A

PROJECT REPORT

ON

AUTOMATIC SPEED CONTROL OF VEHICLES IN RESTRICTED AREAS USING RF AND GSM

Submitted in partial fulfillment of the requirements for the award of degree

DIPLOMA IN ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted

BY

N.LAXMAN	**16219-EC-037**
A SRINIVAS	**16219-EC-001**
G.NAVEEN KUMAR	**16219-EC-019**
M AJAY KUMAR	**16219-EC-032**

Under the esteemed guidance of

Mr. A.NARESH M.Tech

LECTURER

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

RATNAPURI INSTITUTE OF TECHNOLOGY COLLEGE OF POLYTECHNIC

TURKALAKHANAPUR (V), HATHNOORA (M), SANGAREDDY (D) 502296

STATE BOARD OF TECHINCAL EDUCATION & TRAINING

HYDERABAD, TS.

(2016-2019)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

RATNAPURI INSTITUTE OF TECHNOLOGY COLLEGE OF POLYTECHNIC

CERTIFICATE

THIS IS TO CERTIFY THAT THE PROJECT REPORT TITLED **“ AUTOMATIC SPEED CONTROL OF VEHICLES IN RESTRICTED AREAS USING RF AND GSM”** *“SUBMITTED BY THE FOLLOWING STUDENTS FOR THE AWARD OF DIPLOMA IN ELECTRONICS AND COMMUNICATION ENGINEERING IS RECORD OF BONAFIDE WORK CARRIED OUT BY THEM.*

BY

N .LAXMAN	**16219-EC-037**
A SRINIVAS	**16219-EC-001**
G.NAVEEN KUMAR	**16219-EC-019**
M AJAY KUMAR	**16219-EC-032**

In partial fulfillment of the requirements for the award of Diploma in

Electronics and Communication Engineering

During year (2015-2018)

-------------------------- ----------------------

Guide Head of the department

------------------------- --------------------------

Internal examiner External examiner

ACKNOWLEDGEMENT

A leaflet acknowledgment cannot represent our true feeling for all these persons.

We are extremely grateful to our principal and director **Prof.Dr.Col.V.R.R.DATLA** *sir*

For guidance and encouragement gives as the opportunity to work on this project we thank him for helping us throughout the period of this course.

We are thankful to our project Guide **Mr. G.KRANTHI KUMAR M.Tech** lecturer *in Department of Electronics and Communication Engineering for her timely and consistent support through the extraction of this project.*

We wish to express Head of Electronics and Communication department who showed motherly Affection at each moment of the project work

And we wish to express our Regards to **Mr. G.KRANTHI KUMAR M.Tech.**

We express our Gratitude to our faculty members **Mr. K.Rambabu,** Mr. M.**Naveen, Mr. A.Naresh, Miss N.Evajeline** and **Miss. G.Priyanka** *who has_helped us for the successful completion of our project.*

We wish to express our profound thanks to **A.RAGHU** *sir, administrative officer for helping us during the project work.*

We accord our highest Respect to our Beloved **PARENTS** *for support and encouragement.*

BY

N.LAXMAN	**16219-EC-037**
A. SRINIVAS	**16219-EC-001**
G.NAVEEN KUMAR	**16219-EC-019**
M.AJAY KUAMR	**16219-EC-032**

DECLARATION

This is to certify that the work reported in this present thesis titled "**AUTOMATIC SPEED CONTROL OF VEHICLES IN RESTRICTED AREAS USING RF AND GSM**" "*Submitted by the following*" *is the record of the work done by me and my batch.*

BY

NAME	PIN
N.LAXMAN	**16219-EC-037**
A SRINIVAS	**16219-EC-001**
G.NAVEEN KUMAR	**16219-EC-019**
M.AJAY KUMAR	**16219-EC-032**

ABSTRACT

This project has an aim to control the speed of any vehicles automatically in cities and also in restricted areas such schools, parks, hospitals and in speed limited areas etc. Nowadays in a fast moving world all the peoples are not have self-control. Such peoples are driving vehicles in a high speed. So the police are not able to monitor all those things. This paper provides a way for how to control the speed without harming others. Driver does not control anything during such places; controls are taken automatically by the use of electronic system. In this project we using RF for indicating the speed limit areas it is placed front and back of the restricted zones. RF receiver is placed inside the vehicle. Speed is acquired by the help of speedometer in the vehicle. The controller compares the speed. If it exceeds the limited speed the controller alerts the driver and controls taken automatically. If they does not respond that message an information along with the vehicle number is transmitted to the nearest police station by the use of GSM and penalty amount is collected in the nearest toll gate.

