**ACCIDENT DETECTION USING GPS AND GSM IN FOG**

**A**

**MAJOR PROJECT REPORT**

**Submitted in fulfilment of the award of Degree of Bachelor of Technology in**

**Electronics and Communication Engineering**

**Submitted**

**by**

**V. Sai Kumar 14951A0490**

**B. Hema 14951A0471**

**R. Bhavana 14951A0466**

**G. Manikanta 15955A0414**

**Under the Supervision of**

**Mr. S Rambabu**

**ASSISTANT PROFESSOR**



**Department of Electronics and Communication Engineering**

**INSTITUTE OF AERONAUTICAL ENGINEERING**

**(Autonomous)**

DUNIDIGAL- 500043, HYDERABAD, TELANGANA STATE

**(2017-2018)**

|  |  |
| --- | --- |
|  | **INSTITUTE OF AERONAUTICAL ENGINEERING**  **(Autonomous)**  DUNDIGAL-500043, HYDERABAD, TELANGANA STATE |

**CERTIFICATE**

This is to certify that the work embodies in this dissertation entitled, **“Accident Detection Using Gps And Gsm In Fog”** being submitted by **V.SaiKumar (14951A0490), B.Hema (14951A0471), R.Bhavana (14951A0466) and G.Manikanta(15955A0414),** for partial fulfilment of the requirement for the award of ‘**Bachelor of Technology** in **Electronics and Communication Engineering** discipline to **Institute of Aeronautical Engineering, Dundigal , Hyderabad** during the academic year 2017 - 2018 is a record of bonafide piece of work, undertaken by them in the supervision of the undersigned**.**

**Approved and Supervised by**

**Signature**

**(S.RAMBABU)**

Asst. Professor,ECE

**Forwarded by**

**(Dr. L.V.N Prasad) HOD ECE**

**Principal**

**EXTERNAL EXAMINER**

****

**INSTITUTE OF AERONAUTICAL ENGINEERING**

**(Autonomous)**

DUNDIGAL, HYDERABAD - 500043, TELANGANA STATE

**Department of Electronics and Communication Engineering**

**DECLARATION**

We, the students of “**Bachelor of Technology in Electronics and Communication Engineering Branch”, session: 2014 -2018,** Institute of Aeronautical Engineering, Dundigal, Hyderabad, hereby declare that the work presented in this Project Work entitled “**Accident Detection Using Gps And Gsm In Fog”** is the outcome of our own bonafide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics. It contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

**V. Sai Kumar 14951A0490**

**B. Hema 14951A0471**

**R. Bhavana 14951A0466**

**G. Manikanta 15955A0414**

**Date:**

**ACKNOWLEDGEMENTS**

We are greatly indebted to my project guide, **S.Rambabu**, Assistant Professor, Department of Electronics and Communication Engineering, for his invaluable guidance and inspiration which have sustained us to accomplish our work successfully.

We extend our deep gratitude to **S.Rambabu**, Assistant Professor, for his directions, constructive suggestions and continuous support throughout our project work. It helped us as a source of encouragement and support all through the project work.

We have a great pleasure in expressing our sincere thanks to **Prof. V.R.Seshagiri Rao, Head of the department**, who ignited our hidden potential, built career, inculcated self-confidence, sincerity and discipline within us and gave a path of success.

It is our pleasure to acknowledge gratefully to the **Management and Principal**, for their inspiration, valuable suggestions and keen interest during our work.

We are grateful to the teaching and non- teaching faculty members of the Department of Electronics and Communication Engineering, for their encouragement and the facilities provided during our project work.

We appreciate the arduous tasks of our **friends, near** and **dear** who injected patience, fortitude to overcome the challenges that have come our way.

We extend our whole hearted thanks to those who worked above and beyond this long-life time journey.

**V. Sai Kumar 14951A0490**

**B. Hema 14951A0471**

**R. Bhavana 14951A0466**

**G. Manikanta 15955A0414**

**ABSTRACT**

The damage caused to the roads due to Natural calamities cannot be easily predicted. This phenomenon can be explained by accidents in its prone areas which can also be caused due to Smog blocking the vision of the commuters. This model can be used for the sole purpose to overcome this situation. This model can also be used to divert traffic caused due to congestion, blocking of roads etc. This model uses a sample of five poles which can cover an initial distance to indicate the alert .It can be implemented by using Arduino UNO, 7812 IC, Laser light Indicators, Siren and Zigbee. As normal LED indicator cannot display in bad weather conditions for that Laser light indicators can be used as it has better capability to display even in fog. Zigbee is used to establish wireless communication as it can be used to activate unauthorized access to the system. The Software used to implement is Arduino IDE. There will not be any need to construct a separate pole for this model as it can be attached to the existing poles present on the road. It can be used mainly in places with bad weather conditions. It can be useful in accident prone areas. It can be used in metropolitan cities where more traffic is present.

|  |  |  |  |
| --- | --- | --- | --- |
| **CONTENTS** | | | |
| **Chapter** | **Description** | | **Page No.** |
| **1** | **Introduction** | |  |
|  | 1.1 | Embedded System | 1 |
|  | 1.2 | Characteristics Of An Embedded System | 1 |
|  | 1.3 | Advantages | 2 |
| **2** | **Description** | |  |
|  | 2.1 | Block diagram | 4 |
| **3** | **ARM 7** | |  |
|  | 3.1 | Introduction | 5 |
|  | 3.2 | Block Diagram | 7 |
|  | 3.3 | Functional Diagram | 8 |
|  | 3.4 | Signal Description | 8 |
| **4** | **8051 Architecture** | |  |
|  | 4.1 | Computer Software | 15 |
|  | 4.2 | Microcontroller | 15 |
|  | 4.3 | 8051Block Diagram | 17 |
|  | 4.4 | Pin Diagram | 20 |
|  | 4.5 | Pin Configuration | 22 |
|  | 4.6 | Addressing Modes | 27 |
|  | 4.7 | Instruction Types | 30 |
| **5** | **Global System for Mobile communication** | |  |
|  | 5.1 | Introduction | 32 |
|  | 5.2 | Why GSM | 33 |
|  | 5.3 | GSM Architecture | 33 |
|  | 5.4 | GSM Networks Areas | 35 |
|  | 5.5 | GSM Specifications | 35 |
|  | 5.6 | GSM Addressing And Identifiers | 37 |
|  | 5.7 | Mobile Subscriber ISDN Number | 38 |
|  | 5.8 | GSM Operations | 39 |
|  | 5.9 | GSM Protocol Stack | 41 |
| **6** | **Global Positioning System** | |  |
|  | 6.1 | Introduction | 43 |
|  | 6.2 | Segments Of Gps | 45 |
|  | 6.3 | Space Segments | 45 |
|  | 6.4 | Observation Principle And Signal Structure | 47 |
|  | 6.5 | Structure Of The GPS Navigation Data | 49 |
|  | 6.6 | International Limitation Of The System Accuracy | 50 |
|  | 6.7 | Basic Concept Of Gps Receiver And Its Components | 50 |
|  | 6.8 | Applications Of GPS |  |
| **7** | **Liquid Crystal Displays** | |  |
|  | 7.1 | Cathode Ray Tubes | 53 |
|  | 7.2 | Basics Of LCD Displays | 53 |
|  | 7.3 | **Basic Structure Of An LCD** | 53 |
|  | 7.4 | Types Of LCD/Liquid Crystal Displays. | 54 |
|  | 7.5 | Nematic Phase LCD | 54 |
|  | 7.6 | Making Of LCD | 55 |
|  | 7.7 | Colour Liquid Crystal Display | 56 |
| **8** | **Buzzer** | |  |
|  | 8.1 | General Description | 57 |
|  | 8.2 | Specifications | 57 |
|  | 8.3 | Pin Configuration | 57 |
|  | 8.4 | Schematic Diagram | 57 |
| **9** | **Vibration Sensor** | |  |
|  | 9.1 | Tilt Switches | 58 |
|  | 9.2 | Basic Stats | 59 |
| **10** | **MAX 232** | |  |
|  | 10.1 | Introduction | 60 |
|  | 10.2 | Serial port protocol | 60 |
|  | 10.3 | Dual Drivers and Dual Receivers | 60 |
|  | 10.4 | Pin Description | 62 |
| **11** | **KEIL Software** | |  |
|  | 11.1 | Introduction | 65 |
|  | 11.2 | Software Development Tools | 65 |
|  | 11.3 | RTOS and Middleware Components | 66 |
|  | 11.4 | Microcontroller Architectures | 66 |
|  | 11.5 | Classic and Extended 8051 Devices | 67 |
|  | 11.6 | 8051 Development Tool Support | 69 |
|  | 11.7 | µVision IDE | 70 |
|  | 11.8 | µVision Debugger | 71 |
|  | 11.9 | Simulation | 72 |
|  | 11.10 | Executing Code | 73 |
| **12** | **Code** | | 75 |
| **13** | **Results** | | 80 |
| **14** | **Advantages and application** | | 82 |
| **15** | **Conclusion** | | 83 |
| **16** | **References** | | 84 |

|  |  |  |
| --- | --- | --- |
| **List of Figures** | | |
| **S.No** | **Figure Name** | **Page No** |
| 2.1 | Block Diagram | 4 |
| 3.1 | Block Diagram Of ARM7 | 7 |
| 3.2 | Functional Diagram Of ARM7 | 8 |
| 4.1 | 8051 Architecture | 19 |
| 4.2 | Pin Diagram Of 8051 Micro Controller | 20 |
| 5.1 | Components Of The GSM Architecture | 34 |
| 5.2 | Gsm Network | 35 |
| 6.1 | The Global Positioning System (GPS), 21-Satellite Configuration | 45 |
| 6.2 | Schematic View Of A Block II GPS Satellite | 48 |
| 6.3 | Generation Of GPS Signals | 50 |
| 6.4 | Major Components Of A GPS Receiver | 52 |
| 7.1 | Working Of LCD | 57 |
| 8.1 | Schematic Diagram | 58 |
| 9.1 | Tilt Switch | 59 |
| 10.1 | Pin Diagram Of Max 232 | 63 |
| 11.1 | Memory Layout Of A Classic 8051 Device | 69 |

**CHAPTER 1**

**INTRODUCTION**

A system is an arrangement in which all its unit assemble work together according to a set of rules. It can also be defined as a way of working, organizing or doing one or many tasks according to a fixed plan. For example, a watch is a time displaying system. Its components follow a set of rules to show time. If one of its parts fails, the watch will stop working. So we can say, in a system, all its subcomponents depend on each other.

**1.1 Embedded System**

As its name suggests, Embedded means something that is attached to another thing. An embedded system can be thought of as a computer hardware system having software embedded in it. An embedded system can be an independent system or it can be a part of a large system. An embedded system is a microcontroller or microprocessor based system which is designed to perform a specific task. For example, a fire alarm is an embedded system; it will sense only smoke.

An embedded system has three components:

* It has hardware.
* It has application software.
* It has Real Time Operating system (RTOS) that supervises the application software and provide mechanism to let the processor run a process as per scheduling by following a plan to control the latencies. RTOS defines the way the system works. It sets the rules during the execution of application program. A small scale embedded system may not have RTOS.

So we can define an embedded system as a Microcontroller based, software driven, reliable, real-time control system.

**1.2 Characteristics of an Embedded System**

* Single-functioned – An embedded system usually performs a specialized operation and does the same repeatedly. For example: A pager always functions as a pager.
* Tightly constrained – All computing systems have constraints on design metrics, but those on an embedded system can be especially tight. Design metrics is a measure of an implementation's features such as its cost, size, power, and performance. It must be of a size to fit on a single chip, must perform fast enough to process data in real time and consume minimum power to extend battery life.
* Reactive and Real time – Many embedded systems must continually react to changes in the system's environment and must compute certain results in real time without any delay. Consider an example of a car cruise controller; it continually monitors and reacts to speed and brake sensors. It must compute acceleration or de-accelerations repeatedly within a limited time; a delayed computation can result in failure to control of the car.
* Microprocessors based – It must be microprocessor or microcontroller based.
* Memory – It must have a memory, as its software usually embeds in ROM. It does not need any secondary memories in the computer.
* Connected – It must have connected peripherals to connect input and output devices.
* HW-SW systems – Software is used for more features and flexibility. Hardware is used for performance and security.

**1.3 Advantages**

* Easily Customizable
* Low power consumption
* Low cost
* Enhanced performance

**CHAPTER 2**

**DESCRIPTION**

The damage caused on the roads due to Natural calamities cannot be easily predicted. This phenomenon can be explained by accidents in its prone areas which can also be caused due to Smog blocking the vision of the commuters.

The high demand of automobiles has also increased the traffic hazards and the road accidents. Life of the people is under high risk. This is because of the lack of best emergency facilities available in our country. An automatic alarm device for vehicle accidents is introduced in this paper. This design is a system which can detect accidents in significantly less time and sends the basic information to first aid centre within a few seconds covering geographical coordinates, the time and angle in which a vehicle accident had occurred. This alert message is sent to the rescue team in a short time, which will help in saving the valuable lives. A Switch is also provided in order to terminate the sending of a message in rare case where there is no casualty, this can save the precious time of the medical rescue team. When the accident occurs the alert message is sent automatically to the rescue team and to the police station. The message is sent through the GSM module and the location of the accident is detected with the help of the GPS module. The accident can be detected precisely with the help of both Micro electro mechanical system (MEMS) sensor and vibration sensor. The Angle of the rolls over of the car can also be known by the message through the MEMS sensor. This application provides the optimum solution to poor emergency facilities provided to the roads accidents in the most feasible way.

Accidents are increasing at a large pace, and various technologies are being introduced to reduce the accidents.

The indication is displayed to the commuter using a laser light.

This model can also be used to divert traffic caused due to congestion, blocking of roads etc. This model uses a sample of five poles which can cover an initial distance to indicate the alert.

There are certain specifications that are favourable enough to indicate, even in harsh weather conditions like Fog etc.

Output Power: 100MW/300MW.

Input Current: 80MA-200MA.

Laser wavelength: 635nm5nm.

Light angle: 15 Degree.

Working Temperature: -20-60C.

**2.1 Block Diagram**

Figure 2.1 :Block Diagram

There will not be any need to construct a separate pole for this model as it can be attached to the existing poles present on the road.

As it is attached to the existing poles, the power supply is also taken from those existing poles.

As we use Zigbee, the indication to the Laser light can be done without any particular switch.

At present criteria, we cannot detect where the accident has occurred and hence no information related to it, leading to the death of an individual. The research work is going on for tracking the position of the vehicle even in dark clumsy areas where there is no network for receiving the signals. In this project GPS is used for tracking the position of the vehicle, GSM is used for sending the message and the ARM controller is used for saving the mobile number in the EEPROM and sends the message to it when an accident has been detected. A laser light blinks as well as buzzer gives a beep sound upto a specified time.

**CHAPTER 3**

**ARM 7**

**3.1 Introduction**

The ARM7 is part of the Advanced RISC Machines (ARM) family of general purpose 32-bit microprocessors, which offer very low power consumption and price for high performance devices. The architecture is based on Reduced Instruction Set Computer (RISC) principles, and the instruction set and related decode mechanism are much simpler in comparison with micro programmed Complex Instruction Set Computers. This results in a high instruction throughput and impressive real-time interrupt response from a small and cost-effective chip. The instruction set comprises eleven basic instruction types:

* Two of these make use of the on-chip arithmetic logic unit, barrel shifter and multiplier to perform high-speed operations on the data in a bank of 31 registers, each 32 bits wide;
* Three classes of instruction control data transfer between memory and the registers, one optimized for flexibility of addressing, another for rapid context switching and the third for swapping data
* Three instructions control the flow and privilege level of execution.
* Three types are dedicated to the control of external coprocessors which allow the functionality of the instruction set to be extended off-chip in an open and uniform way.

