ACCIDENT DETECTION AND HANDLING IN VEHICLES USING GPS

A PROJECT REPORT

Submitted by

P.ARUN PRAKASS	71203106005
JEMIN SADISH SHETH	71203106017
K.S.KARTHIK	71203106018
N.KARTHIK	71203106302

In partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

In

ELECTRONICS AND COMMUNICATION ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY,

COIMBATORE

ANNA UNIVERSITY:: CHENNAI 600 025

APRIL 2007

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"ACCIDENT DETECTION AND HANDLING IN VEHICLES USING GPS"** is the bonafide work of **"P. ARUN PRAKASS, JEMIN SADISH SHETH, K.S. KARTHIK, N. KARTHIK"** who carried out the project work under my supervision.

SIGNATURE

(Dr. Rajeswari Mariappan Ph.D.,)

HEAD OF THE DEPARTMENT & SUPERVISOR

Department of Electronics and Communication Engineering,

Kumaraguru College of Technology,

Chinnavedampatti,

Coimbatore-641 006.

The candidates with University Register Nos. 71203106005, 71203106017, 71203106018, 71203106302 were examined by us in the project Viva- voce examination held on ________________

Internal Examiner **External Examiner**

LIST OF TABLES

TABLE NO.	TITLE	PAGE NO.
4.1	Standard Characteristics of NMEA	22
4.2	GPGGA Message contents	25
7.1	Features of AT89C51	37
8.1	RS-232 Pin Details	53
8.2	MAX232 DIP Package Pin Layout	57

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
2.1	Accident Detection and Handling- Block Diagram	9
3.1	Scientech Receiver	16
5.1	Compact Magnetic Mount Antenna	29
7.1	Block diagram of a typical microcontroller	35
7.2	Pin diagram of a AT89C51	37
7.3	Block diagram of KM6264B	38
7.4	KM6264B Pin Description	40
8.1	RS-232 Pin Diagram	52
8.2	MAX232 Pin Diagram	56
8.3	MAX232 Pin Diagram With Details	56
10.1	Output of the project	69

	3.2	GPS Satellite Message	15
	3.3	Satellite Acquisition and Time to First Fix	16
	3.4	Garage Search Strategy	18
	3.5	Position Accuracy	19
	3.6	Standard Operating Modes	19
	3.7	System Architecture	20
4		**NATIONAL MARINE ELECTRONICS ASSOCIATION PROTOCOL**	**21**
	4.1	The NMEA 0183 Communication Interface	22
	4.2	NMEA 0183 MESSAGE FORMAT	23
	4.3	NMEA 0183 Message Options	24
	4.4	Interruption of GPS Signal	24
5		**GPS TIMING AND ANTENNAS**	**26**
	5.1	GPS Timing	27
	5.2	GPS Antennas	28
	5.3	Compact Magnetic-Mount Antenna	28
	5.4	Mounting	29
6		**VIBRATION SENSOR**	**30**
	6.1	Introduction	31
	6.2	Working	31

7		**MICROCONTROLLER**	**33**
	7.1	Introduction to Microcontrollers	34
	7.2	Advantages of Microcontrollers	34
	7.3	About 8051 Microcontrollers	36
	7.4	AT89C51 (Atmel)	37
	7.5	Pin Diagram of AT89C51	37
	7.6	Block diagram of AT89C51	38
	7.7	Description	39
	7.8	External Data Memory	39
	7.9	KM6264B Pin Description	40
	7.10	Function of microcontroller in our project	40
	7.11	Microcontroller code	41
8		**SERIAL COMMUNICATION**	47
	8.1	Introduction	48
	8.2	Issues in Communication	48
	8.3	Modes of Serial Data Transmission	50
	8.4	RS-232 Standard	51
	8.5	MAX232	55
9		**PARENT TERMINAL**	**58**
	9.1	Introduction	59
	9.2	RF Transmitter-Receiver	59
	9.3	Visual Basic coding: An overview	59
	9.4	VB Code	60

10	**CONCLUSION**	**67**
10.1	Result	68
10.2	Future Enhancements	68
10.3	Output of the Project	69
APPENDIX		**71**
REFERENCES		**78**

LIST OF ABBREVATIONS

GPS	Global Positioning System
DOD	Department Of Defence
PRN	Pseudo Random Noise
C/A	Course/Acquistion
2D	Two Dimensional
3D	Three Dimensional
I/O	Input Output
NMEA	National Marine Electronics Association
ASCII	American Standard Code for Information Interchange
DGPS	Differential Global Positioning System
PDOP	Position Dilution of Precision
HDOP	Horizontal Dilution of Precision
VDOP	Vertical Dilution of Precision
RAM	Random Access Memory
ROM	Read Only Memory
ADC	Analog to Digital Converter
DAC	Digital to Analog Converter
EPROM	Erasable Programmable Read Only Memory
PCB	Printed Circuit Board
RF	Radio Frequency
TX	Transmitter
RX	Receiver
OP-AMP	Operational Amplifier
VB	Visual Basic

CHAPTER 1

1. INTRODUCTION

1.1 GPS

The Global Positioning System (GPS) is a world wide radio-navigation system formed from a constellation of 24 satellites and their ground stations. GPS uses these "man made stars" as reference points to calculate positions accurate to a matter of meters. In fact, with advanced form of GPS measurements can be made to better than a centimeter! In a sense it's like giving every square meter on the planet unique address. GPS receivers have been miniaturized to just a few integrated circuits and so are becoming very economical. And that makes the technology accessible to virtually everyone. These days GPS is finding its way into cars, boats, planes, construction equipments, movie making gear, farm machinery, even laptop computers. Soon GPS will become almost as basic as the telephone.

1.2 GPS works in five logical steps:

1. The basis of GPS is trilateration from satellites
2. To "trilaterate," a GPS receiver measures distance using the travel time of radio signals.
3. To measure the travel time, GPS needs very accurate timing which it achieves with some tricks.
4. Along with distance, it is to be known exactly where the satellites are in space. High orbits and careful monitoring are the secrets.
5. Finally delays which the signal experiences as it travels through the atmosphere should be corrected

Improbable as it may seem, the whole idea behind GPS is to use satellites in space as reference points for locations here on earth. By very, very accurately measuring the distance from three satellites it is possible to "trilaterate" the position anywhere on earth.

1.3 Trilaterating from Satellites:

Trilateration can be done by measuring distances from three satellites. Suppose the distance measured from a satellite is 11,000 miles. Knowing that the GPS receiver is 11,000 miles from a particular satellite narrows down all possible locations the receiver could be in the whole universe to the surface of a sphere that is centered on this satellite and has a radius of 11,000 miles. If the distance measured from the second satellite is 12,000 miles. This tells that the GPS receiver is not only on the first sphere but also on a sphere that's 12,000 miles from the second satellite. Or in other words, it is somewhere on the circle where these two spheres intersect. If the measurement from a third satellite is made and found to be 13,000 miles from that one, that narrows the receiver's position down even further, to the two points where the 13,000 mile sphere cuts through the circle that's the intersection of the first two spheres.

So by ranging from three satellites, the receiver's position can be narrowed down to just two points in space. To decide which one is the true location a fourth measurement can be made. But usually one of the two points is a ridiculous answer (either too far from earth or moving at an impossible velocity) and can be rejected without a measurement. The position is calculated from distance measurements to at least three satellites. It is done by timing how long it takes for a signal sent from the satellite to arrive at the receiver.

1.4 Distance Measurement:

The distance can be calculated as follows:

Velocity x Time = Distance

In the case of the GPS the measured signal is the radio signal, so the velocity is going to be the speed of the light or roughly 186.000 miles per second. The problem is measuring the travel time. The timing problem is tricky. First, the times are going to be awfully short. If a satellite were right overhead the travel time would be something like 0.06 seconds. So some precise clocks are needed. Only the satellites and receivers use something called "Pseudo Random Code"-which is probably easier.

1.5 Pseudo Random Code:

The Pseudo Random Code is a fundamental part of GPS. Physically it's just a very complicated digital code, or in other words, a complicated sequence of "on" and "off" pulses. The signal is so complicated that it almost looks like random electrical noise. Hence the name "Pseudo Random". There are several good reasons for that complexity: first, the complex pattern makes sure that the receiver doesn't accidentally synchronize up to some other signal. The pattern's are so complex that it's highly unlikely a stray signal will have exactly the same shape. Since each satellite has its own unique Pseudo-Random Code this complexity also guarantees that the receiver won't accidentally pick up another satellite's signal. So all the satellites can use the same frequency without jamming each other. And it makes it more difficult for a hostile force to jam the system. In fact the Pseudo Random Code gives the DOD a way to control access to the system. But there's another reason for the complexity of the Pseudo Random Code, a reason that's crucial to make the GPS economical. The codes make it possible to use "information theory" to "amplify" the GPS signal. And that's why GPS receivers don't need big satellite dishes to receive the GPS signals.

1.6 Getting Perfect Timing:

On the satellite side, timing is almost perfect because they have incredibly precise atomic clocks on board. Both the satellite and the receiver need to be able to precisely synchronize their Pseudo Random codes to make the system work. If the receiver needed atomic clocks (which cost upwards of $50k to $100k) GPS would be lame duck technology.

The designers of GPS came up with a brilliant trick that lets to get much less accurate clocks in the receivers. This trick is one of the key elements of the GPS and as an added side benefit it means that every GPS receiver is essentially an atomic-accuracy clock. The secret to perfect timing is to make an extra satellite measurement. If three perfect measurements can locate a point in 3-dimensional space, then four imperfect measurements can do the same thing. This idea is so perfect to the working of GPS. If the receiver's clock were perfect, then all the satellites range would intersect at single point but with imperfect clock, a fourth measurement, done as cross-check, will not intersect with the first three. Since any offset from universal time will affect all the measurements, the receiver looks for a single correction factor that it can subtract from all its timing measurement that would cause them all to intersect at a single point. That correction brings the receiver's clock back into sync with universal time and the atomic accuracy time is obtained. Once it has that correction it applies to all the rest of its measurements and now the precise positioning is obtained. One of the consequence of this principle is that any decent GPS receiver will need to have at least four channels so that it can make the four measurements simultaneously. With the pseudo-random code as a rock solid timing sync pulse, and this extra measurement trick to get perfectly synchronized to universal time, everything needed to measure the distance to a satellite in space is obtained.

1.7 Knowing where a Satellite is in Space:

That 11,000 mile altitude is actually a benefit in this case, because something that high is very clear of the atmosphere. And that means it will orbit according to very simple mathematics. The air force has injected each GPS satellite into a very precise orbit, according to the GPS master plan. On the ground all GPS receivers have an almanac programmed into their computers that tells them where in sky each satellite is, moment by moment.

The basic orbits are quite exact but just to make things perfect the GPS satellites are constantly monitored by the Department of Defense. They use very precise radar to check each satellite's exact altitude, position and speed.

The errors they are checking for are called "ephemeris errors" because they affect the satellite's orbit or "ephemeris". These errors are caused by gravitational pulls from the moon and the sun and by the pressure of solar radiation on the satellites. Once the DOD (Department Of Defense) has measured a satellite's exact position, they relay that information back up to the satellite itself. The satellite then includes this new corrected position information in the timing signals its broadcasting. So a GPS signal is more than just pseudo-random code for timing purposes.

CHAPTER 2

2. ACCIDENT DETECTION AND HANDLING

2.1 General Block Diagram and Description

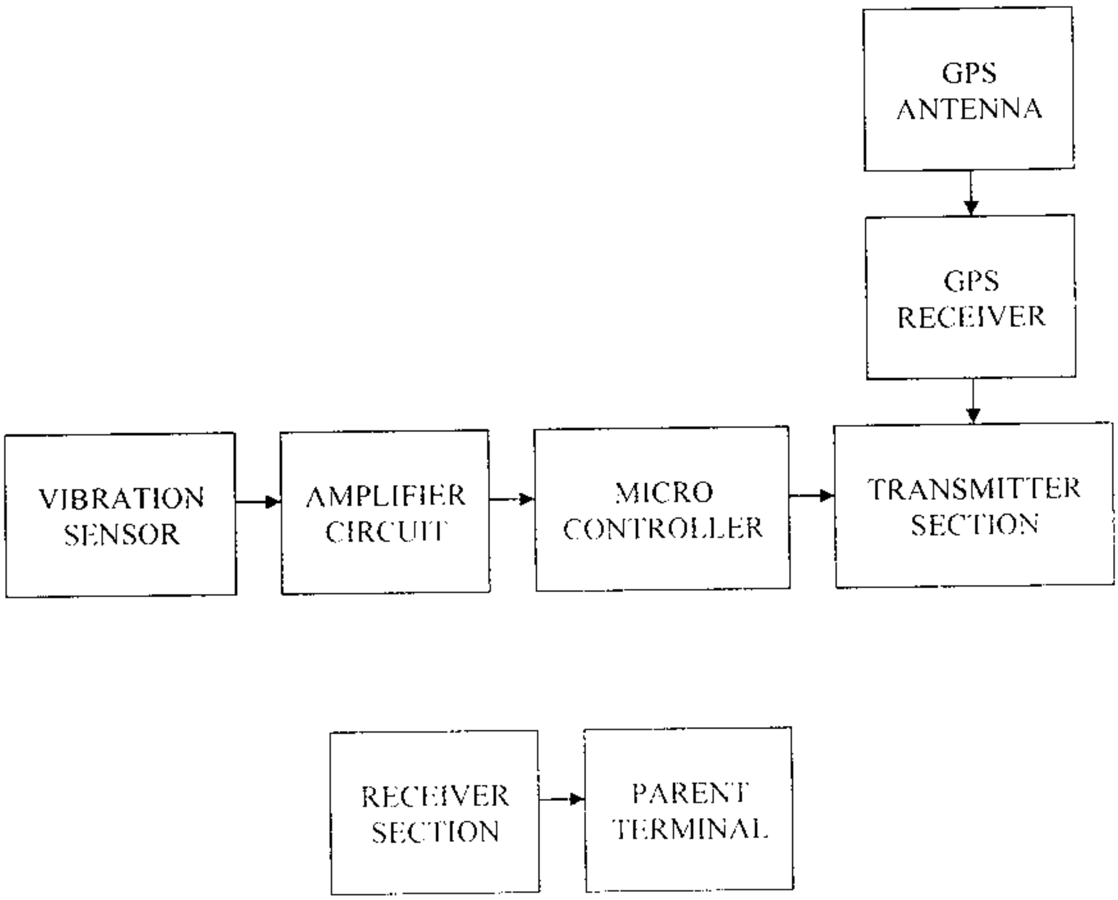


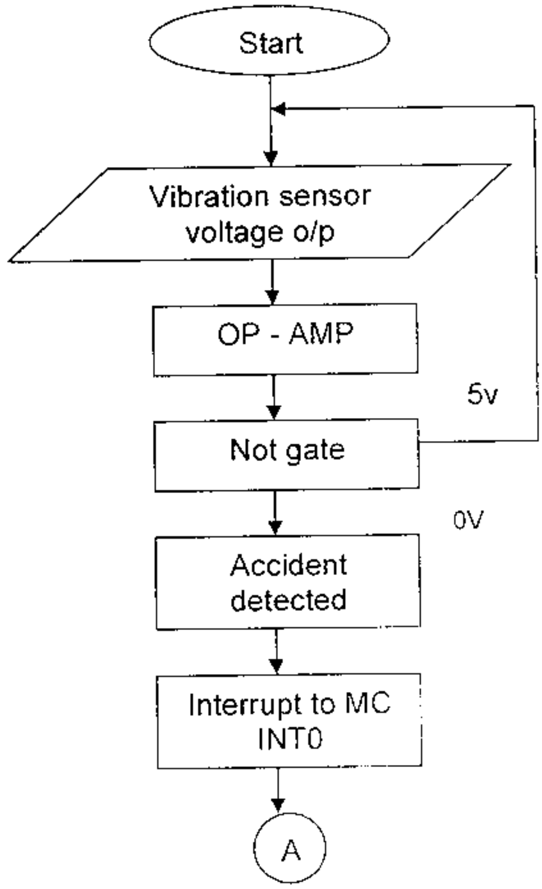
Fig 2.1 Accident Detection and Handling- Block Diagram