LIST OF FIGURES

LIST OF TABLES

INDEX

S.NO	CONTENTS	PG.NO

CHAPTER-1

INTRODUCTION (1-5)

CHAPTER-2

EMBEDDED SYSTEMS (6-18)

CHAPTER-3

MICROCONTROLLER

CHAPTER-4

HARDWARE IMPLEMENTATION OF THE PROJECT

CHAPTER-5

FIRMWARE IMPLEMENTATION OF THE PROJECT DESIGN KEIL COMPUTER

CHAPTER-6

ADAVANTAGES AND APPLICATIONS

CHAPTER-7

CONCLUSION AND FUTURE SCOPE

CHAPTER-1

INTRODUCTION

1.1 INTRODUCTION:

This project has an aim to control the speed of any vehicles automatically in cities and also in restricted areas such schools, parks, hospitals and in speed limited areas etc. Nowadays in a fast moving world all the peoples are not have self-control. Such peoples are driving vehicles in a high speed. So the police are not able to monitor all those things. This paper provides a way for how to control the speed without harming others. Driver does not control anything during such places; controls are taken automatically by the use of electronic system. In this project we using RF for indicating the speed limit areas it is placed front and back of the restricted zones. RF receiver is placed inside the vehicle. Speed is acquired by the help of speedometer in the vehicle. The controller compares the speed. If it exceeds the limited speed the controller alerts the driver and controls taken automatically. If they does not respond that message an information along with the vehicle number is transmitted to the nearest police station by the use of GSM and penalty amount is collected in the nearest toll gate.

1.2 Following are the advantages of Embedded Systems:

1. They are designed to do a specific task and have real time performance constraints which must be met.
2. They allow the system hardware to be simplified so costs are reduced.
3. They are usually in the form of small computerized parts in larger devices which serve a general purpose.

The program instructions for embedded systems run with limited computer hardware resources, little memory and small or even non-existent keyboard or screen.

1.3 SOFTWARE AND HARDWARE TOOLS:

Software Tools:

1. Keil compiler

Hardware Tools:

1. Microcontroller AT89S52.
2. LCD Display
3. Power Supply
4. GSM
5. RF Receiver
6. Switches
7. DC Motor
8. SMTP

1.4 BLOCK DIAGRAM:

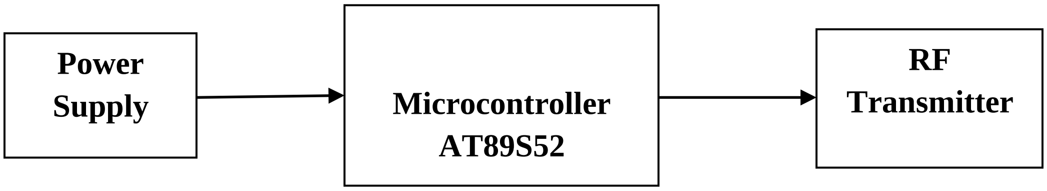


Fig: 1.1 Transmitter block diagram.