The ARM instruction set is a good target for compilers of many different high-level languages. Where required for critical code segments, assembly code programming is also straightforward, unlike some RISC processors which depend on sophisticated compiler technology to manage complicated instruction interdependencies.

Pipelining is employed so that all parts of the processing and memory systems can operate continuously. Typically, while one instruction is being executed, its successor is being decoded, and a third instruction is being fetched from memory.

The memory interface has been designed to allow the performance potential to be realised without incurring high costs in the memory system. Speed critical control signals are pipelined to allow system control functions to be implemented in standard low-power logic, and these control signals facilitate the exploitation of the fast local access modes offered by industry standard dynamic RAMs.

ARM7 has a 32 bit address bus. All ARM processors share the same instruction set, and ARM7 can be configured to use a 26 bit address bus for backwards compatibility with earlier processors.

ARM7 is a fully static CMOS implementation of the ARM which allows the clock to be stopped in any part of the cycle with extremely low residual power consumption and no loss of state.

**Notation:**

0x - marks a Hexadecimal quantity

**BOLD** - external signals are shown in bold capital letters

binary - where it is not clear that a quantity is binary it is followed by the word binary

* 1. **Block Diagram**

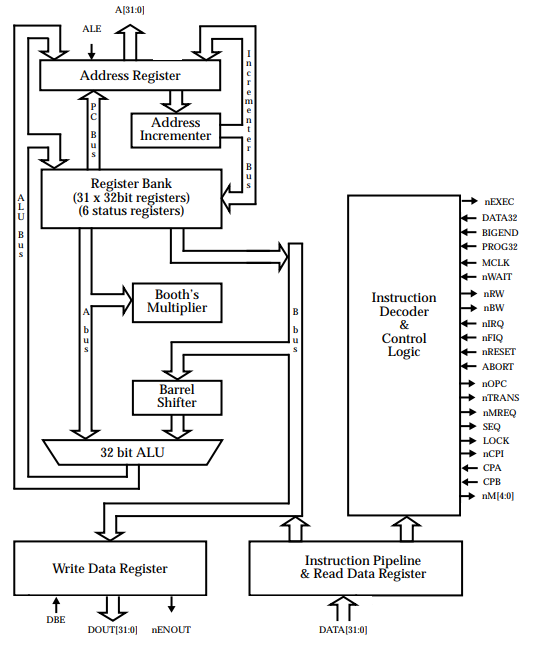


Figure 3.1 : Block Diagram of ARM7

* 1. **Functional Diagram**

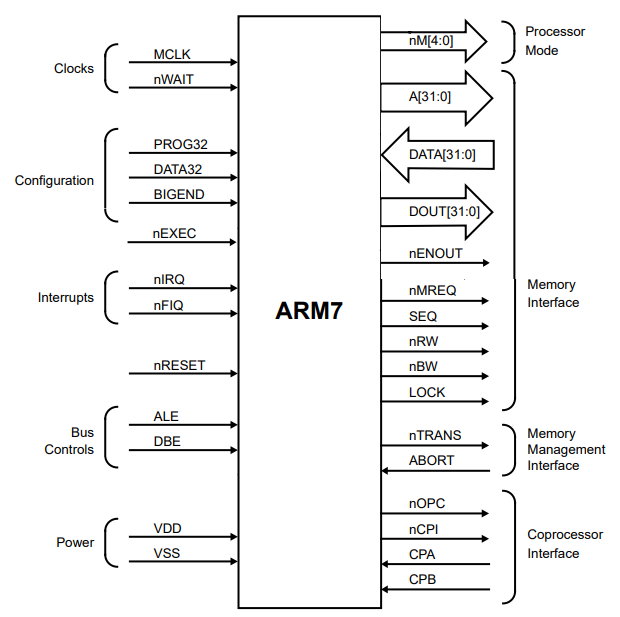
****

Figure 3.2 : Functional Diagram of ARM7

**3.4 Signal Description**

**A[31:0]**

Addresses. This is the processor address bus. If ALE (address latch enable) is HIGH, the addresses become valid during phase 2 of the cycle before the one to which they refer and remain so during phase 1 of the referenced cycle. Their stable period may be controlled by ALE as described below.

**ABORT**

Memory Abort. This is an input which allows the memory system to tell the processor that a requested access is not allowed.

**ALE**

Address latch enable. This input is used to control transparent latches on the address outputs. Normally the addresses change during phase 2 to the value required during the next cycle, but for direct interfacing to ROMs they are required to be stable to the end of phase 2. Taking ALE LOW until the end of phase 2 will ensure that this happens. This signal has a similar effect on the following control signals: nBW, nRW, LOCK, nOPC and nTRANS. If the system does not require address lines to be held in this way, ALE must be tied HIGH. The address latch is static, so ALE may be held LOW for long periods to freeze addresses.

**BIGEND**

Big Endian configuration. When this signal is HIGH the processor treats bytes in memory as being in Big Endian format. When it is LOW memory is treated as Little Endian. ARM processors which do not have selectable Endianism (ARM2, ARM2aS, ARM3, ARM61) are Little Endian.

**CPA**

Coprocessor absent. A coprocessor which is capable of performing the operation that ARM7 is requesting (by asserting nCPI) should take CPA LOW immediately. If CPA is HIGH at the end of phase 1 of the cycle in which nCPI went LOW, ARM7 will abort the coprocessor handshake and take the undefined instruction trap. If CPA is LOW and remains LOW, ARM7 will busy-wait until CPB is LOW and then complete the coprocessor instruction.

**CPB**

Coprocessor busy. A coprocessor which is capable of performing the operation which ARM7 is requesting (by asserting nCPI), but cannot commit to starting it immediately, should indicate this by driving cPB HIGH. When the coprocessor is ready to start it should take CPB LOW. ARM7 samples CPB at the end of phase 1 of each cycle in which nCPI is LOW.

**DATA[31:0]**

Data bus in. During read cycles (when nRW = 0), the input data must be valid before the end of phase 2 of the transfer cycle

**DATA32**

32 bit Data configuration. When this signal is HIGH the processor can access data in a 32 bit address space using address lines A[31:0]. When it is LOW the processor can access data from a 26 bit address space using A[25:0]. In this latter configuration the address lines A[31:26] are not used. Before changing DATA32, ensure that the processor is not about to access an address greater that 0x3FFFFFF in the next cycle.

**DBE**

Data bus enable. When DBE is LOW the write data buffer is disabled. When DBE goes HIGH the write data buffer is free to be enabled during the next actual write cycle. DBE facilitates data bus sharing for DMA ARM7 Data Sheet 6

**DOUT[31:0]**

Data bus out. During write cycles (when nRW = 1), the output data will become valid during phase 1 and remain so throughout phase 2 of the transfer cycle.

**LOCK**

Locked operation. When LOCK is HIGH, the processor is performing a “locked” memory access, and the memory controller must wait until LOCK goes LOW before allowing another device to access the memory. LOCK changes while MCLK is HIGH, and remains HIGH for the duration of the locked memory accesses. It is active only during the data swap (SWP) instruction. The timing of this signal may be modified by the use of ALE in a similar way to the address, please refer to the ALE description. This signal may also be driven to a high impedance state by driving ABE LOW.

**MCLK**

Memory clock input. This clock times all ARM7 memory accesses and internal operations. The clock has two distinct phases - phase 1 in which MCLK is LOW and phase 2 in which MCLK (and nWAIT) is HIGH. The clock may be stretched indefinitely in either phase to allow access to slow peripherals or memory. Alternatively, the nWAIT input may be used with a free running MCLK to achieve the same effect.

**nBW**

Not byte/word. This is an output signal used by the processor to indicate to the external memory system when a data transfer of a byte length is required. The signal is HIGH for word transfers and LOW for byte transfers and is valid for both read and write cycles. The signal will become valid during phase 2 of the cycle before the one in which the transfer will take place. It will remain stable throughout phase 1 of the transfer cycle. The timing of this signal may be modified by the use of ALE in a similar way to the address, please refer to the ALE description. This signal may also be driven to a high impedance state by driving ABE LOW.

**nCPI**

Not Coprocessor instruction. When ARM7 executes a coprocessor instruction, it will take this output LOW and wait for a response from the coprocessor. The action taken will depend on this response, which the coprocessor signals on the CPA and CPB inputs.

**nENOUT**

Not enable data outputs. This is an output signal used by the processor to indicate that a write cycle is taking place, so the DOUT[31:0] data should be sent to the memory system. It may be used to enable the DOUT[31:0] bus through tri-state buffers onto the DATA[31:0] bus if the system requirement is for a bidirectional data bus. nENIN I NOT enable input. This signal may be used in conjunction with nENOUT to control the data bus during write cycles.

**nFIQ**

Not fast interrupt request. This is an asynchronous interrupt request to the processor which causes it to be interrupted if taken LOW when the appropriate enable in the processor is active. The signal is level sensitive and must be held LOW until a suitable response is received from the processor.

**nIRQ**

Not interrupt request. As nFIQ, but with lower priority. May be taken LOW asynchronously to interrupt the processor when the appropriate enable is active.

**nM[4:0]**

Not processor mode. These are output signals which are the inverses of the internal status bits indicating the processor operation mode.

**nMREQ**

Not memory request. This signal, when LOW, indicates that the processor requires memory access during the following cycle. The signal becomes valid during phase 1, remaining valid through phase 2 of the cycle preceding that to which it refers.

**nOPC**

Not op-code fetch. When LOW this signal indicates that the processor is fetching an instruction from memory; when HIGH, data (if present) is being transferred. The signal becomes valid during phase 2 of the previous cycle, remaining valid through phase 1 of the referenced cycle. The timing of this signal may be modified by the use of ALE in a similar way to the address, please refer to the ALE description. This signal may also be driven to a high impedance state by driving ABE LOW.

**nRESET**

Not reset. This is a level sensitive input signal which is used to start the processor from a known address. A LOW level will cause the instruction being executed to terminate abnormally. When nRESET becomes HIGH for at least one clock cycle, the processor will re-start from address 0. nRESET must remain LOW (and nWAIT must remain HIGH) for at least two clock cycles. During the LOW period the processor will perform dummy instruction fetches with the address incrementing from the point where reset was activated. The address will overflow to zero if nRESET is held beyond the maximum address limit.

**nRW**

Not read/write. When HIGH this signal indicates a processor write cycle; when LOW, a read cycle. It becomes valid during phase 2 of the cycle before that to which it refers, and remains valid to the end of phase 1 of the referenced cycle. The timing of this signal may be modified by the use of ALE in a similar way to the address, please refer to the ALE description. This signal may also be driven to a high impedance state by driving ABE LOW.

**nTRANS**

Not memory translate. When this signal is LOW it indicates that the processor is in user mode. It may be used to tell memory management hardware when translation of the addresses should be turned on, or as an indicator of non-user mode activity. The timing of this signal may be modified by the use of ALE in a similar way to the address, please refer to the ALE description. This signal may also be driven to a high impedance state by driving ABE LOW.

**nWAIT**

Not wait. When accessing slow peripherals, ARM7 can be made to wait for an integer number of MCLK cycles by driving nWAIT LOW. Internally, nWAIT is ANDed with MCLK and must only change when MCLK is LOW. If nWAIT is not used it must be tied HIGH.

**PROG32**

32 bit Program configuration. When this signal is HIGH the processor can fetch instructions from a 32 bit address space using address lines A[31:0]. When it is LOW the processor fetches instructions from a 26 bit address space using A[25:0]. In this latter configuration the address lines A[31:26] are not used for instruction fetches. Before changing PROG32, ensure that the processor is in a 26 bit mode, and is not about to write to an address in the range 0 to 0x1F (inclusive) in the next cycle. Name Type Description Table 1: Signal Description (Continued) ARM7 Data Sheet 8 Key to Signal Types: I - Input O - Output P - Power

**SEQ**

Sequential address. This output signal will become HIGH when the address of the next memory cycle will be related to that of the last memory access. The new address will either be the same as or 4 greater than the old one. The signal becomes valid during phase 1 and remains so through phase 2 of the cycle before the cycle whose address it anticipates. It may be used, in combination with the low-order address lines, to indicate that the next cycle can use a fast memory mode and/or to bypass the address translation system. The signal becomes valid during phase 1, remaining valid through phase 2 of the cycle preceding that to which it refers

**VDD**

Power supply. These connections provide power to the device.

**VSS**

Ground. These connections are the ground reference for all signals. It is active only during the data swap (SWP) instruction.

**CHAPTER 4**

**8051 ARCHITECHURE**

**4.1 Computer Software**

A set of instructions written in a specific sequence for the computer to solve a specific task is called a program and software is a collection of such programs. The program stored in the computer memory in the form of binary numbers is called machine instructions. The machine language program is called object code. An assembly language is a mnemonic representation of machine language. Machine language and assembly language are low level languages and are processor specific. The assembly language program the programmer enters is called source code. The source code (assembly language) is translated to object code (machine language) using assembler. Programs can be written in high level languages such as C, C++ etc. High level language will be converted to machine language using compiler or interpreter. Compiler reads the entire program and translate into the object code and then it is executed by the processor. Interpreter takes one statement of the high level language as input and translate it into object code and then executes.

**4.2 Microcontroller**

A **microcontroller** is a small and low-cost microcomputer, which is designed to perform the specific tasks of embedded systems like displaying microwave’s information, receiving remote signals, etc.

The general microcontroller consists of the processor, the memory (RAM, ROM, EPROM), Serial ports, peripherals (timers, counters), etc.

The assembly language program the programmer enters is called source code. The source code (assembly language) is translated to object code (machine language) using assembler. Programs can be written in high level languages such as C, C++ etc. High level language will be converted to machine language using compiler or interpreter. Compiler reads the entire program and translate into the object code and then it is executed by the processor.

## **Difference between Microprocessor and Microcontroller**

The following table highlights the differences between a microprocessor and a microcontroller

|  |  |
| --- | --- |
| **Microcontroller** | **Microprocessor** |
| Microcontrollers are used to execute a single task within an application. | Microprocessors are used for big applications. |
| Its designing and hardware cost is low. | Its designing and hardware cost is high. |
| Easy to replace. | Not so easy to replace. |
| It is built with CMOS technology, which requires less power to operate. | Its power consumption is high because it has to control the entire system. |
| It consists of CPU, RAM, ROM, I/O ports. | It doesn’t consist of RAM, ROM, I/O ports. It uses its pins to interface to peripheral devices. |

## **Types of Microcontrollers**

Microcontrollers are divided into various categories based on memory, architecture, bits and instruction sets. Following is the list of their types −

### Bit

Based on bit configuration, the microcontroller is further divided into three categories.

* **8-bit microcontroller** − This type of microcontroller is used to execute arithmetic and logical operations like addition, subtraction, multiplication division, etc. For example, Intel 8031 and 8051 are 8 bits microcontroller.
* **16-bit microcontroller** − This type of microcontroller is used to perform arithmetic and logical operations where higher accuracy and performance is required. For example, Intel 8096 is a 16-bit microcontroller.
* **32-bit microcontroller** − This type of microcontroller is generally used in automatically controlled appliances like automatic operational machines, medical appliances, etc.