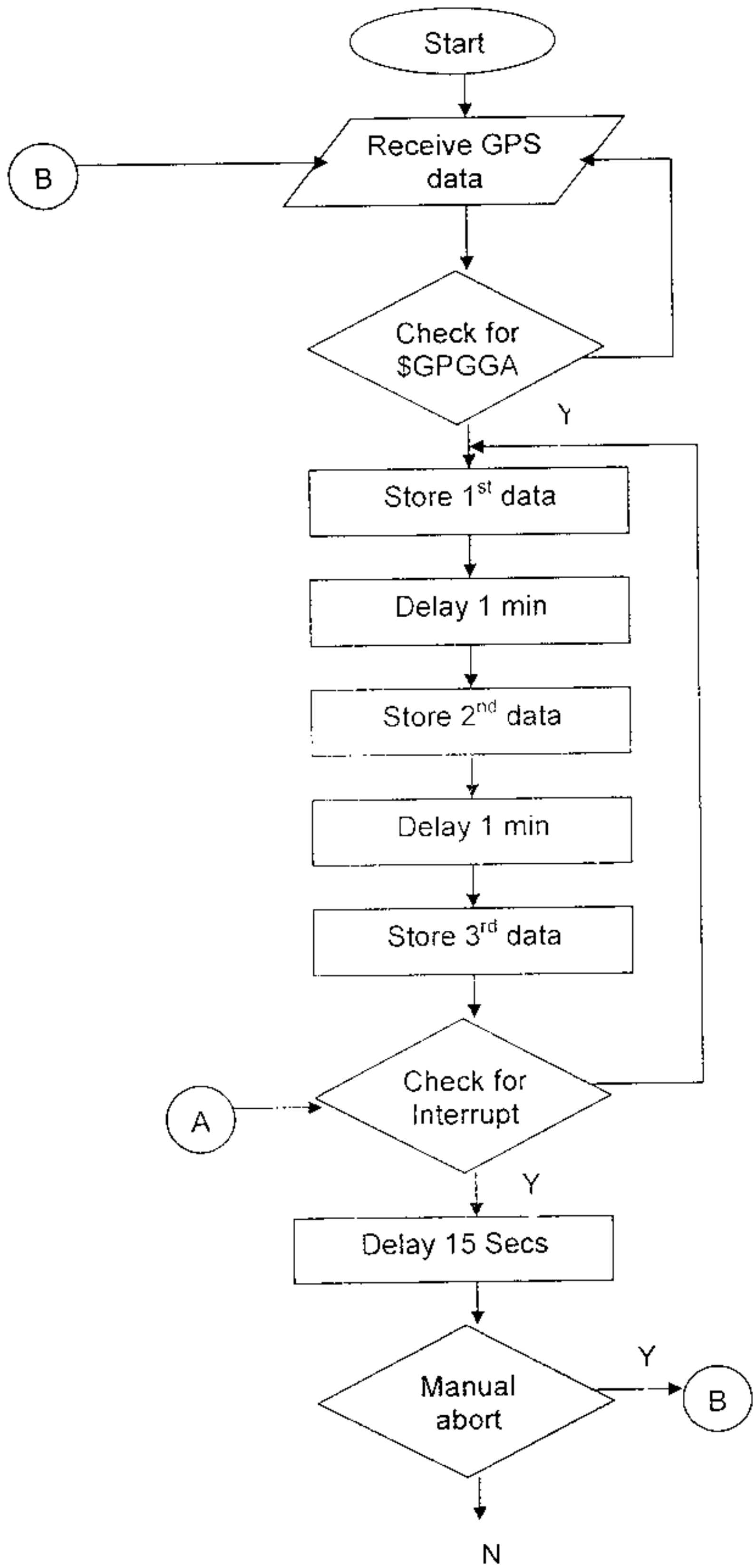
This is the general block diagram of the Accident detection and handling system. It mainly consists of the GPS antenna and receiver. vibration sensor, 8051 microcontroller, specifically AT89C51, RF transmitter and receiver and the parent terminal (computer). The scientech GPS receiver establishes communication with the GPS satellites using Magnetic mount antenna. The power supplies used are two batteries of 6V and two batteries of 9V. The 6V is given to microcontroller and RF receiver. The 9V is given to Op-amp.

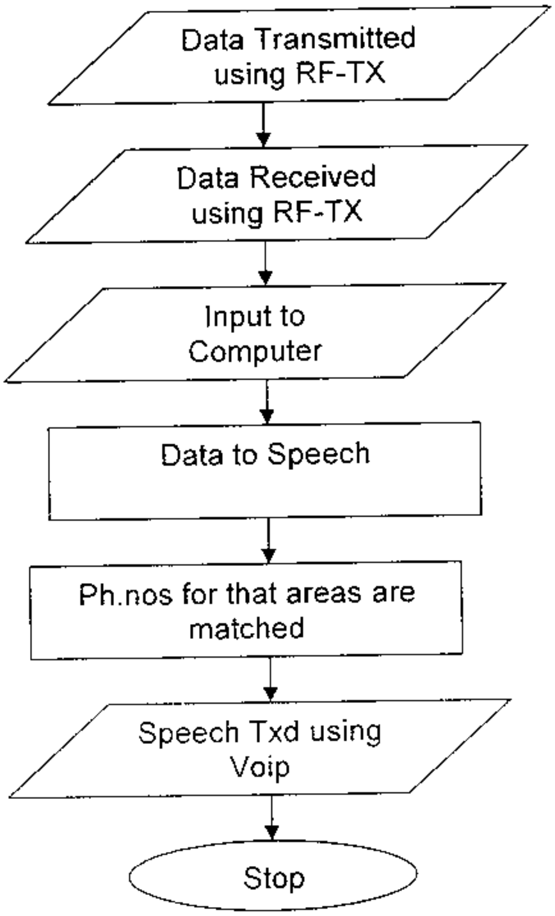
2.2 Algorithm

- Start
- If an accident happens the vibration sensor converts the vibrations into voltage (mV).
- The voltage (mV) is amplified using OP-Amp IC-741.
- The output 5 volts is given to the not gate for producing 0 volt. So that it can be given to hardware interrupt.
- Meanwhile the microcontroller receives GPS signal.
- The microcontroller checks for $GPGGA and stores only that format. Other message formats are neglected.
- Data is stored for every minute in the memory and it is only stored for past three minutes.
- The fourth minute data is stored in the first minute's data location. This is repeated in a cycle.
- If the interrupt is invoked at any time the data in the RAM is transmitted
- The RF Transmitter sends data using ON - OFF key modulation
- The RF receiver receives the data.
- The input is given to the computer.
- The speed, timing, latitude and longitude are processed.
- The data is converted to speech and stored in a file.
- A call is initiated from the computer using voice input.
- The voice is transmitted to the nearest hospital and police station.

2.3 Flowchart




Start
Receive GPS data
B
Check for $GPGGA
Y
Store 1st data
Delay 1 min
Store 2nd data
Delay 1 min
Store 3rd data
A
Check for Interrupt
Y
Delay 15 Secs
Manual abort
Y
B
N


Data Transmitted using RF-TX
Data Received using RF-TX
Input to Computer
Data to Speech
Ph.nos for that areas are matched
Speech Txd using Voip
Stop

CHAPTER 3

3. GPS RECEIVER

3.1 Introduction

This chapter describes the scientech GPS receiver satellite acquisition and tracking processes, performance characteristics and system architecture. The scientech GPS receiver satellite acquisition and tracking algorithms can achieve a position solution without any initialization. The receiver automatically selects and tracks the best combination of satellites to compute position and velocity. As satellites move out of view, the scientech GPS receiver automatically acquires new satellites and includes them in the solution set as required.

3.2 GPS Satellite Message

Every GPS satellite transmits the Coarse/Acquisition (C/A) code and satellite data modulated onto the L1 carrier frequency (1575.42 MHz). The satellite data transmitted by each satellite includes a satellite almanac for the entire GPS system, its own satellite ephemeris and its own clock correction. The satellite data is transmitted in 30 seconds frames. Each frame contains the clock correction and ephemeris for that specific satellite, and two pages of the 50 page GPS system almanac. The almanac is repeated every 12.5 minutes. The ephemeris is repeated every 30 seconds. The system almanac contains information about each of the satellites in the constellation, ionospheric data and special system messages. The GPS system almanac is updated weekly and is typically valid for months. The ephemeris contains detailed orbital information for a specific satellite. Ephemeris data changes hourly, but is valid for up to 4 hours. The GPS control segment updates the system almanac weekly and the ephemeris hourly through three ground based control stations. During normal operation, the scientech GPS receiver module updates its ephemeris and almanac as needed. The performance of the GPS receiver at power on is determined largely by the availability and the accuracy of the satellite ephemeris data and the availability of a GPS system almanac.

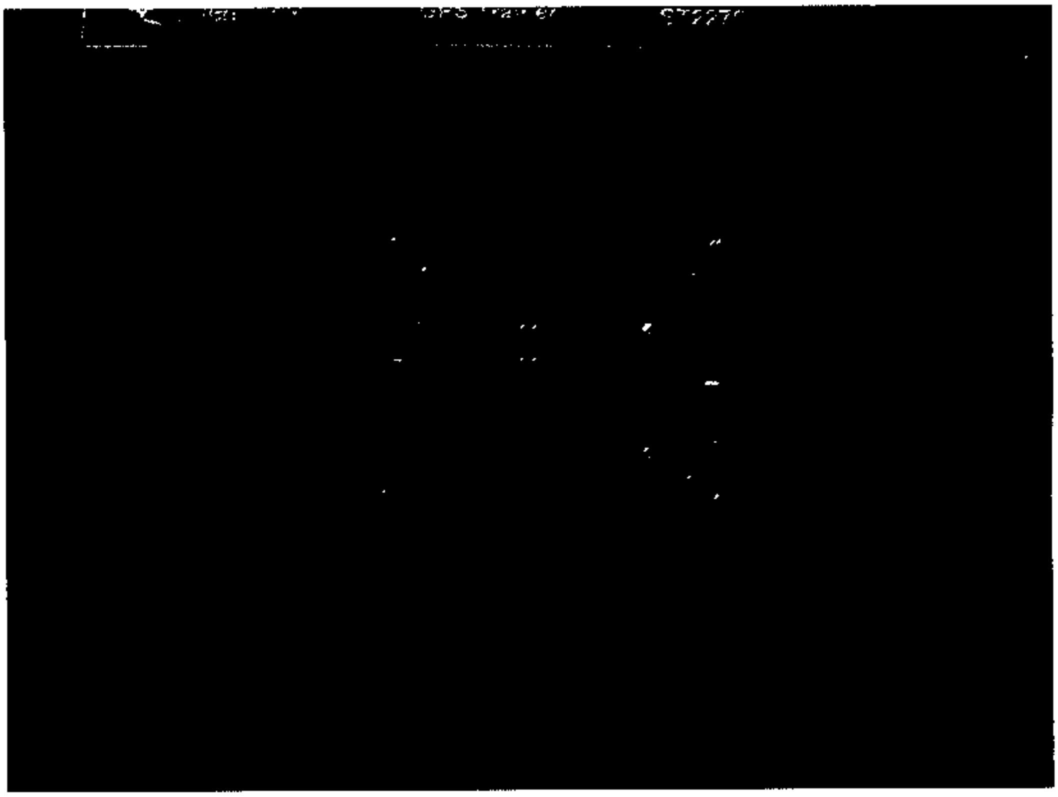


Fig 3.1 Scientech Receiver

3.3 Satellite Acquisition and Time to First Fix

Cold-Start

The term "cold-start" describes the performance of a GPS receiver at power-on when no navigation data is available. "Cold" signifies that the receiver does not have a current almanac, satellite ephemeris, initial position, or time. The cold-start search algorithm applies to a scientech GPS receiver which has no memory of its previous session (i.e., is powered on without the memory backup circuit connected to a source of DC power). In a cold-start condition the receiver automatically selects a set of twelve satellites and dedicates an individual tracking channel to each satellite, to search the Doppler range frequency for each satellite in the set. If none of the twelve selected satellites is acquired after a predetermined period of time (time-out), the receiver will select a new search set of twelve satellites and will repeat the process, until the first satellite is acquired.

