that traffic professionals and travelers never had before. The technologies can be generally grouped into six types.
- **Information Gathering:** These are the technologies that collect information more thoroughly or more frequently than transportation professionals have been able to do in the past. Surveillance and detection cameras, traffic sensors, vehicle probes and infrastructure sensors are examples.
- **Information Sharing:** As personal portable technology matures, an ever-increasing array of devices is available to share travel information. Today, dynamic message signs, highway advisory radio, 511, Internet web sites and specialized warning systems (like fog warnings) are stationary technologies used routinely to share information with travelers.
- **Control:** There are also advanced technologies and software that provide improved methods for controlling and managing traffic. Advanced traffic signal control provides ways of remotely adjusting systems of signals to respond in real-time to changing traffic demands. Lane control signals, ramp meters, transit signal priority, and variable speed limit signs are other technologies that provide other opportunities to control traffic in real-time.

- **Vehicle-based:** This is an area of growing technology innovation. From complex crash avoidance technologies to in-vehicle guidance systems currently on the market, vehicle-based technologies hold promise to dramatically improve safety and give travelers (including commercial drivers) meaningful information about travel conditions to help them avoid bottlenecks and other potentially disruptive congestion situations.
- **Vehicle to roadside to home-base** Sometimes referred to as tracing and tracking technology, this allows the freight operators of commercial carriers to maintain contact with their fleets and the cargo they are moving via satellite systems as well as terrestrial based systems. These systems are expanding in use and experiencing lower per unit costs, and have the extra advantage of addressing security needs as well as productivity and safety needs.
- **Payment:** Electronic toll tags and "smart" cards for transit and parking are technologies that are seeing rapid deployment. These technologies add efficiency to payment operations and expedite traffic flow. Taken together, these technologies enable new ways of managing the transportation system to improve its operation. The technologies themselves are not the answer, but the improved ability to operate the system, enabled by the technologies, is key to addressing congestion.

The opportunity for better managing and operating the transportation system to address congestion comes from combining new technologies with a new focus on operations within transportation agencies. Agencies that have embraced 21st century operations will take advantage of new technologies and apply them to achieve better system performance. Applications like freeway management, arterial management, incident and special event management, work zone mobility and safety management, and road weather management marry technology innovations with a desire to better serve customers through improved mobility. These same

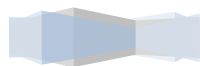
- **Technologies**, supplemented by pre-arrival and clearance information for freight at our borders and intermodal terminals, can improve freight flows as well.

3.2.3 Active Traffic Management:

System opens up UK motorway hard shoulder as an extra traffic lane, it uses CCTV and VMS to control and monitor the traffic's use of the extra lane

3.2.4 Advanced Traffic Management Systems:

Advanced traffic management systems (ATMS) seek to reduce, or at least contain, traffic congestion in urban environments by improving the efficiency of utilization of existing infrastructures. These systems typically seek solutions to congestion problems occurring on urban freeways and surface streets through the deployment of state-of-the-art sensing, communications, and data-processing technologies. Problems considered include both congestion caused by regular traffic patterns (congestion management systems) and



traffic problems caused by stalled vehicles or other unpredictable incidents (incident management systems).

ATMS typically attempt to take advantage of information that can be provided by roadside traffic sensors. These systems typically attempt to use available traffic information to develop optimal traffic control strategies addressing traffic needs at a single intersection, along an arterial or freeway, along a given corridor, or throughout a given area. Real-time solutions capable of automatically adjusting to changes in traffic conditions are often sought. These systems also frequently rely on variable message signs or other information dissemination technologies to provide relevant traffic information and travel recommendations to travelers.

3.2.5 Our Proposed System:

- There are a variety of technologies that are being used to detect the congestion of traffic. Most popular technology is the inductive loop system, the simplest detectors that count the number of vehicles during a unit of time. Using Airborne Camera and the image processing technology, GPS devices and webcam, Radar technology etc. congestion can also be detected. But these technologies have several drawbacks, such as installation problems and cost. Radio Frequency Identification (RFID) is an emerging technology that is still remains largely unexplored in the area of automatic congestion detection. Vehicle detection and counting can be done effectively using RFID technology. In this paper, we propose to design a smart and fully automatic system that will detect the congestion in real time, and subsequently manage it efficiently to ensure smooth traffic flow with the use of Passive RFID devices.
- The current traffic light system is fixed. It has fixed time intervals to pass the traffic from either side of the road, which creates congestion problem and its negative impacts on the environment and the resources. Our Traffic Control System which is implemented using RFID technology solves this problem on real time basis. It will observe the traffic conditions.

Let us say that we have -for example-four intersecting junctions or roads connected to each other, and then our proposed system will take the decision - which is- which one of these roads will be opened for a certain period of time according to traffic density or the number of cars in each junction.



- Our RFID technique deals with a multi-vehicle, Multi-road junction area. It provides an efficient time management scheme, in which a dynamic time schedule is worked out in real time for the passage of each traffic column. The number of vehicles in each column and the routing are properties, upon which the calculations and the judgments are based. One of the most important advantages of this project is that the time of the traffic lights is variable according to the traffic density in the junction...

3.2.6 Advantages of Traffic Control Management Using RFID Compared to Other techniques

The following is a simple comparison between Traditional Traffic Management Systems and our proposed system (Traffic Control Management Using RFID Technology) to show the features and the advantages of our proposed system.

Table 3.1 Comparison between traditional systems and proposed system

Different systems	Traditional Management systems	RFID Technology
Point of view		
Decision based on:	Pre-determined Timer	Congestion
Hardware Used	Sensors or cameras	RFID Readers and Tags
Software Used	Simple Timer	Smarter Software used to detect the most congested Road
Time of traffic:	Constant	Variable according to congestion
Ability of tracking vehicles	Does not exist	Exists
Ability of Identification	Does not exist	Exists
Efficiency according to traffic jam	Lower	higher

From the above comparison it's clear that our proposed system has better efficiency, more features and advantages than traditional Traffic Management Systems.



- **It's also clear that the main advantage of our system is “The Identification” so it's so useful in:**
 - The Ability of Detection of Stolen cars.
 - The ability of identification of Emergency Vehicles like Police cars and Ambulances which have Priority for Passing.
- **Decrease the amount of Traffic jam** or Traffic Congestion.
- The route or junction having the highest capacity has **the priority for passing**.
- Thus, **the toxic gases emissions** will be decreased.
- **The Time Delay** will be decreased.
- **The Fuel Consumption** will be decreased.
- **The time of the traffic lights is variable** according to the traffic density in the junction.
- The following section is organized as follows. Chapter 4 is concerned about how our project could be implemented in the real life (Large Scale) based on some statistics and information from real traffic situations. Chapter 5 illustrates a block diagram or a flow chart about how our proposed traffic control management using RFID scheme works and how the components are connected to each other.

Then we will describe exactly the main tasks in the project which they are:

- Programming (using c#).
- Interface between algorithm and traffic lights (Arduino kit).
- Simulation Tool (Aimsun) on which we built our model.
- Maquette (Small scale) on which we have proposed our project.

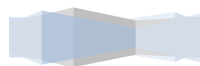


Chapter 4: Large scale

We apply our project on “el nahda” square as the traffic problem is dominant in this place as we pass through it every day and this problem make us waste a lot of time and may make us late for our lectures and meetings. So we think to solve this problem by a traffic management scheme using a new and a suitable technique like the RFID.



Figure 4.1 el nahda square from google earth



4.1 Tag selection:

We evaluate many parameters to select our tags for our implementation, such as tag type, read range, operational environment, data transfer rate, form factor, memory, alternative data storage in case of defective/damaged tags, and what type of product will be tagged.

4.1.1 Read Range:

As we show in chapter one, tags come in three basic types: active, passive, and semi-passive. These tag types differ mainly by their read range and cost. Active tags have long read ranges and high prices compared to passive tags. Semi-passive tags fall somewhere in between, depending on their design and capabilities.

Therefore, Active tags are typically used in real-time locating system and can read from hundreds of feet away.

Significant read range differences exist among the different frequencies used for active tags. Generally, an active tag's read range increases with decreasing frequency used (if the interrogator- and tag-transmitted power stays the same). The longer the wavelength, the farther the wave will propagate depending on available power. For instance, a tag operating at 433 MHz will achieve a farther read than a microwave tag using the same power output. The data transfer speed capabilities of various frequencies apply to active tags as they do to passive tags.

From google earth we measure the width of the street in el nahda square.

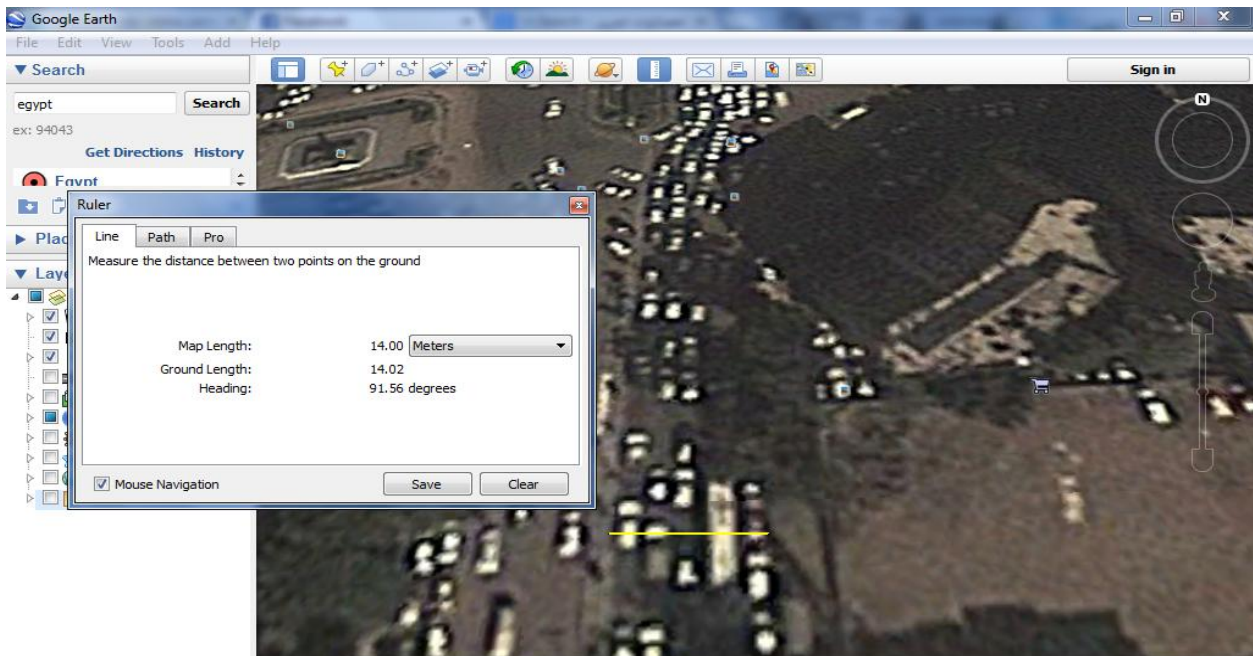


Figure 4.2 the length of street

This point in the process, a frequency should have already been selected

Table 4.1 summary of RFID Operating Frequency

Frequency Type	LF	HF	UHF	Microwave
Frequency range	<135kHz	13.56MHz	860 - 930 MHz	2.45 GHz
Physical coupling	Inductively-coupled systems		Backscatter systems	
Tag use	Passive	Passive	Passive/Semi-passive/Active	Passive /Active
Communication Boundary	Near Field	Near Field	Far-Field	Far-Field
Approximate read range	< 0.5m	< 1m	< 7m (passive) < 80m (semi) > 100m (active)	> 100m (active) < 3m (passive)
Antenna	Coil	Coil	Dipole	Dipole
Effect of liquid	None	Low attenuation	High attenuation	High attenuation
Effect of metal	Disturbance	Disturbance	Attenuation and reflection	Attenuation and reflection
Data rate	< 10 kbit/s	< 100 kbit/s	< 100 kbit/s	< 200 kbit/s

Therefore, we compare between the operating frequencies to get the one which is suitable for our application, and as we need a read range about 15 meters , so we choose the UHF , then we can choose the semi-passive tag as its read range is about 80 meter .

4.1.2 Operational environment

The tags must be able to withstand wide temperature variation and possibly humidity, and as Tag manufacturers provide datasheets for their tags, which will specify all of the environmental conditions in which the tag can operate.

4.1.3 Data Transfer Rate

The tag's data transfer rate requirement is driven by how much time the tag can spend in a particular read zone or how much data will need to be read from or written to the tag. The data transfer rate increases as the frequency of the system increases. Low frequency has the slowest data transfer rate; microwave has the highest data transfer rate.

So, in our application we need the tag to be with high data transfer data to take the decision faster and due to the above table UHF has a data rate about 100 Kbps which is suitable.



As the active technology is often used for enhanced range so that the data can be processed before the vehicle passes through the route.

4.1.4 Form Factor

Many forms of tags are available on the market. Our base in our choice is on the tagging volume and our application technology as well as the necessary protection from the environment and physical wear.

4.1.5 Memory

Memory is also a prime consideration during the design selection phase. We will choose a tag that provides an appropriate amount of memory depending on how much data and what type of data a user needs to write to or read from the tag. Active tags and semi-passive tags can contain larger amounts of memory than passive tags, because these tags can keep a memory bank powered even when it is not being queried over a long period of time.

And as application in our we want to save a large amount of data like the driver's ID, type of the car, color of the car ...etc. So the active tags and semi passive is suitable for our application.

4.1.6 Type of Product

When selecting the tag that we will use in our implementation, we consider the type of product that will be tagged. Some tags are specifically designed for products containing metal, others are specifically designed for products containing high moisture materials, and some generic tags are built for application onto corrugate that is used for packaging.

At our implementation we will use a metal tag as we used it on a car,

Tags designed for use on metal (as well as other specialized tags intended for use on glass, plastics, wood, and other materials) will have a calculated mismatch in the resistance of the chip and antenna. When the tag is applied in close proximity to metal, the detuning that occurs will bring these numbers much closer together. However, this tag cannot be used on nonmetallic surfaces because it would not get detuned and therefore would not perform according to specifications. The same considerations apply to any other specialized tags.

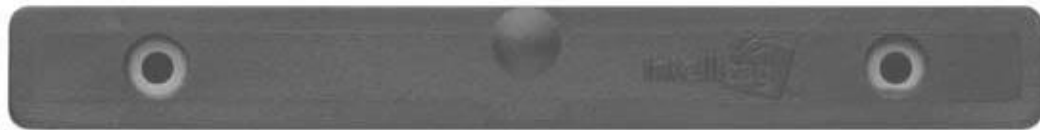


Figure 4.3 show a Tag designed for use in a metal



4.1.7 Region

As the global RFID market emerges, it will be necessary for tags to be manufactured that will respond reliably in all regions of the world within the UHF band from 860 to 960 MHz. In the past, tags were designed to resonate at a specific frequency, usually the center of the targeted band. Now manufacturers are working to create tags with a center band with a very wide range.

4.2 Readers Selection

After choosing the appropriate tag for our installation, we must ensure that the reader we select can operate on the appropriate frequency, can support the protocols of the tag, has the capability to integrate with the back-end system through the appropriate interfaces, and can survive the conditions associated with the environment within which it will operate.

The frequency of the reader should have already been chosen when you selected an operational frequency for your tags. The frequency of the reader must match the frequency of the tag or your system will not work.

The protocols supported by the reader must all match the tags you selected for your implementation. Likewise, if you are accepting tags from trading partners, your reader must also support the protocols that your trading partners use. Several multi-protocol readers on the market allow us to choose one or more protocols, according to our needs. However, if we choose more than one protocol, the reader will perform slower because it has to run through all chosen protocols when scanning for the tags in the reader zone.

In addition, it is a good idea to ensure that your readers can upgrade or change, fairly even power response across 860–960 MHz protocols (usually through firmware upgrades) based on changes in the market, so that you do not end up with an obsolete reader as new protocols emerge. Readers can be also selected according to their communication capabilities.

Many readers today have general-purpose I/O (GPIO) ports that can be configured to drive specific applications and that have direct reader capabilities with light stacks, light break sensors, and a wide variety of I/O devices. Also, many readers have internal processors and operating systems that can run software directly on the reader, which lets it act as an edge server that will filter and aggregate data prior to handing it off to the back-end system.

The interfaces the reader supports must match the network that you have installed currently at a facility. If you have a wireless network employed and want to add your readers to it, they must support the security methods used on your network. If the readers do not have wireless capability built in, external wireless adapters can be used that connect to any standard Ethernet port, which are present in most readers.

Another important consideration is the environment in which the reader is being installed. To see what happens when the environmental factors are not properly considered, see Figure 4.4.



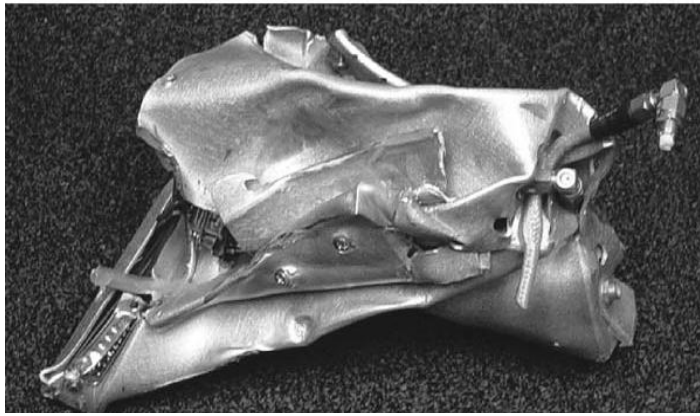


Figure 4.4 remains of a reader when environmental factors are too intense
So, we compare between a lots of products of a distinct companies.

4.2.1 Our reader:

UHF Long Range Reader ID ISC.LRU3500

SPECIAL FEATURES:

- Robust metal housing for use in industrial environment
- Read Range up to 16 m (53 ft)
- Identification of up to 330 Tags / Second in Dense Reader Mode
- 4 Watt Output Power (only LRU3500)
- Power over Ethernet (only LRU3500)
- USB-Port for WLAN-Stick or external Memory Stick
- 5 Inputs and 5 Outputs suit industrial needs
- Linux Operating System

Description:

The UHF Long Range Readers ID ISC.LRU3500 are one of the most powerful reader of the product line OBID i-scan® UHF. ID ISC.LRU3500 is licensed according to ETSI, FCC and IC and is characterized by the following features:

- High receiver sensitivity cares for an enlarged and at the same time homogeneous tag detection range
- Support of Transponders according to EPC Class1 Gen2 and ISO 18000-6-C (ISO 18000-6-B possible on demand)



- Reader protection against fault conditions like antenna shortcut, antenna mismatching and electrostatic discharge
- Robust aluminum die case housing for usage in rough and industrial environments
- Increase of enclosure rating to IP 64 due to optional available connector sealing cap for the connector block
- Quick installation due to easy access to interfaces and antenna ports
- Antenna Port Indication: Display of active antennas (green), read events (blue) and possible antenna mismatching (red) via 4 separate LED's
- Full support for the UHF Multiplexer ID ISC.ANT.UMUX to be used in antenna systems with a maximum number of 2.048 antennas
- Various configuration options for software and hardware
- ACC (Application Connectivity Controller) with Linux operation system for installation of application software directly on the reader platform
- Hardware interface ports: Ethernet, RS232, RS485, USB and an USB-Host for WLAN dongle or memory stick; additionally the reader offers a Wiegand / Data-Clock interface to be used only in Scan Mode for data transmission from reader to host.
- Readout of RSSI data for localization of identified transponders
- High Read Rate for fast and reliable identification of large transponder populations in Dense Reader Mode.

ID ISC.LRU3500

- Power supply: 24 V DC ($\pm 5\%$) or Power over Ethernet (PoE)
- Output power: max. 4 W max. 1 W with PoE
- Read range: 16 m
- Applications: For operation in particularly disturbed and metallic environments
High tag population (> 150)
- Radio Licence :
 - EN 302 208, FCC 47 CFR Part 15,
 - iC RSS-GEN und RSS-210
 - Ready for upcoming Radio Regulations
- The maximum Read Range is depending on the used antenna, the antenna cable, the used transponder and the environmental conditions.