### Memory Based on the memory configuration, the microcontroller is further divided into two categories.

* **External memory microcontroller** − This type of microcontroller is designed in such a way that they do not have a program memory on the chip. Hence, it is named as external memory microcontroller. For example: Intel 8031 microcontroller.
* **Embedded memory microcontroller** − This type of microcontroller is designed in such a way that the microcontroller has all programs and data memory, counters and timers, interrupts, I/O ports are embedded on the chip. For example: Intel 8051 microcontroller.

### Instruction Set

Based on the instruction set configuration, the microcontroller is further divided into two categories.

* **CISC** − CISC stands for complex instruction set computer. It allows the user to insert a single instruction as an alternative to many simple instructions.
* **RISC** − RISC stands for Reduced Instruction Set Computers. It reduces the operational time by shortening the clock cycle per instruction.

## **Applications of Microcontrollers**

Microcontrollers are widely used in various different devices such as −

* Light sensing and controlling devices like LED.
* Temperature sensing and controlling devices like microwave oven, chimneys.
* Fire detection and safety devices like Fire alarm.
* Measuring devices like Volt Meter.

**4.3 The 8051 Architecture**

Introduction Salient features of 8051 microcontroller are given below.

Eight bit CPU

• On chip clock oscillator

• 4Kbytes of internal program memory (code memory) [ROM]

• 128 bytes of internal data memory [RAM]

• 64 Kbytes of external program memory address space.

• 64 Kbytes of external data memory address space.

• 32 bi directional I/O lines (can be used as four 8 bit ports or 32 individually addressable I/O lines) Two 16 Bit Timer/Counter :T0, T1

• Full Duplex serial data receiver/transmitter

• Four Register banks with 8 registers in each bank.  
• Sixteen bit Program counter (PC) and a data pointer (DPTR)

• 8 Bit Program Status Word (PSW)

• 8 Bit Stack Pointer

• Five vector interrupt structure (RESET not considered as an interrupt.)

• 8051 CPU consists of 8 bit ALU with associated registers like accumulator ‘A’ , B register, PSW, SP, 16 bit program counter, stack pointer. ALU can perform arithmetic and logic functions on 8 bit variables.

• 8051 has 128 bytes of internal RAM which is divided into Working registers [00 – 1F] o Bit addressable memory area [20 – 2F] o General purpose memory area (Scratch pad memory) [30-7F]

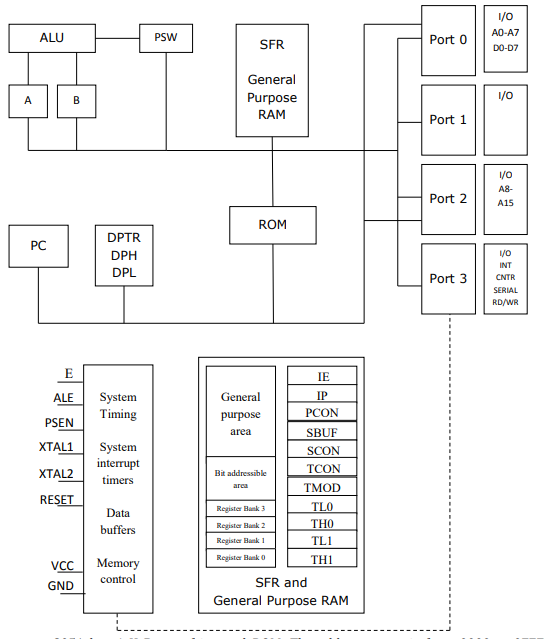


Figure 4.1 : 8051 Architecture

• 8051 has 4 K Bytes of internal ROM. The address space is from 0000 to 0FFFh. If the program size is more than 4 K Bytes 8051 will fetch the code automatically from external memory. Accumulator is an 8 bit register widely used for all arithmetic and logical operations.

• Accumulator is also used to transfer data between external memory. B register is used along with Accumulator for multiplication and division. A and B registers together is also called MATH registers.

• PSW (Program Status Word). This is an 8 bit register which contains the arithmetic status of ALU and the bank select bits of register banks. CY AC F0 RS1 RS0 OV - P CY - carry flag AC - auxiliary carry flag F0 - available to the user for general purpose RS1,RS0 - register bank select bits OV - overflow P - parity Stack Pointer (SP) – it contains the address of the data item on the top of the stack. Stack

• May reside anywhere on the internal RAM. On reset, SP is initialized to 07 so that the default stack will start from address 08 onwards. Data Pointer (DPTR) – DPH (Data pointer higher byte), DPL (Data pointer lower byte).

• This is a 16 bit register which is used to furnish address information for internal and external program memory and for external data memory. Program Counter (PC) – 16 bit PC contains the address of next instruction to be executed.

• On reset PC will set to 0000. After fetching every instruction PC will increment by one.

**4.4 Pin Diagram:**

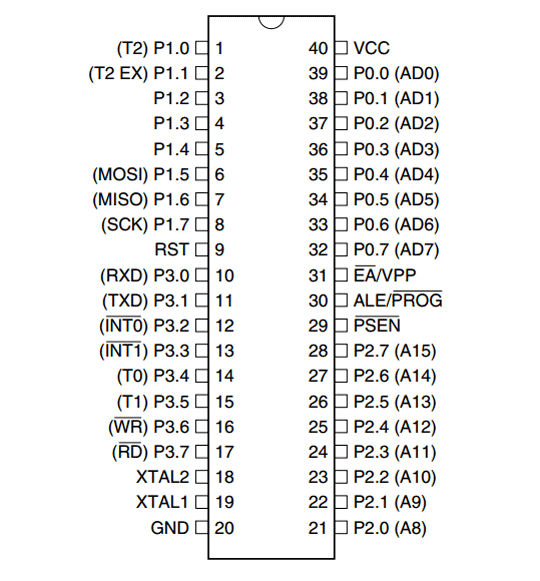
****

Figure 4.2 : Pin Diagram Of 8051 Micro Controller

The pin diagram of 8051 microcontroller consists of 40 pins as shown below. A total of 32 pins are set away into four Ports such as P0, P1, P2 and P3. Where, each port contains 8 pins. Therefore, the microcontroller 8051’s pin diagram and explanation is given below.

* **Port1 (Pin1 to Pin8):**Port1 includes pin1.0 to pin1.7 and these pins can be configured as  input or  output pins.
* **Pin 9 (RST):**Reset pin is used to Reset 8051 Microcontroller by giving a positive pulse to this Pin.
* **Port3 (Pin 10 to 17):**The Port3 Pins are similar to port1 pins and can be used as universal Input or output pins. These pins dual-function Pins and the function of each Pin is given as:
* **Pin 10 (RXD):** RXD pin is a Serial Asynchronous Communication Input or [Serial synchronous Communication](https://www.elprocus.com/serial-peripheral-interface-spi-communication-protocol/) Output.
* **Pin 11 (TXD):** Serial Asynchronous Communication Output or Serial Synchronous Communication clock Output.
* **Pin 12 (INT0):** [Input of Interrupt](https://www.elprocus.com/types-of-interrupts-in-8051-microcontroller-and-interrupt-programming/) 0
* **Pin 13 (INT1):** Input of Interrupt 1
* **Pin 14 (T0):** Input of Counter 0 clock
* **Pin 15 (T1):** Input of Counter 1 clock
* **Pin 16 (WR):** Writing Signal to write content on external RAM.
* **Pin 17 (RD):** Reading Signal to read contents of external RAM.
* **Pin 18 and 19 (XTAL2, XTAL1):** X2 and X1 pins are input output pins for the oscillator. These pins are used to connect an internal oscillator to the microcontroller.
* **Pin 20 (GND):** Pin 20 is a ground pin.
* **Port2 (Pin 21 to Pin28):** Port2 includes pin21 to pin28 which can be configured as Input Output Pins. But, this is only possible  when we don’t use any external memory. If we use external memory, then these pins will work as high order address bus (A8 to A15).
* **Pin 29 (PSEN):** This pin is used to enable external program memory. If we use an external ROM for storing the program, then logic 0 appears on it, which indicates Micro controller to read data from the memory.
* **Pin 30 (ALE):** Address Latch Enable pin is an active high-output signal. If we use multiple memory chips, then this pin is used to distinguish between them. This Pin also gives program pulse input during programming of EPROM.
* **Pin 31 (EA):**If we have to use multiple memories then the application of logic 1 to this pin instructs the Microcontroller to read data from both memories: first internal and then external.
* **Port 0 (Pin 32 to 39):** Similar to the port 2 and 3 pins, these pins can be used as input output pins when we don’t use any external memory. When ALE or Pin 30 is at 1, then this port is used as data bus: when the ALE pin is at 0, then this port is used as a lower order address bus (A0 to A7)
* **Pin40 (VCC):** This VCC pin is used for power supply.

8051 microcontrollers have 4 I/O ports each of 8-bit, which can be configured as input or output. Hence, total 32 input/output pins allow the microcontroller to be connected with the peripheral devices.

**4.5 Pin configuration**

The pin can be configured as 1 for input and 0 for output as per the logic state.

**Input / Output (I/O) pin**

All the circuits within the microcontroller must be connected to one of its pins except P0 port because it does not have pull-up resistors built-in.

**Input pin**

Logic 1 is applied to a bit of the P register. The output FE transistor is turned off and the other pin remains connected to the power supply voltage over a pull-up resistor of high resistance.

**Port 0**

The P0 (zero) port is characterized by two functions −

When the external memory is used then the lower address byte (addresses A0A7) is applied on it, else all bits of this port are configured as input/output.

When P0 port is configured as an output then other ports consisting of pins with built-in pull-up resistor connected by its end to 5V power supply, the pins of this port have this resistor left out.

### Input Configuration

If any pin of this port is configured as an input, then it acts as if it “floats”, i.e. the input has unlimited input resistance and in-determined potential.

### Output Configuration

When the pin is configured as an output, then it acts as an “open drain”. By applying logic 0 to a port bit, the appropriate pin will be connected to ground (0V), and applying logic 1, the external output will keep on “floating”.

In order to apply logic 1 (5V) on this output pin, it is necessary to build an external pullup resistor.

### Port 1

P1 is a true I/O port as it doesn’t have any alternative functions as in P0, but this port can be configured as general I/O only. It has a built-in pull-up resistor and is completely compatible with TTL circuits.

### Port 2

P2 is similar to P0 when the external memory is used. Pins of this port occupy addresses intended for the external memory chip. This port can be used for higher address byte with addresses A8-A15. When no memory is added then this port can be used as a general input/output port similar to Port 1.

### Port 3

In this port, functions are similar to other ports except that the logic 1 must be applied to appropriate bit of the P3 register.

## **Pins Current Limitations**

* When pins are configured as an output (i.e. logic 0), then the single port pins can receive a current of 10mA.
* When these pins are configured as inputs (i.e. logic 1), then built-in pull-up resistors provide very weak current, but can activate up to 4 TTL inputs of LS series.
* If all 8 bits of a port are active, then the total current must be limited to 15mA (port P0: 26mA).
* If all ports (32 bits) are active, then the total maximum current must be limited to 71mA.

Interrupts are the events that temporarily suspend the main program, pass the control to the external sources and execute their task. It then passes the control to the main program where it had left off.

8051 has 5 interrupt signals, i.e. INT0, TFO, INT1, TF1, RI/TI. Each interrupt can be enabled or disabled by setting bits of the IE register and the whole interrupt system can be disabled by clearing the EA bit of the same register.

## **IE (Interrupt Enable) Register**

This register is responsible for enabling and disabling the interrupt. EA register is set to one for enabling interrupts and set to 0 for disabling the interrupts. Its bit sequence and their meanings are shown in the following figure.

IE Register

|  |  |  |
| --- | --- | --- |
| EA | IE.7 | It disables all interrupts. When EA = 0 no interrupt will be acknowledged and EA = 1 enables the interrupt individually. |
| - | IE.6 | Reserved for future use. |
| - | IE.5 | Reserved for future use. |
| ES | IE.4 | Enables/disables serial port interrupt. |
| ET1 | IE.3 | Enables/disables timer1 overflow interrupt. |
| EX1 | IE.2 | Enables/disables external interrupt1. |
| ET0 | IE.1 | Enables/disables timer0 overflow interrupt. |
| EX0 | IE.0 | Enables/disables external interrupt0. |

## **IP (Interrupt Priority) Register**

We can change the priority levels of the interrupts by changing the corresponding bit in the Interrupt Priority (IP) register as shown in the following figure.

* A low priority interrupt can only be interrupted by the high priority interrupt, but not interrupted by another low priority interrupt.
* If two interrupts of different priority levels are received simultaneously, the request of higher priority level is served.
* If the requests of the same priority levels are received simultaneously, then the internal polling sequence determines which request is to be serviced.

IP Register

|  |  |  |
| --- | --- | --- |
| - | IP.6 | Reserved for future use. |
| - | IP.5 | Reserved for future use. |
| PS | IP.4 | It defines the serial port interrupt priority level. |
| PT1 | IP.3 | It defines the timer interrupt of 1 priority. |
| PX1 | IP.2 | It defines the external interrupt priority level. |
| PT0 | IP.1 | It defines the timer0 interrupt priority level. |
| PX0 | IP.0 | It defines the external interrupt of 0 priority level. |

## **TCON Register**

TCON register specifies the type of external interrupt to the microcontroller

The 8255A is a general purpose programmable I/O device designed to transfer the data from I/O to interrupt I/O under certain conditions as required. It can be used with almost any microprocessor.

It consists of three 8-bit bidirectional I/O ports (24I/O lines) which can be configured as per the requirement.

**Ports of 8255A**

8255A has three ports, i.e., PORT A, PORT B, and PORT C.

* **Port A** contains one 8-bit output latch/buffer and one 8-bit input buffer.
* **Port B** is similar to PORT A.
* **Port C** can be split into two parts, i.e. PORT C lower (PC0-PC3) and PORT C upper (PC7-PC4) by the control word.

These three ports are further divided into two groups, i.e. Group A includes PORT A and upper PORT C. Group B includes PORT B and lower PORT C. These two groups can be programmed in three different modes, i.e. the first mode is named as mode 0, the second mode is named as Mode 1 and the third mode is named as Mode 2.

## **Operating Modes**

8255A has three different operating modes −

* **Mode 0** − In this mode, Port A and B is used as two 8-bit ports and Port C as two 4-bit ports. Each port can be programmed in either input mode or output mode where outputs are latched and inputs are not latched. Ports do not have interrupt capability.
* **Mode 1** − In this mode, Port A and B is used as 8-bit I/O ports. They can be configured as either input or output ports. Each port uses three lines from port C as handshake signals. Inputs and outputs are latched.
* **Mode 2** − In this mode, Port A can be configured as the bidirectional port and Port B either in Mode 0 or Mode 1. Port A uses five signals from Port C as handshake signals for data transfer. The remaining three signals from Port C can be used either as simple I/O or as handshake for port B.

**4.6 Addressing Modes**

The 8051 instructions use eight addressing modes. These are:

1. Register

2. Direct

3. Indirect

4. Immediate

5. Relative

6. Absolute

7. Long

8. Indexed

**1. Register Addressing**

In this mode the data, which the instruction operates on, is in one of eight registers labelled R0 to R7 (Rn, in general). These registers are to be found in one of four register banks, only one of which can be active at any one time. The active bank may be selected by using bit 3 and bit 4 of the PSW (rs0 & rs1). On power-up or reset, the default register bank is bank 0. The format of an instruction using register addressing: For example, to logically OR the contents of accumulator A with that of register R3, the following instruction is used: ORL A, R3 and the op-code is 01001011B. The upper five bits, 01001, indicate the instruction, and the lower three bits, 011, the register.