As satellites are acquired, the receiver automatically collects ephemeris and almanac data. The scientech GPS receiver uses the knowledge gained from acquiring a specific satellite to eliminate other satellites, those below the horizon, from the search set. This strategy speeds the acquisition of additional satellites required to achieve the first position fix. The cold-start search sets are established to ensure that at least three satellites are acquired within the first two time-out periods. As soon as three satellites are found, the receiver will compute an initial position fix. The typical time to first fix is less than 2 minutes. A complete system almanac is not required to achieve a first position fix.

Warm Start

In a warm-start condition the receiver has been powered down for at least one hour but has stored a current almanac, an initial position, and time, in memory. When connected to an external back-up power source (battery back-up), the scientech GPS receiver retains the almanac, approximate position, and time to aid in satellite acquisition and reduce the time to first fix. When an external back-up battery is not used, the SiRF binary protocol allows the almanac, an initial position, and time to be uploaded to the receiver via the serial port, to initiate a warm start. During a warm start, the scientech GPS receiver identifies the satellites which are expected to be in view, given the system almanac, the initial position and the approximate time. The receiver calculates the elevation and expected Doppler shift for each satellite in this expected set and directs the eight tracking channels in a parallel search for these satellites. The warm start time to first fix, when the receiver has been powered down for more than 60 minutes (i.e. the ephemeris data is old), is usually less than 45 seconds.

Hot Start

A hot start strategy applies when the scientech GPS receiver has been powered down for less than 60 minutes, and the almanac, position, ephemeris, and time are valid. The hot start search strategy is similar to a warm start, but since the ephemeris data in memory is considered current and valid, the acquisition time is typically less than 20 seconds.

3.4 Garage Search Strategy

During a warm start search, the scientech GPS receiver knows which satellites to search for, based on the system almanac, the initial position (last known position) and the current time. In some cases, the receiver may not be able to acquire the expected satellite signals (e.g., a vehicle parked in a garage or a vessel in a covered berth). If the receiver does not acquire the expected set of satellites within 5 minutes of power-on, some of the eight tracking channels will continue to search for the expected satellites (warm search) while the remaining channels are directed in a cold start search. This strategy minimizes the time to first fix in cases where the stored almanac, position and time are invalid. The stored information is flushed from memory, if the cold start search proves effective and the warm search fails.

3.5 Position Accuracy

GPS position accuracy is degraded by atmospheric distortion, satellite geometry, satellite clock errors, and receiver clock errors. Effective models for atmospheric distortion of satellite signals have been developed to minimize the impact of tropospheric and ionospheric effects. The impact of satellite clock errors is minimized by each satellite used in the position solution.

3.6 Standard Operating Modes

The tracking mode controls the allocation of the receiver`s tracking channels and the method used for computing position fixes.

Fix Modes

The scientech GPS receiver offers three positioning modes: 2D Manual, 3D Manual, and Automatic 2D/3D. Automatic 2D/3D is the default mode for the scientech GPS receiver. The positioning mode can be modified in receivers accepting SiRF commands.

3.7 System Architecture

The scientech GPS receiver uses twelve processing channels operating on the L1 frequency of 1575.42 MHz and using the coarse acquisition (C/A) code. The module uses custom integrated circuitry designed by Trimble to track the GPS satellite signals. These ICs also contain support circuitry to the navigation processor.

An integrated 32-bit microprocessor is used for tracking, computing a position, and performing the I/O operations. The scientech GPS receiver receives the amplified GPS satellite signals through the antenna feed line connector and passes them to the RF down converter.

A highly stable crystal reference oscillator operating at 12.504 MHz is used by the down converter to produce the signals used by the 12-channel signal processor. The 12-channel signal processor tracks the GPS satellite signals and extracts the carrier code information as well as the navigation data at 50 bits per second. Operation of the tracking channels is controlled by the navigation processor.

The tracking channels are used to track the highest twelve satellites above the horizon. The navigation processor will then use the optimum satellite combination to compute a position. The navigation processor also manages the ephemeris and almanac data for all of the satellites, and performs the data I/O.

CHAPTER 4

4. NATIONAL MARINE ELECTRONICS ASSOCIATION PROTOCOL

NMEA 0183 is a simple, yet comprehensive ASCII protocol which defines both the communication interface and the data format. Since it is a well established industry standard, NMEA 0183 has also gained popularity for use in applications other than marine electronics. The scientech receiver supports the latest release of NMEA 0183, Version 3.0. The primary change in release 3.0 is the addition of the mode indicators in the GLL and RMC messages. For those applications requiring output only from the GPS receiver, NMEA 0183 is a popular choice since, in many cases, an NMEA 0183 software application code already exists.

4.1 The NMEA 0183 Communication Interface

NMEA 0183 allows a single source (talker) to transmit serial data over a single twisted wire pair to one or more receivers (listeners). The table below lists the standard characteristics of the NMEA 0183 data transmissions.

SIGNAL CHARACTERISTIC	NMEA STANDARD
Baud Rate	4800
Data Bits	8
Parity	None (Disabled)
Stop Bits	1

Table 4.1 Standard Characteristics of NMEA

4.2 NMEA 0183 MESSAGE FORMAT

The NMEA 0183 protocol covers a broad array of navigation data. This broad array of information is separated into discrete messages which convey a specific set of information. The entire protocol encompasses over 50 messages, but only a sub-set of these messages apply to a GPS receiver like the scientech GPS receiver. The NMEA message structure is described below.

$IDMSG, D1, D2, D3, D4, ……, Dn*CS[CR] [LF]

Where:

"$"	signifies the start of a message.
ID	The talker identification is a two letter mnemonic which describes the source of the navigation information. The GP identification signifies a GPS source.
MSG	The message identification is a three letter mnemonic which describes the message content and the number and order of the data fields.
","	Commas serve as delimiters for the data fields.
Dn	Each message contains multiple data fields which are delimited by commas.
"*"	The asterisk serves as a checksum delimiter.
CS	The checksum field contains two ASCII characters which indicate the hexadecimal value of the checksum.
[CR][LF]	The carriage return **[CR]** and line feed **[LF]** combination terminate the message.

NMEA 0183 messages vary in length, but each message is limited to 79 characters or less. This length limitation excludes the "$" and the [CR] [LF]. The data field block, including delimiters, is limited to 74 characters or less.

4.3 NMEA 0183 Message Options

The scientech GPS receiver can output any or all of the messages. These messages are output at a milli seconds interval with the "GP" talker ID and checksums. These messages are output at all times during operation, with or without a fix. If a different set of messages has been selected and this setting has been stored in Flash memory the default messages are permanently replaced until the receiver is returned to the factory default settings.

4.4 Interruption of GPS Signal

If the GPS signal is interrupted temporarily, the NMEA will continue to be output according to the user-specified message list and output rate. Position and velocity fields will be blank until the next fix, but most other fields will be filled.

GGA – Geographic Position – Fix information

The GGA message contains time of the position fix, the latitude and longitude of the present vehicle position, the fix quality, number of satellites being tracked, horizontal dilution of position, altitude, and height of geoid.

$GPGGA, hhmmss.sss, llll.lll, a, yyyyy.yyy, a, b, nn, d.d, aaa.a, M, gg.g, M, , , *cs<CR><LF>

The below table describes the elements in the GGA message format.

FIELD#	DESCRIPTION
1	UTC of position (when UTC offset has been decoded by the receiver)
2,3	Latitude, N (North) or S (South)
4,5	Longitude, E (East) or W (West)
6	Fix Quality: 0=Invalid, 1= GPS fix, 2=DGPS fix, 3=PPS fix, 4=Real Time Kinematic, 5=float RTK, 6=estimated (Dead Reckoning) (2.3 Feature), 7=Manual input mode, 8=Simulation mode
7	No of satellites being tracked
8	Horizontal dilution of position
9,10	Altitude, Meters
11,12	Height of geoid, Meters
13,14	Empty field (Last DPGS update, DGPS station number)
cs	Checksum

Table 4.2 GPGGA Message contents

CHAPTER 5

5. GPS TIMING AND ANTENNAS

5.1 GPS Timing

In many timing applications, such as time/frequency standards, site synchronization systems and event measurement systems, GPS receivers are used to discipline local oscillators. The GPS constellation consists of 24 orbiting satellites. Each GPS satellite contains a highly-stable atomic (Cesium) clock, which is continuously monitored and corrected by the GPS control segment. Consequently, the GPS constellation can be considered a set of 24 orbiting clocks with worldwide 24-hour coverage. GPS receivers use the signals from these GPS "clocks" to correct its internal clock, which is not as stable or accurate as the GPS atomic clocks. GPS receivers like the scientech GPS receiver output a highly accurate timing pulse (PPS) generated by its internal clock, which is constantly corrected using the GPS clocks. This timing pulse is synchronized to UTC within ±50 ns. In addition to serving as a highly accurate stand-alone time source, GPS receivers are used to synchronize distant clocks in communication or data networks. This synchronization is possible since all GPS satellite clocks are corrected to a common master clock. Therefore, the relative clock error is the same, regardless of which satellite or satellites are used. The position and time errors are related by the speed of light. Therefore, a position error of 100 meters corresponds to a time error of approximately 333 ns. The hardware and software implementation affects the GPS receiver's PPS accuracy level. The receiver's clocking rate determines the PPS steering resolution. The scientech GPS receiver clocking rate is 12.504 MHz. this rate corresponds to a steering resolution of ±40 ns.

5.2 GPS Antennas

The antenna receives the GPS satellite signals and passes them to the receiver. The GPS signals are spread spectrum signals in the 1575 MHz range and do not penetrate conductive or opaque surfaces. Therefore, the antenna must be located outdoors with a clear view of the sky. The scientech GPS receiver requires an active antenna. The received GPS signals are approximately -130dBm, at the surface of the earth. Trimble's active antenna includes a preamplifier that filters and amplifies the GPS signals before delivery to the receiver.

A Compact Magnetic-Mount GPS Antenna with a 5 m cable and an MCX or SMA connector. This antenna provides for a flexible, movable installation. The MCX or SMA output connector mates to the Hirose connector on the scientech GPS module with an optional RF transition cable. The antenna with the MCX connector is supplied with the Starter Kit. The MCX connector on the end of the antenna cable mates to the MCX connector in the front of the Starter Kit interface unit.

5.3 Compact Magnetic-Mount Antenna

3 V active micropatch antenna with magnetic mount. Cable length: 5m, Dim: 42mm W x 50.5mm L x 13.8mm H (1.65" x 1.99" x 0.55"), Connectors: MCX or SMA; mates through the optional RF transition cable to the on-module RF connector.

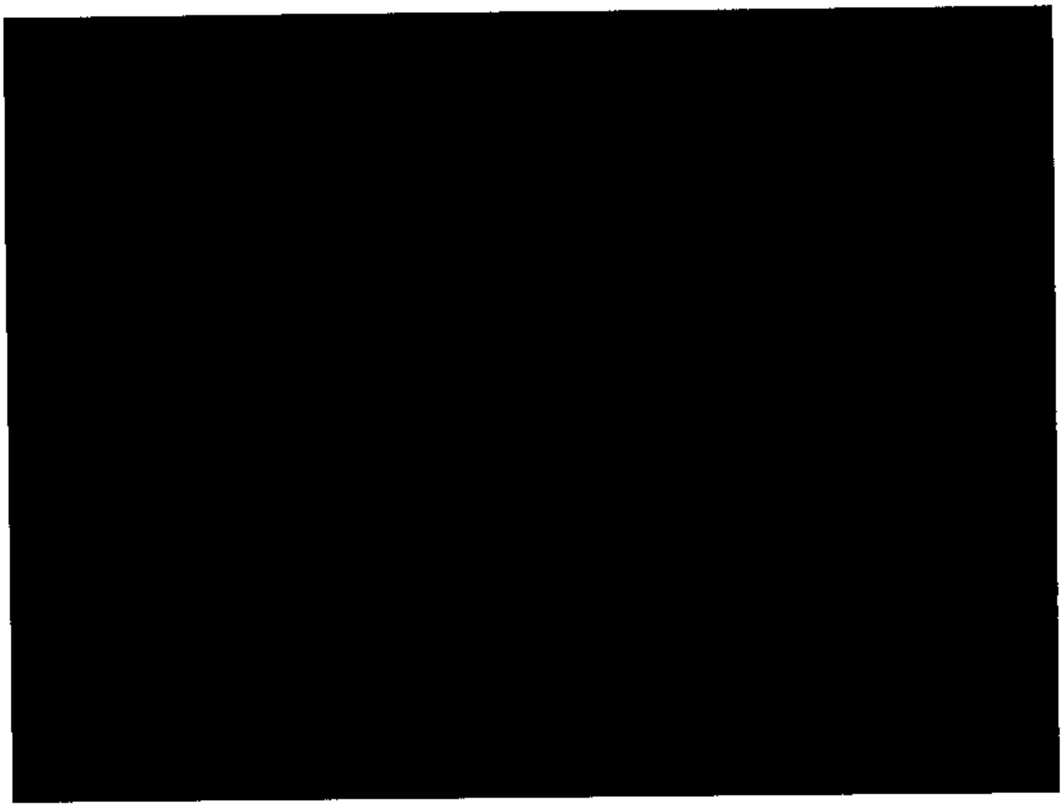


Fig 5.1 Compact Magnetic Mount Antenna

5.4 Mounting

The scientech GPS PCB is encased in a plastic enclosure. The enclosure acts as a protective case. There are four mounting solder tabs on the bottom of the enclosure. When the surface-mount mating connector is used, the mounting tabs may be used for securing the scientech GPS module on the user's PCB. When the cable strip I/O connector scheme is used, the connector side of the scientech GPS module will be faced up and the mounting tabs will be on the top of the module away from PCB.

CHAPTER 6

6. VIBRATION SENSOR

6.1 Introduction

The vibration sensor works on the principle of piezoelectric effect. Piezoelectricity is the ability of crystals and certain ceramic materials to generate a voltage in response to applied mechanical stress. Piezoelectricity was discovered by Pierre Curie and the word is derived from the Greek piezein, which means to squeeze or press. The piezoelectric effect is reversible in that piezoelectric crystals, when subjected to an externally applied voltage, can change shape by a small amount. (For instance, the deformation is about 0.1% of the original dimension in PZT.) The effect finds useful applications such as the production and detection of sound, generation of high voltages, electronic frequency generation, microbalance, and ultra fine focusing of optical assemblies.