Table 4.2 Technical Data

Mechanical Data		Features	
Housing	Aluminum, powder coated	Supported transponder types	EPC Class1 Gen2 ISO 18000-6-C (Upgrade Code) ISO 18000-6-B (on demand)
Dimensions	260 mm x 157 mm x 65 mm (10.23 x 6.18 x 2.56 inch)	Signaler	16 LEDs for diagnosis of reader operation and antenna status
Weight	2.000 g	Supply Voltage on Antenna Outputs (only LRU3500)	24 V DC / 200 mA
Protection Class	IP 53, IP 64 (with protection cap)*	Other Features	Anti-Collision Real Time Clock RSSI
Color	RAL9003 Signal-White	Environmental Conditions	
Electrical Data		Temperature	
Power Supply	24 V DC ($\pm 10\%$) or Power over Ethernet (PoE)**	- Operation	-25 °C to 55 °C
Power Consumption	max. 35 VA	- Storage	-25 °C to 50 °C (PoE) -25 °C to 85 °C
Operating Frequencies		Humidity	5 % to 95 % (non-condensing)
- Version EU:	865 MHz to 868 MHz	Vibration	EN 60068-2-6 10 Hz to 150 Hz: 0,075 mm / 1 g
- Version FCC:	902 MHz to 928 MHz	Shock	EN 60068-2-27 Acceleration: 30 g
Output Power		Applicable Standards	
- LRU3000	300 mW to max. 2 W	Radio Regulation	
- LRU3500	300 mW to max. 4 W 300 mW to max. 1 W (PoE)	- Europe	EN 302 208
Antenna Connector	4 x SMA-Female (50 Ohm); integrated Multiplexer	- USA	FCC 47 CFR Part 15
RF-Diagnosis	RF-channel monitoring Antenna SWR control internal overheating control	- Canada	IC RSS-GEN, RSS-210
Outputs		EMC	EN 301 489
- 2 Optocoupler	max. 24 V DC / 30 mA	Safety	
- 3 Relays	max. 24 V DC / 1 A switching current, 2 A permanent current	- Low Voltage	EN 60950
Inputs		- Human Exposure	EN 50364
- 5 Optocoupler	5 V DC to 10 V DC / 20 mA max. 24 V DC / 20 mA with additional external series resistor		
Interfaces	RS232, RS485, Ethernet, USB, USB-Host for WLAN-Stick or external Memory-Stick, Data-Clock***	* Optionally a connector sealing cap is available which covers the connectors, offers a pull relief for the connected cables and guarantees enclosure rate IP 64.	
Protocol-Modes	ISO Host Mode, Scan Mode, Notification Mode, Buffered Read Mode	** PoE only with ID ISC.LRU3500	
Operating System	Linux (Kernel 3.0) (64 MB RAM, 256 MB FLASH)	*** The reader offers a Wiegand / Data-Clock interface to be used only in Scan Mode for data transmission from reader to host.	
		Note:	FEIG ELECTRONIC reserves the right to change specification without notice at any time.



4.2.2 Reader Installation

You should follow these rules when installing readers:

- Mount readers securely so that they are not disturbed, bumped, or damaged during regular business operations.
- Leave 5 to 6 inches of clearance on all sides of the reader, except on the bottom of a reader that has shock-absorbing mounts (the rubber feet attached to the bottom of most interrogators). This clearance is necessary in order to avoid cable strain when plugging cables into the reader, especially if you run them out of the box through conduit with the rest of the cables.
- Ensure that the reader is placed and installed in accordance with the recommendations of the manufacturer to keep your warranty valid.
- If environmental enclosures (such as IP or NEMA enclosures) are required for the protection of the readers, make sure that you use a correct type of enclosure and that you install it correctly.
- Place readers in a way that will make the status indicator lights or LEDs visible to an operator or other personnel; this can assist you with exception processing and troubleshooting. If you are using an enclosure, you may need to install a window or a light panel/stack for notification of a device failure.

4.3 Antenna Selection:

After choosing the appropriate reader for our application, now we have to select the antennas. Antennas are selected based on their polarization, gain, ruggedness, size, cable, and mounting options.

4.3.1 Polarization

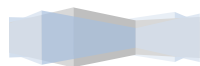
Antennas can be either circularly or linearly polarized. Circular antennas are used in implementations for which the orientation of a tag cannot be assured.

Circular antennas typically have a shorter range than their linear counterparts; however, they will assure that the tag is readable regardless of the angle/orientation of the product/container at which the antenna is directed.

Linear antennas read larger distances than circular antennas but require tagged objects to be in the same orientation every time they are presented to the antenna. A good example of a linear antenna's use would be on a manufacturing line, where machines are placing objects on conveyors for movement through a facility. If a machine places any object on a conveyor line, each object should be placed in the same orientation and position on the line. Linear antennas offer better penetration of dense objects—consider, for example, a company reading tags through rolls of paper. Bulk paper is dense and contains a lot of moisture; therefore, a linear antenna stands a much better chance of penetrating the whole roll than a circular antenna.

4.3.2 Gain

The gain of an antenna plays a huge role in the results your interrogation zone will achieve. A higher gain means that you will be able to read tags at a greater distance with this antenna. However, higher gain also means that the antenna will provide a narrower beam; therefore, if the tags are in motion while being read, they will spend less time in a



narrower beam than they will in a wider beam. If you need to read tags at relatively short distance but want to achieve a longer dwell time, you should use an antenna with a lower gain, because this will cause the beam to be wider but with a relatively short read range. Low- and high-gain antenna beams are represented in Figures 4.5 and 4.6.

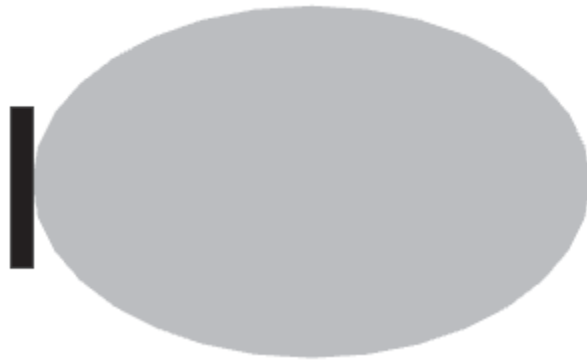


Figure 4.5 Low gain antenna beams



Figure 4.6 High gain antenna beams

HF and LF antennas are not directional. They propagate a signal equally in a manner that is perpendicular to the coil of the antenna. To increase the range of the antenna, you must increase the size of the coil or the power provided to it.

The read range of these frequencies is driven by the practical size for an antenna.

Installation and maintenance of huge antennas is more costly and more problematic than simply using a frequency more suited for a specific application.

4.3.3 Ruggedness

An antenna must be selected based on its abilities to withstand its environment.

Lightweight and inexpensive antennas can be used in retail environments that are temperature controlled and have protection built around the antenna. For example, the antennas can be built into a countertop for a cash register, where they will never be touched by the goods while tags are being read.

However, if you are mounting antennas at a warehouse dock door, they are often exposed to sun, rain, snow, wind, and other elements. The antennas are also exposed to vehicles driving by and personnel working nearby, all of which can accidentally run into them. A rugged antenna may be needed on baggage handling devices that can be used indoors or



outdoors. For such conditions, you must select a rugged antenna if you expect it to survive without additional protection. Covers can be also used to protect exposed antennas.

4.3.4 Size

The size of the antenna must also be considered. The size must be practical and fit the application you are designing. If the antenna is to be embedded within an object, obviously the antenna must be smaller than that object.

Although bistatic readers offer better performance in difficult read situations, their use may be impractical due to a relatively large size of a bistatic antenna compared to a monostatic antenna. A bistatic reader needs two antennas (in separate cases or joined in one case) to function, while the monostatic reader will utilize one smaller antenna.

4.3.5 Mounting Options

Mounting options available with the antenna could also help you with your selection. Check out what the specific antenna models offer (holes for screws, flexible joints, and so on) to determine which mounting option will be the best for your system. You may want to use screws, zip ties, adhesives, industrial Velcro, or other options. Breakaway antenna mounts are similar to the breakaway mirrors installed on cars—these are certainly advisable if you know that the antennas may be occasionally hit by a vehicle or other equipment. Mounting options will not change your antenna's performance but will be important for installation and long-term maintenance.

Antennas can be mounted in readymade dock stands that hold from two to four antennas (and a reader and certain I/Os), which are protected by an impact-resistant cover. Antenna mounts designed for forklifts embed antennas near the mast of the truck and offer a good degree of protection as well, ensuring antennas are close to the goods the truck is carrying and yet protected enough to expect a reasonable lifespan. For antennas on conveyors, extruded aluminum frames can be constructed that surround a section of conveyor to allow antennas to be safely placed in close proximity.

4.3.6 Our Antenna:

UHF RFID Antennas ID ISC.ANT. U600/270

SPECIAL FEATURES

- Circular Polarization for optimized reading performance
- Available for EU and FCC Frequencies
- IP 65 for outdoor applications
- Unique design
- Usable with all available UHF readers
- ID ISC.ANT.U600/270: 30° Beam width, particularly suitable



DESCRIPTION

- The antennas of the OBID i-scan® UHF series are specially designed for use in the different UHF frequency bands. A separate version is available for the European frequency band in the range from 865 MHz to 868 MHz as well as for the FCC frequency band from 902 MHz to 928 MHz
- The circular polarization of the antenna allows an identification of transponders in different orientation. E.g. an intelligent alignment of the antennas within a gate allows a 3-dimensional identification of transponders.
- With the combination of exclusive design and high protection class IP65 the antenna can be used for indoor applications as well as outdoor applications.
- Each Antenna convinces with its individual properties. For nearly each kind of application a suitable antenna is available.
- The development of customized antennas is possible on demand.

Another specification

Dimensions:590 x 270 x 57 mm³

Weight:2.200 g

Protection class:IP65

Gain:approx.11 dBic

3 dB beam width:30° x 65°

Polarization:circular

Antenna connection:SMA socket (50 Ohm)

Temperature range Operation Storage:

- 25 °C up to 55 °C

- 25 °C up to 80 °C



4.4 Algorithm of the Large Scale:

- In our implementation on “nahda” square, we will use 4 readers and each reader will be connected to 2 antennas which cover the area of 15 meters.
- We will put each reader away from the entrance of the road to measure the length of the congestion.
- In every cycle the 4 roads have to be activated, let’s assume the maximum time for each cycle is about 120 seconds, then, the maximum time for each state is about 30 seconds.

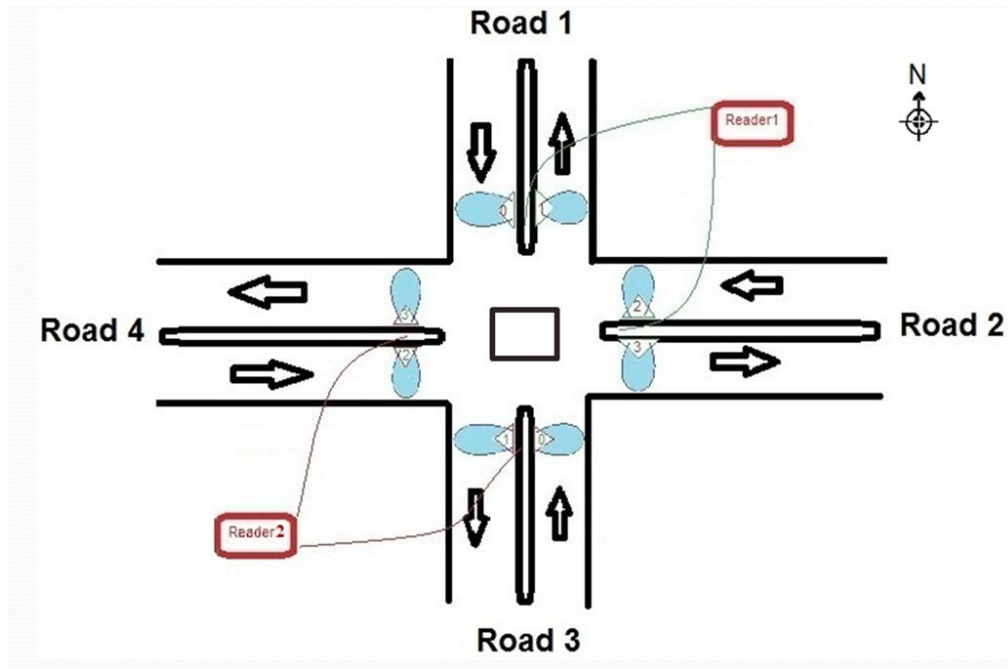


Figure 4.7

Now assume that we will compare between the 4 roads and Road 3 is the most crowded.



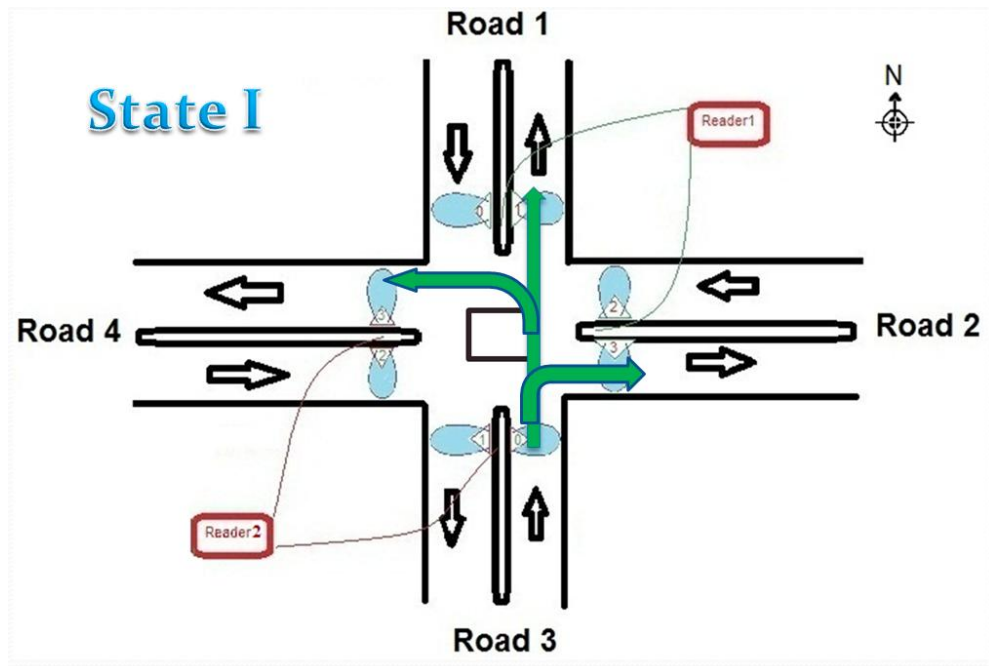


Figure 4.8

When the reader at road 4 reads more capacity of cars, we will close the traffic signal which is in the square, and opens the main way. (road 1 to road 3).

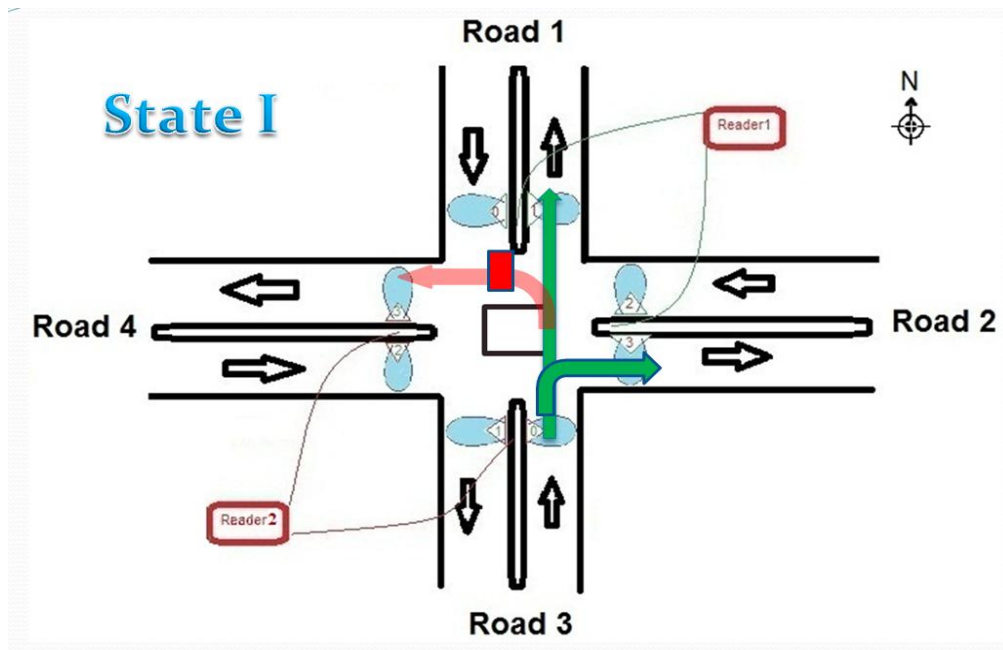
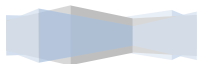


Figure 4.9



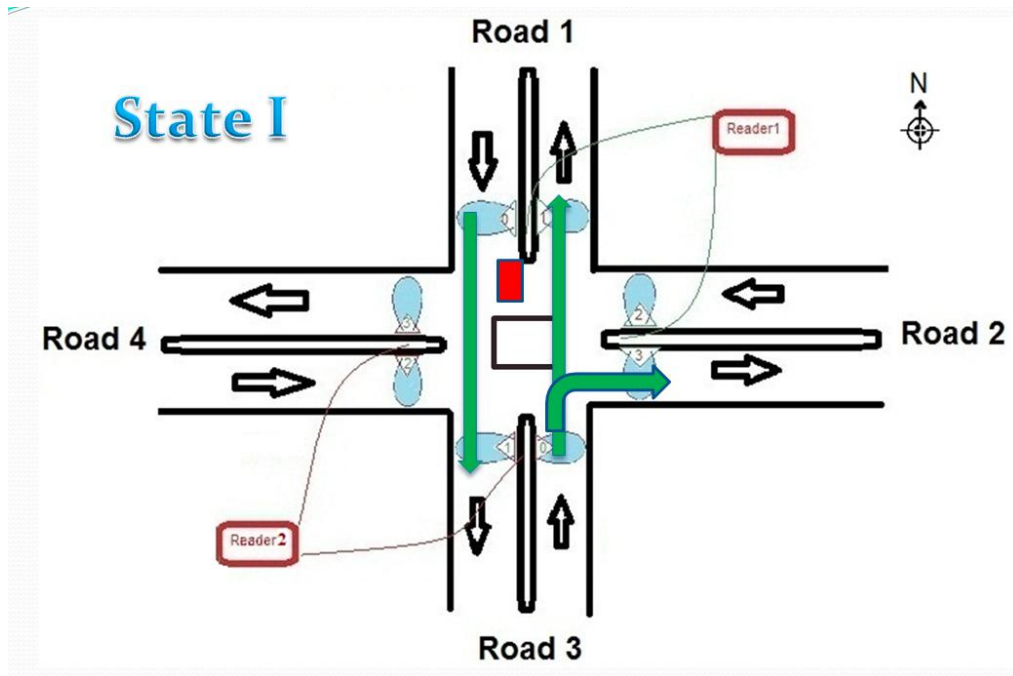


Figure 4.10

If the congestion was in road 1 not road 4, then we will compare between the other 3 roads and do the same steps again.