**2. Direct Addressing**

Instructions using direct addressing consists of two bytes: op-code and address. Such instructions can access any on-chip variable or hardware register. Note that the most significant bit of the direct address determines which area in the on-chip is to be accessed. An address between 00H and 7FH accesses a location in the low-order on-chip RAM. Any address with bit 7 = 1 refers to one of the special function registers. It is not necessary to remember the addresses of these special function registers. The assembler usually understands and converts the mnemonic of a special function register, e.g. P2 for Port 2, into the appropriate address. An example of a direct addressing instruction is MOV P1, A which transfer the content of the accumulator to Port 1. The direct address of Port 1 (90H) is determined by the assembler and inserted as byte 2 of the instruction. The source of the data, the accumulator, is specified implicitly in the op-code. The complete encoding of this instruction is

1. **Indirect Addressing**

In this mode of addressing the instruction performs an operation on the data whose address is contained in register R0 or R1. Instructions using indirect addressing are single byte instructions. In 8051 assembly language the symbol @ before R0 or R1 denotes indirect addressing. An example of an indirect addressing instruction is SUBB A, @R0 This instruction performs the operation: (A) ¬ (A) – (C) – ((R0)).

1. **Immediate Addressing**

In an instruction that uses immediate addressing, the operand of the instruction is given as the byte that follows the op-code. The operand may be a numeric constant, a symbolic variable, or an arithmetic expression using constants, symbols, and operators. In assembly language we use the symbol # before an operand to denote immediate addressing. An example of an instruction using immediate addressing is

**ANL A, #77**

which performs the operation: **(A) ¬ (A) · #77**

1. **Relative Addressing**

Sometimes this is also called program counter relative addressing. This addressing mode is used only with certain jump instructions. A relative address (or offset) is an 8-bit signed value, which is added to the program counter to form the address of the next instruction executed. The range for such a jump instruction is –128 to +127 locations. Although the range is rather limited, relative addressing does offers the advantage of providing position-independent code (since absolute addresses are not used). For example, the instruction

**JZ rel**

performs the following operations:

**(PC) ¬ (PC) + 2**

**IF (A) = 0**

**THEN (PC) ¬ (PC) + rel**

**ELSE continue**

The branch destination is computed by adding the signed relative-displacement in the second instruction byte to the PC, after incrementing the PC twice.

1. **Absolute Addressing**

There are only two instructions that use this addressing: **ACALL** (absolute call) and **AJMP** (absolute jump). These instructions perform branching within the current 2K page of program memory. The branch address is obtained by successively concatenating the five high-order bits of the program counter, bits 5 – 7 of the op-code, and the second byte of the instruction. The diagram illustrate how this is done: Note that the branch destination address is within the same 2K page of program memory because the highest most five address bits are the same as those in the program counter before the branch is taken.

1. **Long Addressing**

Only two instructions use this addressing mode. These instructions are **LCALL** **addr16** and **LJMP** **addr16**. Both of these are three byte instructions with the op-code being the first byte and the following two bytes are the address high-byte and address low-byte respectively. These instructions enable the program to branch to anywhere within the full 64 K-bytes of program memory address space.

1. **Indexed Addressing**

In this mode the 16-bit address in a base register is added to a positive offset to form an effective address for the jump indirect instruction **JMP @A+DPTR**, and the two move code byte instructions **MOVC A,@A+DPTR** and **MOVC A,@A+PC**. The base register in the jump instruction is the data pointer and the positive offset is held in the accumulator. For the move instructions the base register can either be the data pointer or the program counter, and again the positive offset is in the accumulator. The operations of these three instructions are as follows:

**JMP @A+DPTR (PC) ¬ (A) +(DPTR)**

**MOVC A,@A+DPTR (A) ¬ ((A) + (DPTR))**

**MOVC A,@A+PC (PC) ¬ (PC) + 1**

**(A) ¬ ((A) + (PC))**

**4.7 Instruction Types**

The 8051 instructions are divided among five functional groups:

1. Arithmetic

2. Logical

3. Data transfer

4. Boolean variable

5. Program branching

**1. Arithmetic Instructions**

With the arithmetic instructions four addressing modes may be used. These modes are direct, indirect, register and immediate. For instance the add-with-carry instruction ADDC A, operand2 has the following four forms:

**2. Logical Instructions**

The logical instructions can perform Boolean operations on the data contained either in the accumulator or in an internal RAM location. Those logical instructions that use the accumulator as one of the operands have the same addressing modes as those found in arithmetic instruction.

**Note**

That in addition to the ORL A, direct instruction we also have the equivalent “mirror” instruction ORL direct, A (OR the accumulator to the direct byte). All such instructions execute in twelve oscillator clock periods. Apart from the logical instructions that use the accumulator as one of the operands, there are three logical instructions that perform Boolean operations directly on any byte in the internal data memory without going through the accumulator.

These three instructions take 24-oscillator clock period to execute. Note that these instructions perform what is known as “read-modify-write” operation. In a “read-modify-write” instruction the datum in the direct address location is first read, then the logical operation is performed on the read datum with the immediate byte, and finally the result of the logical operation is written back to the direct address location. The logical group of instructions also contains four rotate instructions, which operate on the contents of the accumulator, and a swap instruction **(SWAP A)**. The swap instruction is useful in BCD arithmetic manipulations.

1. **Data Transfer Instructions**

This group contains the largest number of instructions that enable us to move data within the internal RAM, move data between the internal RAM and external RAM, and three instructions that allow us to manipulate look-up tables. The following table contains some examples of data transfer instructions. Note that the stack in the 8051 is implemented in the on-chip RAM. Unlike the stack implementations in other microprocessors the stack grows “upwards” in memory, i.e. towards higher memory addresses. The execution of the PUSH instruction first increments the stack pointer, and then copies the indicated byte into the stack.

1. **Boolean Instructions**

The 8051 has a range of Boolean variable manipulating instructions which enable us to set or reset individual bits within some of the locations in the internal RAM, and some of the special function registers.

**CHAPTER 5**

**Global System For Mobile Communication**

**5.1 Introduction**

If you are in Europe or Asia and using a mobile phone, then most probably you are using GSM technology in your mobile phone.

* GSM stands for Global System for Mobile Communication. It is a digital cellular technology used.
* For transmitting mobile voice and data services. The concept of GSM emerged from a cell-based mobile radio system at Bell Laboratories in the early 1970s. GSM is the name of a standardization group established in 1982 to create a common European
* Mobile telephone standard. GSM is the most widely accepted standard in telecommunications and it is implemented
* Globally GSM is a circuit-switched system that divides each 200 kHz channel into eight 25 kHz timeslots.
* GSM operates on the mobile communication bands 900 MHz and 1800 MHz in most parts of the world. In the US, GSM operates in the bands 850 MHz and 1900 MHz. GSM owns a market share of more than 70 percent of the world's digital cellular subscribers.
* GSM makes use of narrowband Time Division Multiple Access (TDMA) technique for transmitting signals.
* GSM was developed using digital technology. It has an ability to carry 64 kbps to 120 Mbps of data rates.
* Presently GSM supports more than one billion mobile subscribers in more than 210 countries
* Throughout the world GSM provides basic to advanced voice and data services including roaming service.
* Roaming is the ability to use your GSM phone number in another GSM network. GSM digitizes and compresses data, then sends it down through a channel with two other streams of user data, each in its own timeslot.

**5.2 Why GSM?**

Listed below are the features of GSM that account for its popularity and wide accepatance. Improved spectrum efficiency

• International roaming

• Low-cost mobile sets and base stations (BSs)

• High-quality speech

• Compatibility with Integrated Services Digital Network (ISDN) and other telephone company

• services Support for new services

**5.3 GSM Architecture**

A GSM network comprises of many functional units. These functions and interfaces are explained in this chapter. The GSM network can be broadly divided into: The Mobile Station (MS) GSM is the most secured cellular telecommunications system available today. GSM has its security methods standardized. GSM maintains end-to-end security by retaining the confidentiality of calls and anonymity of the GSM subscriber. Temporary identification numbers are assigned to the subscriber’s number to maintain the privacy of the user. The privacy of the communication is maintained by applying encryption algorithms and frequency hopping that can be enabled using digital systems and signalling.

The GSM network authenticates the identity of the subscriber through the use of a challenge response mechanism. A 128-bit Random Number (RAND) is sent to the MS. The MS computes the 32-bit Signed Response (SRES) based on the encryption of the RAND with the authentication algorithm (A3) using the individual subscriber authentication key (Ki). Upon receiving the SRES from the subscriber, the GSM network repeats the calculation to verify the identity of the subscriber.

• The Base Station Subsystem (BSS)

• The Network Switching Subsystem (NSS)

• The Operation Support Subsystem (OSS)

• Given below is a simple pictorial view of the GSM architecture

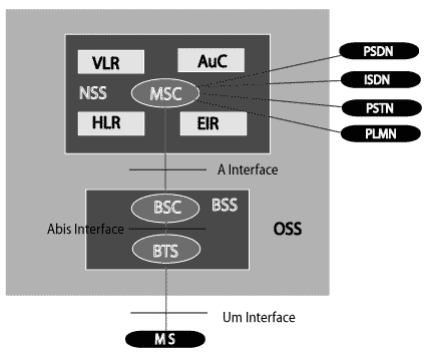


Figure 5.1 : Components of the GSM Architecture

The additional components of the GSM architecture comprise of databases and messaging systems’ functions:

* Home Location Register (HLR)
* Visitor Location Register (VLR)
* Equipment Identity Register (EIR)
* Authentication Center (AuC)
* SMS Serving Center (SMS SC)
* Gateway MSC (GMSC)
* Chargeback Center (CBC)
* Transcoder and Adaptation Unit (TRAU)

The following diagram shows the GSM network along with the added elements:

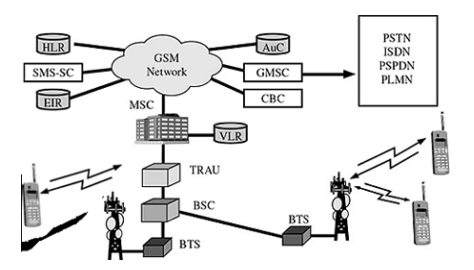


Figure 5.2 : GSM Network

**5.4 GSM Network Areas**

In a GSM network, the following areas are defined:

* **Cell:** Cell is the basic service area; one BTS covers one cell. Each cell is given a Cell Global Identity (CGI), a number that uniquely identifies the cell.
* **Location Area:** A group of cells form a Location Area (LA). This is the area that is paged when a subscriber gets an incoming call. Each LA is assigned a Location Area Identity (LAI). Each LA is served by one or more BSCs.
* **MSC/VLR Service Area**: The area covered by one MSC is called the MSC/VLR service area.
* **PLMN:** The area covered by one network operator is called the Public Land Mobile Network(PLMN). A PLMN can contain one or more MSCs.

**5.5 GSM Specifications**

The requirements for different Personal Communication Services (PCS) systems differ for each PCS network. Vital characteristics of the GSM specification are listed below: **Modulation:**

Modulation is the process of transforming the input data into a suitable format for the transmission medium. The transmitted data is demodulated back to its original form at the receiving end. The GSM uses Gaussian Minimum Shift Keying (GMSK) modulation method. **Access Methods:**

Radio spectrum being a limited resource that is consumed and divided among all the users, GSM devised a combination of TDMA/FDMA as the method to divide the bandwith among the users. In this process, the FDMA part divides the frequency of the total 25 MHz bandwidth into 124 carrier frequencies of 200 kHz bandwidth. Each BS is assigned with one or multiple frequencies, and each of this frequency is divided into eight timeslots using a TDMA scheme. Each of these slots are used for both transmission as well as reception of data. These slots are separated by time so that a mobile unit doesn’t transmit and receive data at the same time.

**Transmission Rate**:

The total symbol rate for GSM at 1 bit per symbol in GMSK produces 270.833 K symbols/second. The gross transmission rate of a timeslot is 22.8 Kbps. GSM is a digital system with an over-the-air bit rate of 270 kbps.

**Frequency Band**

The uplink frequency range specified for GSM is 933–960 MHz (basic 900 MHz band only). The downlink frequency band is 890–915 MHz (basic 900 MHz band only).

**Channel Spacing**

Channel spacing indicates the spacing between adjacent carrier frequencies. For GSM, it is 200 kHz.

**Speech Coding**

For speech coding or processing, GSM uses Linear Predictive Coding (LPC). This tool compresses the bit rate and gives an estimate of the speech parameters. When the audio signal passes through a filter, it mimics the vocal tract. Here, the speech is encoded at 13 kbps.

**Duplex Distance**

Duplex distance is the space between the uplink and downlink frequencies. The duplex distance for GSM is 80 MHz, where each channel has two frequencies that are 80 MHz apart.

**Miscellaneous**

* Frame duration: 4.615 mS
* Duplex Technique: Frequency Division Duplexing (FDD) access mode previously known as WCDMA.
* Speech channels per RF channel: 8

**5.6 GSM Addressing and Identifiers**

GSM treats the users and the equipment in different ways. Phone numbers, subscribers, and equipment identifiers are some of the known ones. There are many other identifiers that have been well-defined, which are required for the subscriber’s mobility management and for addressing the remaining network elements. Vital addresses and identifiers that are used in GSM are addressed below.

The International Mobile Station Equipment Identity (IMEI) looks more like a serial number which distinctively identifies a mobile station internationally. This is allocated by the equipment manufacturer and registered by the network operator, who stores it in the Entrepreneurs-in Residence (EIR). By means of IMEI, one recognizes obsolete, stolen, or non-functional equipment.

Following are the parts of IMEI:

• Type Approval Code (TAC): 6 decimal places, centrally assigned.

• Final Assembly Code (FAC): 6 decimal places, assigned by the manufacturer.

• Serial Number (SNR): 6 decimal places, assigned by the manufacturer.

• Spare (SP): 1 decimal place.

• Thus, IMEI = TAC + FAC + SNR + SP. It uniquely characterizes a mobile station and gives clues about the manufacturer and the date of manufacturing.

**International Mobile Subscriber Identity**

Every registered user has an original International Mobile Subscriber Identity (IMSI) with a valid IMEI stored in their Subscriber Identity Module

IMSI comprises of the following parts:

• **Mobile Country Code (MCC):** 3 decimal places, internationally standardized.

• **Mobile Network Code (MNC):** 2 decimal places, for unique identification of a mobile network within the country.

• **Mobile Subscriber Identification Number (MSIN):** Maximum 10 decimal places, identification number of the subscriber in the home mobile network.

**5.7 Mobile Subscriber ISDN Number**

The authentic telephone number of a mobile station is the Mobile Subscriber ISDN Number (MSISDN). Based on the SIM, a mobile station can have many MSISDNs, as each subscriber is assigned with a separate MSISDN to their SIM respectively. Listed below is the structure followed by MSISDN categories, as they are defined based on international ISDN number plan:

• Country Code (CC) : Up to 3 decimal places.

• National Destination Code (NDC): Typically 2–3 decimal places

• Subscriber Number (SN): Maximum 10 decimal places.