6.2 Working

When an accident (vibration) occurs the piezoelectric crystal generates output as voltage. This voltage is in the range of a 250-600mV.

The output from sensor is given as input to the operational amplifier. The operational amplifier operates as an non-inverting amplifier. When input voltage exceeds 200mV the op-amp starts working. Negative feed back is given to the op-amp. The voltage output obtained from op amp is 8V. The ouput is limited to 5V using a zener diode.

Then the 5V is given as input to the NOT gate. So that it produces 0V output. This can be taken and given to the INT0 pin of microcontroller. If interrupt is invoked, the micro controller identifies it as an accident.

CHAPTER 7

7. Microcontroller

7.1 Introduction to Microcontrollers

A Microcontroller consists of a powerful CPU tightly coupled with memory (RAM, ROM, or EPROM), various I/O features such as Serial Ports, Parallel Ports, Timer/Counters, Interrupt Controller, data acquisition interfaces, Analog to Digital Converter (ADC), Digital to Analog Converter (DAC), everything integrated on to a single Silicon Chip.

It does not mean that any microcontroller should have all the above said features on a chip, depending on the need and area of application for which it is designed. The on chip features present in it may or may not include all the individual sections said above. Any microcomputer system requires memory to store a sequence of instructions making up a program, parallel port or serial port for communicating with an external system, timer / counter for control purposes like generating time delays, Baud rate for the serial port, apart from the controlling unit called Central Processing Unit.

7.2 Advantages of Microcontrollers

If a system is developed with a microprocessor, the designer has to go for external memory such as RAM, ROM, or EPROM and peripherals and hence the size of PCB will be large enough to hold all the required peripherals. But the microcontroller has got all these peripheral facilities on a single chip so development of similar system with a microcontroller reduces PCB size and cost of the design.

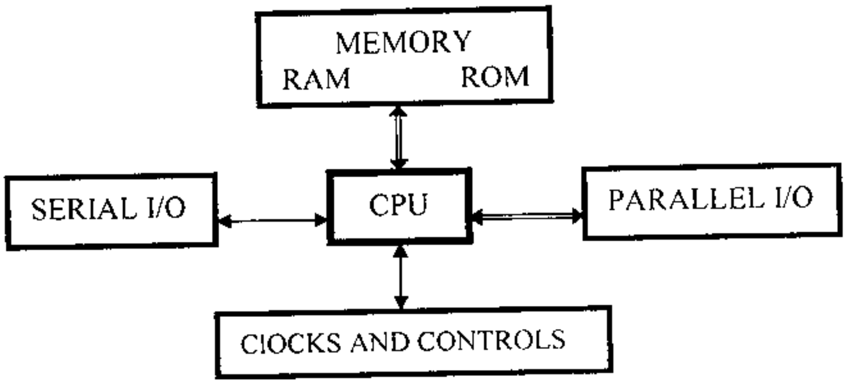


Fig 7.1 Block diagram of a typical microcontroller

The above block diagram shows a typical microcontroller. which is a true computer on a chip. The design incorporates all of the features found in a microprocessor arithmetic and logic unit. program counter, stack pointer and registers. It also has added the other feature needed to make a complete computer: RAM, ROM, serial ports. parallel ports counters and clock circuit.

A microcontroller is a general purpose device, but one that is meant to read data, perform limited calculations on that data and control its environment based on those calculations. The prime use of a microcontroller is to control the operation of a machine using a fixed program that is stored in ROM and that does not change over the lifetime of the system. The microcontroller is connected with getting data from and to its own pins. The architecture and instruction set are optimized to handle data in bit and byte size. Usually a single design that can be used in as many applications as possible is done. The microprocessor design accomplishes this goal by having a very flexible and extensive range of multi byte instructions. Much of the activity in the microprocessor has to do moving code and data from external memory of the Central Processing Unit. The architecture features working registers that can be programmed to take part in the memory access process, and the instruction set is aimed at expediting this activity in order to improve throughput. The pins that connect the microprocessor to external memory are unique, each having a single function.

7.3 About 8051 Microcontrollers

The Intel 8051 is a Harvard architecture single chip microcontroller (µC) which was developed by Intel in 1980 for use in embedded systems. It was extremely popular in the 1980s and early 1990s, but today it has largely been superseded by a vast range of enhanced devices with 8051-compatible processor cores that are manufactured by more than 20 independent manufacturers including Atmel, Infineon Technologies, Maxim IC (via its Dallas Semiconductor subsidiary), NXP (formerly Philips Semiconductor), Winbond, Silicon Laboratories, Texas Instruments and Cypress Semiconductor. Intel's official designation for the 8051 family of µCs is MCS 51.

Intel's original 8051 family was developed using NMOS technology, but later versions, identified by a letter "C" in their name, e.g. 80C51, used CMOS technology and were less power-hungry than their NMOS predecessors - this made them eminently more suitable for battery-powered devices.

7.4 AT89C51 (Atmel)

Features of AT89C51

Key Features	AT89C51
Operating Frequency	0-24MHz
FLASH program memory (8-bit words)	4K
Internal RAM (8-bit words)	128 Bytes
Interrupts	6
I/O Ports	Port 0,1,2,3 (32 lines)
Timers/Counters (16-bit words)	Timer 0,1
Serial Communications	Port 3
Parallel Communication	Port 0,1,2
Packaging	40-pin DIP

Table 7.1 Features of AT89C51

7.5 Pin Diagram of AT89C51

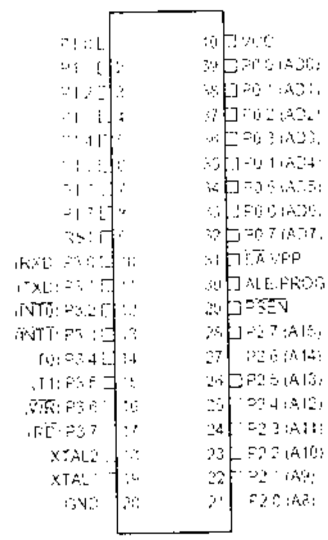


Fig 7.2 Pin diagram of a AT89C51

7.7 Description

The AT89C51 is a low-power, high-performance CMOS 8-bit microcomputer with 4K bytes of Flash programmable and erasable read only memory (PEROM). The endurance of the ROM is 1000 Write/Erase cycles. The device is manufactured using Atmel's high-density nonvolatile memory technology and is compatible with the industry-standard MCS-51 instruction set and pin out. The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C51 is a powerful microcomputer which provides a highly-flexible and cost-effective solution to many embedded control applications.

7.8 External Data Memory

Since the internal data memory in AT89C51 is only 128 bytes, external data memory of 8k has been used. The RAM used is a 28 pin DIP (KM6264B).

Features

- 8Kx8 bit Low Power CMOS Static RAM
- Process Technology: CMOS
- Organization: 8K x 8
- Power Supply Voltage: Single 5V ± 10%
- Low Data Retention Voltage: 2V(Min)
- Three state output and TTL Compatible
- Package Type:28-DIP

7.9 KM6264B Description

The KM6264B family is fabricated by Samsung's advanced CMOS process technology. The family can support various operating temperature ranges and has various package types for user flexibility of system design. The family also support low data retention voltage for battery back-up operations with low data retention current.

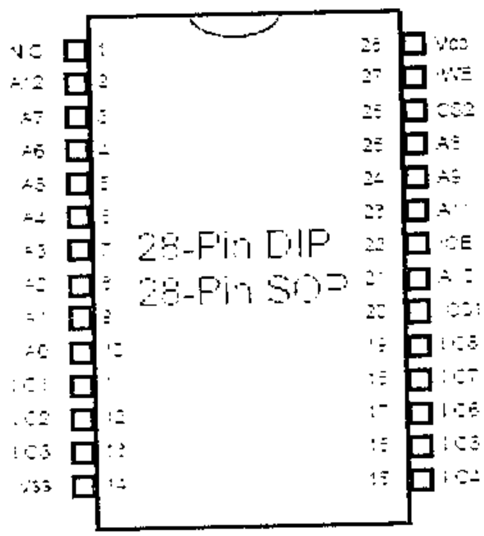


FIG 7.4 KM6264B Pin Description

7.10 Function of microcontroller in our project

In our project the microcontroller is interfaced with Global positioning system to receive the GPS data. The GPS produces datas of different formats but only one particular data format is required for identifying the latitude, longitude and speed. The GPS generates and transmits information at 4800bps. So the I/O port is set at baud rate of 4800 bps. We need the data format starting with $GPGGA so the ASCII value of it is compared when the GPS data is received. Only that data format is stored in the RAM. Consecutive set of datas are not stored, instead only one set of data is stored for each minute. Only three previous data are stored in RAM and the data received at the fourth minute is again stored in the location where first data was stored. This is repeated like a cycle.

Meanwhile if one of the hardware interrupts (INT0) is invoked in the microcontroller then the interrupt service routine is run. The interrupt is set to edge triggering. The program control is transferred to respective interrupt vector location. In that the program execution is transferred to the interrupt module using ACALL. In this a delay of fifteen seconds is produced. So that if the accident occurred is a minor one the driver could manually stop the transmission of accident's information. Else the information stored in the RAM is sent in the transmit pin of microcontroller. Then the data is transmitted over wireless using RF transmitter.

7.11 Microcontroller code

```
SJMP START      * Goto starting of the program *
NOP             * No operation *
ACALL INT       * Goto interrupt subroutine when interrupt occurs *
NOP
NOP
NOP
NOP
NOP
NOP
RETI             *return to main program after interrupt is complete *
NOP
NOP
NOP
```

```
NOP
NOP
NOP
NOP
RETI
NOP
NOP
NOP
NOP
NOP
NOP
RETI
NOP
NOP
NOP
NOP
NOP
NOP
NOP
RETI
NOP
NOP
NOP
NOP
NOP
NOP
NOP
MOV SP,#30H                 * Move stack pointer to 30h *
MOV IE,#85H                 * Enable interrupt int 0 and int 1*
MOV IP,#00H                 * No interrupt priority *
MOV TCON,#0AH               * One interrupt edge triggered and one
                            level triggered *
MOV DPTR,#0000H             * Move 0000h to dptr *
MOV A,#30H                  * Move ascii value of 0 to a *
MOV R1,#03H                 * Move 3 to r1*
TWO: MOV R0,#0DCH           * Move dc to r0 *
ONE: MOVX @DPTR,A           * Write '0' to all data locations one by
                            one *
INC DPTR
DJNZ R0,ONE
DJNZ R1,TWO
MOV R1,#00H
MOV R0,#00H
```

```
MOV R2,#01H
MOV DPTR,#0000H
MOV TMOD,#20H               * Timer1 auto reload mode *
MOV TH1,#0FAH               * Set 4800 baud rate *
MOV SCON,#50H               * 8 bit 1 stop bit no parity *
SETB TR1                    * Enable timer *
HERE: JNB RI,HERE           * Receive one byte *
MOV A,SBUF                  * Move received byte to accumulator *
CJNE R0,#00H,CHECK1
CJNE A,#24H,CLEAR           * Checking for $ *
SJMP KUT
CHECK1: CJNE R0,#01H,CHECK2
CJNE A,#47H,RES1            * Checking for G *
SJMP KUT
CHECK2: CJNE R0,#02H,CHECK3
CJNE A,#50H,RES1            * Checking for P *
SJMP KUT
CHECK3: CJNE R0,#03H,CHECK4
CJNE A,#47H,RES1            * Checking for G *
SJMP KUT
CHECK4: CJNE R0,#04H,CHECK5
CJNE A,#47H,RES1            * Checking for G *
SJMP KUT
CHECK5: CJNE R0,#05H,CHECK6
CJNE A,#41H,RES1            * Checking for A *
SJMP KUT
CHECK6:  CJNE R0,#06H,RES1
CHECK6I: CJNE A,#2AH,GETDATA
MOVX @DPTR,A
INC DPTR
CLR RI
HERE2: JNB RI,HERE2
MOV A,SBUF
MOVX @DPTR,A
INC DPTR
CLR RI
HERE3: JNB RI,HERE3
MOV A,SBUF
MOVX @DPTR,A
INC DPTR
ACALL SHIFT                 * Shift the memory locations *
ACALL BLINK                 * Introduce delay *
RES1: MOV R0,#0FFH          * Reset module *
```

```
KUT: INC R0
CLEAR: CLR RI
SJMP HERE
GETDATA: MOVX @DPTR,A          * Module to receive the GPS
                               data*
INC DPTR
CLR RI
HERE1: JNB RI,HERE1
MOV A,SBUF
SJMP CHECK6I
SHIFT: CJNE R2,#00H,JEM1       * Shifting memory locations after
                               receiving in the first minute *
INC R2
MOV DPTR,#0000H                * First memory location *
RET
JEM1: CJNE R2,#01H,JEM2
INC R2
MOV DPTR,#00DCH                * Second memory location *
RET
JEM2: MOV R2,#00H
MOV DPTR,#01B8H                * Third memory location *
RET
BLINK: MOV A,#30H
MOV B,#75H
ACALL SOFTIME                  * Call delay sub routine *
RET                            * Ret after delay is complete *
SOFTIME: PUSH 07H              * This is the delay sub routine *
PUSH ACC
ORL A,B
CJNE A,#00H,OK
POP ACC
SJMP DONE
OK: POP ACC
TIMER: MOV R7,#0A6H
ONEMIL: NOP
NOP
NOP
NOP
DJNZ R7,ONEMIL
NOP
NOP
DEC A
CJNE A,#0FFH,NOROLL
```

```
DEC B
NOROLL: CJNE A,#00H,TIMER
CJNE A,B,TIMER
DONE: POP 07H
RET                         * Return after delay subroutine is complete *
INT:                        * This is interrupt sub routine *
MOV A,#34H
MOV B,#21H
ACALL SOFTIME
MOV SBUF,54#H          * Move T to SBUF *
HERE4: JNB TI,HERE4 * Send data *
CLR TI
MOV SBUF,#4EH          * Move N to SBUF *
HERE5: JNB TI,HERE5 * Send data *
CLR TI
MOV SBUF,#33H          * Move 3 to SBUF *
HERE6: JNB TI,HERE6 * Send data *
CLR TI
MOV SBUF,#38H          * Move 8 to SBUF *
HERE7: JNB TI,HERE7 * Send data *
CLR TI
MOV SBUF,#4AH          * Move J to SBUF *
HERE8: JNB TI,HERE8 * Send data *
CLR TI
MOV SBUF,#34H          * Move 4 to SBUF *
HERE9: JNB TI,HERE9 * Send data *
CLR TI
MOV SBUF,#39H          * Move 9 to SBUF *
HERE10: JNB TI,HERE10 * Send data *
CLR TI
MOV SBUF,#39H          * Move 9 to SBUF *
HERE11: JNB TI,HERE11 * Send data *
CLR TI
MOV SBUF,#31H          * Move 1 to SBUF *
HERE12: JNB TI,HERE12 * Send data *
CLR TI
MOV DPTR,#0000H      * Send all the GPS data out *
MOV R3,#03H
KS: MOV R4,#0DCH
NK: MOVX A,@DPTR
MOV SBUF,A
KCT: JNB TI,KCT
CLR TI
```

```
INC DPTR
DJNZ R4,NK
DJNZ R3,KS
END1: SJMP END1
END                         * End of Program *
```

CHAPTER 8

8. SERIAL COMMUNICATION

8.1 Introduction:

Communication refers to meaningful exchange of information between the communicating entities. Therefore, in communication, it is concerned with all issues relating to exchange of information in the form of a dialog, e.g., dialog discipline, interpretation of messages, and acknowledgements.