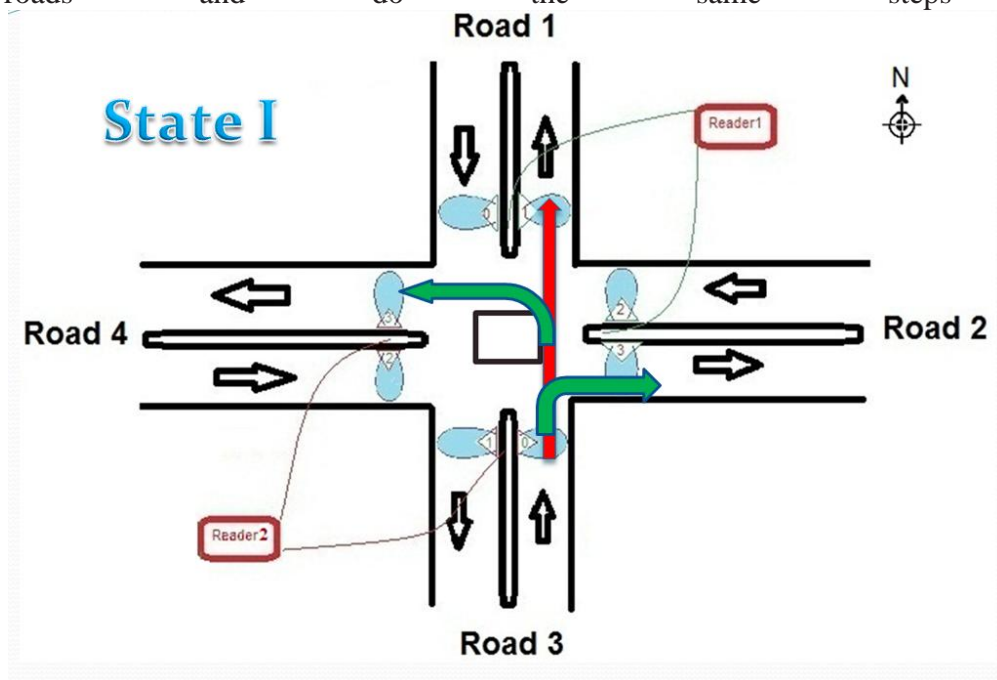


Figure 4.11



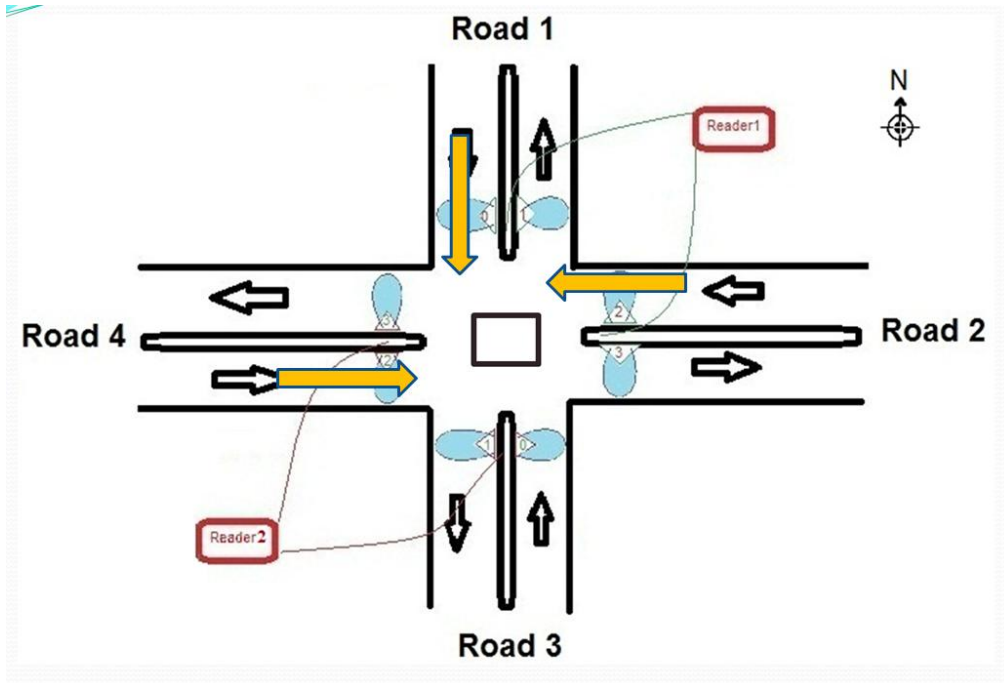


Figure 4.12

Assuming that road 2 is the second crowded road, What if neither road 4 nor road 3 becomes crowded? In this case we will wait till time out occurs , and compare between the crowded from the last 2 roads .

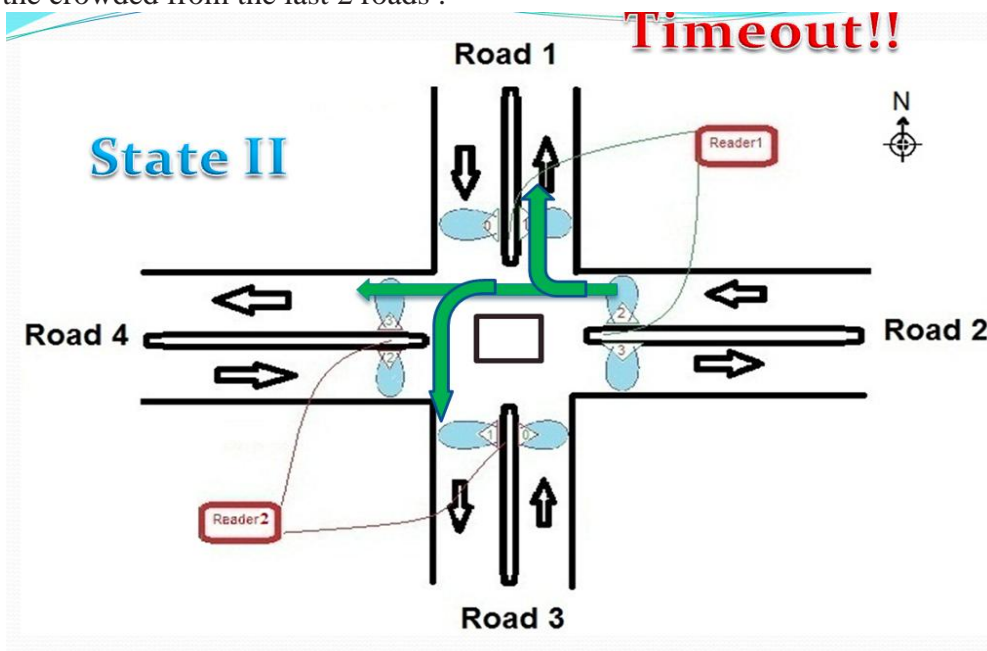


Figure 4.13



When we are in state 2 and there is a police car or any emergency cars in another road, this is an exception case and we will open this road till the emergency cars get out from this road and then return to our state.

There is another exception when a reader in any road reads the ID of a stolen car , then a message will appear in the police office.

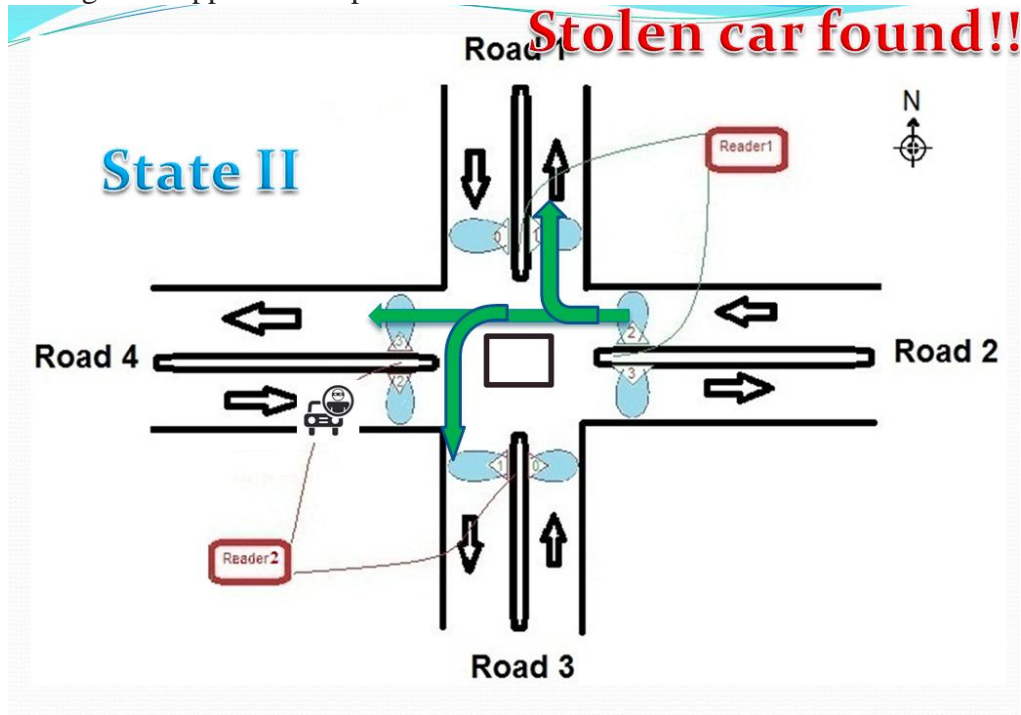


Figure 4.14

In the next cycle, if there were cars in the road did not pass in the first cycle then we will check if these cars were not detected by the readers in other roads, then these cars will be saved in the database to measure the length of the crowded road by the reads of the reader plus the cars which were saved in the database from the first cycle.



Chapter5: System Implementation

5.1 Project outcome

Our project is divided into 2 parts: 1- Small scale
2- Large scale

5.1.1 Small scale:

Our small scale is Maquette of 1.20*1.20 meter wooden box, above it, a drawing of 8 roads (4 in and 4 out) with 6 Traffic lights for the 4 roads and 2 in the middle. At the beginning of every road there is a reader which detects and knows the number of tags (cars). We first initialize readers using cp-start program given from the company. The 8 readers send the data to the laptop (our processing unit) using 8 Ethernet cables and 2 switches. At the laptop, Traffic listener program collects the data from the readers and put it in database SQL server 2008 R2 where it is stored.

At **visual basic program using c#.net** programming, we connect between our program and the data base to collect the number of tags in each reader. To start our program we enter 4 values:

- a- Time at which no congestion is detected in the road, we have to close the road so that other roads will be opened.
- b- Time at which car remains at a reader we consider that road congested.
- c- Tag-ID refers to important vehicles like ambulances, police cars... etc. So that, if the reader detects one of those readers, it will open the road containing this car.
- d- Tag-ID refers to stolen car so it can be tracked. Then we must press reset button at the user interface (to delete previous stored data in the database), finally we must press start button, as shown in figure5.1.



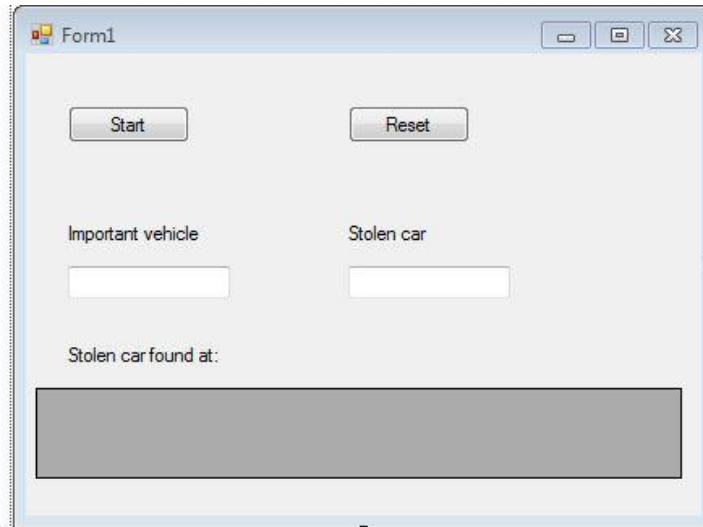


Figure5.1 Fields required at the beginning of the program

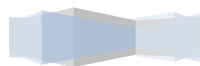
At this moment, processing starts to run the algorithm to know whether there is any congestion in the opened road, if so, we compare the other 3 roads, then the most congested road is now the one that will be opened and we close the previous road.

–Our algorithm will be discussed in details in the next section–.

To do so, we must open the green light for the new opened road and the red light for the closed one, so the C# send the signal to Arduino program using serial port command in the c#.

The Arduino program receives the signals from C#, and sends special signals using USB port to the kit.

The kit is connected with 6 boards, each, containing 2 leds- green led and red led- , it illuminates according to the given signal from the kit.



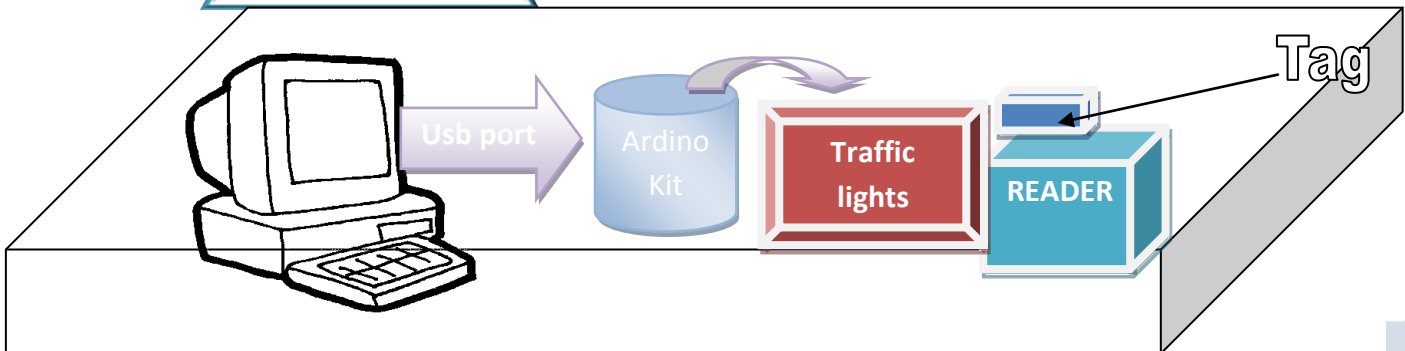
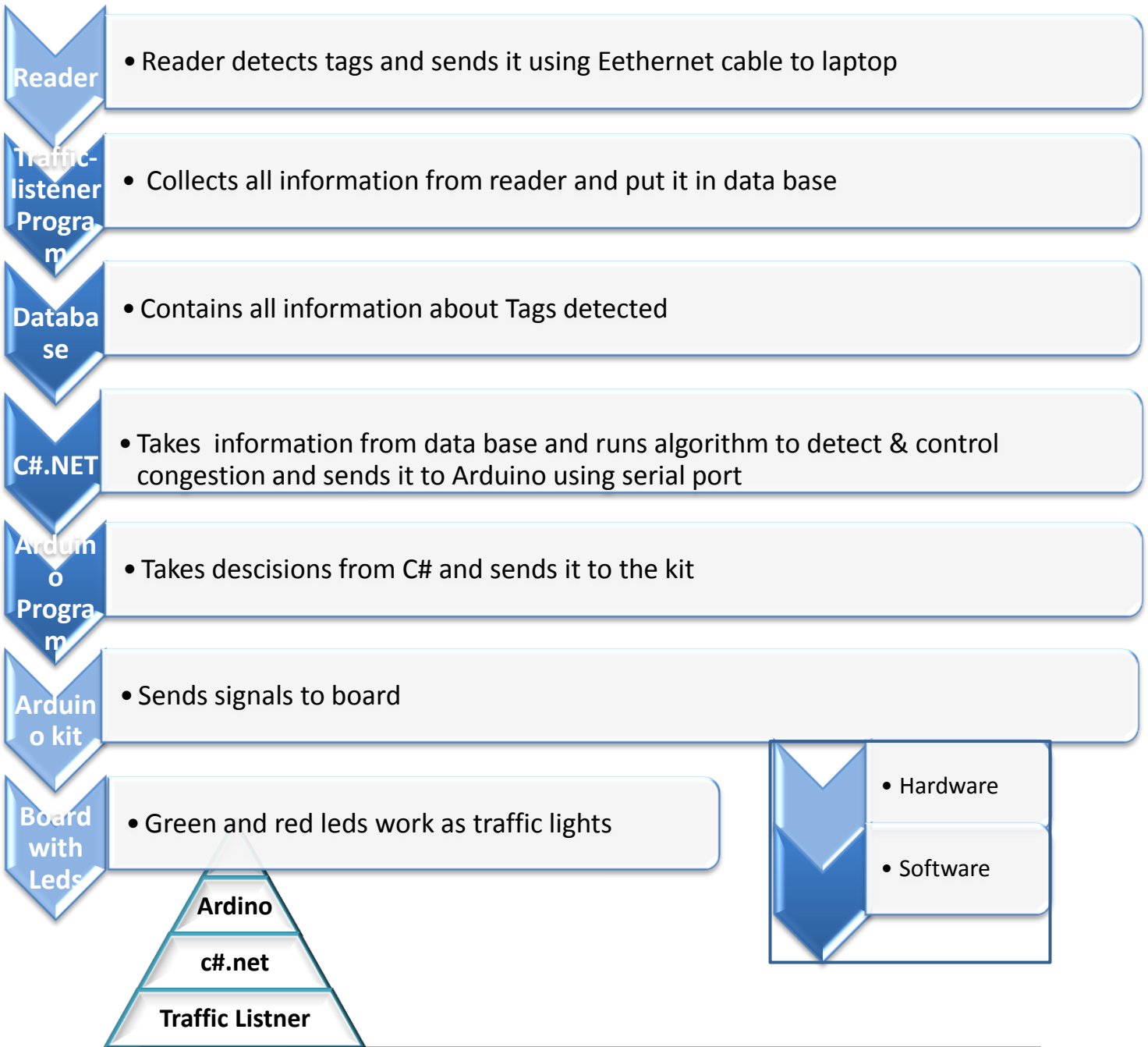


Figure5.2 All blocks of the system

➤ **Components used in small scale:**

1. Maquette of 1.2*1.20.



Figure5.3 the maquette

2. 8 readers from OG-TECH Company.
3. Tags also from OG-TECH Company

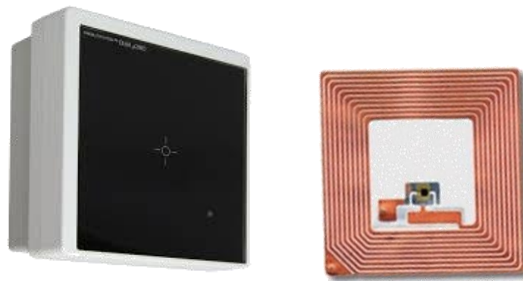


Figure5.4 Reader and tag

- 4- 2 switches and Ethernet cables.

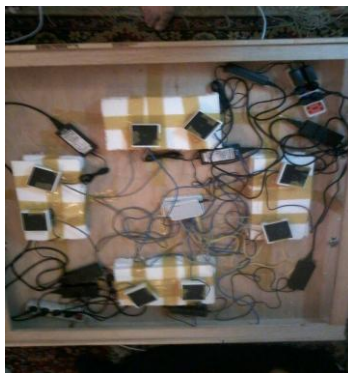
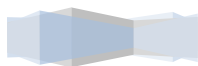


Figure5.5 Switches and Ethernet cables

Note: These figures don't include all materials used but the main components.



5.1.2 Large scale:

In our project we don't consider only small scale Maquette, but also we considered large scale calculation so that we proved that our project can be made in reality as hoped. For the calculation of the large scale dimension, you can refer to chapter 3 in this book(s) which considers all details.

And to simulate the large scale we used AIMSUN program in which we made it like the same small scale Maquette but on the large scale dimensions which are known from the previous calculations in chapter 3, we used the same algorithm and entered random number of cars to ensure the efficiency of our system.

5.2 Algorithm

5.2.1 Explanation of the algorithm

As Mentioned in the previous section, Data received from the 8 readers using Traffic listener program made by OG-Tech Company, then it's saved in database called Traffic listener. At visual basic program using c#.net programming, we wrote our program, it consists of 3 main parts:

- 1- Connecting it with the database and collecting data.
- 2- Running our algorithm.
- 3- Sending decisions to Arduino program.

For our algorithm, it consists of 4 states each represents an open road and 3 closed ones. I first will discuss the flow chart of our algorithm then I will discuss each part of the program but I will take only 1 state as an example, as the 3 other states are similar. First of all we connect the database with visual basic. To collect all stored information about the 8 readers and the read tags and time at which have been read. Then when we run the program, a message box appears to user asking him to enter 4 values:

a- time at which no congestion is detected in the road, we have to close the road so that other roads will be opened.

b- Time at which car remains at a certain reader we consider that road congested.

c- Tag-ID refers to important vehicles like ambulance, police car... etc. So that if reader detects one of those, it opens the road containing this car.

d- Tag-ID refers to stolen car so it can be tracked.