**Mobile Station Roaming Number**:

Mobile Station Roaming Number (MSRN) is an interim location dependent ISDN number, assigned to a mobile station by a regionally responsible Visitor Location Register (VLA). Using MSRN, the incoming calls are channelled to the MS. The MSRN has the same structure as the MSISDN. Country Code (CC) : of the visited network.

• National Destination Code (NDC): of the visited network

• Subscriber Number (SN): in the current mobile network.

**Location Area Identity**

Within a PLMN, a Location Area identifies its own authentic Location Area Identity (LAI). The LAI hierarchy is based on international standard and structured in a unique format as mentioned below:

• Country Code (CC): 3 decimal places.

• Mobile Network Code (MNC): 2 decimal places.

• Location Area Code (LAC): maximum 5 decimal places or maximum twice 8 bits coded in hexadecimal (LAC < FFFF).

**Temporary Mobile Subscriber Identity**

Temporary Mobile Subscriber Identity (TMSI) can be assigned by the VLR, which is responsible for the current location of a subscriber. The TMSI needs to have only local significance in the area handled by the VLR. This is stored on the network side only in the VLR and is not passed to the Home Location Register (HLR). Together with the current location area, the TMSI identifies a subscriber uniquely. It can contain up to 4 × 8 bits.

**Local Mobile Subscriber Identity**

Each mobile station can be assigned with a Local Mobile Subscriber Identity (LMSI), which is an original key, by the VLR. This key can be used as the auxiliary searching key for each mobile station within its region. It can also help accelerate the database access. An LMSI is assigned if the mobile station is registered with the VLR and sent to the HLR. LMSI comprises of four octets (4x8 bits).

**Cell Identifier**

Using a Cell Identifier (CI) (maximum 2 × 8) bits, the individual cells that are within an LA can be recognized. When the Global Cell Identity (LAI + CI) calls are combined, then it is uniquely defined.

**5.8 GSM Operations**

Once a Mobile Station initiates a call, a series of events takes place. Analyzing these events can give an insight into the operation of the GSM system.

**Mobile Phone to Public Switched Telephone Network (PSTN)**

When a mobile subscriber makes a call to a PSTN telephone subscriber, the following sequence of events takes place:

1. The MSC/VLR receives the message of a call request.

2. The MSC/VLR checks if the mobile station is authorized to access the network. If so, the mobile station is activated. If the mobile station is not authorized, then the service will be denied.

3. MSC/VLR analyzes the number and initiates a call setup with the PSTN.

4. MSC/VLR asks the corresponding BSC to allocate a traffic channel (a radio channel and a timeslot).

5. The BSC allocates the traffic channel and passes the information to the mobile station.

6. The called party answers the call and the conversation takes place.

7. The mobile station keeps on taking measurements of the radio channels in the present cell and the neighbouring cells and passes the information to the BSC. The BSC decides if a handover is required. If so, a new traffic channel is allocated to the mobile station and the handover takes place. If handover is not required, the mobile station continues to transmit in the same frequency.

**PSTN to Mobile Phone**

When a PSTN subscriber calls a mobile station, the following sequence of events takes place:

1. The Gateway MSC receives the call and queries the HLR for the information needed to route the call to the serving MSC/VLR.

2. The GMSC routes the call to the MSC/VLR.

3. The MSC checks the VLR for the location area of the MS.

4. The MSC contacts the MS via the BSC through a broadcast message, that is, through a paging request.

5. The MS responds to the page request.

6. The BSC allocates a traffic channel and sends a message to the MS to tune to the channel. The MS generates a ringing signal and, after the subscriber answers, the speech connection is established.

7. Handover, if required, takes place, as discussed in the earlier case.

To transmit the speech over the radio channel in the stipulated time, the MS codes it at the rate of 13 Kbps. The BSC transcodes the speech to 64 Kbps and sends it over a land link or a radio link to the MSC. The MSC then forwards the speech data to the PSTN. In the reverse direction, the speech is received at 64 Kbps at the BSC and the BSC transcodes it to 13 Kbps for radio transmission. GSM supports 9.6 Kbps data that can be channeled in one TDMA timeslot. To supply higher data rates, many enhancements were done to the GSM standards (GSM Phase 2 and GSM Phase 2+).

**5.9 GSM Protocol Stack**

GSM architecture is a layered model that is designed to allow communications between two different systems. The lower layers assure the services of the upper-layer protocols. Each layer passes suitable notifications to ensure the transmitted data has been formatted, transmitted, and received accurately.

**MS Protocols**

Based on the interface, the GSM signalling protocol is assembled into three general layers:

1. Layer 1: The physical layer. It uses the channel structures over the air interface.

2. Layer 2: The data-link layer. Across the Um interface, the data-link layer is a modified version of the Link Access Protocol for the D channel (LAP-D) protocol used in ISDN, called Link Access Protocol on the Dm channel (LAP-Dm). Across the A interface, the Message Transfer Part (MTP), Layer 2 of SS7 is used.

3. Layer 3: GSM signalling protocol’s third layer is divided into three sublayers:

• Radio Resource Management (RR)

• Mobility Management (MM), and

• Connection Management (CM).

**MS to BTS Protocols**

The RR layer is the lower layer that manages a link, both radio and fixed, between the MS and the MSC. For this formation, the main components involved are the MS, BSS, and MSC. The responsibility of the RR layer is to manage the RR-session, the time when a mobile is in a dedicated mode, and the radio channels including the allocation of dedicated channels. The MM layer is stacked above the RR layer. It handles the functions that arise from the mobility of the subscriber, as well as the authentication and security aspects. Location management is concerned with the procedures that enable the system to know the current location of a powered on MS so that incoming call routing can be completed.

The CM layer is the topmost layer of the GSM protocol stack. This layer is responsible for Call Control, Supplementary Service Management, and Short Message Service Management. Each of these services are treated as individual layer within the CM layer. Other functions of the CC sublayer include call establishment, selection of the type of service (including alternating between services during a call), and call release.

**BSC Protocols**

The BSC uses a different set of protocols after receiving the data from the BTS. The Abis interface is used between the BTS and BSC. At this level, the radio resources at the lower portion of Layer 3 are changed from the RR to the Base Transceiver Station Management (BTSM). The BTS management layer is a relay function at the BTS to the BSC.

The RR protocols are responsible for the allocation and reallocation of traffic channels between the MS and the BTS. These services include controlling the initial access to the system, paging for MT calls, the handover of calls between cell sites, power control, and call termination. The BSC still has some radio resource management in place for the frequency coordination, frequency allocation, and the management of the overall network layer for the Layer 2 interfaces.

To transit from the BSC to the MSC, the BSS mobile application part or the direct application part is used, and SS7 protocols is applied by the relay, so that the MTP 1-3 can be used as the prime architecture.

**MSC Protocols**

At the MSC, starting from the BSC, the information is mapped across the A interface to the MTP Layers 1 through 3. Here, Base Station System Management Application Part (BSS MAP) is said to be the equivalent set of radio resources. The relay process is finished by the layers that are stacked on top of Layer 3 protocols, they are BSS MAP/DTAP, MM, and CM. This completes the relay process. To find and connect to the users across the network, MSCs interact using the control signalling network.

**CHAPTER 6**

**Global Positioning System**

**6.1 Introduction**

Traditional methods of surveying and navigation resort to tedious field and astronomical observation for deriving positional and directional information. Diverse field conditions, seasonal variation and many unavoidable circumstances always bias the traditional field approach. However, due to rapid advancement in electronic systems, every aspect of human life is affected to a great deal. Field of surveying and navigation is tremendously benefited through electronic devices. Many of the critical situations in surveying/navigation are now easily and precisely solved in short time.

Astronomical observation of celestial bodies was one of the standard methods of obtaining coordinates of a position. This method is prone to visibility and weather condition and demands expert handling. Attempts have been made by USA since early 1960’s to use space based artificial satellites. System TRANSIT was widely used for establishing a network of control points over large regions.

Establishment of modern geocentric datum and its relation to local datum was successfully achieved through TRANSIT. Rapid improvements in higher frequently transmission and precise clock signals along with advanced stable satellite technology have been instrumental for the development of global positioning system.

The NAVSTAR GPS (Navigation System with Time and Ranging Global Positioning System) is a satellite based radio navigation system providing precise three- dimensional position, course and time information to suitably equipped user. GPS has been under development in the USA since 1973. The US department of Defence as a worldwide navigation and positioning resource for military as well as civilian use for 24 hours and all weather conditions primarily developed it. In its final configuration, NAVSTAR GPS consists of 21 satellites (plus 3 active spares) at an altitude of 20200 km above the earth’s surface (Fig. 1). These satellites are so arranged in orbits to have atleast four satellites visible above the horizon anywhere on the earth, at any time of the day.

GPS Satellites transmit at frequencies L1=1575.42 MHz and L2=1227.6 MHz modulated with two types of code viz. P-code and C/A code and with navigation message. Mainly two types of observable are of interest to the user. In pseudo ranging the distance between the satellite and the GPS receiver plus a small corrective term for receiver clock error is observed for positioning whereas in carrier phase techniques, the difference between the phase of the carrier signal transmitted by the satellite and the phase of the receiver oscillator at the epoch is observed to derive the precise information.

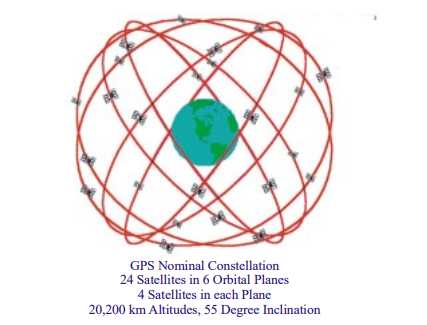


Figure 6.1 : The Global Positioning System (GPS), 21-satellite configuration

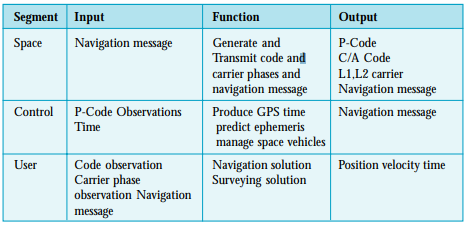
The GPS satellites act as reference points from which receivers on the ground detect their position. The fundamental navigation principle is based on the measurement of pseudo ranges between the user and four satellites. Ground stations precisely monitor the orbit of every satellite and by measuring the travel time of the signals transmitted from the satellite four distances between receiver and satellites will yield accurate position, direction and speed. Though three-range measurements are sufficient, the fourth observation is essential for solving clock synchronization error between receiver and satellite. Thus, the term “pseudoranges” is derived. The secret of GPS measurement is due to the ability of measuring carrier phases to about 1/100 of a cycle equaling to 2 to 3 mm in linear distance. Moreover the high frequency L1 and L2 carrier signal can easily penetrate the ionosphere to reduce its effect. Dual frequency observations are important for large station separation and for eliminating most of the error parameters.

GPS has been designed to provide navigational accuracy of ±10 m to ±15 m. However, sub meter accuracy in differential mode has been achieved and it has been proved that broad varieties of problems in geodesy and geodynamics can be tackled through GPS.

Versatile use of GPS for a civilian need in following fields have been successfully practiced viz. navigation on land, sea, air, space, high precision kinematics survey on the ground, cadastral surveying, geodetic control network densification, high precision aircraft positioning, photogrammetric without ground control, monitoring deformations, hydro graphic surveys, active control survey and many other similar jobs related to navigation and positioning,. The outcome of a typical GPS survey includes geocentric position accurate to 10 m and relative positions between receiver locations to centimetre level or better.

**6.2 Segments of GPS**

For better understanding of GPS, we normally consider three major segments viz. space segment, Control segment and User segment. Space segment deals with GPS satellites systems, Control segment describes ground based time and orbit control prediction and in User segment various types of existing GPS receiver and its application is dealt Table 2 gives a brief account of the function and of various segments along with input and output information. Functions of various segments of GPS



GLONASS (Global Navigation & Surveying System) a similar system to GPS is being developed by former Soviet Union and it is considered to be a valuable complementary system to GPS for future application.

**6.3 Space Segment**

Space segment will consist 21 GPS satellites with an addition of 3 active spares. These satellites are placed in almost six circular orbits with an inclination of 55 degree. Orbital height of these satellites is about 20,200 km corresponding to about 26,600 km from the semi major axis. Orbital period is exactly 12 hours of sidereal time and this provides repeated satellite configuration every day advanced by four minutes with respect to universal time. Final arrangement of 21 satellites constellation known as “Primary satellite constellation” There are six orbital planes A to F with a separation of 60 degrees at right ascension (crossing at equator). The position of a satellite within a particular orbit plane can be identified by argument of latitude or mean anomaly M for a given epoch.

GPS satellites are broadly divided into three blocks : Block-I satellite pertains to development stage, Block II represents production satellite and Block IIR are replenishment/spare satellite.

Under Block-I, NAVSTAR 1 to 11 satellites were launched before 1978 to 1985 in two orbital planes of 63-degree inclination. Design life of these prototype test satellites was only five years but the operational period has been exceeded in most of the cases. The first Block-II production satellite was launched in February 1989 using channel Douglas Delta 2 booster rocket. A total of 28 Block-II satellites are planned to support 21+3 satellite configuration. Block-II satellites have a designed lifetime of 5-7 years. To sustain the GPS facility, the development of follow-up satellites under Block-II R has started. Twenty replenishment satellites will replace the current block-II satellite as and when necessary. These GPS satellites under Block-IR have additional ability to measure distances between satellites and will also compute ephemeris on board for real time information gives a schematic view of Block-II satellite. Electrical power is generated through two solar panels covering a surface area of 7.2 square meter each. However, additional battery backup is provided to provide energy when the satellite moves into earth’s shadow region. Each satellite weighs 845kg and has a propulsion system for positional stabilization and orbit maneuvers. GPS satellites have a very high performance frequency standard with an accuracy of between 1X10-12 to 1X10-13 and are thus capable of creating precise time base.

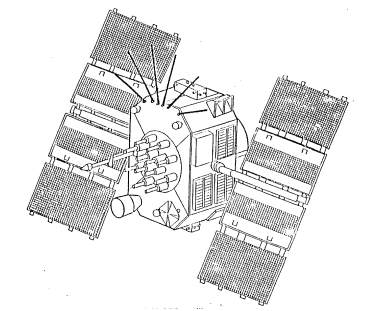


Figure 6.2: Schematic view of a Block II GPS satellite

Block-I satellites were partly equipped with only quartz oscillators but Block-II satellites have two caesium frequency standards and two rubidium frequency standards. Using fundamental frequency of 10.23 MHz, two carrier frequencies are generated to transmit signal codes.

**6.4 Observation Principle and Signal Structure**

NAVSTAR GPS is a one-way ranging system i.e. signals are only transmitted by the satellite. Signal travel time between the satellite and the receiver is observed and the range distance is calculated through the knowledge of signal propagation velocity. One way ranging means that a clock reading at the transmitted antenna is compared with a clock reading at the receiver antenna. But since the two clocks are not strictly synchronized, the observed signal travel time is biased with systematic synchronization error. Biased ranges are known as pseudo ranges. Simultaneous observations of four pseudo ranges are necessary to determine X, Y, Z coordinates of user antenna and clock bias.

Real time positioning through GPS signals is possible by modulating carrier frequency with Pseudorandom Noise (PRN) codes. These are sequence of binary values (zeros and ones or +1 and -1) having random character but identifiable distinctly.