8.2 Issues in Communication:

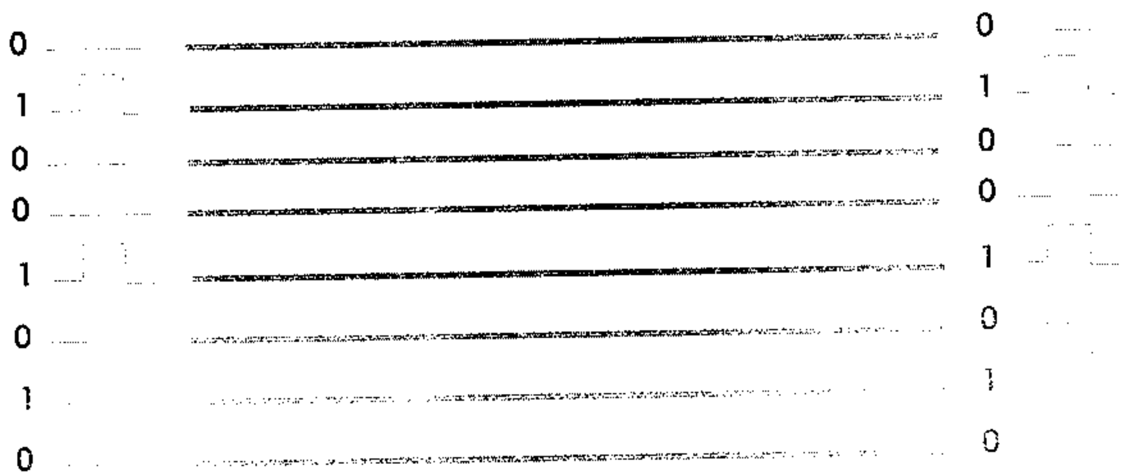
- Types of data communication
- Modes of data transmission
- Transmission Media
- Serial Interface standards

Data Transmission:

- There is always the need to exchange data, commands and other control information between a computer and its terminals, or between two computers.
- This information is in the form of bits. Data transmission refers to the movement of bits over some physical medium connecting two or more digital devices.
- The two options of transmitting the bits are:
  - ✓ Parallel Communication
  - ✓ Serial Communication

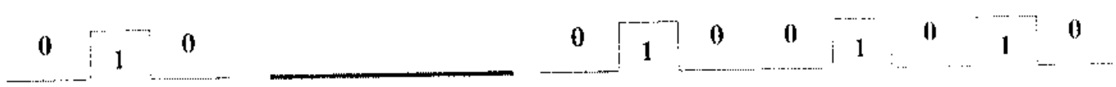
Parallel Transmission:

If more than 1 bit of information is transmitted over the data transmission medium at a time then it is considered as a parallel communication.



Serial Transmission:

- The physical connection determines how many bits (1's or 0's) can be transmitted in a single instance of time.
- If only 1 bit of information can be transmitted over the data transmission medium at a time then it is considered a serial communication.



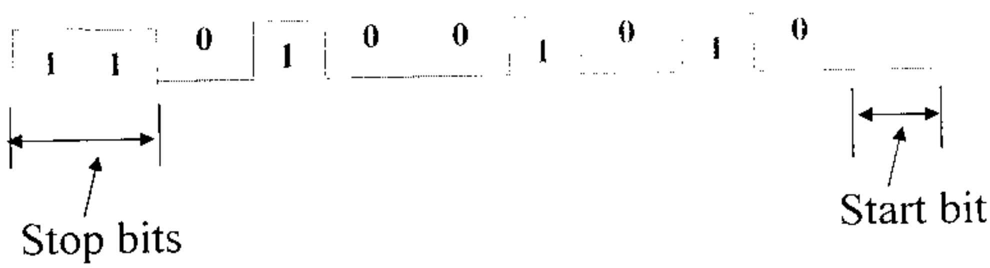
8.3 Modes of Serial Data Transmission:

Asynchronous Transmission:

- An action is called asynchronous when the agent performing an action does so whenever it wishes.
- Asynchronous transmission refers to the case when the sending Node commences transmission of bytes at any instant of time.
- Only one byte is sent at a time and there is no time relation between consecutive bytes, i.e., after sending a byte, the next byte can be sent after an arbitrary delay.

Start & Stop Bits In Asynch Serial Communication:

- Due to the arbitrary delay between consecutive bytes, the time occurrences of the clock pulses at the receiving end need to be synchronized repeatedly for each byte.
- This is usually achieved by providing two extra bits, a Start bit at the beginning and a Stop bit at the end of a byte.



7.6 Block diagram of AT89C51

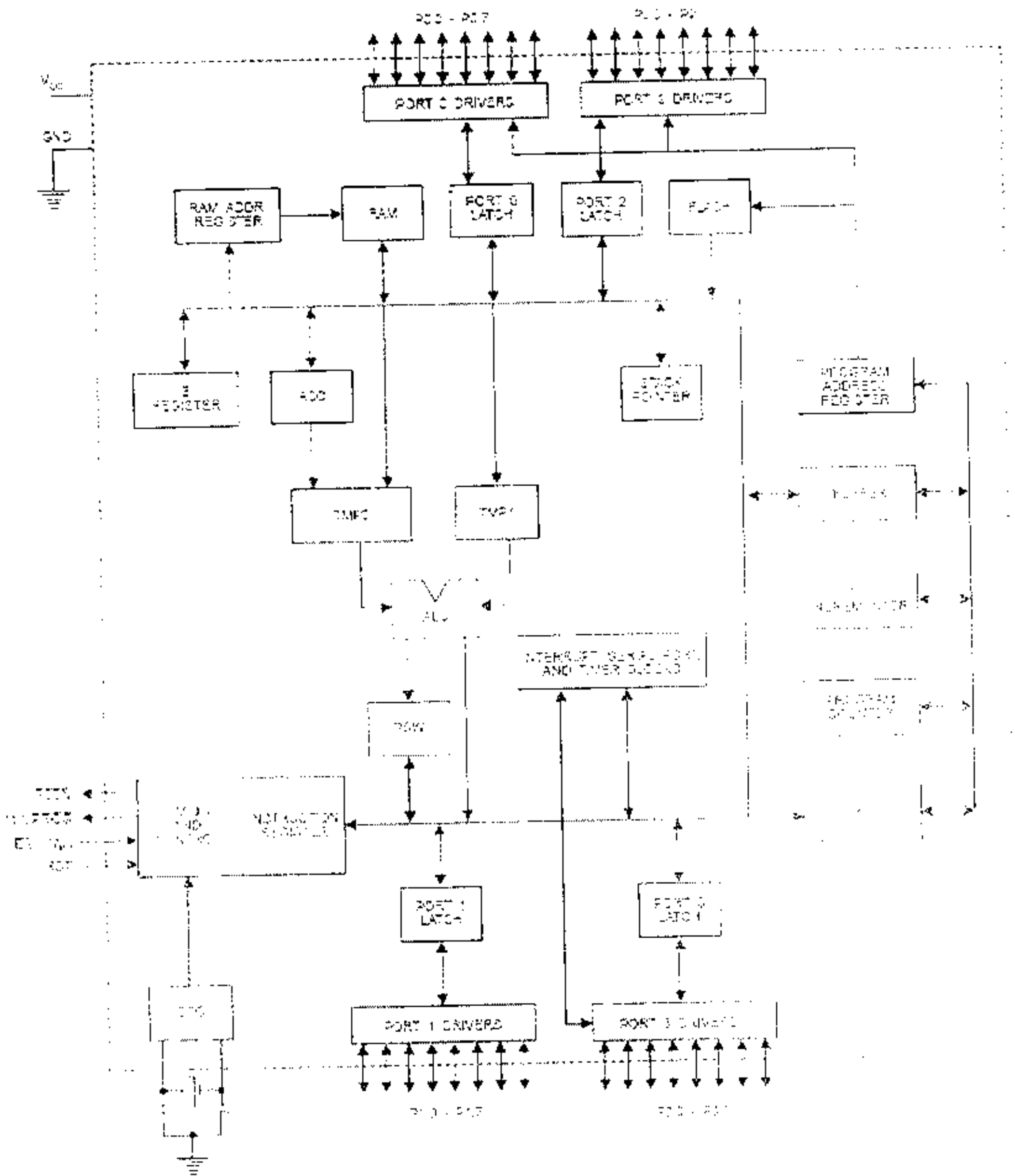


Fig 7.3 Block diagram of AT89C51

Synchronous Serial Transmission:

- A synchronous action, unlike an asynchronous action, is carried out under the control of a timing source.
- In synchronous transmission, bits are always synchronized to a reference clock irrespective of the bytes they belong to. There are no Start or Stop bits.
- Bytes are transmitted in a Block in a continuous stream of bits.
- Continuous transmission of bits enables the receiver to extract the clock from the incoming electrical signal.

8.4 RS-232 Standard:

- ✓ Most widely used serial I/O interfacing standard.
- ✓ Used in PCs and numerous types of equipments.
- ✓ It is not compatible with I/O voltage levels of TTL logic family

Voltage Levels in RS-232:

- ✓ Logic high (1) represented as -3 to -25V.
- ✓ Logic low (0) represented as +3 to +25V.
- ✓ -3 to +3v not defined.

RS-232 Interface:

RS-232 was introduced in 1960, and is currently the most widely used communication protocol. It is simple, inexpensive to implement, and though relatively slow; it is more than adequate for most simple serial communication devices such as keyboards and mice.

- RS-232 is a single-ended data transmission system, which means that it uses a single wire for data transmission.
- Since useful communication is generally two way, a two-wire system is employed, one to transmit and one to receive.
- Signals are processed by determining whether they are positive or negative when compared with a ground.

PC's Serial Port:

IBM PC/compatible computers based on x86 (8086, 286, 386, 486 and Pentium) microprocessors normally have two COM (read: COMMUNICATION) ports. Both COM ports have RS – 232 type connectors. Many PCs use one each of the DB – 25 and DB – 9 RS – 232 connectors. The COM ports are designated as COM 1, COM 2 etc..... One can utilize COM X port of a PC for serial communication experiments, where X designates a particular port.

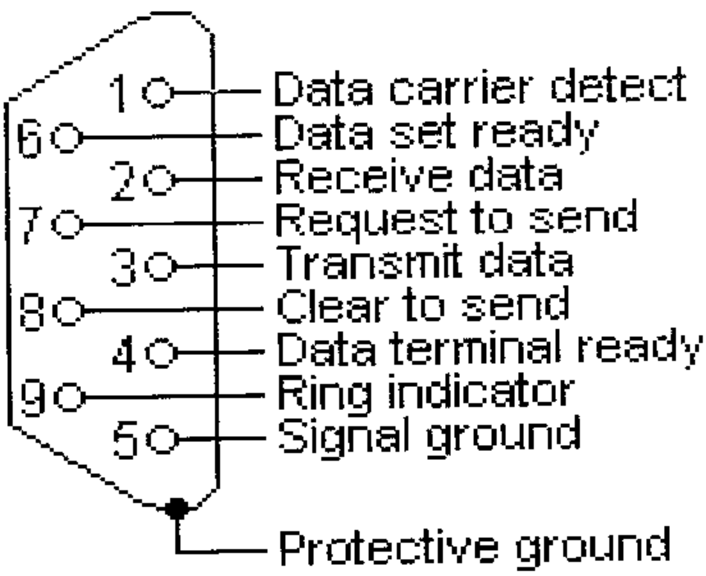


Fig 8.1 RS-232 Pin Diagram

PIN NO	SIGNAL
1	Carrier Detect (CD) (from DCE) Incoming signal from a modem
2	Received Data (RD) Incoming Data from a DCE
3	Transmitted Data (TD) outgoing Data to a DCE
4	Data Terminal Ready (DTR) Outgoing handshaking signal
5	Signal Ground Common reference voltage
6	Data Set Ready (DSR) Incoming handshaking signal
7	Request To Send (RTS) Outgoing flow control signal
8	Clear To Send (CTS) Incoming flow control signal
9	Ring Indicator (RI) (from DCE) Incoming signal from a modem

Table 8.1 RS-232 Pin Details

RS232 Control Lines:

The RS232 standard describes the functions carried out by several control signals between the DTE and the DCE. The following control signals implement most of the important functions of an R232 DTE to DCE link.

Data terminal ready (DTR):

This is a signal from the DTE to the DCE. When asserted, DTR indicates that the DTE is ready to accept data from the DCE. In systems with a modem, it maintains the connection and keeps the channel open. If DTR is negated, the communication path is broken. In everyday terms, negating DTR is the same as hanging up a phone.