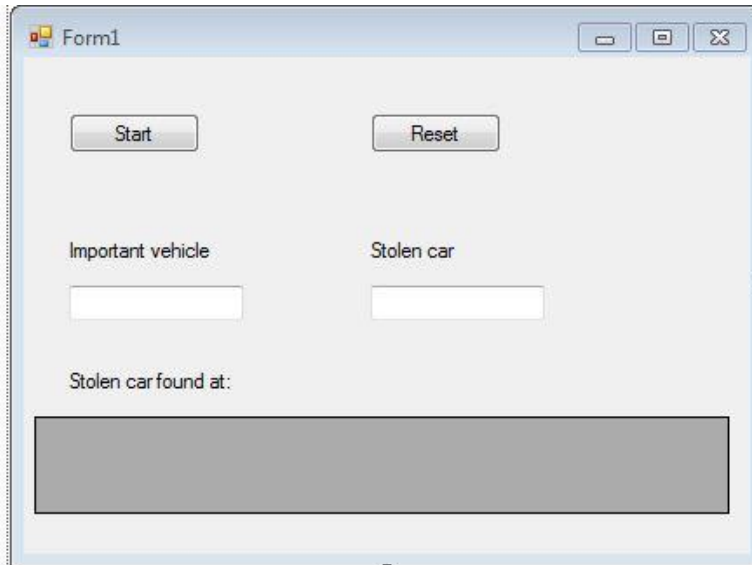


Figure5.6

Then the program uses the last 2 field's values to check whether there is a stolen car or an important vehicle or not

- 1- If a stolen vehicle is found , it displays a message box called (stolen car found) , then data at which this car found at , will appear in a data grid view1
- 2- If an important vehicle is found , it goes immediately to the state containing the important vehicle

Then a while loop starts the first state runs as shown by making the appropriate traffic lights green

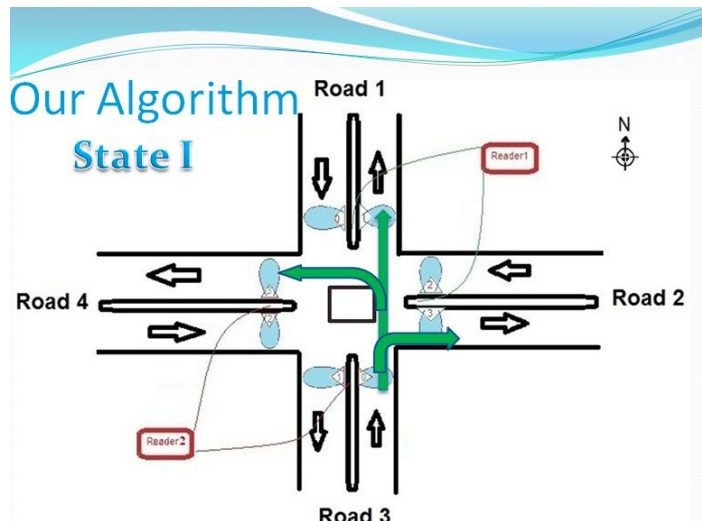


Figure5.7

After we checked if the secondary road congested, if so we open the opposing road

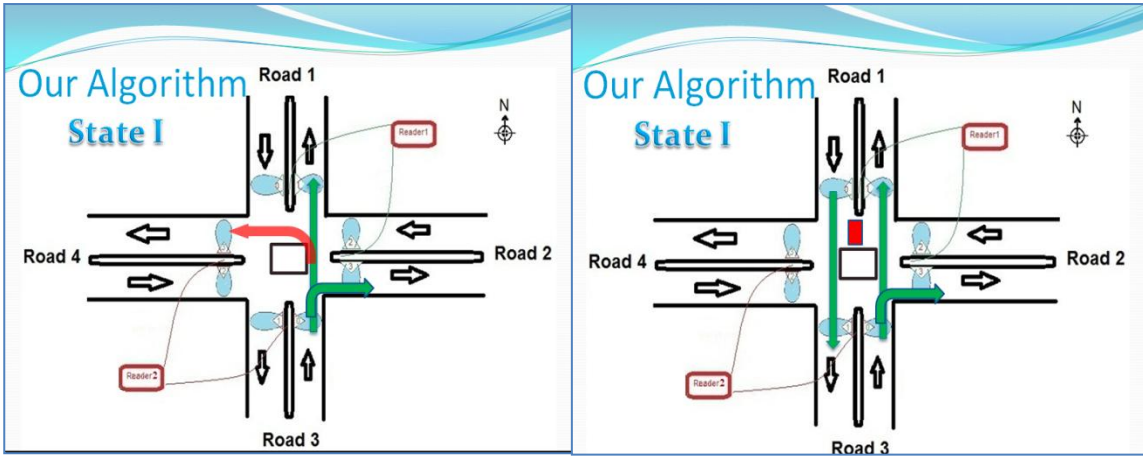


Figure5.8

And start the loop again. If not but the main road is the congested one, we compare the other roads to see which road containing greatest number of cars so to be opened. Before we go to the state containing largest number of cars, we must first be sure that it is not previously opened in this cycle... (In each cycle the 4 roads must be opened but not a must that they open in order). So we made 4 flags, each for a road, once a road is opened its flag is true, if the 4 flags are opened which means that we have completed a cycle, we must turn all flags to false.

Returning to our runtime program, after it checked which road containing greatest number of cars. it first checks its flag, if false it goes to this new state, but if it is true, it checks again the next road containing next greatest number of tags and also checks its flag and decide to which state it will go. Before it goes to any state, there is a mini state, in which all traffic lights turn red, only ones in the square turns green; its function is to evacuate the square from the cars before it goes to the next state.

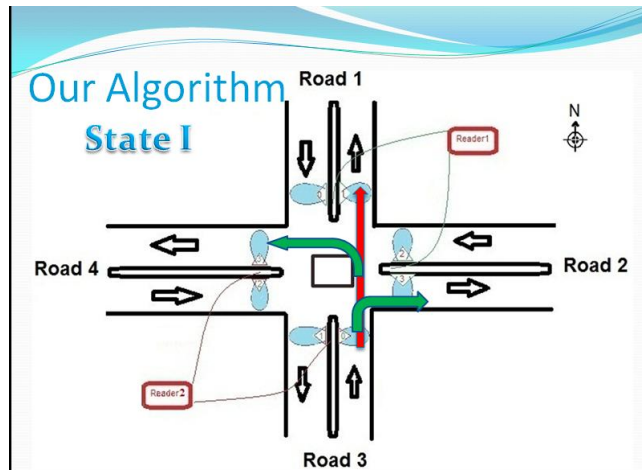


Figure5.9

5.2.2 Flow chart

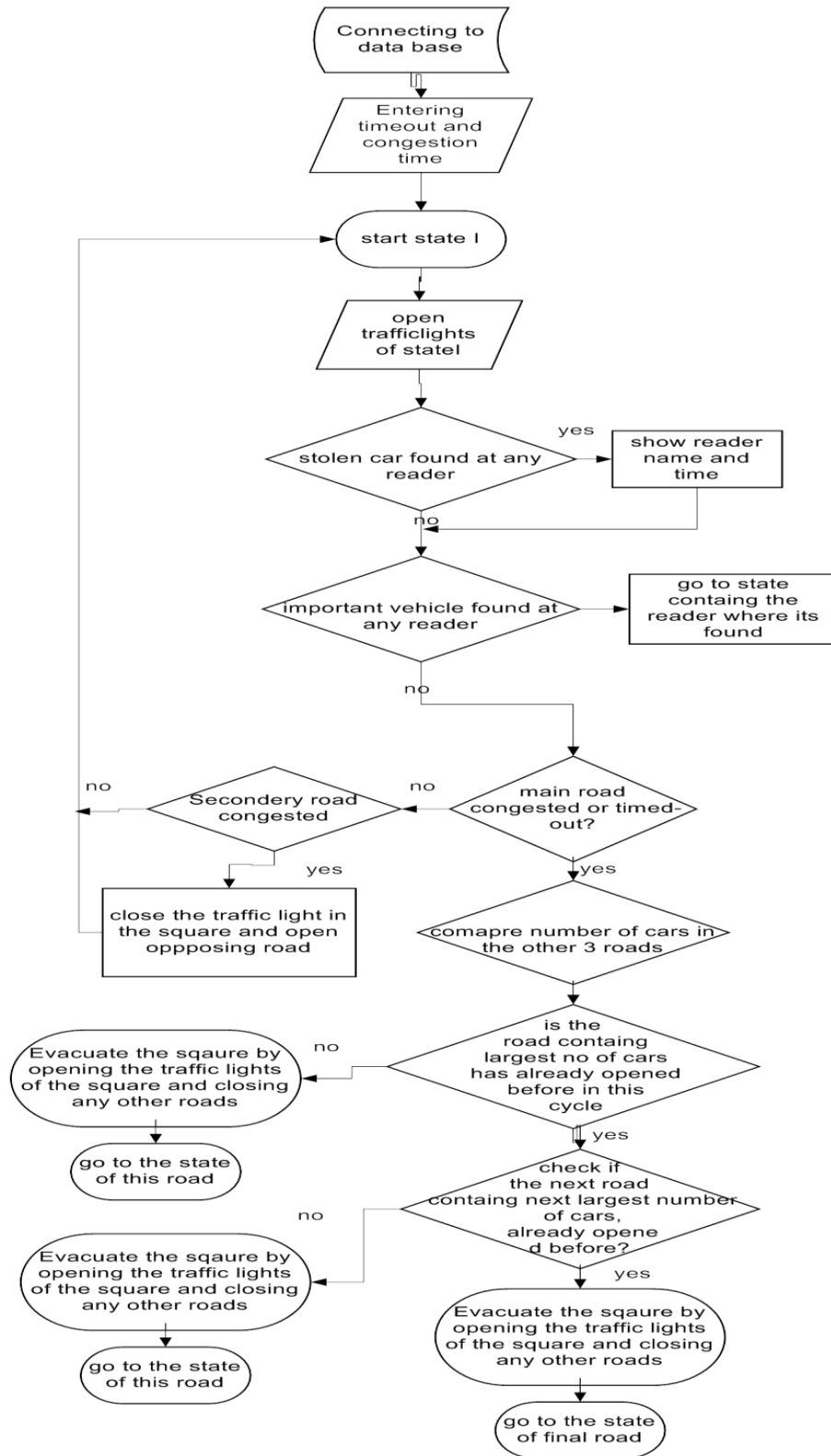


Figure5.10



5.2.3 Explanation of the code:

```

{ public partial class Form1 : Form
  { GP_TrafficListenerEntities ctx = new GP_TrafficListenerEntities();
    public Form1()
    { InitializeComponent();
      MessageBox.Show("please enter time out & time to detect congestion before
you press start");
      MessageBox.Show("please enter ID in case of stolen car or important vehicle..if
there is not please enter 0"); }

    SerialPort port = new SerialPort("COM4", 9600);

    bool timeout1;
    bool fstop;
    System.Timers.Timer timerClock = new System.Timers.Timer();
    private void button1_Click(object sender, EventArgs e)
    {
      fstop = false;
      //////////////////////////////////////
      int condet = Convert.ToInt32(textBox2.Lines[0]);
      int w2t_elashara = Convert.ToInt32(textBox1.Lines[0]);
      int w2t_elashara_2 = w2t_elashara * 1000;
      timerClock.Elapsed += new ElapsedEventHandler(OnTimer);
      timerClock.Interval = w2t_elashara_2;
      timerClock.Enabled = true;
      //////////////////////////////////////
      bool FI, FII, FIII, FIIII;
      bool FST;
      FST = false;
      FI = FII = FIII = FIIII = false;
      //////////////////////////////////////
      state1:
      DateTime currentTime1 = DateTime.Now;//getting the current time
      DateTime timeBefore1 = currentTime1.AddSeconds(-
timerClock.Interval);//current time- timeout time
      var query = from p in ctx.TagLogs where p.InFieldTime <= currentTime1 select p;
      foreach (var x in query)
        ctx.DeleteObject(x);
      ctx.SaveChanges();
    }
  }
}

```

To get information from database

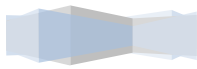
To make user enter the 4 needed values

To initialize serial port to the arduino program

To initialize timer of the timeout
And converting entered value in Timeout field into integer to be used by the timer

The 4 flags of indicating starting of each state

Deleting all previous data at beginning of every state



//

```
if (textBox4.Lines[0] != "0") {
    var stln = textBox4.Lines[0];
    var minaaa = (from minaaa in ctx.TagLogs where (minaaa.TagID == stln) select
minaaa).Count();
    var touty =(from minaaa in ctx.TagLogs where (minaaa.TagID == stln) select
minaaa);
    if (minaaa >= 1)
    {
    if (FST == false)
    {
    MessageBox.Show("stolen car found ");
    dataGridView1.DataSource = touty;
    MessageBox.Show("HORAAAI ");
    FST = true;
    }}}
}
```

Checking the 8 readers if anyone detects the stolen car (value in textbox4) ..If found

- 1- Message box(found)
- 2- View it on data grid view
- 3- 3- another message box as to make the gui respond faster

FST is a flag so that not make message ox

//

```
if (textBox3.Lines[0] != "0") {
    var papawahba = from papa in ctx.TagLogs where papa.ReaderIP ==
"192.168.10.12" select papa.TagID ;
    foreach (var elem in papawahba)
    { if (elem.ToString() == textBox3.Lines[0]) goto stateII; }
    var papawahba2 = from papa2 in ctx.TagLogs where papa2.ReaderIP ==
"192.168.10.14" select papa2.TagID;
    foreach (var elem2 in papawahba2)
    { if (elem2.ToString() == textBox3.Lines[0]) goto stateIII; }
    var papawahba3 = from papa3 in ctx.TagLogs where papa3.ReaderIP ==
"192.168.10.16" select papa3.TagID;
    foreach (var elem3 in papawahba3)
    { if (elem3.ToString() == textBox3.Lines[0]) goto stateIII; } }
```

Checking if an important vehicle (value contained in textbox3) found in any reader

If found go to its state

//

```
timeout1 = true;
if (!port.IsOpen) { port.Open(); } port.Write("h");
if (!port.IsOpen) { port.Open(); } port.Write("i");
if (!port.IsOpen) { port.Open(); } port.Write("j");
if (!port.IsOpen) { port.Open(); } port.Write("l");
if (!port.IsOpen) { port.Open(); } port.Write("a");
if (!port.IsOpen) { port.Open(); } port.Write("e"); port.Close();
```

Sending appropriate signals to arduino program using the serial port..As example (h) signal means red light in the seconded road

//

```
while ((fstop == false)) {
    if (textBox4.Lines[0] != "0") {
    var stln1 = textBox4.Lines[0];
```



```

var minaa1 = (from minaaa1 in ctx.TagLogs where (minaaa1.TagID ==
stln1) select minaaa1).Count();
var touty1 = (from minaaa1 in ctx.TagLogs where (minaaa1.TagID == stln1)
select minaaa1);

```

Checking again the stolen car and the important vehicle

```

if (minaa1 >= 1)
    {
    if (FST == false)
        {
        MessageBox.Show("stolen car found ");
        dataGridView1.DataSource = touty1;
        MessageBox.Show("HORAAAI ");
        FST = true;}}
if (textBox3.Lines[0] != "0")
    {
    var papawahba = from papa in ctx.TagLogs where papa.ReaderIP ==
"192.168.10.12" select papa.TagID;
    foreach (var elem in papawahba)
        { if (elem.ToString() == textBox3.Lines[0]) goto stateII; }
    var papawahba2 = from papa2 in ctx.TagLogs where papa2.ReaderIP ==
"192.168.10.14" select papa2.TagID;
    foreach (var elem2 in papawahba2)
        { if (elem2.ToString() == textBox3.Lines[0]) goto stateIII; }
    var papawahba3 = from papa3 in ctx.TagLogs where papa3.ReaderIP ==
"192.168.10.16" select papa3.TagID;
    foreach (var elem3 in papawahba3)
        { if (elem3.ToString() == textBox3.Lines[0]) goto stateIII; }
}

```

FI = true;////////// turning the flag of the first state to true.

```

var result = (from t in ctx.TagLogs where (((t.OutFieldTime.Second -
t.InFieldTime.Second) > Math.Abs(condet)) && (t.ReaderIP == "192.168.10.15")) select
t).Count();

```

Checking if time out or the main road congested (if there is any car remains in the main road time more than the congestion time)

```

if ((timeout1 == false) || (result >= 1))
{ DateTime currentTime = DateTime.Now;
  DateTime timeBefore = currentTime.AddSeconds(-2);
  /// comparing II III IIII
  var cmp2 = (from t2 in ctx.TagLogs where ((t2.ReaderIP ==
"192.168.10.12") && ((timeBefore <= (t2.OutFieldTime)) && ((t2.OutFieldTime) <=
currentTime))) select t2).Count();
  var cmp3 = (from t3 in ctx.TagLogs where ((t3.ReaderIP ==
"192.168.10.14") && ((timeBefore <= (t3.OutFieldTime)) && ((t3.OutFieldTime) <=
currentTime))) select t3).Count();
  var cmp4 = (from t4 in ctx.TagLogs where ((t4.ReaderIP ==
"192.168.10.16") && ((timeBefore <= (t4.OutFieldTime)) && ((t4.OutFieldTime) <=
currentTime))) select t4).Count();
}

```

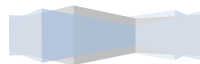
Getting all number of tags in each reader to know which one contains greatest number of cars

//////////if road 2 has the greatest number of tags

```

if ((cmp2 >= cmp3) && (cmp2 >= cmp4))//II state

```



```

{
if (FI == true && FII == true && FIII == true && FIIII == true)
{
    FI = FII = FIII = FIIII = false; int milliseconds = 5000;
    if (!port.IsOpen) { port.Open(); } port.Write("g");
    if (!port.IsOpen) { port.Open(); } port.Write("h");
    if (!port.IsOpen) { port.Open(); } port.Write("i");
    if (!port.IsOpen) { port.Open(); } port.Write("j"); ;
    if (!port.IsOpen) { port.Open(); } port.Write("e");
    if (!port.IsOpen) { port.Open(); } port.Write("f");
port.Close();
        Thread.Sleep(milliseconds);//makes the program
            // sleeps for 5 seconds
        goto stateII;}

```

If the 4 flags true which means the cycle has completed, a new cycle should start, so turn all flags to false and go to the sub state in which we must evacuate the square by turning the lights of the square green and the other lights red, then wait 5 seconds

////////////////////////////////////

```

else {
if (FII == true) {
if (cmp3 > cmp4)//stateIII {
if (FIII == true) {
int                milliseconds                =
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}

```

If we find that state 2 has already opened before and the cycle didn't finish yet,
 And we found that the next congested road is road3
 We first check its flag..if it true we go to state 4

////////////////////////////////////

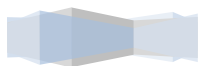
```

else
{
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}

```

Else if flag 3 is false we go to state 3
 But after passing but evacuating the square mini state

////////////////////////////////////



```

else{
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII} }
else
{///wait 5 sec
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII;
}}}

```

Returning to condition when we found that state 2 has already opened before and the cycle didn't finish yet, But we found that the next congested road is road4 We go to state 4 But after passing but evacuating the square mini state

Returning to condition when we found that not all flags are true which means that cycle not completed yet ,But we found that road 2 flag false. We go to state 2 But after passing but evacuating the square ministate

```

////////////////////////////////////
////

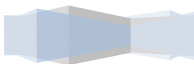
```

```

else if ((cmp3 >= cmp2) && (cmp3 >= cmp4)) {
if (FI == true && FII == true && FIII == true && FIIII == true) {
FI = FII = FIII = FIIII = false; int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}
else {
if (FIII == true) {
if (cmp2 > cmp4)//stateII
{
if (FII == true) {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();

```

The same as above, but on condition that road 3 is the most congested one



```

if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}
else {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII;}}
else {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;// stateIII }}
else
{///wait 5 sec
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}
////////////////////////////////////
else if ((cmp4 >= cmp3) && (cmp4 >= cmp2))//stateIII
{ if (FI == true && FII == true && FIII == true && FIIV == true) {
FI = FII = FIII = FIIV = false; int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();

```

The same as above, but on condition that road 4 is the most congested one.



```

Thread.Sleep(milliseconds);
goto stateIII; }
else {
if (FIII == true) {
if (cmp2 > cmp3)//stateII
{
if (FII == true) {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}
else {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII;}}
else{ int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}
else
{///wait 5 sec
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}}}

```



```

else{
var result1 = (from t5 in ctx.TagLogs where (((t5.OutFieldTime.Second -
t5.InFieldTime.Second) > Math.Abs(condet)) && (t5.ReaderIP == "192.168.10.17"))
select t5).Count();
if (result1 >= 1) {
if (!port.IsOpen) { port.Open(); } port.Write("k");// port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("c"); port.Close();
}}
}

```

```

.....
public void OnTimer(Object source, ElapsedEventArgs e)
{
timeout1 = false; }

```

Timer function indicates the timeout, onctimeout happens; it turns timeout flag to false.

```

.....
private void button2_Click(object sender, EventArgs e)
{
fstop = true;
timerClock.Enabled = false;
DateTime currentTime1 = DateTime.Now;
GP_TrafficListenerEntities ctx = new GP_TrafficListenerEntities();
var query = from p in ctx.TagLogs where p.InFieldTime<=currentTime1 select p;
foreach (var x in query)
ctx.DeleteObject(x);
ctx.SaveChanges();
}}

```

Reset Button, its function to delete every entry in the database



5.3 Simulation

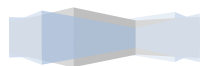


5.3.1 Introduction

Aimsun is traffic modeling software that allows you to model anything from a single bus lane to an entire region; Aimsun stands out for the exceptionally high speed of its simulations and for fusing travel demand modeling, static and dynamic traffic assignment with mesoscopic, microscopic and hybrid simulation – all within a single software application.