Thus pseudo ranges are derived from travel time of an identified PRN signal code. Two different codes viz. P-code and C/A code are in use. P means precision or protected and C/A means clear/acquisition or coarse acquisition. P- code has a frequency of 10.23 MHz. This refers to a sequence of 10.23 million binary digits or chips per second.

This frequency is also referred to as the chipping rate of P-code. Wavelength corresponding to one chip is 29.30m. The P-code sequence is extremely long and repeats only after 266 days. Portions of seven days each are assigned to the various satellites. As a consequence, all satellite can transmit on the same frequency and can be identified by their unique one-week segment. This technique is also called as Code Division Multiple Access (CDMA). P-code is the primary code for navigation and is available on carrier frequencies L1 and L2.

The C/A code has a length of only one millisecond; its chipping rate is 1.023 MHz with corresponding wavelength of 300 meters. C/A code is only transmitted on L1 carrier.

GPS receiver normally has a copy of the code sequence for determining the signal propagation time. This code sequence is phase-shifted in time step by-step and correlated with the received code signal until maximum correlation is achieved. The necessary phase-shift in the two sequences of codes is a measure of the signal travel time between the satellite and the receive antennas. This technique can be explained as code phase observation.

For precise geodetic applications, the pseudo ranges should be derived from phase measurements on the carrier signals because of much higher resolution. Problems of ambiguity determination are vital for such observations. The third type of signal transmitted from a GPS satellite is the broadcast message sent at a rather slow rate of 50 bits per second (50 bps) and repeated every 30 seconds. Chip sequence of P-code and C/A code are separately combined with the stream of message bit by binary addition i.e. the same value for code and message chip gives 0 and different values result in 1.

With a bit rate of 50 bps and a cycle time of 30 seconds, the total information content of a navigation data set is 1500 bits. The complete data frame is subdivided into five sub frames of six-second duration comprising 300 bits of information. Each sub frame contains the data words of 30 bits each. Six of these are control bits. The first two words of each sub frame are the Telemetry Work (TLM) and the C/A-P-Code Hand over Work (HOW). The TLM work contains a synchronization pattern, which facilitates the access to the navigation data

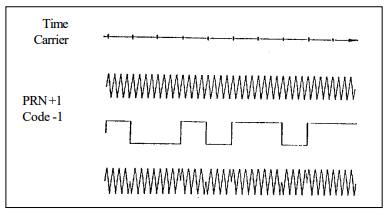
The main features of all three signal types used in GPS observation via carrier, code and data signals. The signal structure permits both the phase and the phase shift (Doppler effect) to be measured along with the direct signal propagation. The necessary bandwidth is achieved by phase modulation of the PRN code.

Figure 6.3 : Generation of GPS Signals

**6.5 Structure of the GPS Navigation Data**

Structure of GPS navigation data (message) is shown in Fig. 7. The user has to decode the data signal to get access to the navigation data. For on line navigation purposes, the internal processor within the receiver does the decoding. Most of the manufacturers of GPS receiver provide decoding software for post processing purposes.

With a bit rate of 50 bps and a cycle time of 30 seconds, the total information content of a navigation data set is 1500 bits. The complete data frame is subdivided into five sub frames of six-second duration comprising 300 bits of information. Each sub frame contains the data words of 30 bits each. Six of these are control bits. The first two words of each sub frame are the Telemetry Work (TLM) and the C/A-P-Code Hand over Work (HOW). The TLM work contains a synchronization pattern, which facilitates the access to the navigation data.

|  |  |
| --- | --- |
| **Data Block I** | appears in the first sub frame and contains the clock coefficient/bias. |
| **Data Block II** | appears in the second and third sub frame and contains all necessary parameters for the computation of the satellite coordinates |
| **Data Block III** | appears in the fourth and fifth sub frames and contains the almanac data with clock and ephemeris parameter for all available satellite of the GPS system. This data block includes also ionospheric correction parameters and particular alphanumeric information for authorized users. |

Unlike the first two blocks, the sub frame four and five are not repeated every 30 seconds.

**6.6 International Limitation of the System Accuracy**

Since GPS is a military navigation system of US, a limited access to the total system accuracy is made available to the civilian users. The service available to the civilians is called Standard Positioning System (SPS) while the service available to the authorized users is called the Precise Positioning Service (PPS). Under current policy the accuracy available to SPS users is 100m, 2DRMS and for PPS users it is 10 to 20 meters in 3D. Additional limitation viz. Anti-Spoofing (AS), and Selective Availability (SA) was further imposed for civilian users. Under AS, only authorized users will have the means to get access to the P-code. By imposing SA condition, positional accuracy from Block-II satellite was randomly offset for SPS users. Since May 1, 2000 according to declaration of US President, SA is switched off for all users.

The GPS system time is defined by the caesium oscillator at a selected monitor station. However, no clock parameter are derived for this station. GPS time is indicated by a week number and the number of seconds since the beginning of the current week. GPS time thus varies between 0 at the beginning of a week to 6,04,800 at the end of the week. The initial GPS epoch is January 5, 1980 at 0 hours Universal Time. Hence, GPS week starts at Midnight (UT) between Saturday and Sunday. The GPS time is a continuous time scale and is defined by the main clock at the Master Control Station (MCS). The leap seconds is UTC time scale and the drift in the MCS clock indicate that GPS time and UTC are not identical. The difference is continuously monitored by the control segment and is broadcast to the users in the navigation message. Difference of about 7 seconds was observed in July, 1992.

**6.7 Basic Concept Of Gps Receiver And Its Components**

The main components of a GPS receiver are shown in Fig. 10. These are:

* Antenna with pre-amplifier
* RF section with signal identification and signal processing
* Micro-processor for receiver control, data sampling and data processing
* Precision oscillator
* Power supply
* User interface, command and display panel
* Memory, data storage

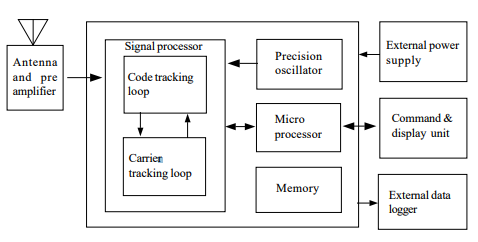
**Antenna**

Figure 6.4 : Major components of a GPS receiver

Sensitive antenna of the GPS receiver detects the electromagnetic wave signal transmitted by GPS satellites and converts the wave energy to electric current] amplifies the signal strength and sends them to receiver electronics.

**Microprocessor**

To control the operation of a GPS receiver, a microprocessor is essential for acquiring the signals, processing of the signal and the decoding of the broadcast message. Additional capabilities of computation of on-line position and velocity, conversion into a given local datum or the determination of waypoint information are also required. In future more and more user relevant software will be resident on miniaturized memory chips.

**Precision Oscillator**

A reference frequency in the receiver is generated by the precision oscillator. Normally, less expensive, low performance quartz oscillator is used in receivers since the precise clock information is obtained from the GPS satellites and the user clock error can be eliminated through double differencing technique when all participating receivers observe at exactly the same epoch. For navigation with two or three satellites only an external high precision oscillator is used. The keypad is used to enter commands, external data like station number or antenna height or to select a menu operation. The display indicates computed coordinates, visible satellites, data quality indices and other suitable information.

**Power Supply**

First generation GPS receivers consumed very high power, but modern receivers are designed to consume as little energy as possible. Most receivers have an internal rechargeable Nickel-Cadmium battery in addition to an external power input. Caution of low battery signal prompts the user to ensure adequate arrangement of power supply.

**Accuracy**

In general, an SPS receiver can provide position information with an error of less than 25 meter and velocity information with an error less than 5 meters per second. Upto 2 May 2000 U.S Government has activated Selective Availability (SA) to maintain optimum military effectiveness. Selective Availability inserts random errors into the ephemeris information broadcast by the satellites, which reduces the SPS accuracy to around 100 meters.

**Differential Theory**

Differential positioning is technique that allows overcoming the effects of environmental errors and SA on the GPS signals to produce a highly accurate position fix. This is done by determining the amount of the positioning error and applying it to position fixes that were computed from collected data.

**6.8 Applications of GPS**

* Providing Geodetic control.
* Survey control for Photogrammetric control surveys and mapping.
* Finding out location of offshore drilling.
* Pipeline and Power line survey.
* Navigation of civilian ships and planes.
* Crustal movement studies.
* Offshore positioning: shipping, offshore platforms, fishing boats etc.

**CHAPTER 7**

**Liquid Crystal Displays**

**7.1 Cathode Ray Tubes**

We always use devices made up of Liquid Crystal Displays (LCDs) like computers, digital watches and also DVD and CD players. They have become very common and have taken a giant leap in the screen industry by clearly replacing the use of Cathode Ray Tubes (CRT). CRT draws more power than LCD and are also bigger and heavier. All of us have seen an LCD, but no one knows the exact working of it. Let us take a look at the working of an LCD.

We get the definition of LCD from the name “Liquid Crystal” itself. It is actually a combination of two states of matter – the solid and the liquid. They have both the properties of solids and liquids and maintain their respective states with respect to another. Solids usually maintain their state unlike liquids who change their orientation and move everywhere in the particular liquid. Further studies have showed that liquid crystal materials show more of a liquid state than that of a solid. It must also be noted that liquid crystals are more heat sensitive than usual liquids. A little amount of heat can easily turn the liquid crystal into a liquid. This is the reason why they are also used to make thermometers.

## **7.2 Basics of LCD Displays**

The liquid-crystal display has the distinct advantage of having a low power consumption than the LED. It is typically of the order of microwatts for the display in comparison to the some order of mill watts for LEDs. Low power consumption requirement has made it compatible with MOS integrated logic circuit. Its other advantages are its low cost, and good contrast. The main drawbacks of [**LCD**](http://www.circuitstoday.com/lcd-liquid-crystal-displays)s are additional requirement of light source, a limited temperature range of operation (between 0 and 60° C), low reliability, short oper­ating life, poor visibility in low ambient lighting, slow speed and the need for an ac drive.

**7.3 Basic structure of an LCD**

A liquid crystal cell consists of a thin layer (about 10 u m) of a liquid crystal sand­wiched between two glass sheets with transparent elec­trodes deposited on their inside faces. With both glass sheets transparent, the cell is known astransitive type cell. When one glass is transparent and the other has a reflective coating, the cell is called reflective type. The LCD does not produce any illumination of its own. It, in fact, depends entirely on illumination falling on it from an external source for its visual effect

### 7.4 Types of LCD/Liquid Crystal Displays.

**Two types of display available are dynamic scatter­ing display and field effect display.**

When dynamic scattering display is energized, the molecules of energized area of the display become tur­bulent and scatter light in all directions. Consequently, the activated areas take on a frosted glass appearance resulting in a silver display. Of course, the unenergized areas remain translucent.

Field effect LCD contains front and back polarizer’s at right angles to each other. Without electrical exci­tation, the light coming through the front polarizer is rotated 90° in the fluid.

Now, let us take a look at the different varieties of liquid crystals that are available for industrial purposes. The most usable liquid crystal among all the others is the nematic phase liquid crystals.

### 7.5 Nematic Phase LCD

The greatest advantage of a nematic phase liquid crystal substance is that it can bring about predictable controlled changes according to the electric current passed through them. All the liquid crystals are according to their reaction on temperature difference and also the nature of the substance.

Twisted Nematics, a particular nematic substance is twisted naturally. When a known voltage is applied to the substance, it gets untwisted in varying degrees according to our requirement. This in turn is useful in controlling the passage of light. A nematic phase liquid crystal can be again classified on the basis in which the molecules orient themselves in respect to each other. This change in   orientation mainly depends on the director, which can be anything ranging from a magnetic field to a surface with microscopic grooves. Classification includes Smectic and also cholesteric. Smectic can be again classified as smectic C, in which the molecules in each layer tilt at an angle from the previous layer. Cholesteric, on the other hand has molecules that twist slightly from one layer to the next, causing a spiral like design. There are also combinations of these two called Ferro-electric liquid crystals (FLC), which include cholesteric molecules in a smectic C type molecule so that the spiral nature of these molecules allows the microsecond switching response time. This makes FLCs to be of good use in advanced displays.

Liquid crystal molecules are further classified into thermotropic and lyotropic crystals. The former changes proportionally with respect to changes in pressure and temperature. They are further divided into nematic and isotropic. Nematic liquid crystals have a fixed order of pattern while isotropic liquid crystals are distributed randomly.  The lyotropic crystal depends on the type of solvent they are mixed with. They are therefore useful in making detergents and soaps.

**7.6 Making of LCD**

* Though the making of LCD is rather simple there are certain facts that should be noted while making it.
* The basic structure of an LCD should be controllably changed with respect to the applied electric current.
* The light that is used on the LCD can be polarized.
* Liquid crystals should be able to both transmit and change polarized light.
* There are transparent substances that can conduct electricity.

To make an LCD, you need to take two polarized glass pieces. The glass which does not have a polarized film on it must be rubbed with a special polymer which creates microscopic grooves in the surface. It must also be noted that the grooves are on the same direction as the polarizing film. Then, all you need to do is to add a coating of nematic liquid crystals to one of the filters. The grooves will cause the first layer of molecules to align with the filter’s orientation. At right angle to the first piece, you must then add a second piece of glass along with the polarizing film. Till the uppermost layer is at a 90-degree angle to the bottom, each successive layer of TN molecules will keep on twisting. The first filter will naturally be polarized as the light strikes it at the beginning.   Thus the light passes through each layer and is guided on to the next with the help of molecules. When this happens, the molecules tend to change the plane of vibration of the light to match their own angle. When the light reaches the far side of the liquid crystal substance, it vibrates at the same angle as the final layer of molecules. The light is only allowed an entrance if the second polarized glass filter is same as the final layer. Take a look at the figure below.

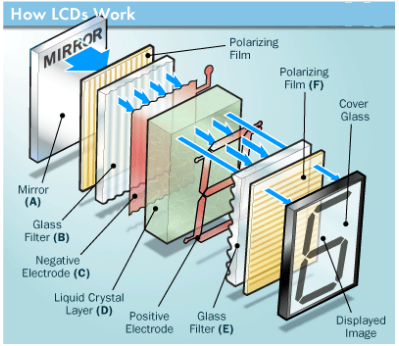
The main principle behind liquid crystal molecules is that when an electric current is applied to them, they tend to untwist. This causes a change in the light angle passing through them. This causes a change in the angle of the top polarizing filter with respect to it. So little light is allowed to pass through that particular area of LCD. Thus that area becomes darker comparing to others.

Figure 7.1 : Working of LCD

**7.7 Colour Liquid Crystal Display**

Colour LCDs are those that can display pictures in colours. For this to be possible there must be three sub-pixels with red, green and blue colour filters to create each colour pixel. For combining these sub-pixels these LCDs should be connected to a large number of transistors. If any problem occurs to these transistors, it will cause a bad pixel.

One of the main disadvantages of these types of LCDs is the size. Most manufacturers try to reduce the height than gain it. This is because more transistors and greater pixels will be needed to increase the length. This will increase the probability of bad pixels.  It is very difficult or also impossible to repair a LCD with bad pixels. This will highly affect the sale of LCDs.

**CHAPTER 8**

**BUZZER**

**8.1 General Description**

A buzzer or beeper is an audio signalling device, which may be mechanical, electromechanical, or piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers and confirmation of user input such as a mouse click or keystroke. Buzzer is an integrated structure of electronic transducers, DC power supply, widely used in computers, printers, copiers, alarms, electronic toys, automotive electronic equipment, telephones, timers and other electronic products for sound devices. Active buzzer 5V Rated power can be directly connected to a continuous sound, this section dedicated sensor expansion module and the board in combination, can complete a simple circuit design, to "plug and play."