Data set ready (DSR):

This is a signal from the DCE to the DTE, which indicates the readiness of the DCE. When this signal is asserted, the DCE is able to receive from the DTE. DSR indicates that the DCE (usually a modem) is switched on and is in its normal functioning mode (as opposed to its self-test mode).

Request to send (RTS):

This is a signal from the DTE to the DCE. When asserted, RTS indicates to the DCE that the DTE wishes to transmit data to it.

Clear to send (CTS):

This is a signal from the DCE to the DTE and, when asserted, indicates that the DCE is ready to receive data from the DTE.

Rxd:

It is used to receive the serial data transmitted from the external Microcontroller or circuit.

Txd:

It is used to transmit the serial data to the external Microcontroller or circuit from the PC.

PC Connection to RS232:

The RS232 standard is not TTL compatible; therefore, it requires the line driver such as MAX232 chip to convert the RS232 voltage levels to TTL levels and vice versa.

8.5 MAX232:

RS232 does not use the conventional 0 and 5v implemented in TTL and CMOS designs. Drivers have to supply +5 to +15v for logic 0 and -5 to -15v for logical; this is performed by IC MAX232. This means that extra power supplies are needed to drive the RS232 voltage levels. Typically a +12 and a -12v power supply are used to drive the RS232 outputs. The first level charge pump essentially doubles the standard +5v power supply to provide the voltage level necessary for driving logic0. A second charge pump inverts this voltage and provides the voltage level necessary for driving logic1. These two charge pumps allow the RS232 interface products to operate from a single +5v supply.

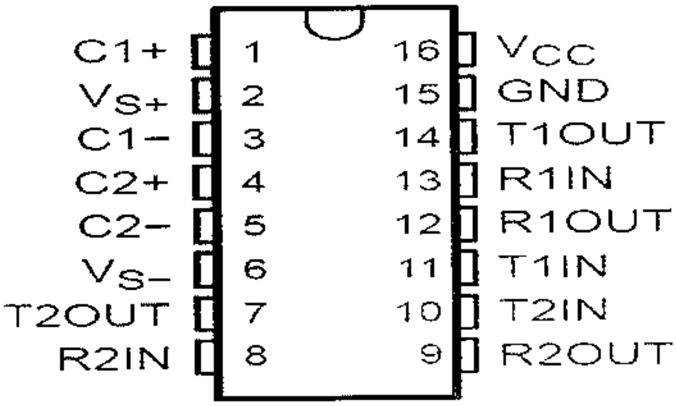


Fig 8.2 MAX232 Pin Diagram

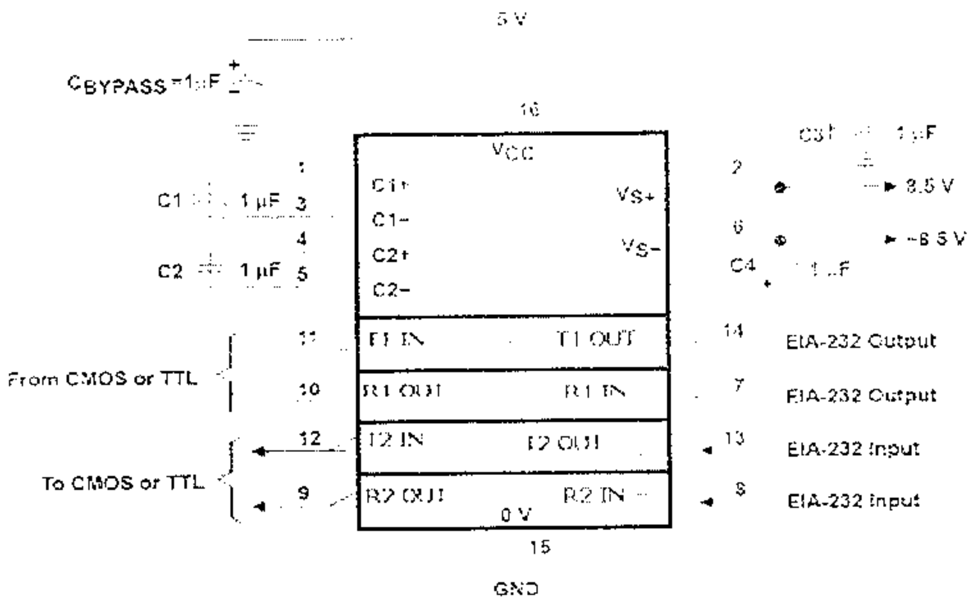


Fig 8.3 MAX232 Pin Diagram With Details

No.	Name	Purpose	Signal Voltage
1	C1+	+ connector for capacitor C1	capacitor should stand at least 16V
2	V+	output of voltage pump	+10V
3	C1-	- connector for capacitor C1	capacitor should stand at least 16V
4	C2+	+ connector for capacitor C2	capacitor should stand at least 16V
5	C2-	- connector for capacitor C2	capacitor should stand at least 16V
6	V-	output of voltage pump / inverter	-10V
7	T2out	Driver 2 output	RS-232
8	R2in	Receiver 2 input	RS-232
9	R2out	Receiver 2 output	TTL
10	T2in	Driver 2 input	TTL
11	T1in	Driver 1 input	TTL
12	R1out	Receiver 1 output	TTL
13	R1in	Receiver 1 input	RS-232
14	T1out	Driver 1 output	RS-232
15	GND	Ground	0V
16	VCC	Power supply	+5V

Table 8.2 MAX232 DIP Package Pin Layout

CHAPTER 9

9. PARENT TERMINAL

9.1 Introduction

Parent terminal here means a computer. The parent terminal is connected to the RF Rx using a RS232 Cable. The GPS data received is manipulated and processed in the computer and the required output is obtained. Visual basic programming is used for everything.

9.2 RF Transmitter-Receiver

RF TX – RX uses 433.3 Mhz frequency for data transmission from the microcontroller to the parent terminal. The tx – rx uses ON-OFF key modulation for transmission and reception. The microcontroller sends data from transmit pin is directly fed in to the data in pin of the transmitter. It is highly immune to noise signals and it has a gain of almost 65dB. The VCC for both transmitter and receiver is +5V. For attaining exact frequency for transmission the transmitter uses PLL circuit for frequency locking. The range of the RF tx-rx is almost 300 feet. The system we developed is only a prototype. If a real system is developed to fix in the car then a transmitter for very long range can be deployed or satellite can be used for data transmission. In the receiver section the output from the data out pin of receiver is given to the line driver so as to feed in to the RS232 cable.

9.3 Visual Basic coding: An overview

First of all a login password is asked using which a user can sign in. After successful sign in the application is started. The Text Aloud software and Globe7 are also started. Then the application is ready and waits for the GPS data. As soon as an accident occurs the GPS data is received. The GPS data is parsed and the latitude, longitude, altitude information is separated. The car no is identified based on which the colour, model are identified from the database. The velocity of the vehicle is also calculated. The nearest hospital is identified using the database available and the corresponding hospital and police station are called using the phone numbers available in the database.

9.4 VB Code

```
Option Explicit
  Dim i, j, ind As Integer
  Dim len, big, small, cal As Integer
  Dim qual(2), nos(2), hdop(2) As Integer
  Dim tim(2), hr(2), min(2), sec(2) As Integer
  Dim lat(2), lon(2), alt(2), geo(2)  As Double
  Dim model, carno, phno As String
  Dim a, b As Double
  Dim gps As String

Private Sub Form_Load()
  Form1.Caption = "Enter the Password"
  Form1.Height = 800
  Form1.Width = 3000
  Form1.WindowState = 0
  Form1.Left = 250
  Form1.Top = 250
  Text2.Height = 800
```

```
  Text2.Width = 4000
  Text2.PasswordChar = "*"
  MSComm1.CommPort = 1
  MSComm1.Settings = "4800,n,8,1"
  MSComm1.PortOpen = True
  j = 1
  Timer1.Enabled = False
  Timer1.Interval = 10000
  a = Shell("C:\Program Files\TextAloud\textaloudmp3.exe",
  vbNormalFocus)
  b = Shell("C:\Program Files\Globe7\globe7.exe", vbNormalFocus)
End Sub

Private Sub Text2_KeyPress(KeyAscii As Integer)
  If KeyAscii = 13 Then
    Open "c:\pass" For Input As #1
      While Not EOF(1)
        Line Input #1, temp$
        If temp$ = Text2.Text Then
          Text2.Visible = False
          Text1.Visible = True
          Command2.Visible = True
          Form1.WindowState = 2
          End
        Else
          MsgBox "Wrong password type again"
          Text2.Text = ""
        End If
      Wend
```

```
    Close #1
  End If
End Sub

Private Sub MSComm1_OnComm()
  If MSComm1.CommEvent = comEvReceive Then
    Text1.Text = "An accident has occurred"
    Text1.BackColor = &HFF&
    gps=MSComm1.input
      For i=0 to 2 Step 1
        len = InStr(gps, ",")
        gps = Mid(gps,len)
        len = InStr(gps,",")
        tim(i) = Left(gps,len)
        gps = Mid(gps,len)
        len = InStr(gps,",")
        lat(i)= Left(gps,len)
        gps = Mid(gps,len)
        len = InStr(gps,",")
        lon(i) = Left(gps,len)
        gps = Mid(gps,len)
        len = InStr(gps,",")
        qual(i) = Left(gps,len)
        gps = Mid(gps,len)
        len = InStr(gps,",")
        nos(i) = Left(gps,len)
        gps = Mid(gps,len)
        len = InStr(gps,",")
        hdop(i) = Left(gps,len)
```

```
      gps = Mid(gps,len)
      len = InStr(gps,",")
      alt(i) = Left(gps,len)
      gps = Mid(gps,len)
      len = InStr(gps,",")
      geo(i) = Left(gps,len)
      gps = Mid(gps,len)
    Next i
OPEN "c:\carno.txt" FOR INPUT as #1
WHILE NOT EOF(1)
   LINE INPUT #1, temp$
   len = InStr(temp$,",")
   carno = Left(temp$,len)
   If carno = gps Then
      gps = Mid(temp$,len)
      len = InStr(temp$,",")
      col = Left(temp$,len)
      gps = Mid(temp$,len)
      len = InStr(temp$,",")
      model = Left(temp$,len)
   End If
WEND
CLOSE #1
For i = 0 to 2 Step 1
   hr(i) = tim(i)/10000
   min(i) = (tim(i)/100) Mod 100
   sec(i) = tim(i) Mod 100
Next i
big = tim(0)
```

```
    For i = 1 to 2 Step 1
      If big<tim(i) Then ind=i
    Next i
    OPEN "c:\db.txt" FOR INPUT as #1
    WHILE NOT EOF(1)
      LINE INPUT #1, temp$1
      LINE INPUT #1, temp$2
      LINE INPUT #1, temp$3
      cal=sqr((abs(temp$1 - lat(ind))) * (abs(temp$1 - lat(ind))) +
      (abs(temp$2 - lon(ind))) * (abs(temp$2  -lon(ind))))
      If cal<small Then phno = temp$3
     WEND
     CLOSE #1
     Timer1.Enabled = True
  End If
End Sub

Private Sub Timer1_Timer()
  Timer1.Enabled = False
  AppActivate "TextAloud"
  If j = 1 Then
    SendKeys "^(n)"
    SendKeys "Emergency! An accident has taken place at"
    SendKeys lat(ind)
    SendKeys " degrees north,"
    SendKeys lon(ind)
    SendKeys " degrees east and "
    SendKeys alt(ind)
    SendKeys " Height. "
```

```
  SendKeys "The car is a "
  SendKeys col
  SendKeys "colour"
  SendKeys model
  SendKeys "whose number is"
  SendKeys carno
  AppActivate "Globe7"
  SendKeys phno
  SendKeys "~"
End If
SendKeys "^({F3})"
SendKeys "^(%(s))"
SendKeys "^(%(s))"
Timer1.Interval = 32000
Timer1.Enabled = True
j = 2
ElseIf j = 2 Then
  Timer1.Enabled = False
  SendKeys "%(s)u"
  SendKeys "{tab}{tab}{tab}~"
  Timer1.Interval = 5000
  Timer1.Enabled = True
  j = 3
ElseIf j = 3 Then
  SendKeys "%(f)x"
  Me.Visible = True
  AppActivate "GPS monitor"
  SendKeys "%({F4})"
End If
```

```
End Sub

Private Sub Command2_Click()
  End
End Sub
```