Aimsun allows you to carry out traffic operations assessments of any scale and complexity. The applications are endless but some of the most common are:

1. Assessment and optimization of Transit Signal Priority (TSP) and Bus Rapid Transit (BRT) schemes.
2. Feasibility studies for High Occupancy Vehicle (HOV) and High Occupancy Toll (HOT) lanes.
3. Impact analysis of infrastructure design such as highway corridor improvement/construction.
4. Environmental impact analysis.
5. Toll and road pricing.
6. Evaluation of travel demand management (TDM) strategies.
7. Signal control plan optimization (Aimsun-TRANSYT link) and adaptive control evaluation.
8. Safety analysis.
9. Evaluation of Variable Speed policies and other Intelligent Transportation Systems (ITS).
10. Highway Capacity Manual (HCM) analysis.
11. Work zone management.



5.3.2 Top features

- **Travel demand modeling brings true integration**

Four-step demand modeling is an important functionality for Aimsun software and another step towards true integration. You can now start a transportation modelling project from scratch, entering raw geographical and socio-economic data and follow the process all the way through to simulation without recourse to a separate package.

With Aimsun you no longer have to interface with an external traffic demand model for importing data as a starting point or as a means of redistributing demand from one mode to another or to estimate the impact that capacity changes have on overall demand levels.

Being able to use a single software application for an entire project means vastly increased cost-efficiency, consistency and quality for Aimsun users.

- **Speed**

One of Aimsun's most outstanding features is its speed: our microscopic simulator is the fastest on the market by far and the finest microsimulation tool available in the world for large-scale projects.

Aimsun's multithreaded software architecture speeds up the simulation process to such an extent that it completely redefines what can be included in a dynamic model, be that a major city or a large and complex highway network.

Even on a laptop, the Aimsun microsimulator can run a model of the whole of Singapore with 10,580 intersections and 4,483 km of lanes 2-3 times faster than real time.

- **Openness**

Aimsun's modular architecture and programming capabilities make it open to developing interfaces with a broad range of third party tools. Anything is possible, from traffic signal optimization with the TRANSYT-Aimsun Link to emissions modeling with VERSIT+micro or even providing realistic virtual traffic conditions for the Oktal SCANer driving simulation engine.

Aimsun fits right in with your existing software and can exchange data with the most popular CAD, GIS, transport modelling, signal optimization and adaptive control software tools.

- **OpenStreetMap Importer**

Another feature generating great interest is the OpenStreetMap Importer. This gives you amazing independence, allowing you to start a new project by importing the geometry of any place in the world in a matter of seconds without having to rely on anyone to provide the right material, or depending on another model's network. It's just you and your internet connection.



- **3ds Max® integration**

FZP Exporter exports Aimsun simulations to Autodesk® 3ds Max® for high-quality three dimensional presentations. With this technology users can immerse themselves in a very rich virtual environment with high-fidelity terrain and structures. The user has the ability to attach to or detach from a vehicle and fly around the database, zoom in and zoom out, freeze the simulation, and resume.

5.3.3 Working steps

First of all we settled on a certain algorithm as shown in chapter after that we simulated our algorithm by using Aimsun to determine its efficiency.

The following session I am going to build a traffic control system on a simple square by using Aimsun simulator to show our algorithm.

- **Step 1**

The first step is sketching the entrance and exit streets of the square with required number of lanes.

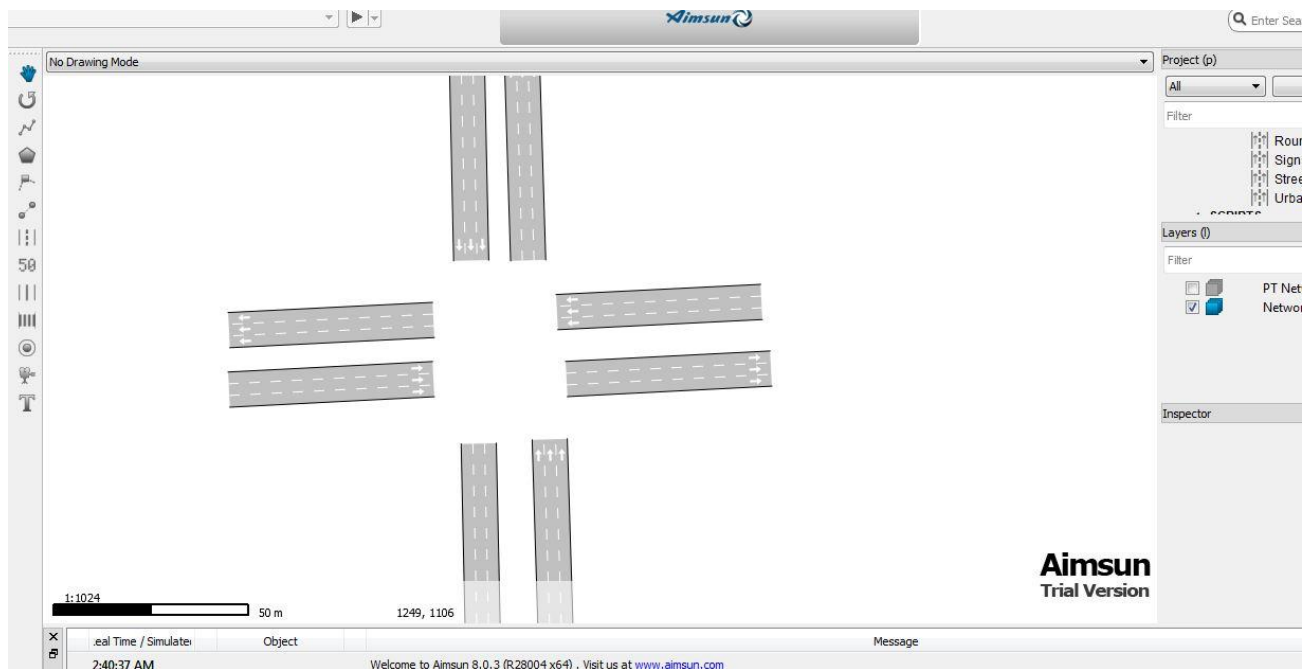


Figure 5.11 step 1 in simulation



- **Step 2**

The second step is creating a node. I.e. A node is a point or an area in the network where vehicles change their direction and/or disperse. Hence, a node has one or more origin sections and one or more destination sections.

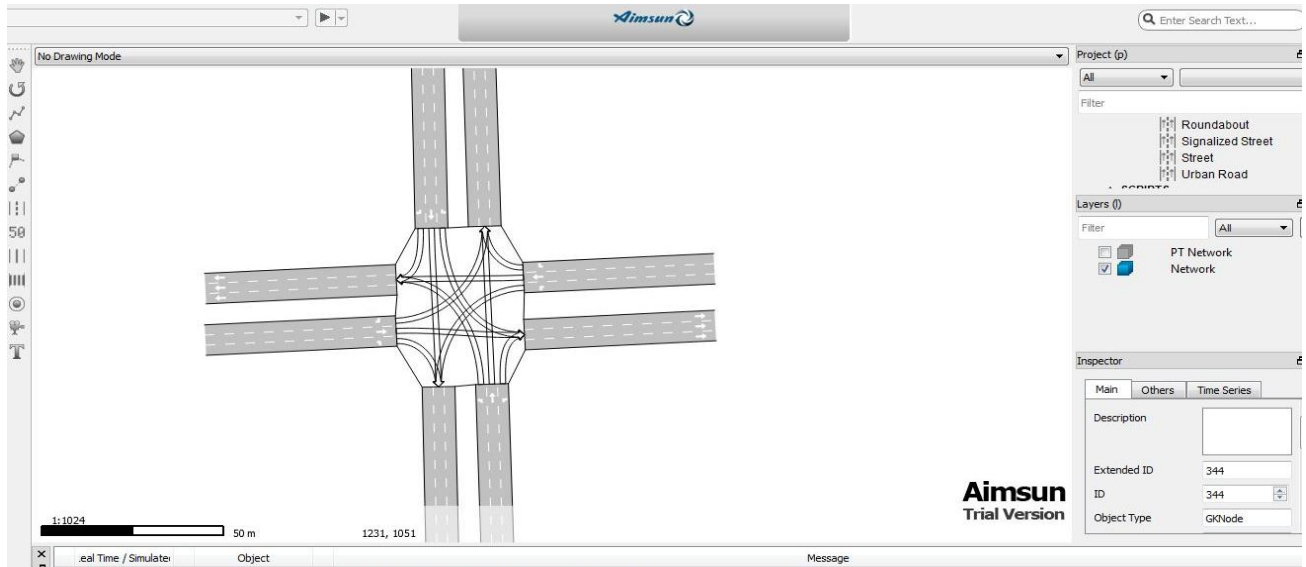


Figure 5.12 step 2 in simulation

- **Step 3**

The third step is creating a signal group. I.e. A signal group is the set of turn movements given right of way when a given traffic light turns green.

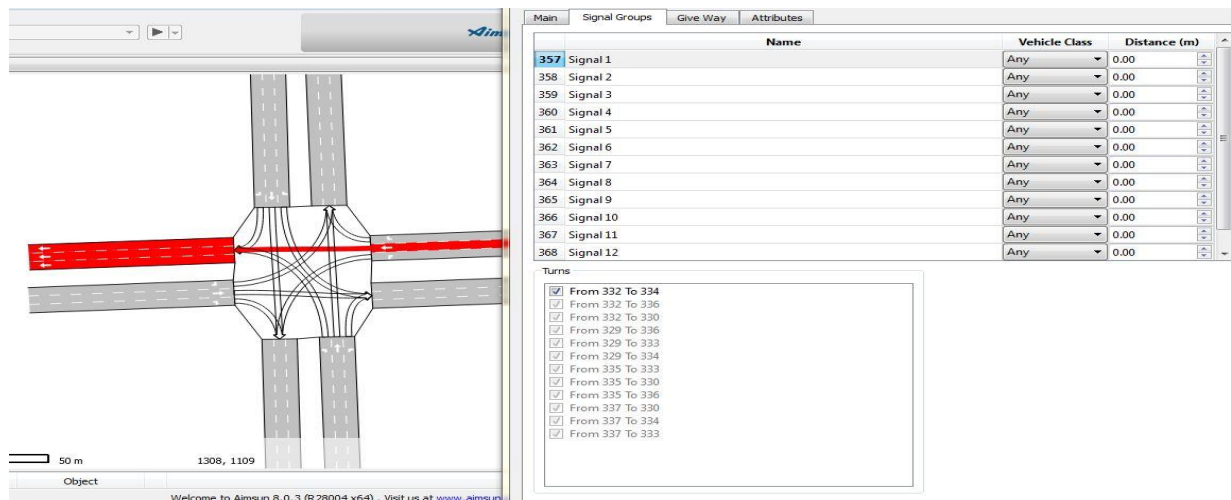


Figure 5.13 signal 1 in step 3 in simulation



signal 2

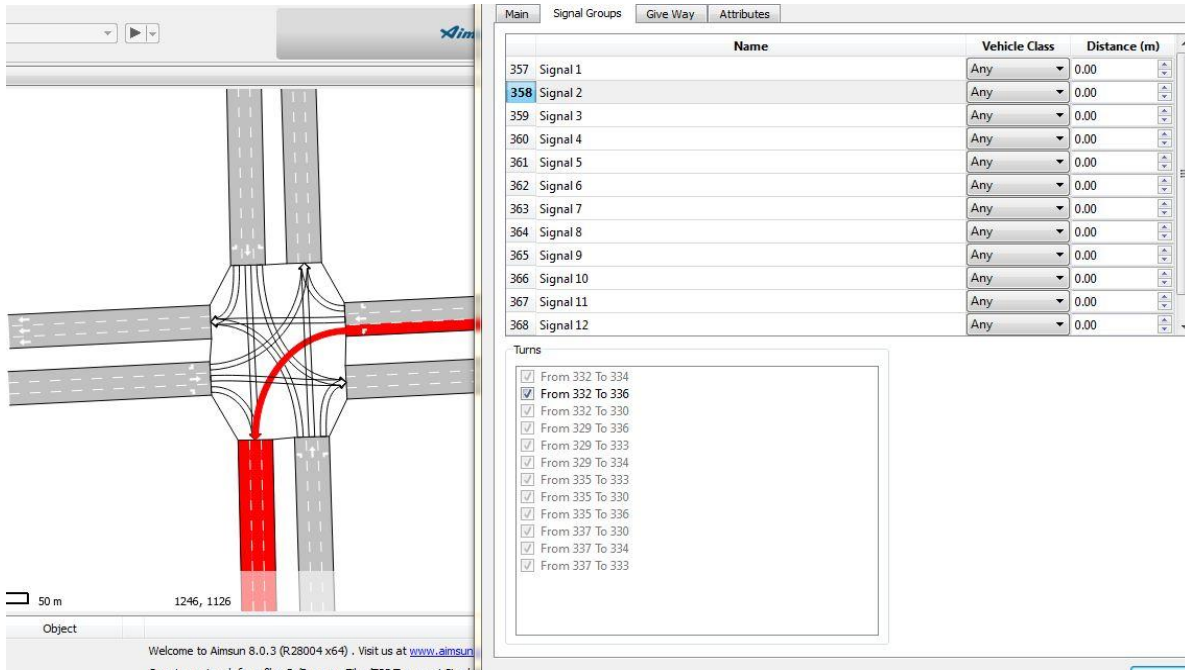


Figure 5.14 signal 2 in step 3 in simulation

- **Step 4**

The fourth step is adding centroids. I.e. A centroid is a source and/or a sink of vehicles. They are used to define O/D matrices, as origin and destination points of the trips.

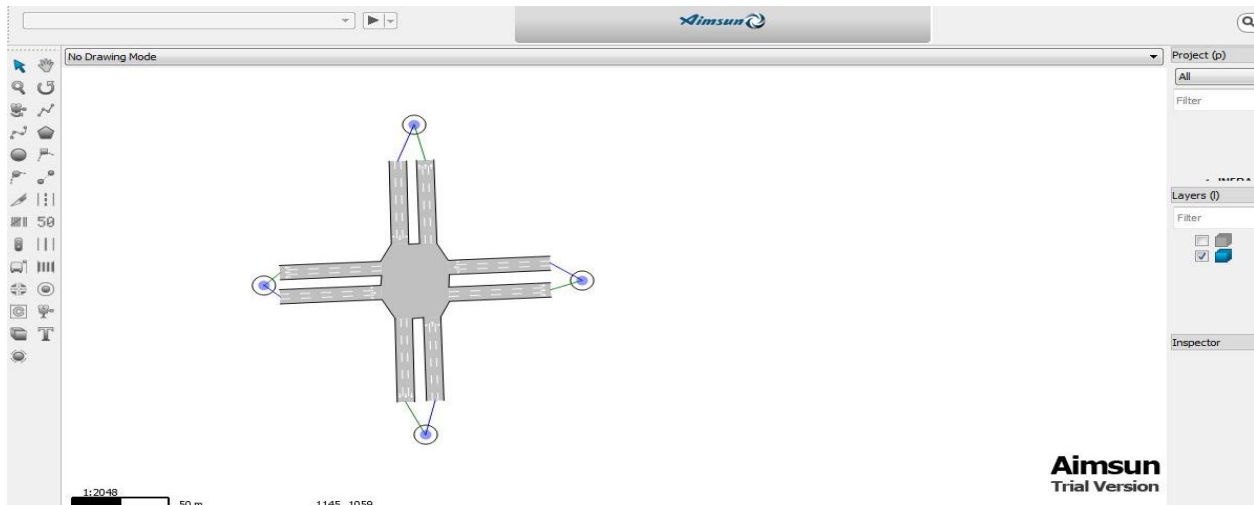


Figure 5.15 step 4 in simulation

- **Step 5**

The fifth step is adding detectors. I.e. Many types of detectors exist: pressure, magnetic, loop, video, and others, but all of them are characterized by their measuring capabilities. Measuring capabilities include vehicle count, presence, speed, occupancy, density, headway and, as a special one, whether the detected vehicle is an equipped one. The detector will always distinguish between different vehicle types.

A detector can be positioned at any point in a section, and its width can extend to more than one lane. A section can have no, one, or multiple detectors. The only restriction is that a detector cannot be shared by more than one section.

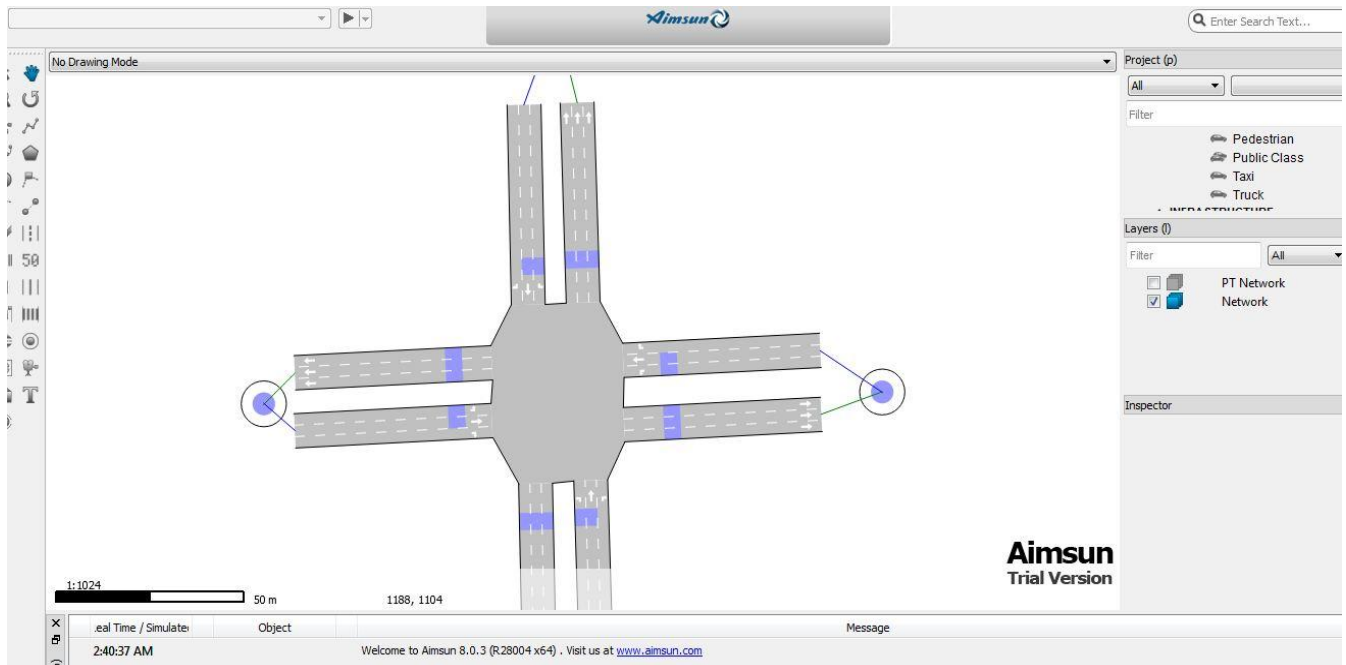
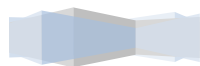


Figure 5.16 step 5 in simulation

The blue labels on each street in the above figure are the detectors.