**8.2 Specifications**

* On-board passive buzzer
* On-board 8550 triode drive
* Can control with single-chip microcontroller IO directly
* Working voltage: 5V
* Board size: 22 (mm) x12 (mm)

**8.3 Pin Configuration**

* VCC
* Input
* Ground
  1. **Schematic Diagram**

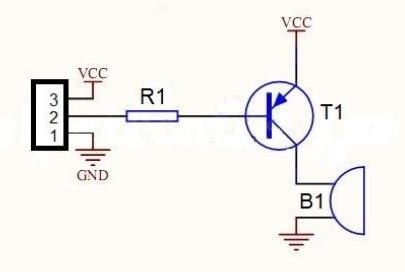


Figure 8.1 : Schematic Diagram

**CHAPTER 9**

**VIBRATION SENSOR**

**9.1 Tilt Switches**

Tilt sensors allow you to detect orientation or inclination. They are small, inexpensive, low-power and easy-to-use. If used properly, they will not wear out. Their simplicity makes them popular for toys, gadgets and appliances. Sometimes they are referred to as "mercury switches", "tilt switches" or "rolling ball sensors" for obvious reasons. They are usually made by a cavity of some sort (cylindrical is popular, although not always) and a conductive free mass inside, such as a blob of mercury or rolling ball. One end of the cavity has two conductive elements (poles). When the sensor is oriented so that that end is downwards, the mass rolls onto the poles and shorts them, acting as a switch throw.

Tilt switches used to be made exclusively of mercury, but are rarer now since they are recognized as being extremely toxic. The benefits of mercury is that the blob is dense enough that it doesn’t bounce and so the switch isn’t susceptible to vibrations. On the other hand, ball-type sensors are easy to make, wont shatter, and pose no risk of pollution.

Figure 9.1 : 1 Tilt Switch

While not as precise or flexible as a full accelerometer, tilt switches can detect motion or orientation simply. Another benefit to them is that the big ones can switch power on their own. Accelerometers, on the other hand, output digital or analog voltage that must then be analyzed with extra circuitry. They are usually made by a cavity of some sort (cylindrical is popular, although not always) and a conductive free mass inside, such as a blob of mercury or rolling ball. This switch is normally an open circuit when it is at rest in any position, but becomes a momentary short when it is vibrated. When the switch is vibrated, your detector should notice the momentary short, and set off an ultrasonic alarm for 5-10 seconds

**9.2 Basic Stats**

These stats are for the tilt sensor in the Adafruit shop which is very much like the 107-2006-EV . Nearly all will have slightly different sizes & specifications, although they all pretty much work the same. If there's a datasheet, you'll want to refer to it

* Size: Cylindrical, 4mm (0.16") diameter & 12mm (0.45") long.
* Sensitivity range: > +-15 degrees
* Lifetime: 50,000+ cycles (switches)
* Power supply: Up to 24V, switching less than 5mA

Design a vibration detector that uses the mercury switch. This switch is normally an open circuit when it is at rest in any position, but becomes a momentary short when it is vibrated. When the switch is vibrated, your detector should notice the momentary short, and set off an ultrasonic alarm for 5-10 seconds. Since the alarm is not audible to humans, you will need to provide a visual indicator for when the alarm is activated. We are suggesting that you use an LED for this purpose. For testing purposes, we would also like your oscillator to have a variable frequency so that it may be adjusted down into the human audible range.

**CHAPTER 10**

**MAX 232**

**10.1 Introduction**

The MAX232 is a hardware layer protocol converterIC manufactured by the Maxim Corporation. Commonly known as a RS-232 Transceiver, it consists of a pair of drivers and a pair of receivers. At a very basic level, the driver converts TTL and CMOS voltage levels to TIA/EIA-232-E levels, which are compatible for serial port communications. The receiver performs the reverse conversion.

Used in embedded microcontroller systems, and computers, this IC has been one of the most popular components in production for well over two decades. If you have a microcontroller circuit that requires communication through a serial port, then this is the chip to use. This is a versatile IC, which is one of those wonderful components that solve so many signal conversion problems.

**10.2 Serial Port Protocol**

RS-232 is a serial communication protocol defined by the EIA/TIA-232-E specification in 1962. The letters "RS" refer to Recommended Standards.

This protocol requires a voltage between -3 V to -15 V to represent binary 1, and a voltage between +3 V to +15 V to represent binary 0. For CMOS and TTL communication, this is incompatible since TTL uses 5 V to represent binary 1 and 0 V to represent binary 0. This chip therefore performs the necessary protocol conversion of the electrical voltage levels in both directions.

One interesting thing to note is that RS232 communication also consists of control signals such as request-to-send (RTS), data-terminal-ready (DTR), data-set-ready (DSR), and clear-to-send (CTS). Usually, for simple projects, it is only the transmit (Tx), and receive (Rx) lines that are converted.

## **10.3 Dual Drivers and Dual Receivers**

As you can see, there are two drivers, and two receivers in the MAX232 IC package. This can be confusing for students, as it makes the chip appear more complicated than it really is, but it is actually very easy since for most applications we generally use only one driver and one receiver.

I tend to use pin 7, pin 8, pin 9, and pin 10 for most of my circuits because I am used to using those. The other driver and receiver are not used, and these extra circuits are very useful as spares.

The RS-232(X) is a communication cable, commonly used for transferring and receiving the serial data between two devices.  It supports both synchronous and asynchronous data transmissions. Many devices in the industrial environment are still using RS-232 communication cable. Rs-232 cable is used to identify the difference of two signal levels between logic 1 and logic 0. The logic 1 is represented by the -12V and logic 0 is represented the +12V.  The RS-232 cable works at different baud rates like 9600 bits/s, 2400bits/s, 4800bits/s etc. The RS-232 cable has two terminal devices namely Data Terminal Equipment and Data communication Equipment. Both device will sends and receives the signals. The  data terminal equipment is computer terminal and data communication Equipment is modems, or controllers etc.

Now a day’s most of personal computers have two serial ports and one parallel port (RS232). These two types of ports are used for communicating with external devices and they work in different ways. The parallel port sends and receives the 8-bit data at a time over eight separate wires and this transfers the data very quickly, the parallel ports are typically used to connect a printer to [a PC](http://www.edgefxkits.com/pc-based-electrical-load-control).

A serial port sends and receives one bit data at a time over one wire and it transfer data very slowly. The RS-232 stands for recommended slandered and 232 is [a number](http://www.edgefxkits.com/energy-meter-billing-with-load-control-over-gsm-with-user-programable-number-features) X is indicates the latest version like RS-232c, RS232s.

The most commonly used type of serial cable connectors are 9-pin connector DB9 and 25-pin connector DB-25. Each of them may be male or female type. Now a days most of computers use DB9 connector for asynchronous data exchange. The maximum length of RS-232 cable is 50ft.

**10.4 Pin Description**

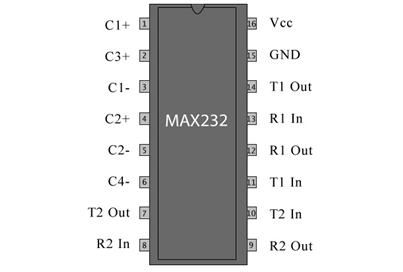


Figure 10.1 :Pin Diagram of max 232

It is a 25-pin connector, each pin has its own function is as follows.

**PIN 1*: (*Protective Ground);** It is a ground Pin.

**PIN 2:**Transmit Data.

**PIN 3:**Receive Data.

**PIN 2 & PIN 3:**These pins are most important pins for data transmitting and receiving. The 1 & 2-pins are used to data transmitting and pin-3 used to data receiving purpose.

**PIN 4**: Request to send.

**Pin 5***:*Clear to send.

**PIN 6*:***Data Set Ready.

**PIN20*:***Data terminal Ready.

**PIN 4, PIN 5, PIN 6, PIN 20:**These pins are the handshaking pins(flow of control).Normally terminals cannot transmit the data until clear to send transmission is received from the DCE.

**PIN 7:**This pin is the common reference for all signals, including data, timing, and control signals. The DCE and DTE works properly across the serial interface and the pin-7 must be connected both ends without interface would not work.

**PIN 8**: This pin also known as received line signal detect or carrier detect. This signal is activated when a suitable carrier is established between the local and remote DCE devices.

**PIN9:**This pin is DTE serial connector, this signal follows the incoming ring to an extent. Normally this signal is used by DCE auto answer mode.

**PIN 10:**Test Pin.

**PIN 11:**stand by select**.**

**PIN 12:**Data Carrier Detect.

**PIN 13:**Clear to send.

**PIN 14:**Transmit data.

**PIN 15:**Transmit clock.

**PIN 17:**Receive clock.

**PIN 24:**External Clock.

**PIN 15, 17, 24**; Synchronous modems use the signals on these pins. These pins are control bit timing.

**PIN 16:**Receive data.

**PIN 18:**Test Pin.

**PIN 19:**Request to send.

**PIN 21: (**Signal Quality Detector); This pin Indicates the quality of the received carrier signal because the transmitting modem must be send 0 or either 1 at each bit time , the modem controls the timing of the bits from the DTE.

**PIN 22: (**Ring Indicator): The ringing indicator means the DCE informs the DTE that the phone is ringing. All the modems designed for direct connected to the phone network equipped with auto answer.

**PIN 23:**Data Signal Rate Detector

**CHAPTER 11**

**KEIL SOFTWARE**

**11.1 Introduction**

Getting Started, describes the µVision IDE, µVision Debugger and Analysis Tools, the simulation, and debugging and tracing capabilities. In addition to describing the basic behaviour and basic screens of µVision, this book provides a comprehensive overview of the supported microcontroller architecture types, their advantages and highlights, and supports you in selecting the appropriate target device. This book incorporates hints to help you to write better code. As with any Getting Started book, it does not cover every aspect and the many available configuration options in detail. We encourage you to work through the examples to get familiar with µVision and the components delivered.

The Keil Development Tools are designed for the professional software developer, however programmers of all levels can use them to get the most out of the embedded microcontroller architectures that are supported. Tools developed by Keil endorse the most popular microcontrollers and are distributed in several packages and configurations, dependent on the architecture.

* MDK-ARM: Microcontroller Development Kit, for several ARM7, ARM9, and Cortex-Mx based devices
* PK166: Keil Professional Developer’s Kit, for C166, XE166, and XC2000 devices
* DK251: Keil 251 Development Tools, for 251 devices
* PK51: Keil 8051 Development Tools, for Classic & Extended 8051 devices

In addition to the software packages, Keil offers a variety of evaluation boards, USB-JTAG adapters, emulators, and third-party tools, which completes the range of products. The following illustrations show the generic component blocks of µVision in conjunction with tools provided by Keil, or tools from other vendors, and the way the components relate.

**11.2 Software Development Tools**

Like all software based on Keil’s µVision IDE, the toolsets provide a powerful, easy to use and easy to learn environment for developing embedded applications. They include the components you need to create, debug, and assemble your C/C++ source files, and incorporate simulation for microcontrollers and related peripherals. The RTX RTOS Kernel helps you to implement complex and time-critical software.

**11.3 RTOS and Middleware Components**

These components are designed to solve communication and real-time challenges of embedded systems. While it is possible to implement embedded applications without using a real-time kernel, a proven kernel saves time and shortens the development cycle. This component also includes the source code files for the operating system. Hardware Debug Adapters The µVision Debugger fully supports several emulators provided by Keil, and other vendors. The Keil ULINK USB-JTAG family of adapters con nect the USB port of a PC to the target hardware. They enable you to download, test, and debug your embedded application on real hardware.

**11.4 Microcontroller Architectures**

The Keil µVision Integrated Development Environment (µVision IDE) supports three major microcontroller architectures and sustains the development of a wide range of applications.

* **8-bit (classic and extended 8051)** devices include an efficient interrupt system designed for real-time performance and are found in more than 65% of all 8-bit applications. Over 1000 variants are available, with peripherals that include analog I/O, timer/counters, PWM, serial interfaces like UART, I 2C, LIN, SPI, USB, CAN, and on-chip RF transmitter supporting low-power wireless applications. Some architecture extensions provide up to 16MB memory with an enriched 16/32-bit instruction set. The µVision IDE supports the latest trends, like custom chip designs based on IP cores, which integrate application-specific peripherals on a single chip
* **16-bit (Infineon C166, XE166, XC2000)** devices are tuned for optimum real-time and interrupt performance and provide a rich set of on-chip peripherals closely coupled with the microcontroller core. They include a Peripheral Event Controller (similar to memory-to-memory DMA) for high-speed data collection with little or no microcontroller overhead. These devices are the best choice for applications requiring extremely fast responses to external events.
* **32-bit (ARM7 and ARM9 based)** devices support complex applications, which require greater processing power. These cores provide high-speed 32- bit arithmetic within a 4GB address space. The RISC instruction set has been extended with a Thumb mode for high code density. ARM7 and ARM9 devices provide separate stack spaces for high-speed context switching enabling efficient multi-tasking operating systems. Bit addressing and dedicated peripheral address spaces are not supported. Only two interrupt priority levels, - Interrupt Request (IRQ) and Fast Interrupt Request (FIQ), are available.
* **32-bit (Cortex-Mx based)** devices combine the cost benefits of 8-bit and 16-bit devices with the flexibility and performance of 32-bit devices at extremely low power consumption. The architecture delivers state of the art implementations for FPGAs and SoCs. With the improved Thumb2 instruction set, Cortex-Mx1 based microcontrollers support a 4GB address space, provide bit-addressing (bit-banding), and several interrupts with at least 8 interrupt priority levels.

**8051 Architecture Advantages**

* Fast I/O operations and fast access to on-chip RAM in data space
* Efficient and flexible interrupt system
* Low-power operation

8051-based devices are typically used in small and medium sized applications that require high I/O throughput. Many devices with flexible peripherals are available, even in the smallest chip packages.

**ARM7 and ARM9 Architecture Advantages**

* Huge linear address space
* The 16-bit Thumb instruction set provides high code density
* Efficient support for all C integer data types including pointer addressing

ARM7 and ARM9-based microcontrollers are used for applications with large memory demands and for applications that use PC-based algorithms.

**11.5 Classic and Extended 8051 Devices**

8051 devices combine cost-efficient hardware with a simple but efficient programming model that uses various memory regions to maximize code efficiency and speed-up memory access.