CHAPTER 10

10. CONCLUSION

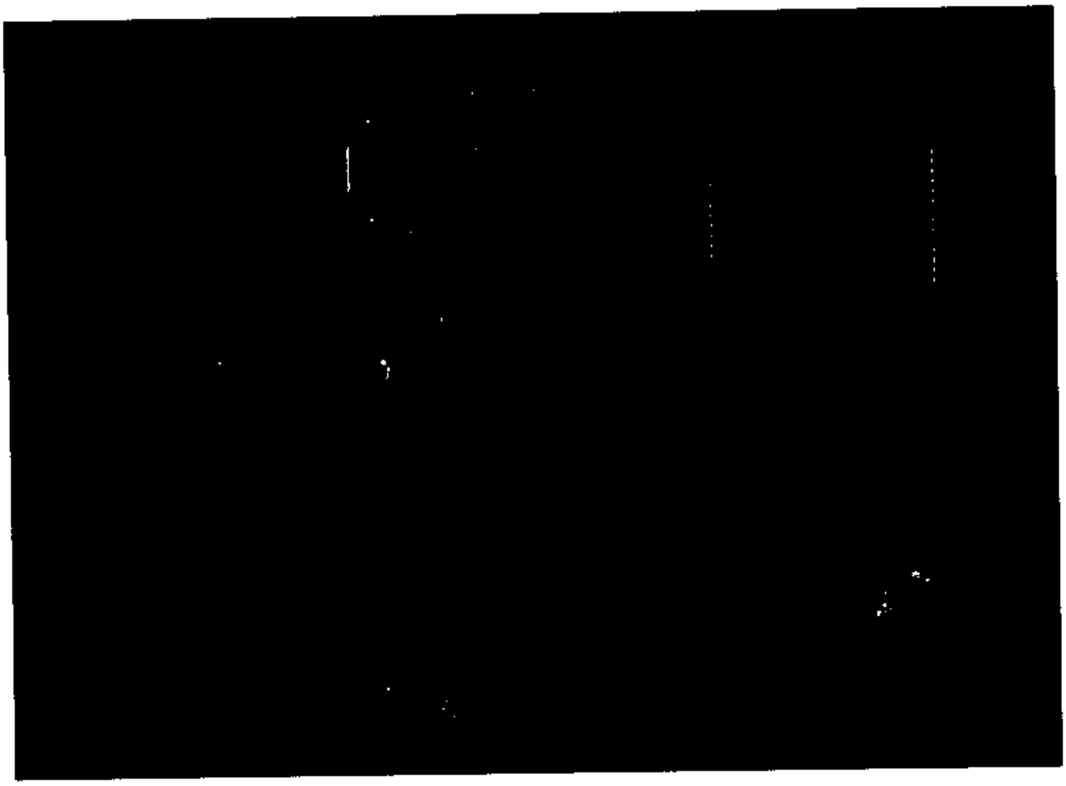
10.1 Result

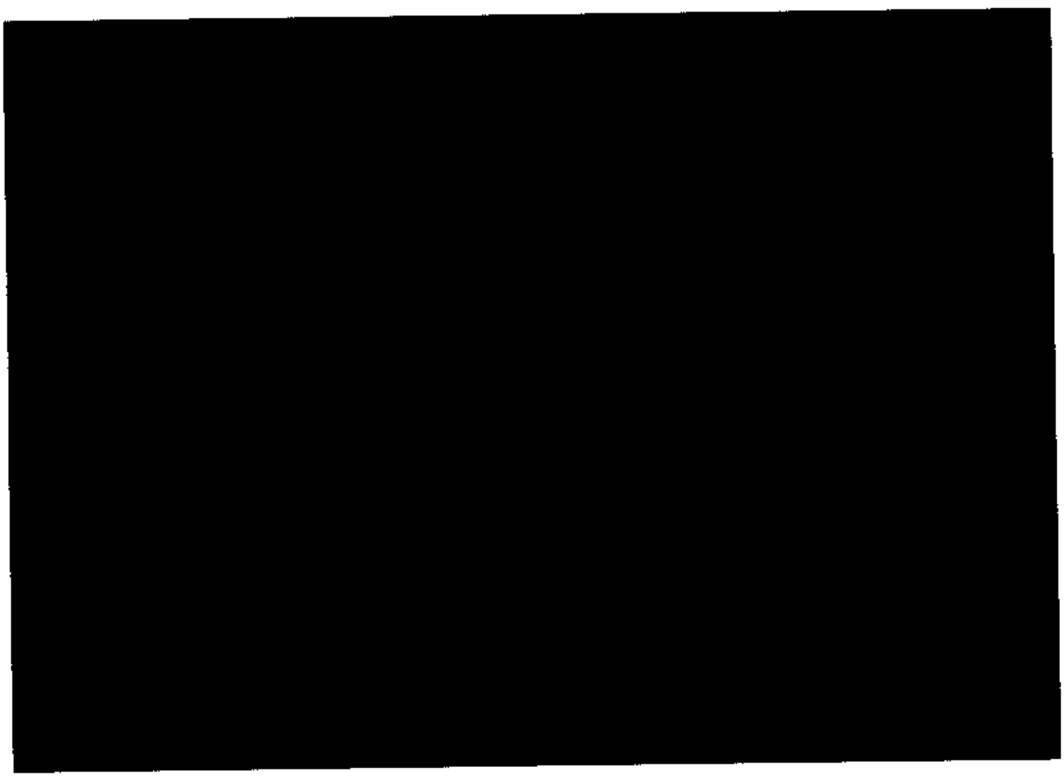
The results of the project are displayed in the figures below. All the modules worked and the required output was obtained. This was checked in different places in the college. The phone numbers corresponding to those locations were called automatically, with the respective data. This could be programmed for different hospitals and police station around the city. This project brings about an effective accident detection and handling scheme.

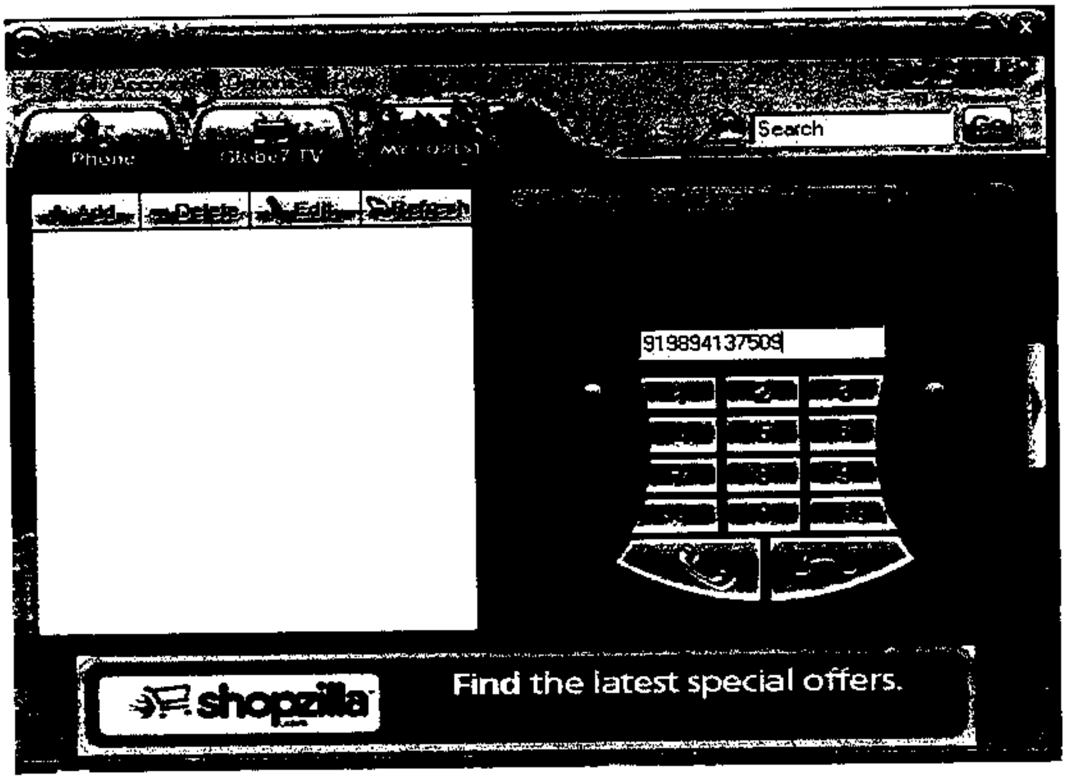
10.2 Future Enhancements

- Google K-maps can be used to show the exact location.
- Super sense technology GPS receiver can be used
- Artificial intelligence programming can be done to locate the nearest hospital
- Database of the accidents occurred in that area can be created automatically
- Many different sensors can be deployed in different location of the car so that puncture detection and some other faults can be detected and informed to nearest garage.

10.3 Output of the Project







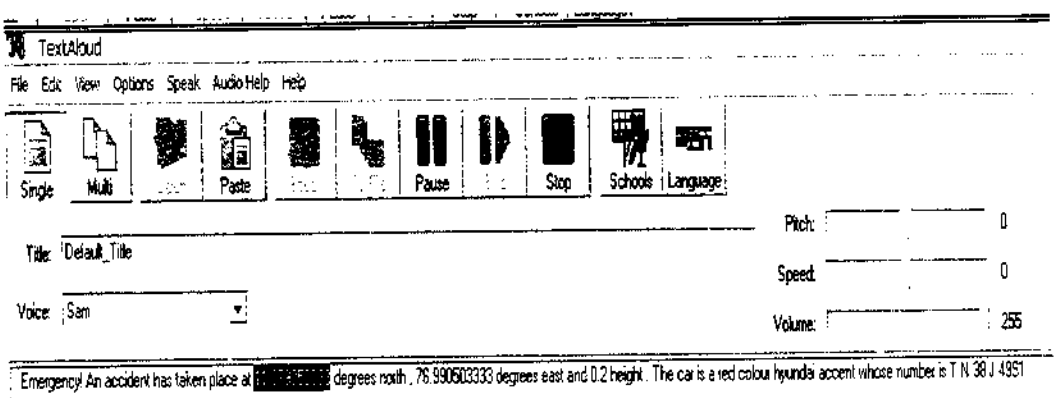
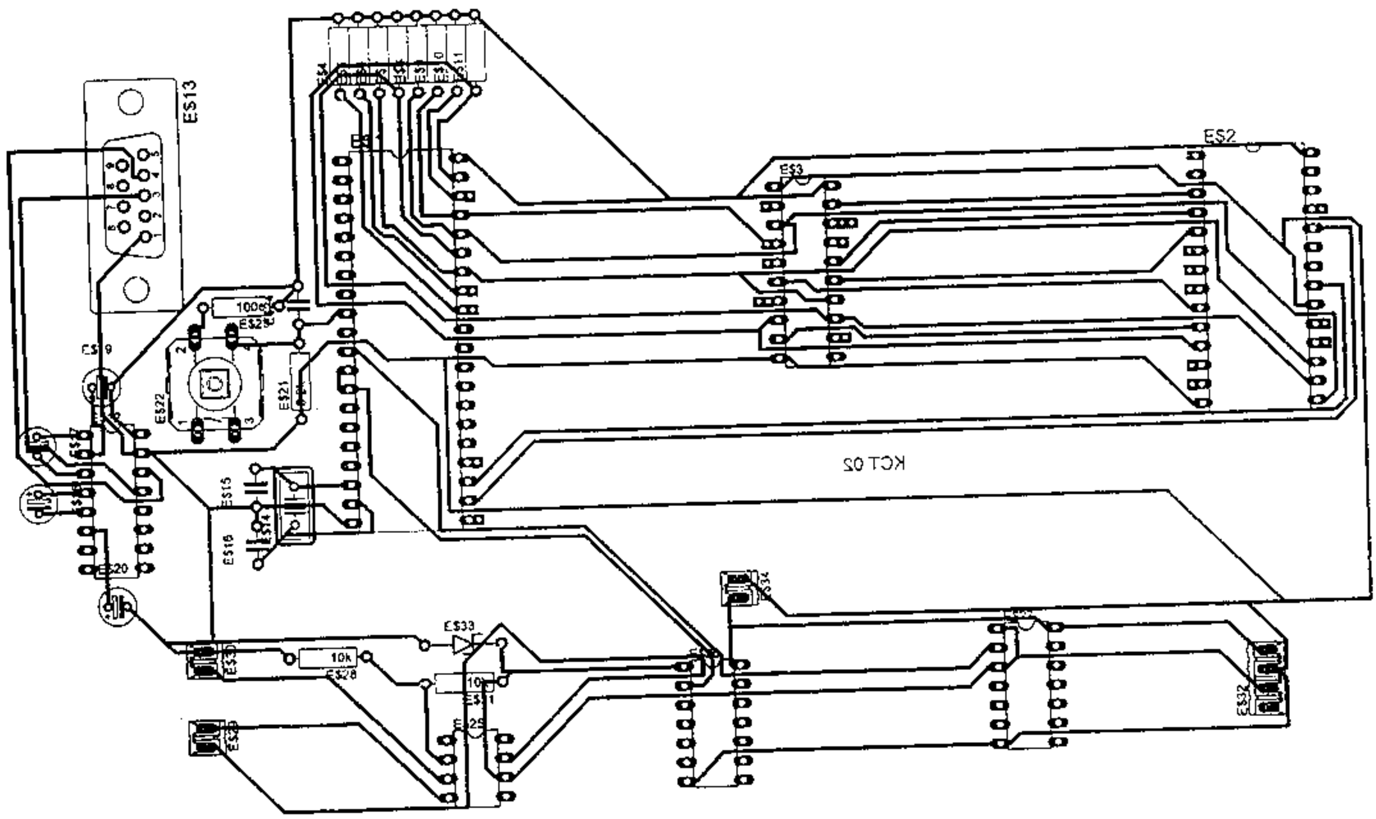


FIG 10.1 Output of the project

APPENDIX

APPENDIX

PCB DESIGN LAYOUT



Instruction Set Summary

	0	1	2	3	4	5	6	7
0	NOP	JBC bit,rel [3B, 2C]	JB bit, rel [3B, 2C]	JNB bit, rel [3B, 2C]	JC rel [2B, 2C]	JNC rel [2B, 2C]	JZ rel [2B, 2C]	JNZ rel [2B, 2C]
1	AJMP (P0) [2B, 2C]	ACALL (P0) [2B, 2C]	AJMP (P1) [2B, 2C]	ACALL (P1) [2B, 2C]	AJMP (P2) [2B, 2C]	ACALL (P2) [2B, 2C]	AJMP (P3) [2B, 2C]	ACALL (P3) [2B, 2C]
2	LJMP addr16 [3B, 2C]	LCALL addr16 [3B, 2C]	RET [2C]	RETI [2C]	ORL dir, A [2B]	ANL dir, A [2B]	XRL dir, a [2B]	ORL C, bit [2B, 2C]
3	RR A	RRC A	RL A	RLC A	ORL dir, #data [3B, 2C]	ANL dir, #data [3B, 2C]	XRL dir, #data [3B, 2C]	JMP @A + DPTR [2C]
4	INC A	DEC A	ADD A, #data [2B]	ADDC A, #data [2B]	ORL A, #data [2B]	ANL A, #data [2B]	XRL A, #data [2B]	MOV A, #data [2B]
5	INC dir [2B]	DEC dir [2B]	ADD A, dir [2B]	ADDC A, dir [2B]	ORL A, dir [2B]	ANL A, dir [2B]	XRL A, dir [2B]	MOV dir, #data [3B, 2C]
6	INC @R0	DEC @R0	ADD A, @R0	ADDC A, @R0	ORL A, @R0	ANL A, @R0	XRL A, @R0	MOV @R0, @data [2B]
7	INC @R1	DEC @R1	ADD A, @R1	ADDC A, @R1	ORL A, @R1	ANL A, @R1	XRL A, @R1	MOV @R1, #data [2B]
8	INC R0	DEC R0	ADD A, R0	ADDC A, R0	ORL A, R0	ANL A, R0	XRL A, R0	MOV R0, #data [2B]
9	INC R1	DEC R1	ADD A, R1	ADDC A, R1	ORL A, R1	ANL A, R1	XRL A, R1	MOV R1, #data [2B]
A	INC R2	DEC R2	ADD A, R2	ADDC A, R2	ORL A, R2	ANL A, R2	XRL A, R2	MOV R2, #data [2B]
B	INC R3	DEC R3	ADD A, R3	ADDC A, R3	ORL A, R3	ANL A, R3	XRL A, R3	MOV R3, #data [2B]
C	INC R4	DEC R4	ADD A, R4	ADDC A, R4	ORL A, R4	ANL A, R4	XRL A, R4	MOV R4, #data [2B]
D	INC R5	DEC R5	ADD A, R5	ADDC A, R5	ORL A, R5	ANL A, R5	XRL A, R5	MOV R5, #data [2B]
E	INC R6	DEC R6	ADD A, R6	ADDC A, R6	ORL A, R6	ANL A, R6	XRL A, R6	MOV R6, #data [2B]
F	INC R7	DEC R7	ADD A, R7	ADDC A, R7	ORL A, R7	ANL A, R7	XRL A, R7	MOV R7, #data [2B]