- **Step 6**

The sixth step is adding control plans and it is considered the most important step where control plan controls the traffic light and it's duration at each node. The main parameters for a junction are:

- Control type: uncontrolled, fixed control, external or actuated.
- Critical junction flag: a critical junction could be treated differently in an external model.
- Offset: used to set the beginning of the sequence of phases regarding to the control plan's initial time, and therefore synchronize adjacent junctions within the same control plan.
- Phases: determine the different time periods of signal activation.

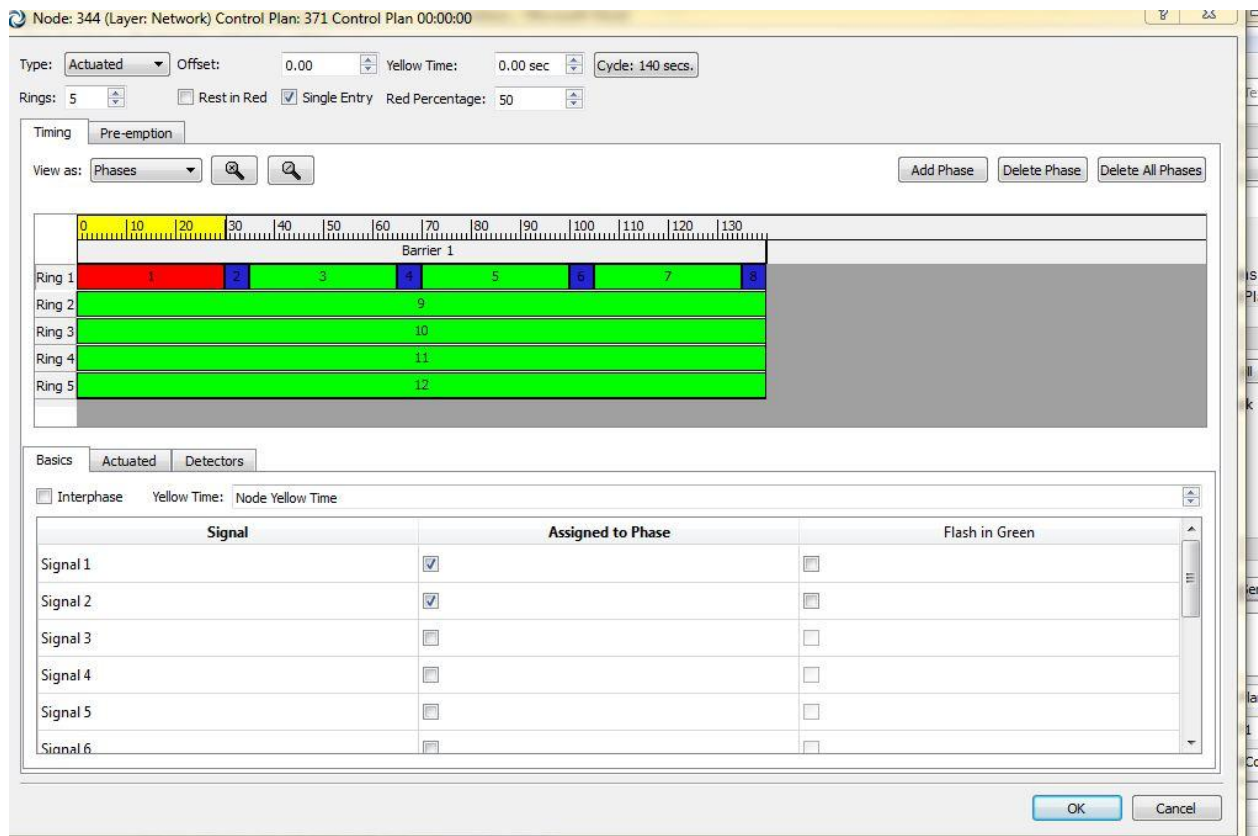


Figure 5.17 step 6 in simulation

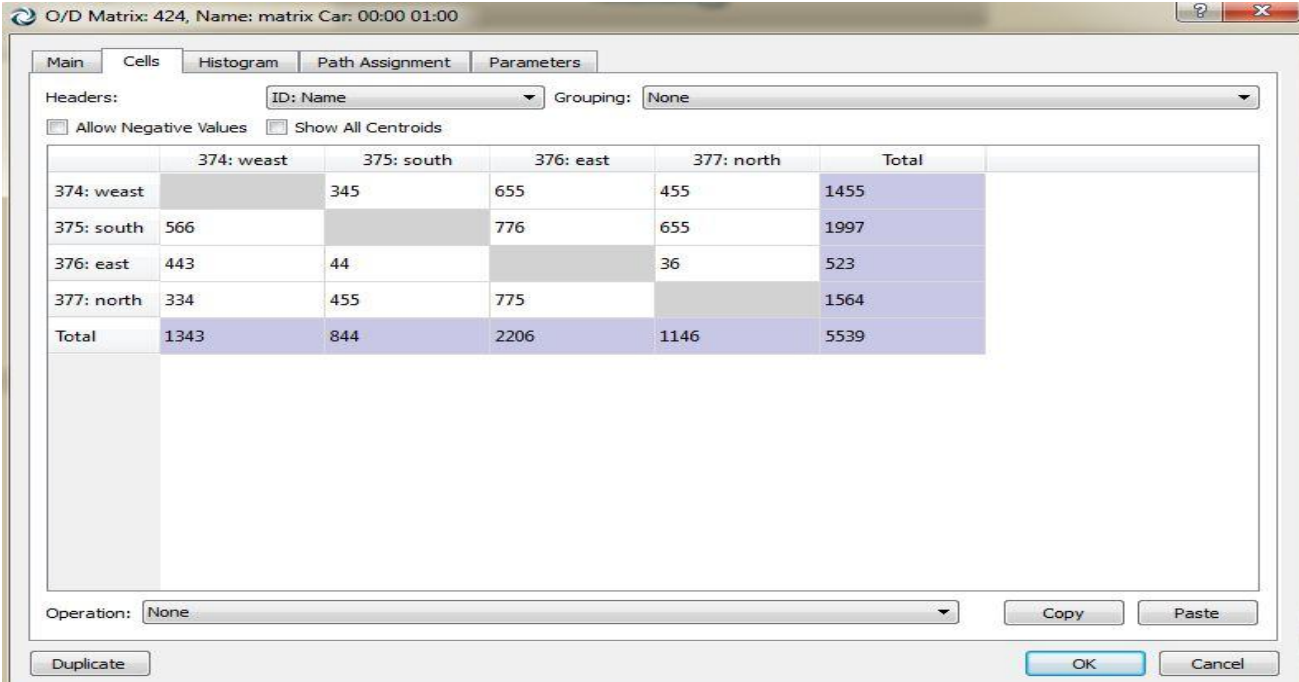


Important notes from the above figure

- Type is actuated which means each phase must has one detector or more and this phase is executed when this detectors detect existing cars according to detectors parameters.
- Each ring has cycle time at which it is divided into phases.
- Phase consists of one or more signals that executed simultaneously.
- Number of rings means that the phases at the same time interval executes with each other.
- The blue time interval is a gap time between each two phases.

• Step 7

The seventh step is adapting the O/D matrix.I.e. The function of O/D matrix is determining the number of cars that is generated at each centroid and to which destination centroid is and determines the type of vehicles (car, trunk, bus, taxi).

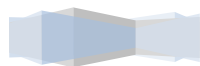


The screenshot shows a software window titled "O/D Matrix: 424, Name: matrix Car: 00:00 01:00". The window has several tabs: "Main", "Cells", "Histogram", "Path Assignment", and "Parameters". The "Main" tab is active. Below the tabs, there are controls for "Headers:" (ID: Name), "Grouping:" (None), and checkboxes for "Allow Negative Values" and "Show All Centroids". The main area contains a table with the following data:

	374: weast	375: south	376: east	377: north	Total
374: weast		345	655	455	1455
375: south	566		776	655	1997
376: east	443	44		36	523
377: north	334	455	775		1564
Total	1343	844	2206	1146	5539

At the bottom of the window, there is an "Operation:" dropdown menu set to "None", and buttons for "Copy", "Paste", "Duplicate", "OK", and "Cancel".

Figure 5.18 step 7 in simulation



- **Step 8**

The eighth step is creating a scenario to simulate our algorithm which is considered the last step.

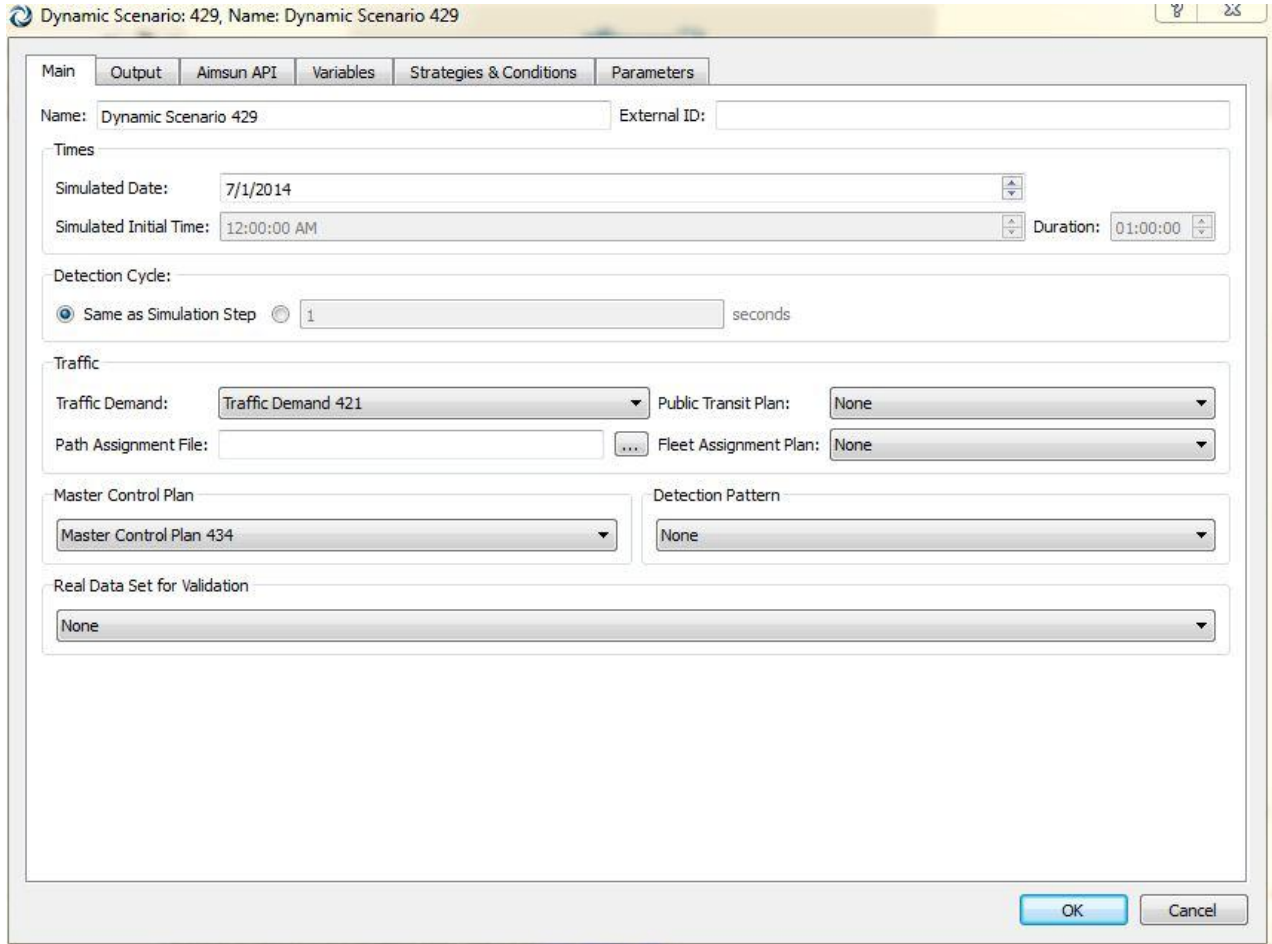
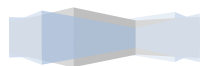


Figure 5.19 step 8 of simulation

Some important notes

- We should determine which master control plan in our scenario.
- We also should determine which traffic demand is used.



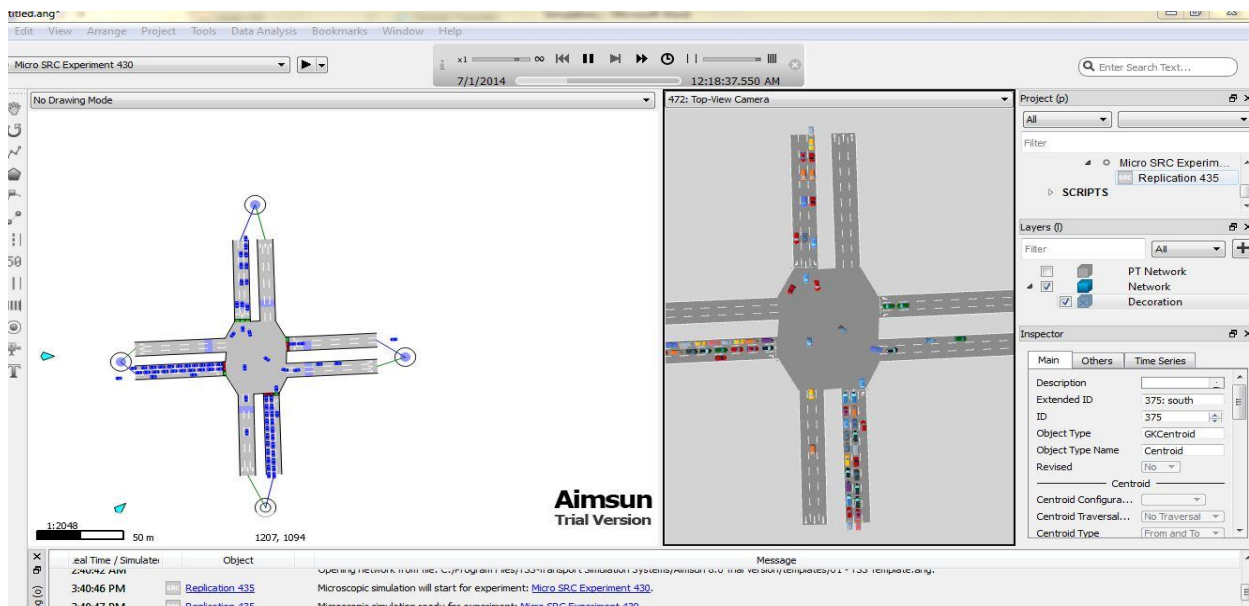
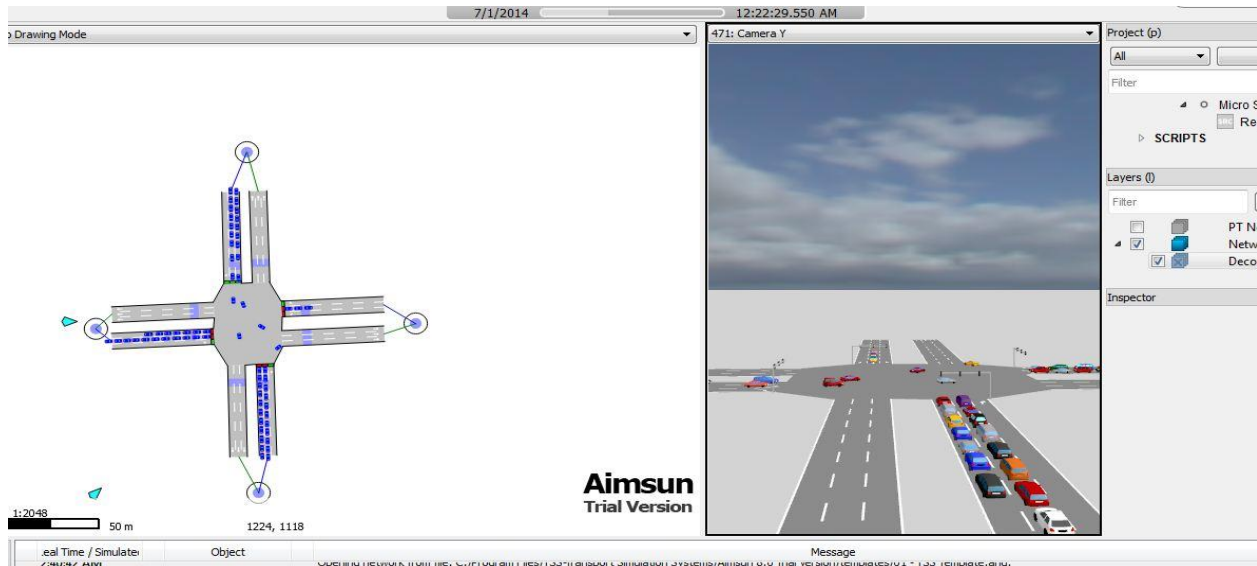
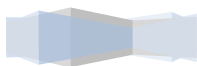


Figure 5.20 some of simulation shots



5.3.4 Challenges

1. We worked on trial version as the costs of licensed versions are very expensive with minimum cost 1,200 euro and maximum cost 4,200 euro according to which version you are going to purchase.
2. The following restrictions are only applicable to the free trial version of the software (the full versions do not have these limitations):
 - Free trial license expires 30 days after installation.
 - Save is disabled.
 - Scripting is not available.
 - Storing experiment data in an external database is disabled.

Save disabling is the most consequence we have faced as after working for hours in simulation steps, the simulator can suddenly stop running, based on that we start from scratch which means wasting a lot of time.

3. Self-learning as few of people around the world can work on such these simulators not like Matlab or C++ for example.
4. Poor sources of references.



5.4 Maquette:

It Consists of the Layout of Our Site settled on a wooden Box, we will talk about both of them ; the Layout and the Box.

A) Layout:

We made it with the help of an architect; he drew it using auto-cad and edited it using PhotoShop CS5. Then we have printed it on Glossy paper and then we used double face tape to stick the two surfaces together.