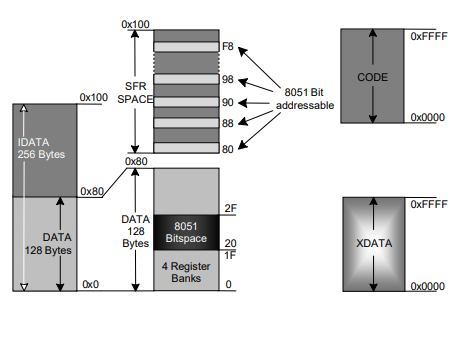


Figure 11.1 : memory layout of a classic 8051 device

The 8051 architecture provides three different physical memory regions:

* **DATA/IDATA** memory includes a 256 Bytes on-chip RAM with register banks and bit-addressable space that is used for fast variable accessing. Some devices provide an extended data (**EDATA**)space with up to 64KB.
* **CODE** memory consists of 64KB ROM space used for program code and constants. The Keil linker supports code banking that allows you to expand the physical memory space. In extended variants, up to 16MB ROM space is available.
* **XDATA** memory has a 64KB RAM space for off-chip peripheral and memory addressing. Today, most devices provide some on-chip RAM that is mapped into XDATA
* **SFR** and **IDATA** memory are located in the same address space but are accessed through different assembler instructions
* For extended devices, the memory layout provides a universal memory map that includes all 8051-memory types in a single 16MByte address region

8051 Highlights

* Fast interrupt service routines with two or four priority levels and up to 32- vectored interrupts
* Four register banks for minimum interrupt prolog/epilog
* Bit-addressable space for efficient logical operations
* 128 Bytes of Special Function Register (SFR) space for tight integration of on-chip peripherals. Some devices extend the SFR space using paging.
* Low-power, high-speed devices up to 100 MIPS are available

**11.6 8051 Development Tool Support**

The Keil C51 Compiler and the Keil Linker/Locator provide optimum 8051 architecture support with the following features and C language extensions.

* Interrupt functions with register bank support are written directly in C
* Bit and bit-addressable variables for optimal Boolean data type support
* Compile-time stack with data overlaying uses direct memory access and gives high-speed code with little overhead compared to assembly programming
* Reentrant functions for usage by multiple interrupt or task threats
* Generic and memory-specific pointers provide flexible memory access
* Linker Code Packing gives utmost code density by reusing identical program sequences
* Code and Variable Banking expand the physical memory address space
* Absolute Variable Locating enables peripheral access and memory sharing

The assembler translates assembly source files into re-locatable object modules and can optionally create listing files with symbol table and cross-reference details. Complete line number, symbol, and type information is written to the generated object files. This information enables the debugger to display the program variables exactly. Line numbers are used for source-level debugging with the µVision Debugger or other third-party debugging tools. Keil assemblers support several different types of macro processors (depending on architecture):

The Standard Macro Processor is the easier macro processor to use. It allows you to define and use macros in your assembly programs using syntax that is compatible with that used in many other assemblers.

The Macro Processing Language or MPL is a string replacement facility that is compatible with the Intel ASM-51 macro processor. MPL has several predefined macro processor functions that perform useful operations like string manipulation and number processing. Macros save development and maintenance time, since commonly used sequences need to be developed once only. Another powerful feature of the assembler’s macro processor is the conditional assembly capability. You can invoke conditional assembly through command line directives or symbols in your assembly program. Conditional assembly of code sections can help achieve the most compact code possible. It also allows you to generate different applications from a single assembly source file.

**11.7 µVision IDE**

The µVision IDE is a window-based software development platform combining a robust editor, Project Manager, and Make Utility tool. µVision supports all the Keil tools including C/C++ Compiler, Macro Assembler, Linker, Library Manager, and Object-HEX Converter. The ARM C/C++ compiler is designed to generate fast and compact code for the ARM7, ARM9 and Cortex-Mx processor architectures; while the Keil ANSI C compilers target the 8051, C166, XE166, and XC2000 architectures. They can generate object code that matches the efficiency and speed of assembly programming. Using a high-level language like C/C++ offers many advantages over assembly language programming .µVision helps expedite the development process by providing:

* **Device Database** for selecting a device and configuring the development tools for that particular microcontroller
* **Project Manager** to create and maintain projects
* **Make Utility** for assembling, compiling, and linking your embedded applications
* Full-featured source code editor
* **Template Editor** that is used to insert common text sequences or header blocks
* **Source Browser** for rapidly exploring code objects, locating and analyzing data in your application
* **Function Browser** for quickly navigating between functions in your program
* Function Outlining for controlling the visual scope within a source file
* Built-in utilities, such as **Find in Files** and functions for commenting and uncommenting source code
* µVision **Simulator** and **Target Debugger** are fully integrated
* **Configuration Wizard** providing graphical editing for microcontroller startup code and configuration files
* Interface to configure **Software Version Control Systems** and third-party utilities
* **Flash Programming Utilities**, such as the family of Keil ULINK USBJTAG Adapters
* **Dialogs** for all development tool settings
* **On-line Help** and links to microcontroller data sheets and user guides

The object-hex converter creates Intel HEX files from absolute object modules that have been created by the linker. Intel HEX files are ASCII files containing a hexadecimal representation of your application program. They are loaded easily into a device program for writing to ROM, EPROM, FLASH, or other programmable memory. Intel HEX files can be manipulated easily to include checksum or CRC data.

**11.8 µVision Debugger**

The µVision Debugger is completely integrated into the µVision IDE. It provides the following features:

* Disassembly of the code on C/C++ source- or assembly-level with program execution in various stepping modes and various view modes, like assembler, text, or mixed mode
* Multiple breakpoint options including access and complex breakpoints
* Bookmarks to quickly find and define your critical spots
* Review and modify memory, variable, and register values
* List the program call tree including stack variables
* Review the status of on-chip microcontroller peripherals
* Debugging commands or C-like scripting functions
* Execution Profiling to record and display the time consumed, as well as the cycles needed for each instruction
* Code Coverage statistics for safety-critical application testing
* Various analyzing tools to view statistics, record values of variables and peripheral I/O signals, and to display them on a time axis
* Instruction Trace capabilities to view the history of executed instructions
* Define personalized screen and window layouts Development Tools

The µVision Debugger offers two operating modes—Simulator Mode and Target Mode. **Simulator Mode** configures the µVision Debugger as a software-only product that accurately simulates target systems including instructions and most on-chip peripherals. In this mode, you can test your application code before any hardware is available. It gives you serious benefits for rapid development of reliable embedded software. The Simulator Mode offers:

* Software testing on your desktop with no hardware environment
* Early software debugging on a functional basis improves software reliability
* Breakpoints that are impossible with hardware debuggers
* Optimal input signals. Hardware debuggers add extra noise
* Single-stepping through signal processing algorithms is possible. External signals are stopped when the microcontroller halts
* Detection of failure scenarios that would destroy real hardware peripherals

**Target Mode1** connects the µVision Debugger to real hardware. Several target drivers are available that interface

* **ULINK JTAG/OCDS** Adapter that connects to on-chip debugging systems
* Monitor that may be integrated with user hardware or that is available on many evaluation boards
* **Emulator** that connects to the microcontroller pins of the target hardware
* **In-System** Debugger that is part of the user application program and provides basic test functions
* **ULINKPro** Adapter a high-speed debug and trace unit connecting to on-chip debugging systems via JTAG/SWD/SWV, and offering Cortex-M3 ETM Instruction Trace capabilities

**11.9 Simulation**

µVision simulates up to 4 GB of memory from which specific areas can be mapped for reading, writing, executing, or a combination of these. In most cases, µVision can deduce the correct memory map from the program object module. Any illegal memory access is automatically trapped and reported. A number of device-specific simulation capabilities are possible with µVision. When you select a microcontroller from the Device Database, µVision configures the Simulator accordingly and selects the appropriate instruction set, timing, and peripherals.

The µVision Simulator:

* Runs programs using the ARM7, ARM9, Thumb, Thumb2, 8051, C166/XE166/XC2000 instruction sets
* Is cycle-accurate and correctly simulates instructions and on-chip peripheral timing, where possible
* Simulates on-chip peripherals of many 8051, C166/XE166/XC2000, ARM7, ARM9, and Cortex-Mx devices
* Can provide external stimulus using the debugger C script language

**11.10 Executing Code**

µVision provides several ways to run your programs. You can

* Instruct the program to run directly to the main C function. Set this option in the Debug tab of the Options for Target dialog.
* Select debugger commands from the Debug Menu or the Debug Toolbar
* Enter debugger commands in the Command Window
* Execute debugger commands from an initialization file
* **Starting the Program** Select the Run command from the Debug Toolbar or Debug Menu or type GO in the Command Window to run the program
* **Stopping the Program** Select Stop from the Debug Toolbar or from the Debug Menu or press the Esc key while in the Command Window
* **Resetting the CPU** Select Reset from the Debug Toolbar or from the Debug – Reset CPU Menu or type RESET in the Command Window to reset the simulated CPU
* The linker/locator combines object modules into a single, executable program. It resolves external and public references and assigns absolute addresses to relocatable program segments.
* The linker includes the appropriate run-time library modules automatically and processes the object modules created by the Compiler and Assembler. You can invoke the linker from the command line or from within the µVision IDE.
* To accommodate most applications, the default linker directives have been chosen carefully and need no additional options. However, it is easy to specify additional custom settings for any application.
* The library manager creates and maintains libraries of object modules (created by the C/C++ Compiler and Assembler). Library files provide a convenient way to combine and reference a large number of modules that may be used by the linker.
* The linker includes libraries to resolve external variables and functions used in applications. Modules from libraries are extracted and added to programs only if required. Modules, containing routines that are not invoked by your program specifically, are not included in the final output.
* Object modules extracted by the linker from a library are processed exactly like other object modules. There are a number of advantages to using libraries: security, speed, and minimized disk space are only a few.
* Libraries provide a vehicle for distributing large numbers of functions and routines without distributing the original source code. For example, the ANSI C library is supplied as a set of library files.

**Chapter 12**

**Code**

#include<reg51.h> //speed control of DC motor

#include<string.h>

#define lcd\_data P2

sbit lcd\_rs = P2^0; // Here we are using LCD in four bit mode that's why LCD's Data pins and control

sbit lcd\_en = P2^1;

sbit led = P1^0; //motor PWM

sbit buzzer = P3^7;

unsigned char gchr='x',chr='x';

void delay(unsigned int value)

{

unsigned int x,y;

for(x=0;x<value;x++)

for(y=0;y<200;y++);

}

void lcdcmd(unsigned char value) // LCD COMMAND

{

// lcd\_data=value; //slcd\_end msb 4 bits

// lcd\_rs=0; //select command register

// lcd\_en=1; //lcd\_enable the lcd to execute command

// delay(10);

// lcd\_en=0;

// delay(10);

lcd\_data=value&(0xf0); //slcd\_end msb 4 bits

lcd\_rs=0; //select command register

lcd\_en=1; //lcd\_enable the lcd to execute command

delay(3);

lcd\_en=0;

lcd\_data=((value<<4)&(0xf0)); //slcd\_end lsb 4 bits

lcd\_rs=0; //select command register

lcd\_en=1; //lcd\_enable the lcd to execute command

delay(3);

lcd\_en=0;

}

void lcd\_init(void)

{

lcdcmd(0x02);

lcdcmd(0x02);

lcdcmd(0x28); //intialise the lcd in 4 bit mode\*/

lcdcmd(0x28); //intialise the lcd in 4 bit mode\*/

lcdcmd(0x0e); //culcd\_rsor blinking

lcdcmd(0x06); //move the culcd\_rsor to right side

lcdcmd(0x01); //clear the lcd

// lcdcmd(0x38);

// lcdcmd(0x0e);

// lcdcmd(0x06);

// lcdcmd(0x01);

// lcdcmd(0x80);

}

void lcddata(unsigned char value)

{

lcd\_data=value&(0xf0); //send msb 4 bits

lcd\_rs=1; //select data register

lcd\_en=1; //enable the lcd to execute data

delay(3);

lcd\_en=0;

lcd\_data=((value<<4)&(0xf0)); //send lsb 4 bits

lcd\_rs=1; //select data register

lcd\_en=1; //enable the lcd to execute data

delay(3);

lcd\_en=0;

delay(3);

}

void msgdisplay(unsigned char b[]) // send string to lcd

{

unsigned char s,count=0;

for(s=0;b[s]!='\0';s++)

{

count++;

if(s==16)

lcdcmd(0xc0);

if(s==32)

{

lcdcmd(1);

count=0;

}

lcddata(b[s]);

}

}

void serinit()

{

TMOD=0x20;

TH1=0xFD; //9600

SCON=0x50;

TR1=1;

}

void sertx(unsigned char tx)

{

SBUF=tx;

while(TI == 0);

TI=0;

}

void sertxs(unsigned char \*tx)

{

for(;\*tx != '\0';tx++)

{

SBUF=\*tx;

while(TI == 0);

TI=0;

}

}

unsigned char receive()

{

unsigned char rx;

while(RI == 0);

rx=SBUF;

RI=0;

return rx;

}

void sie()

{

//ET0 = 0;

ES = 1; /\* allow serial interrupts \*/

EA = 1;

}

void sid()

{

//ET0 = 1;

ES = 0; /\* allow serial interrupts \*/

EA = 0;

}

unsigned char gl=0,gps\_location[25];

void main()

{

int count=0;

unsigned char cnts=0;

P2=0xff;

serinit();

led=1;

buzzer=1;

lcd\_init(); lcdcmd(1);

msgdisplay("Acc Det GSM-GPS");

while(1)

{

do{

gchr = receive();

}while(gchr != '\*');

for(gl=0;gl<24;gl++)

{

gps\_location[gl] = receive();

}

buzzer=0;led=0;

lcdcmd(1);msgdisplay("Accident Occur");delay(1000);

lcdcmd(1);msgdisplay(gps\_location); delay(1000); delay(1000); delay(1000); delay(1000); delay(1000); delay(1000); delay(1000); delay(1000); delay(1000); delay(1000); delay(1000);

buzzer=1;led=1;

lcdcmd(1);

}

}

**CHAPTER 13**

**RESULTS**

**CHAPTER 14**

**ADVANTAGES AND APPLICATIONS**

**Advantages:**

* There will not be any need to construct a separate pole for this model as it can be attached to the existing poles present on the road.
* As it is attached to the existing poles, the power supply is also taken from those existing poles.
* As we use Zigbee, the indication to the Laser light can be done without any particular switch.

**Applications**

* It can be used mainly in places with bad weather conditions.
* It can be useful in accident prone areas.
* It can be used in metropolitan cities where more traffic is present

**CONCLUSION**

\*

This project presents vehicle accident detection and alert system with SMS to the user defined mobile numbers. The GPS tracking and GSM alert based algorithm is designed and implemented with LPC2148 MCU in embedded system domain. The proposed Vehicle accident detection system can track geographical information automatically and sends an alert SMS regarding accident. Experimental work has been carried out carefully. The result shows that higher sensitivity and accuracy is indeed achieved using this project. EEPROM is interfaced to store the mobile numbers permanently. This made the project more user-friendly and reliable. The proposed method is verified to be highly beneficial for the automotive industry.

**REFERENCES**

[1] GPS: Theory and Practice, B. Hofmann-Wellenhof et al., Springer Verlag, 1992, ISBN 3-211-82364-6 and 0-387-82364-6.  
  
[2] Understanding GPS: Principles and Applications (Artech House Telecommunications Library), Elliott D. Kaplan (Editor) / Hardcover / (1996), (USD 99).  
  
[3] GSM Networks: Protocols, Terminology and Implementation by Gunnar Heine.  
  
[4] GSM Switching, Services, and Protocols by Joerg Eberspaecher.  
  
[5] GSM System Engineering (Artech House Mobile Communications Series) by Asha K. Mehrotra.  
  
[6] Working with GSM Network By Cruis Leonardo.

[7] “Automatic traffic accident detection and alarm system” International Journal of Technological Exploration and Learning (IJTEL) Volume 1 Issue 1 (August 2012)

[8] “Automatic accident notification system using gsm and gps modems with 3g technology for video video monitoring” International Journal of Emerging Trends in Electrical and Electronics (IJETEE) Vol. 1, Issue. 2, March-2013.