Note Key: [2B] = 2 Byte, [3B] = 3 Byte, [2C] = 2 Cycle, [4C] = 4 Cycle, Blank = 1 byte/1 cycle

Instruction Set Summary (Continued)

	8	9	A	B	C	D	E	F
0	SJMP REL [2B, 2C]	MOV DPTR,# data 16 [3B, 2C]	ORL C, /bit [2B, 2C]	ANL C, /bit [2B, 2C]	PUSH dir [2B, 2C]	POP dir [2B, 2C]	MOVX A, @DPTR [2C]	MOVX @DPTR, A [2C]
1	AJMP (P4) [2B, 2C]	ACALL (P4) [2B, 2C]	AJMP (P5) [2B, 2C]	ACALL (P5) [2B, 2C]	AJMP (P6) [2B, 2C]	ACALL (P6) [2B, 2C]	AJMP (P7) [2B, 2C]	ACALL (P7) [2B, 2C]
2	ANL C, bit [2B, 2C]	MOV bit, C [2B, 2C]	MOV C, bit [2B]	CPL bit [2B]	CLR bit [2B]	SETB bit [2B]	MOVX A, @R0 [2C]	MOVX @R0, A [2C]
3	MOVC A, @A + PC [2C]	MOVC A, @A + DPTR [2C]	INC DPTR [2C]	CPL C	CLR C	SETB C	MOVX A, @R1 [2C]	MOVX @R1, A [2C]
4	DIV AB [2B, 4C]	SUBB A, #data [2B]	MUL AB [4C]	CJNE A, #data, rel [3B, 2C]	SWAP A	DA A	CLR A	CPL A
5	MOV dir, dir [3B, 2C]	SUBB A, dir [2B]		CJNE A, dir, rel [3B, 2C]	XCH A, dir [2B]	DJNZ dir, rel [3B, 2C]	MOV A, dir [2B]	MOV dir, A [2B]
6	MOV dir, @R0 [2B, 2C]	SUBB A, @R0	MOV @R0, dir [2B, 2C]	CJNE @R0, #data, rel [3B, 2C]	XCH A, @R0	XCHD A, @R0	MOV A, @R0	MOV @R0, A
7	MOV dir, @R1 [2B, 2C]	SUBB A, @R1	MOV @R1, dir [2B, 2C]	CJNE @R1, #data, rel [3B, 2C]	XCH A, @R1	XCHD A, @R1	MOV A, @R1	MOV @R1, A
8	MOV dir, R0 [2B, 2C]	SUBB A, R0	MOV R0, dir [2B, 2C]	CJNE R0, #data, rel [3B, 2C]	XCH A, R0	DJNZ R0, rel [2B, 2C]	MOV A, R0	MOV R0, A
9	MOV dir, R1 [2B, 2C]	SUBB A, R1	MOV R1, dir [2B, 2C]	CJNE R1, #data, rel [3B, 2C]	XCH A, R1	DJNZ R1, rel [2B, 2C]	MOV A, R1	MOV R1, A
A	MOV dir, R2 [2B, 2C]	SUBB A, R2	MOV R2, dir [2B, 2C]	CJNE R2, #data, rel [3B, 2C]	XCH A, R2	DJNZ R2, rel [2B, 2C]	MOV A, R2	MOV R2, A
B	MOV dir, R3 [2B, 2C]	SUBB A, R3	MOV R3, dir [2B, 2C]	CJNE R3, #data, rel [3B, 2C]	XCH A, R3	DJNZ R3, rel [2B, 2C]	MOV A, R3	MOV R3, A
C	MOV dir, R4 [2B, 2C]	SUBB A, R4	MOV R4, dir [2B, 2C]	CJNE R4, #data, rel [3B, 2C]	XCH A, R4	DJNZ R4, rel [2B, 2C]	MOV A, R4	MOV R4, A
D	MOV dir, R5 [2B, 2C]	SUBB A, R5	MOV R5, dir [2B, 2C]	CJNE R5, #data, rel [3B, 2C]	XCH A, R5	DJNZ R5, rel [2B, 2C]	MOV A, R5	MOV R5, A
E	MOV dir, R6 [2B, 2C]	SUBB A, R6	MOV R6, dir [2B, 2C]	CJNE R6, #data, rel [3B, 2C]	XCH A, R6	DJNZ R6, rel [2B, 2C]	MOV A, R6	MOV R6, A
F	MOV dir, R7 [2B, 2C]	SUBB A, R7	MOV R7, dir [2B, 2C]	CJNE R7, #data, rel [3B, 2C]	XCH A, R7	DJNZ R7, rel [2B, 2C]	MOV A, R7	MOV R7, A

Note. Key: [2B] = 2 Byte, [3B] = 3 Byte, [2C] = 2 Cycle, [4C] = 4 Cycle, Blank = 1 byte/1 cycle

Mnemonic		Description	Byte	Oscillator Period
ARITHMETIC OPERATIONS				
ADD	A,Rn	Add register to Accumulator	1	12
ADD	A,direct	Add direct byte to Accumulator	2	12
ADD	A,@Ri	Add indirect RAM to Accumulator	1	12
ADD	A,#data	Add immediate data to Accumulator	2	12
ADDC	A,Rn	Add register to Accumulator with Carry	1	12
ADDC	A,direct	Add direct byte to Accumulator with Carry	2	12
ADDC	A,@Ri	Add indirect RAM to Accumulator with Carry	1	12
ADDC	A,#data	Add immediate data to Acc with Carry	2	12
SUBB	A,Rn	Subtract Register from Acc with borrow	1	12
SUBB	A,direct	Subtract direct byte from Acc with borrow	2	12
SUBB	A,@Ri	Subtract indirect RAM from ACC with borrow	1	12
SUBB	A,#data	Subtract immediate data from Acc with borrow	2	12
INC	A	Increment Accumulator	1	12
INC	Rn	Increment register	1	12
INC	direct	Increment direct byte	2	12
INC	@Ri	Increment direct RAM	1	12
DEC	A	Decrement Accumulator	1	12
DEC	Rn	Decrement Register	1	12
DEC	direct	Decrement direct byte	2	12
DEC	@Ri	Decrement indirect RAM	1	12
INC	DPTR	Increment Data Pointer	1	24
MUL	AB	Multiply A & B	1	48
DIV	AB	Divide A by B	1	48
DA	A	Decimal Adjust Accumulator	1	12

Note 1. All mnemonics copyrighted © Intel Corp 1980

Mnemonic		Description	Byte	Oscillator Period
LOGICAL OPERATIONS				
ANL	A,Rn	AND Register to Accumulator	1	12
ANL	A,direct	AND direct byte to Accumulator	2	12
ANL	A,@Ri	AND indirect RAM to Accumulator	1	12
ANL	A,#data	AND immediate data to Accumulator	2	12
ANL	direct,A	AND Accumulator to direct byte	2	12
ANL	direct,#data	AND immediate data to direct byte	3	24
ORL	A,Rn	OR register to Accumulator	1	12
ORL	A,direct	OR direct byte to Accumulator	2	12
ORL	A,@Ri	OR indirect RAM to Accumulator	1	12
ORL	A,#data	OR immediate data to Accumulator	2	12
ORL	direct,A	OR Accumulator to direct byte	2	12
ORL	direct,#data	OR immediate data to direct byte	3	24
XRL	A,Rn	Exclusive-OR register to Accumulator	1	12
XRL	A,direct	Exclusive-OR direct byte to Accumulator	2	12
XRL	A,@Ri	Exclusive-OR indirect RAM to Accumulator	1	12
XRL	A,#data	Exclusive-OR immediate data to Accumulator	2	12
XRL	direct,A	Exclusive-OR Accumulator to direct byte	2	12
XRL	direct,#data	Exclusive-OR immediate data to direct byte	3	24
CLR	A	Clear Accumulator	1	12
CPL	A	Complement Accumulator	1	12
RL	A	Rotate Accumulator Left	1	12
RLC	A	Rotate Accumulator Left through the Carry	1	12
LOGICAL OPERATIONS (continued)				

Mnemonic		Description	Byte	Oscillator Period
RR	A	Rotate Accumulator Right	1	12
RRC	A	Rotate Accumulator Right through the Carry	1	12
SWAP	A	Swap nibbles within the Accumulator	1	12
DATA TRANSFER				
MOV	A,R_n	Move register to Accumulator	1	12
MOV	A,direct	Move direct byte to Accumulator	2	12
MOV	A,@R_i	Move indirect RAM to Accumulator	1	12
MOV	A,#data	Move immediate data to Accumulator	2	12
MOV	R_n,A	Move Accumulator to register	1	12
MOV	R_n,direct	Move direct byte to register	2	24
MOV	R_n,#data	Move immediate data to register	2	12
MOV	direct,A	Move Accumulator to direct byte	2	12
MOV	direct,R_n	Move register to direct byte	2	24
MOV	direct,direct	Move direct byte to direct	3	24
MOV	direct,@R_i	Move indirect RAM to direct byte	2	24
MOV	direct,#data	Move immediate data to direct byte	3	24
MOV	@R_i,A	Move Accumulator to indirect RAM	1	12
MOV	@R_i,direct	Move direct byte to indirect RAM	2	24
MOV	@R_i,#data	Move immediate data to indirect RAM	2	12
MOV	DPTR,#data16	Load Data Pointer with a 16-bit constant	3	24
MOVC	A,@A+DPTR	Move Code byte relative to DPTR to Acc	1	24
MOVC	A,@A+PC	Move Code byte relative to PC to Acc	1	24
MOVX	A,@R_i	Move External RAM (8-bit addr) to Acc	1	24
DATA TRANSFER (continued)				

Mnemonic		Description	Byte	Oscillator Period
MOVX	A,@DPTR	Move External RAM (16-bit addr) to Acc	1	24
MOVX	@R_i,A	Move Acc to External RAM (8-bit addr)	1	24
MOVX	@DPTR,A	Move Acc to External RAM (16-bit addr)	1	24
PUSH	direct	Push direct byte onto stack	2	24
POP	direct	Pop direct byte from stack	2	24
XCH	A,R_n	Exchange register with Accumulator	1	12
XCH	A,direct	Exchange direct byte with Accumulator	2	12
XCH	A,@R_i	Exchange indirect RAM with Accumulator	1	12
XCHD	A,@R_i	Exchange low-order Digit indirect RAM with Acc	1	12
BOOLEAN VARIABLE MANIPULATION				
CLR	C	Clear Carry	1	12
CLR	bit	Clear direct bit	2	12
SETB	C	Set Carry	1	12
SETB	bit	Set direct bit	2	12
CPL	C	Complement Carry	1	12
CPL	bit	Complement direct bit	2	12
ANL	C,bit	AND direct bit to CARRY	2	24
ANL	C,/bit	AND complement of direct bit to Carry	2	24
ORL	C,bit	OR direct bit to Carry	2	24
ORL	C,/bit	OR complement of direct bit to Carry	2	24
MOV	C,bit	Move direct bit to Carry	2	12
MOV	bit,C	Move Carry to direct bit	2	24
JC	rel	Jump if Carry is set	2	24
JNC	rel	Jump if Carry not set	2	24
JB	bit,rel	Jump if direct Bit is set	3	24
JNB	bit,rel	Jump if direct Bit is Not set	3	24
JBC	bit,rel	Jump if direct Bit is set & clear bit	3	24
PROGRAM BRANCHING				

Mnemonic		Description	Byte	Oscillator Period
ACALL	addr11	Absolute Subroutine Call	2	24
LCALL	addr16	Long Subroutine Call	3	24
RET		Return from Subroutine	1	24
RETI		Return from interrupt	1	24
AJMP	addr11	Absolute Jump	2	24
LJMP	addr16	Long Jump	3	24
SJMP	rel	Short Jump (relative addr)	2	24
JMP	@A+DPTR	Jump indirect relative to the DPTR	1	24
JZ	rel	Jump if Accumulator is Zero	2	24
JNZ	rel	Jump if Accumulator is Not Zero	2	24
CJNE	A,direct,rel	Compare direct byte to Acc and Jump if Not Equal	3	24
CJNE	A,#data rel	Compare immediate to Acc and Jump if Not Equal	3	24
CJNE	R_n,#data rel	Compare immediate to register and Jump if Not Equal	3	24
CJNE	@R_i,#data,rel	Compare immediate to indirect and Jump if Not Equal	3	24
DJNZ	R_n,rel	Decrement register and Jump if Not Zero	2	24
DJNZ	direct,rel	Decrement direct byte and Jump if Not Zero	3	24
NOP		No Operation	1	12

REFERENCES

Books:

1. Satheesh gopi, "Global Positioning System", Tata McGraw-Hill 2000.
2. Kenneth J.Ayala., "The 8051 Microcontroller Architecture Programming and Applications", Penram International Publishing (India). 1996.
3. Muhammad Ali Mazidi, Janice Gillispie Mazidi. "The 8051 Microcontroller, and Embedded Systems", Prentice Hall 2000.

Websites:

1. www.scientech.bz
2. www.palowireless.com
3. www.datasheetcatalog.com
4. www.chipdocs.com