The design shown in Figure 5.21

Specifications:

1-1.20 x 1.20 m^2

2-it consits of 8 streets

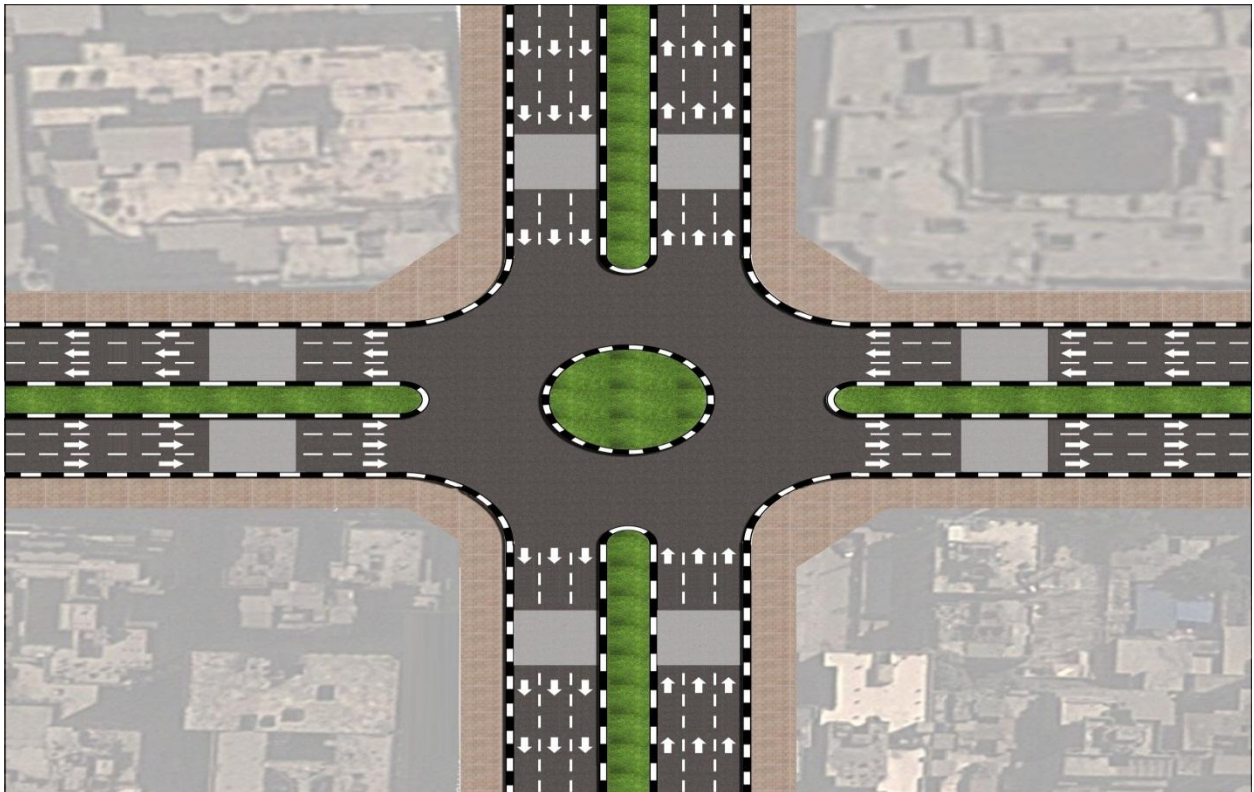


Figure 5.21 layout of the maquette



B) The wooden box

We made it because we need to carry 8 adapters for 8 readers and 2 switches and the fig5.22 , show us what the box carry

We made it with the help of a carpenter and we can open the box from two sides

Specifications:

1-We use counter wood

2-1.20 x 1.20 m^2



Figure 5.22 the wooden Box

We use Foam in Four spots to Have more safety to reader and carry it up to be a part of the street and that show in figure 5.23

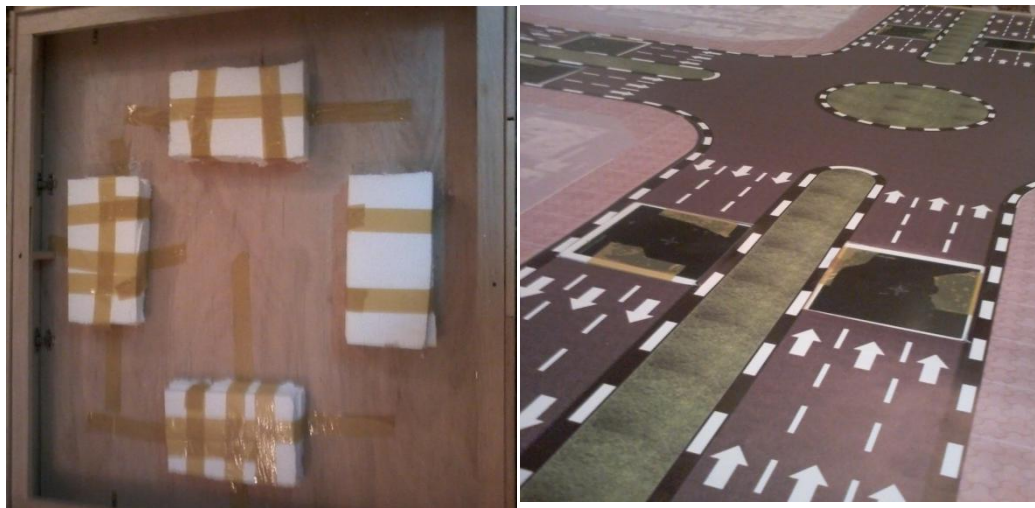


Figure 5.23 foam in the box

And then the final Maquette shown in figure 5.25

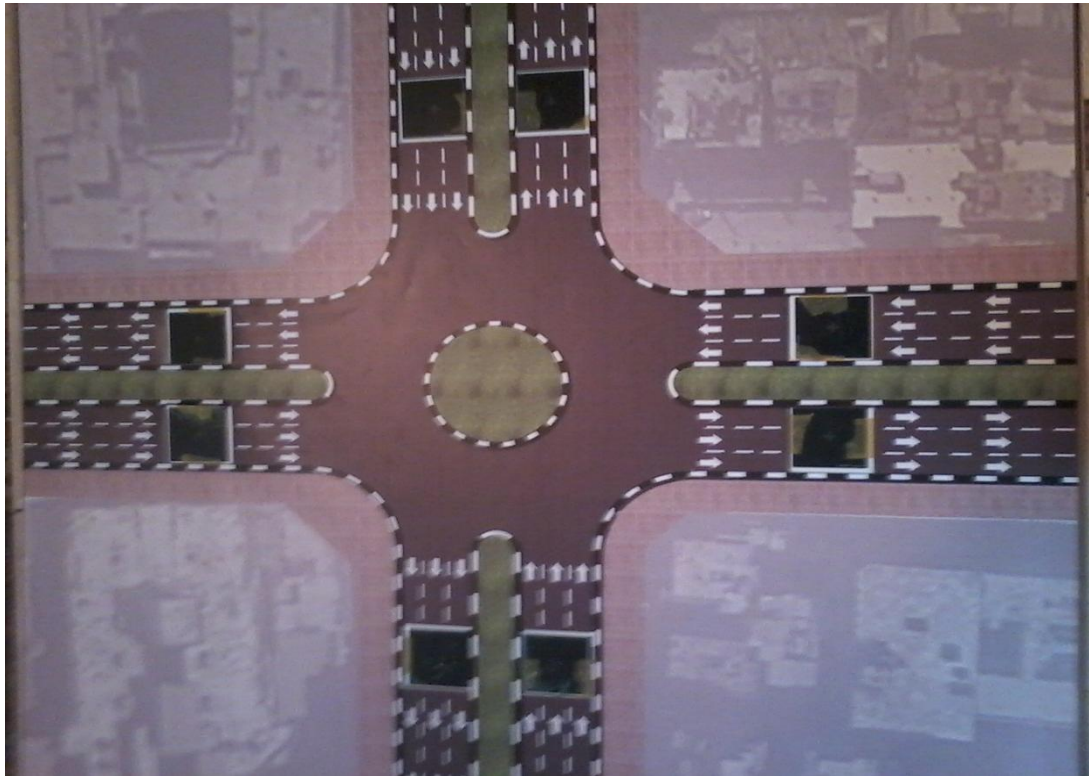


Figure 5.25 the final maquette



5.4 Problems

There are many problems during working in this project :

- First of all, the limitation of our budget was an important problem because the overall system of RFID is very expensive and also we need 8 readers to implement our project.
- We decided to implement our final project as small scale on a maquette , and we actually designed one but it was incompatible to our project.

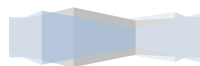


Figure 5.26 The old maquette

- So, we implemented another one to be able to handle all readers, switches, cables...etc.



Figure 5.27 The new maquette



5.6 Deals

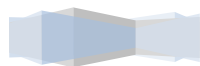
Facing to our budget limit, we were hard-pressed to find a reader chip that includes the anti-collisions feature we needed to implement our system as we envisioned.

- Firstly, OG-TECH Company which is working on The RFID System at which the cost of readers and tags are very expensive, so the overall RFID system will be very expensive. However, the components found at OG-TECH are very compatible with the desired specifications of our project as the anti-collision property of the reader and the compatible tag with the reader.
- Searching on the INTERNET for a reader compatible with our project in many sites as “Alibaba “ and “ Amazon” , but not found.
- Next, at “future ” and “RAM” (Electronic companies) the reader does not satisfy the desired requirements as it doesn’t have the property of anti-collision and other important properties.

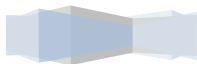


Figure5.28 The reader from RAM

- DR/ Ibrahim Amar (professor in the department of our college) informed us about 2 engineers who have RFID companies. But the readers were very expensive and also didn’t have readers compatible with tags in the low frequency.



- Finally, doctor Hassan Mostafa made deal with OGTECH Company, The deal was:
 - Buying 2 readers.
 - Borrowing 2 readers but with paying insurance on them.
- But we cannot forget the help and support from OG-TECH company which allowed us to borrow their equipment (the other 4 readers) and returned them back at the end of the project.



Chapter 6: CONCLUSION AND FUTURE WORK

The RFID technology may lead to a revolution in traffic management, when it is properly deployed as an intelligent system with suitable algorithm. This technique may revolutionize the management of traffic systems in Egypt, if it is implemented on a large scale commensurate with a comprehensive plan for an automated transport system.

The dynamic management scheme operates in real time and emulates the judgment of a traffic policeman on duty. The efficiency of the system may save many man-hours usually lost in traffic problems.

Accidents may also be prevented and lives can be saved as well as property. Priority emergency tags can be deployed on ambulance, fire, police and other emergency vehicles.

Also the ability of tracking cars for example a stolen car makes the RFID use very effective and important.

The system saves valuable details in the records of the database, which can provides ample and valuable information to planners and investigators. . However, the integration of the databases among the local authorities is a challenge that requires decisions at national level. Data sharing and secure hierarchical access to various levels of databases and protocols must be designed to integrate new information with existing systems. The issues of integration and collaboration may be a subject for future work.

In another direction:

The frequency range can enhance the reading range for RFID tag reading capability. If we used a higher frequency (UHF) then we can use ACTIVE tags in cars, so we can implement our system in the real life using ACTIVE RFID as an evolution to our system. By improving this capability, the development of RFID usage in Egypt will rapidly grow. This advancement may to the invention of more RFID systems and applications, as the cost of the technology gets lower with the expected mass production.

One of its main features is the ability to communicate operation commands from headquarters or any other subsidiary command station to any location in the system via existing infrastructure such as GSM or Internet. This system can enhance the transportation system of the country, by efficient management.

In the previous system, the **ACTIVE RFID** technology was adopted to automatically detect and manage road traffic congestion in near real-time since it is cost effective, easily manageable and reliable. Moreover, this method provides comprehensive way of congestion detection and management as a whole. The simulation results give us a fair idea of the extent of congestion occurred at a particular crossing and how the proposed schemes of congestion management reduce the waiting time significantly.

There are further scope of enhancement and extension of this idea to improve congestion management process ensuring smooth traffic flow in city in busy hours.

There, the **GSM network was used to exchange SMS among the coordinators**, which is not fully reliable. Loss of message or delay may hamper the correct detection of congestion and real time delivery of message. It is better to **design an indigenous network solely dedicated for this process**.

The router and the coordinator needs to be solar powered and weather-proof as they will be kept in outdoor environment and relies on battery power.

Therefore, we can work towards integration of solar cells as supplementary source of energy which can further reduce the cost.

An additional feature can also be added to our system, is the ability of **AUTO-LEARNING**. By using a smarter algorithm, an amount of prediction can be achieved, for example; we can predict the rush hour or the busy hour during the day, we can predict the frequently congested sites and how to manage them in an efficient way and also we can determine more precisely the value of the time for opening the traffic.(time of no congestion)

Finally the RFID offers several benefits, a good many of which you can realize today to a substantial degree with the existing products. Other benefits are also somewhat realizable, and it is expected that improvements in the technology will steadily bring these benefits to a mature level. Undeniably, however, such a collection of unique RFID benefits makes is already a potential enabler of a wide variety of applications. RFID will likely be the preferred technology in the coming days. RFID technology is currently undergoing rapid changes that are expected to provide continuously improving products, steadily bringing the true potential of the technology to the user.



References

[1] <http://www.rfidjournal.com/article/view/1339/>

[2] http://www.aimglobal.org/technologies/RFID/what_is_rfid.asp

[3] [Zhi Chang, Zhangeng Sun & Junbao Gu, "Study on the Technology of the Coal Mining Safety Monitoring System". Modern Applied Science. Vol.3,No.8,\(August 2009\).](#)

[4] ["ASYNCHRONOUS SERIAL COMMUNICATION OVERVIEW." Retrieved 5 March 2011, 2011, from http://www.quatech.com/support/comm-overasyncserial.php.](#)

[5] [Cytron, T. \(2007\). "PIR \(Passive Infra-Red\) Sensor User Manual." Retrieved July 25, 2010, from http://www.cytron.com.my.](#)

[6] <http://www.rfidjournal.com/article/view/1334/>

[7] <http://www.slais.ubc.ca/courses/libr500/04-05->

[wt2/www/A_Farrell/HowDoesRFIDWork.htm](#)

[8] [EPCGRP - Electronic Product Code Group](#)

http://www.epcgrp.com/docs/Demo_v2/index.html

[Last visit: 2006-02-17](#)

[9] [Boutin P: We know what you're buying, MSNBC Interactive 2003](#)

<http://msnbc.msn.com/id/3158851/>

[10] [Krakow, G: Credit on your key ring, MSNBC Interactive 2006](#)

<http://www.msnbc.msn.com/id/3072638/>



[Last visit: 2006-04-07](#)

[\[11\]Comptia.RFID.plus.Certification.Passport.May.2007](#)

[\[12\]http://www.ops.fhwa.dot.gov/aboutus/opstory.htm](http://www.ops.fhwa.dot.gov/aboutus/opstory.htm)

[\[13\]http://traveltips.usatoday.com/effects-traffic-congestion-61043.html](http://traveltips.usatoday.com/effects-traffic-congestion-61043.html)

[\[14\]http://ibtta.org/sites/default/files/The%20Public%20Health%20Costs%20of%20Traffic%20Congestion.pdf](http://ibtta.org/sites/default/files/The%20Public%20Health%20Costs%20of%20Traffic%20Congestion.pdf)

[\[15\]http://en.wikipedia.org/wiki/Traffic_congestion](http://en.wikipedia.org/wiki/Traffic_congestion)

[\[16\]http://www.siemens.com/sustainability/en/environmental-portfolio/products-solutions/mobility/intelligent-traffic-management.htm](http://www.siemens.com/sustainability/en/environmental-portfolio/products-solutions/mobility/intelligent-traffic-management.htm)

[\[17\]https://facultylive.iimcal.ac.in/sites/facultylive.iimcal.ac.in/files/9-real-time.PDF](https://facultylive.iimcal.ac.in/sites/facultylive.iimcal.ac.in/files/9-real-time.PDF)

[\[18\]http://www.ijirset.com/upload/december/6-SMART%20TRAFFIC%20CONTROL.pdf](http://www.ijirset.com/upload/december/6-SMART%20TRAFFIC%20CONTROL.pdf)

[\[19\]http://www.ops.fhwa.dot.gov/aboutus/opstory.htm](http://www.ops.fhwa.dot.gov/aboutus/opstory.htm)

[\[20\]http://en.wikipedia.org/wiki/Traffic_congestion](http://en.wikipedia.org/wiki/Traffic_congestion)

[\[21\]http://www.umtri.umich.edu/our-focus/advanced-traffic-management-systems](http://www.umtri.umich.edu/our-focus/advanced-traffic-management-systems)

[\[22\]http://my.safaribooksonline.com/book/hardware/rfid/0131851373/advantages-of-the-technology/ch02lev1sec2](http://my.safaribooksonline.com/book/hardware/rfid/0131851373/advantages-of-the-technology/ch02lev1sec2)



Appendix

The Code

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Threading;
using System.Timers;
using System.IO.Ports;
using System.IO;
namespace omar
{
public partial class Form1 : Form
{
GP_TrafficListenerEntities ctx = new GP_TrafficListenerEntities();
public Form1()
{
InitializeComponent();
MessageBox.Show("please enter time out & time to detect congestion before you press start");
MessageBox.Show("please enter ID in case of stolen car or important vehicle..if there is not please enter 0"); }
SerialPort port = new SerialPort("COM4", 9600);
bool timeout1;
bool fstop;
System.Timers.Timer timerClock = new System.Timers.Timer();
private void button1_Click(object sender, EventArgs e)    {
fstop = false;
////////////////////////////////////
int condet = Convert.ToInt32(textBox2.Lines[0]);
int w2t_elashara = Convert.ToInt32(textBox1.Lines[0]);
int w2t_elashara_2 = w2t_elashara * 1000;
timerClock.Elapsed += new ElapsedEventHandler(OnTimer);
timerClock.Interval = w2t_elashara_2;
timerClock.Enabled = true;
////////////////////////////////////
bool FI, FII, FIII, FIIII;
bool FST;
FST = false;
FI = FII = FIII = FIIII = false;
////////////////////////////////////
stateI:
DateTime currentTime1 = DateTime.Now;
```



```

DateTime timeBefore1 = currentTime1.AddSeconds(-timerClock.Interval);
var query = from p in ctx.TagLogs where p.InFieldTime <= currentTime1 select p;
foreach (var x in query)
ctx.DeleteObject(x);
ctx.SaveChanges();
if (textBox4.Lines[0] != "0")      {
var stln = textBox4.Lines[0];
var minaaa = (from minaaa in ctx.TagLogs where (minaaa.TagID == stln) select
minaaa).Count();
var touty =(from minaaa in ctx.TagLogs where (minaaa.TagID == stln) select minaaa);
if (minaaa >= 1)      {
if (FST == false)      {
MessageBox.Show("stolen car found ");
dataGridView1.DataSource = touty;
MessageBox.Show("HORAAAI ");
FST = true;
}}
if (textBox3.Lines[0] != "0")      {
var papawahba = from papa in ctx.TagLogs where papa.ReaderIP == "192.168.10.12"
select papa.TagID ;
foreach (var elem in papawahba)
{ if (elem.ToString() == textBox3.Lines[0]) goto stateII; }
var papawahba2 = from papa2 in ctx.TagLogs where papa2.ReaderIP ==
"192.168.10.14" select papa2.TagID;
foreach (var elem2 in papawahba2)
{ if (elem2.ToString() == textBox3.Lines[0]) goto stateIII; }
var papawahba3 = from papa3 in ctx.TagLogs where papa3.ReaderIP ==
"192.168.10.16" select papa3.TagID;
foreach (var elem3 in papawahba3)
{ if (elem3.ToString() == textBox3.Lines[0]) goto stateIII; } }
timeout1 = true;
if (!port.IsOpen) { port.Open(); } port.Write("h");// port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("l"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("a"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); port.Close();
while ((fstop == false))      {
if (textBox4.Lines[0] != "0")      {
var stln1 = textBox4.Lines[0];
var minaaa1 = (from minaaa1 in ctx.TagLogs where (minaaa1.TagID == stln1) select
minaaa1).Count();
var touty1 = (from minaaa1 in ctx.TagLogs where (minaaa1.TagID == stln1) select
minaaa1);
if (minaaa1 >= 1)      {
if (FST == false)      {

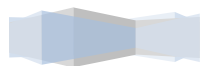
```



```

MessageBox.Show("stolen car found ");
dataGridView1.DataSource = touty1;
MessageBox.Show("HORAAAI ");
FST = true;}}
if (textBox3.Lines[0] != "0")      {
var papawahba = from papa in ctx.TagLogs where papa.ReaderIP == "192.168.10.12"
select papa.TagID;
foreach (var elem in papawahba)
{ if (elem.ToString() == textBox3.Lines[0]) goto stateII; }
var papawahba2 = from papa2 in ctx.TagLogs where papa2.ReaderIP ==
"192.168.10.14" select papa2.TagID;
foreach (var elem2 in papawahba2)
{ if (elem2.ToString() == textBox3.Lines[0]) goto stateIII; }
var papawahba3 = from papa3 in ctx.TagLogs where papa3.ReaderIP ==
"192.168.10.16" select papa3.TagID;
foreach (var elem3 in papawahba3)
{ if (elem3.ToString() == textBox3.Lines[0]) goto stateIII; } }
FI = true;
var result = (from t in ctx.TagLogs where (((t.OutFieldTime.Second -
t.InFieldTime.Second) > Math.Abs(condet)) && (t.ReaderIP == "192.168.10.15")) select
t).Count();
if ((timeout1 == false) || (result >= 1))      {
DateTime currentTime = DateTime.Now;
DateTime timeBefore = currentTime.AddSeconds(-2);
/// comparing II III III
var cmp2 = (from t2 in ctx.TagLogs where ((t2.ReaderIP == "192.168.10.12") &&
((timeBefore <= (t2.OutFieldTime)) && ((t2.OutFieldTime) <= currentTime))) select
t2).Count();
var cmp3 = (from t3 in ctx.TagLogs where ((t3.ReaderIP == "192.168.10.14") &&
((timeBefore <= (t3.OutFieldTime)) && ((t3.OutFieldTime) <= currentTime))) select
t3).Count();
var cmp4 = (from t4 in ctx.TagLogs where ((t4.ReaderIP == "192.168.10.16") &&
((timeBefore <= (t4.OutFieldTime)) && ((t4.OutFieldTime) <= currentTime))) select
t4).Count();
Console.WriteLine(FI);
Console.WriteLine(FII); Console.WriteLine(FIII); Console.WriteLine(FIIII);
if ((cmp2 >= cmp3) && (cmp2 >= cmp4))//II state
{
if (FI == true && FII == true && FIII == true && FIIII == true)      {
FI = FII = FIII = FIIII = false; int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
}
}
}

```




```

Thread.Sleep(milliseconds);
goto stateII;}
else {
if (FII == true)          {
if (cmp3 > cmp4)//stateIII
{
if (FIII == true)          {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}
else          {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}
else{
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII} }
else
{///wait 5 sec
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);

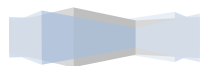
```




```

if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII;}}
else{ int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}
else
{///wait 5 sec
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}}}
else{
var result1 = (from t5 in ctx.TagLogs where (((t5.OutFieldTime.Second -
t5.InFieldTime.Second) > Math.Abs(condet)) && (t5.ReaderIP == "192.168.10.17"))
select t5).Count();
if (result1 >= 1)
{
if (!port.IsOpen) { port.Open(); } port.Write("k");// port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("c"); port.Close();
}}}
//////////////////////////////////////
stateII:
timeout1 = true;
DateTime currentTime2 = DateTime.Now;
DateTime timeBefore2 = currentTime2.AddSeconds(-timerClock.Interval);
var queryy = from p2 in ctx.TagLogs where p2.InFieldTime <= currentTime2 select p2;
foreach (var x in queryy)
ctx.DeleteObject(x);
ctx.SaveChanges();
if (textBox4.Lines[0] != "0"){
var stln2 = textBox4.Lines[0];
var minaa2 = (from minaaa2 in ctx.TagLogs where (minaaa2.TagID == stln2) select
minaaa2).Count();
var touty2 = (from minaaa2 in ctx.TagLogs where (minaaa2.TagID == stln2) select
minaaa2);

```



```

if (minaa2 >= 1) {
if (FST == false) {
MessageBox.Show("stolen car found ");
dataGridView1.DataSource = touty2;
FST = true;
}}}
if (textBox3.Lines[0] != "0") {
var papawahba = from papa in ctx.TagLogs where papa.ReaderIP == "192.168.10.12"
select papa.TagID;
foreach (var elem in papawahba)
{ if (elem.ToString() == textBox3.Lines[0]) goto stateII; }
var papawahba2 = from papa2 in ctx.TagLogs where papa2.ReaderIP ==
"192.168.10.14" select papa2.TagID;
foreach (var elem2 in papawahba2)
{ if (elem2.ToString() == textBox3.Lines[0]) goto stateIII; }
var papawahba3 = from papa3 in ctx.TagLogs where papa3.ReaderIP ==
"192.168.10.16" select papa3.TagID;
foreach (var elem3 in papawahba3)
{ if (elem3.ToString() == textBox3.Lines[0]) goto stateIII; }
}
if (!port.IsOpen) { port.Open(); } port.Write("g");// port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("k"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("b"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
while ((fstop == false)){
if (textBox4.Lines[0] != "0")
{ var stln3 = textBox4.Lines[0];
var minaa3 = (from minaaa3 in ctx.TagLogs where (minaaa3.TagID == stln3) select
minaaa3).Count();
var touty3 = (from minaaa3 in ctx.TagLogs where (minaaa3.TagID == stln3) select
minaaa3);
if (minaa3 >= 1) {
if (FST == false) {
MessageBox.Show("stolen car found ");
dataGridView1.DataSource = touty3;
FST = true;
}}}

if (textBox3.Lines[0] != "0") {
var papawahba = from papa in ctx.TagLogs where papa.ReaderIP == "192.168.10.12"
select papa.TagID;
foreach (var elem in papawahba)

{ if (elem.ToString() == textBox3.Lines[0]) goto stateII; }

```



```

var papawahba2 = from papa2 in ctx.TagLogs where papa2.ReaderIP ==
"192.168.10.14" select papa2.TagID;
foreach (var elem2 in papawahba2)
{ if (elem2.ToString() == textBox3.Lines[0]) goto stateIII; }
var papawahba3 = from papa3 in ctx.TagLogs where papa3.ReaderIP ==
"192.168.10.16" select papa3.TagID;
foreach (var elem3 in papawahba3)
{ if (elem3.ToString() == textBox3.Lines[0]) goto stateIII; }}
FII = true;
var result2 = (from U in ctx.TagLogs where (((U.OutFieldTime.Second -
U.InFieldTime.Second) > Math.Abs(condet)) && (U.ReaderIP == "192.168.10.17"))
select U).Count();
if ((timeout1 == false) || (result2 >= 1))      {
DateTime currentTime = DateTime.Now;
DateTime timeBefore = currentTime.AddSeconds(-2);
///comparing I III IIII
var cmp21 = (from U2 in ctx.TagLogs where ((U2.ReaderIP == "192.168.10.10") &&
((timeBefore <= (U2.OutFieldTime)) && ((U2.OutFieldTime) <= currentTime))) select
U2).Count();
var cmp23 = (from U3 in ctx.TagLogs where ((U3.ReaderIP == "192.168.10.14") &&
((timeBefore <= (U3.OutFieldTime)) && ((U3.OutFieldTime) <= currentTime))) select
U3).Count();
var cmp24 = (from U4 in ctx.TagLogs where ((U4.ReaderIP == "192.168.10.16") &&
((timeBefore <= (U4.OutFieldTime)) && ((U4.OutFieldTime) <= currentTime))) select
U4).Count();
if ((cmp21 >= cmp23) && (cmp21 >= cmp24))//I state
{
if (FI == true && FII == true && FIII == true && FIIII == true)      {
FI = FII = FIII = FIIII = false; goto stateI; int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
}
else      {
if (FI == true)      {
if (cmp23 > cmp24)//stateIII
{
if (FIII == true)      {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();

```



```

if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;
}
else {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII; } }
else {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateI; } }
else if ((cmp23 >= cmp21) && (cmp23 >= cmp24)) //III state
{
if (FI == true && FII == true && FIII == true && FIIII == true) {
FI = FII = FIII = FIIII = false; int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();

```



```

Thread.Sleep(milliseconds);
goto stateIII; }
else {
if (FIII == true) {
if (cmp21 > cmp24)//stateI
{
if (FI == true){
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}
else{
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateI;}}
else {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;// stateIII} }
else {///wait 5 sec
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}} }

```




```

else if ((cmp24 >= cmp23) && (cmp24 >= cmp21))//stateIII
{
if (FI == true && FII == true && FIII == true && FIIII == true)       {
FI = FII = FIII = FIIII = false; int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;          }
else      {
if (FIIII == true)          {
if (cmp21 > cmp23)//stateI
{
if (FI == true)          {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;
}
else      {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateI;}}
else{
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);

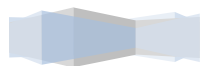
```



```

goto stateIII;// stateIII} }
else
{///wait 5 sec
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}}}
else{
var result3 = (from U5 in ctx.TagLogs where (((U5.OutFieldTime.Second -
U5.InFieldTime.Second) > Math.Abs(condet)) && (U5.ReaderIP == "192.168.10.11"))
select U5).Count();
if (result3 >= 1){
if (!port.IsOpen) { port.Open(); } port.Write("l"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("d"); port.Close();}}
////////////////////////////////////
stateIII:
DateTime currentTime3 = DateTime.Now;
DateTime timeBefore3 = currentTime3.AddSeconds(-timerClock.Interval);
var queryyy = from p3 in ctx.TagLogs where p3.InFieldTime <= currentTime3 select p3;
foreach (var x in queryyy)
ctx.DeleteObject(x);
ctx.SaveChanges();
timeout1 = true;
if (textBox4.Lines[0] != "0"){
var stln4 = textBox4.Lines[0];
var minaaa4 = (from minaaa4 in ctx.TagLogs where (minaaa4.TagID == stln4) select
minaaa4).Count();
var touty4 = (from minaaa4 in ctx.TagLogs where (minaaa4.TagID == stln4) select
minaaa4);
if (minaaa4 >= 1){
if (FST == false){
MessageBox.Show("stolen car found ");
dataGridView1.DataSource = touty4; FST = true;}} }
if (textBox3.Lines[0] != "0"){
var papawahba = from papa in ctx.TagLogs where papa.ReaderIP == "192.168.10.12"
select papa.TagID;
foreach (var elem in papawahba)
{ if (elem.ToString() == textBox3.Lines[0]) goto stateII; }
var papawahba2 = from papa2 in ctx.TagLogs where papa2.ReaderIP ==
"192.168.10.14" select papa2.TagID;
foreach (var elem2 in papawahba2)

```



```

{ if (elem2.ToString() == textBox3.Lines[0]) goto stateIII; }
var papawahba3 = from papa3 in ctx.TagLogs where papa3.ReaderIP ==
"192.168.10.16" select papa3.TagID;
foreach (var elem3 in papawahba3)
{ if (elem3.ToString() == textBox3.Lines[0]) goto stateIII; }}
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("k"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("c"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); //port.Close();
while ((fstop == false)){
if (textBox4.Lines[0] != "0"){
var stln5 = textBox4.Lines[0];
var minaa5 = (from minaaa5 in ctx.TagLogs where (minaaa5.TagID == stln5) select
minaaa5).Count();
var touty5 = (from minaaa5 in ctx.TagLogs where (minaaa5.TagID == stln5) select
minaaa5);
if (minaa5 >= 1){
if (FST == false){
MessageBox.Show("stolen car found ");
dataGridView1.DataSource = touty5; FST = true;}} }
if (textBox3.Lines[0] != "0"){
var papawahba = from papa in ctx.TagLogs where papa.ReaderIP == "192.168.10.12"
select papa.TagID;
foreach (var elem in papawahba)
{ if (elem.ToString() == textBox3.Lines[0]) goto stateII; }
var papawahba2 = from papa2 in ctx.TagLogs where papa2.ReaderIP ==
"192.168.10.14" select papa2.TagID;
foreach (var elem2 in papawahba2)
{ if (elem2.ToString() == textBox3.Lines[0]) goto stateIII; }
var papawahba3 = from papa3 in ctx.TagLogs where papa3.ReaderIP ==
"192.168.10.16" select papa3.TagID;
foreach (var elem3 in papawahba3)
{ if (elem3.ToString() == textBox3.Lines[0]) goto stateIII; }}
FIII = true;
var result4 = (from V in ctx.TagLogs where (((V.OutFieldTime.Second -
V.InFieldTime.Second) > Math.Abs(condet)) && (V.ReaderIP == "192.168.10.11"))
select V).Count();
if ((timeout1 == false) || (result4 >= 1)){
DateTime currentTime = DateTime.Now;
DateTime timeBefore = currentTime.AddSeconds(-2);
var cmp31 = (from V2 in ctx.TagLogs where ((V2.ReaderIP == "192.168.10.10") &&
((timeBefore <= (V2.OutFieldTime)) && ((V2.OutFieldTime) <= currentTime))) select
V2).Count();

```



```

var cmp32 = (from V3 in ctx.TagLogs where ((V3.ReaderIP == "192.168.10.12") &&
((timeBefore <= (V3.OutFieldTime)) && ((V3.OutFieldTime) <= currentTime))) select
V3).Count();
var cmp34 = (from V4 in ctx.TagLogs where ((V4.ReaderIP == "192.168.10.16") &&
((timeBefore <= (V4.OutFieldTime)) && ((V4.OutFieldTime) <= currentTime))) select
V4).Count();
Console.WriteLine("FI");
Console.WriteLine(FI);//true
Console.WriteLine(FII);
Console.WriteLine(FIII);//true
Console.WriteLine(FIIII);
if ((cmp31 >= cmp32) && (cmp31 >= cmp34))//I state
{if (FI == true && FII == true && FIII == true && FIIII == true)
{  FI = FII = FIII = FIIII = false; int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateI;}
Else{
if (FI == true){
if (cmp32 > cmp34)//stateII
{ if (FII == true)
{ int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII; }
else{ int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII;}}
else{ {
int milliseconds = 5000;

```



```

if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}}
else
{///wait 5 sec
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateI;}}}
else if ((cmp32 >= cmp31) && (cmp32 >= cmp34))//II state
{
if (FI == true && FII == true && FIII == true && FIIII == true)
FI = FII = FIII = FIIII = false;{
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII;}}}
else
{
if (FII == true)
if (cmp31 > cmp34)//stateI
{
if (FI == true)
{ int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}
else
{

```



```

int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateI;}}
else
    {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}
else
    {
    ///wait 5 sec
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII;}}
else if ((cmp34 >= cmp32) && (cmp34 >= cmp31))//stateIII
    {
if (FI == true && FII == true && FIII == true && FIIII == true)
{FI = FII = FIII = FIIII = false;
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}
else
    {
if (FIIII == true)
    {
if (cmp31 > cmp32)//stateI

```



```

{
if (FI == true){
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII;          }
else{
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateI;}}
else{
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII;}}
else
{///wait 5 sec
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}}}
else
{
var result5 = (from V5 in ctx.TagLogs where (((V5.OutFieldTime.Second -
V5.InFieldTime.Second) > Math.Abs(condet)) && (V5.ReaderIP == "192.168.10.13"))
select V5).Count();

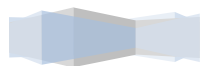
```



```

if (result5 >= 1)
    {
if (!port.IsOpen) { port.Open(); } port.Write("l");// port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("a"); port.Close();}}
////////////////////////////////////
stateIII:
DateTime currentTime4 = DateTime.Now;
DateTime timeBefore4 = currentTime4.AddSeconds(-timerClock.Interval);
var queryyyy = (from p4 in ctx.TagLogs where p4.InFieldTime <= currentTime4 select
p4);
foreach (var x in queryyyy)
ctx.DeleteObject(x);
ctx.SaveChanges();
timeout1 = true;
if (textBox4.Lines[0] != "0")
    {
var stln6 = textBox4.Lines[0];
var minaa6 = (from minaaa6 in ctx.TagLogs where (minaaa6.TagID == stln6) select
minaaa6).Count();
var touty6 = (from minaaa6 in ctx.TagLogs where (minaaa6.TagID == stln6) select
minaaa6);
if (minaa6 >= 1)
    {
if (FST == false)
    {
MessageBox.Show("stolen car found ");
dataGridView1.DataSource = touty6; FST = true;}}
if (textBox3.Lines[0] != "0")
    {
var papawahba = from papa in ctx.TagLogs where papa.ReaderIP == "192.168.10.12"
select papa.TagID;
foreach (var elem in papawahba)
{ if (elem.ToString() == textBox3.Lines[0]) goto stateII; }
var papawahba2 = from papa2 in ctx.TagLogs where papa2.ReaderIP ==
"192.168.10.14" select papa2.TagID;
foreach (var elem2 in papawahba2)
{ if (elem2.ToString() == textBox3.Lines[0]) goto stateIII; }
var papawahba3 = from papa3 in ctx.TagLogs where papa3.ReaderIP ==
"192.168.10.16" select papa3.TagID;
foreach (var elem3 in papawahba3)
{ if (elem3.ToString() == textBox3.Lines[0]) goto stateIII; }}
if (!port.IsOpen) { port.Open(); } port.Write("g");//port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h");//port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i");//port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("l");//port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e");//port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("d"); port.Close();
while ((fstop == false))
    {
if (textBox4.Lines[0] != "0")
    {
var stln7 = textBox4.Lines[0];

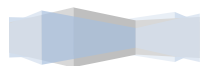
```




```

var minaa7 = (from minaaa7 in ctx.TagLogs where (minaaa7.TagID == stln7) select
minaaa7).Count();
var touty7 = (from minaaa7 in ctx.TagLogs where (minaaa7.TagID == stln7) select
minaaa7);
if (minaa7 >= 1)
    {
if (FST == false)
    {
MessageBox.Show("stolen car found ");
dataGridView1.DataSource = touty7; FST = true; } }
if (textBox3.Lines[0] != "0"){
var papawahba = from papa in ctx.TagLogs where papa.ReaderIP == "192.168.10.12"
select papa.TagID;
foreach (var elem in papawahba)
    { if (elem.ToString() == textBox3.Lines[0]) goto stateII; }
var papawahba2 = from papa2 in ctx.TagLogs where papa2.ReaderIP ==
"192.168.10.14" select papa2.TagID;
foreach (var elem2 in papawahba2)
    { if (elem2.ToString() == textBox3.Lines[0]) goto stateIII; }
var papawahba3 = from papa3 in ctx.TagLogs where papa3.ReaderIP ==
"192.168.10.16" select papa3.TagID;
foreach (var elem3 in papawahba3)
    { if (elem3.ToString() == textBox3.Lines[0]) goto stateIII; }
    }
FIIII = true;
var result6 = (from O in ctx.TagLogs where (((O.OutFieldTime.Second -
O.InFieldTime.Second) > Math.Abs(condet)) && (O.ReaderIP == "192.168.10.13"))
select O).Count();
if ((result6 >= 1) || (timeout1 == false)){
DateTime currentTime = DateTime.Now;
DateTime timeBefore = currentTime.AddSeconds(-2);
///comparing II III IIII
var cmp41 = (from O2 in ctx.TagLogs where ((O2.ReaderIP == "192.168.10.10") &&
((timeBefore <= (O2.OutFieldTime)) && ((O2.OutFieldTime) <= currentTime))) select
O2).Count();
var cmp42 = (from O3 in ctx.TagLogs where ((O3.ReaderIP == "192.168.10.12") &&
((timeBefore <= (O3.OutFieldTime)) && ((O3.OutFieldTime) <= currentTime))) select
O3).Count();
var cmp43 = (from O4 in ctx.TagLogs where ((O4.ReaderIP == "192.168.10.14") &&
((timeBefore <= (O4.OutFieldTime)) && ((O4.OutFieldTime) <= currentTime))) select
O4).Count();
if ((cmp41 >= cmp42) && (cmp41 >= cmp43))//I state
    {
if (FI == true && FII == true && FIII == true && FIIII == true)
    {
FI = FII = FIII = FIIII = false;
    {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();

```



```

if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII; } }
else{
if (FI == true) {
if (cmp42 > cmp43)//stateII
{
if (FII == true){
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII; }
else { int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII; } }
else{ {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII; } } }
else { //wait 5 sec
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();

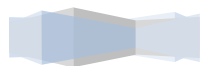
```



```

Thread.Sleep(milliseconds);
goto stateI;}}
else if ((cmp42 >= cmp41) && (cmp42 >= cmp43))//II state
{
if (FI == true && FII == true && FIII == true && FIIII == true)      {
FI = FII = FIII = FIIII = false;          {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII;}}
else          {
if (FII == true)          {
if (cmp41 > cmp43)//stateI
{
if (FI == true)          {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}
else          {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}
else{{
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();

```



```

if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateI;}}
else
{///wait 5 sec
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII;}}
else if ((cmp43 >= cmp42) && (cmp43 >= cmp41))//stateIII
{
if (FI == true && FII == true && FIII == true && FIIII == true)
FI = FII = FIII = FIIII = false;
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}
else
{
if (FIII == true)
if (cmp41 > cmp42)//stateI
{
if (FI == true)
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII;
}
else
{
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();

```



```

if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateI;}}
else{ {
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateII;}} }
else {///wait 5 sec
int milliseconds = 5000;
if (!port.IsOpen) { port.Open(); } port.Write("g"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("h"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("i"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("j"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("e"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("f"); port.Close();
Thread.Sleep(milliseconds);
goto stateIII;}}}}
else{
var result7 = (from O5 in ctx.TagLogs where (((O5.OutFieldTime.Second -
O5.InFieldTime.Second) > Math.Abs(condet)) && (O5.ReaderIP == "192.168.10.15"))
select O5).Count();
if (result7 >= 1)
{
if (!port.IsOpen) { port.Open(); } port.Write("k"); //port.Close();
if (!port.IsOpen) { port.Open(); } port.Write("b"); port.Close();}}}}
////////////////////////////////////
////////////////////////////////////
public void OnTimer(Object source, ElapsedEventArgs e)
{
timeout1 = false; }
////////////////////////////////////
private void button2_Click(object sender, EventArgs e)
{
fstop = true;
timerClock.Enabled = false;
DateTime currentTime1 = DateTime.Now;
GP_TrafficListenerEntities ctx = new GP_TrafficListenerEntities();
var query = from p in ctx.TagLogs where p.InFieldTime<=currentTime1 select p;

```



```
foreach (var x in query)
ctx.DeleteObject(x);
ctx.SaveChanges();  } }
```