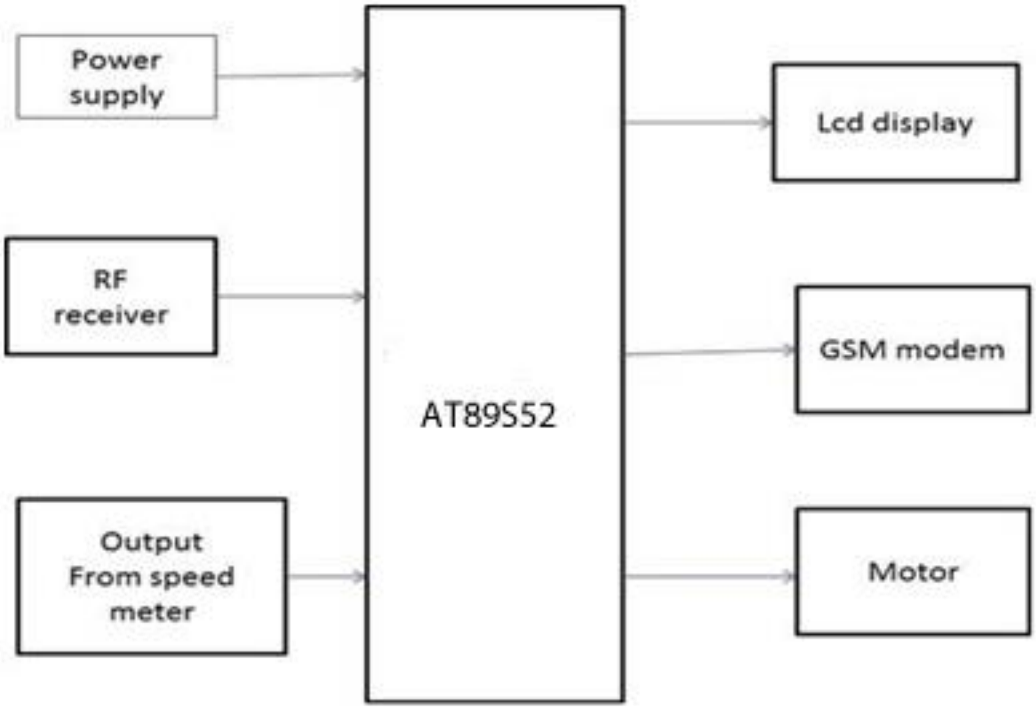
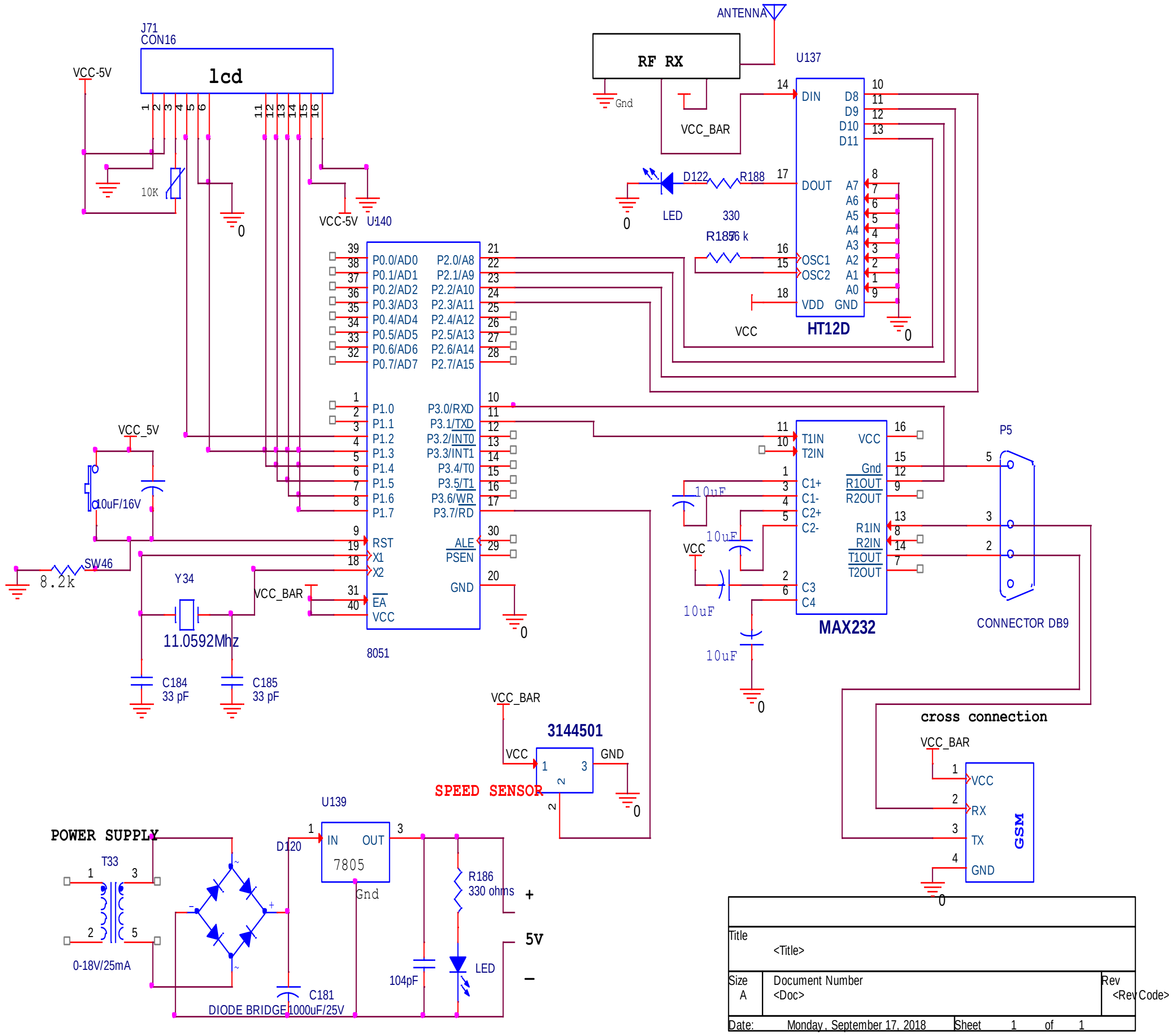
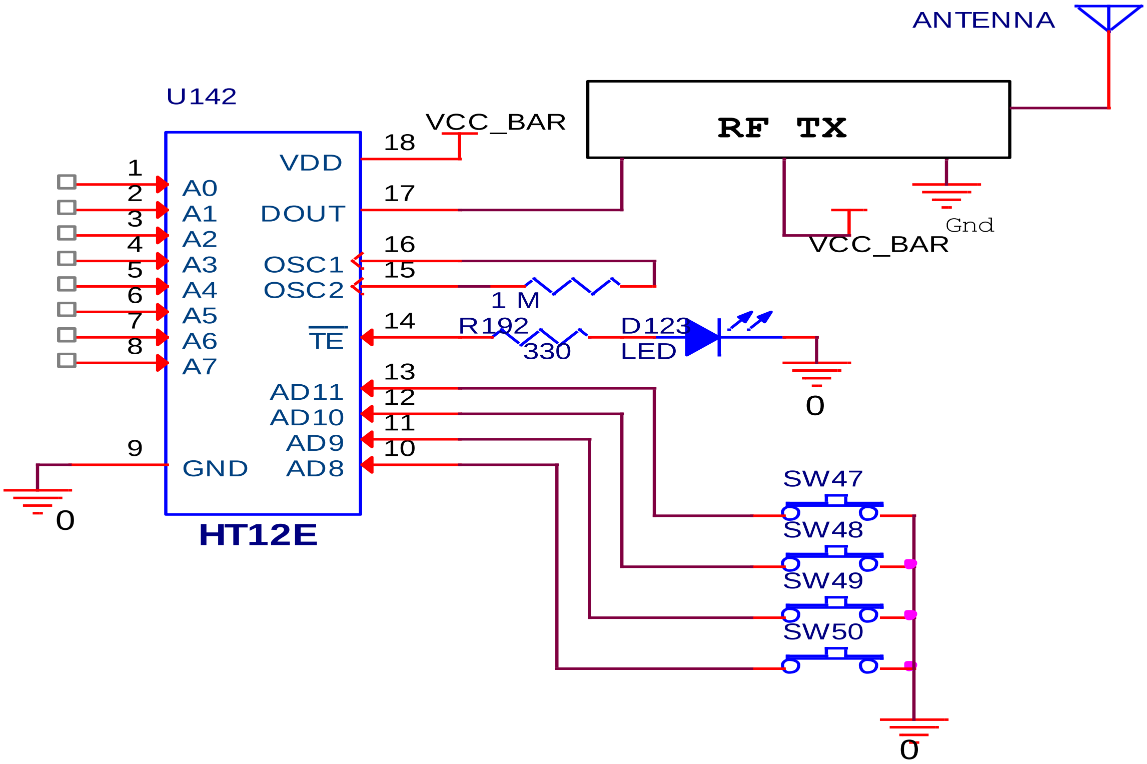


Fig 1.2 Receiver block diagram

1.5 SCHEMATIC DIAGRAM:



RF TRANSMITTER PART



CHAPTER 2

EMBEDDED SYSTEMS

2.1 Embedded Systems:

An embedded system is a computer system designed to perform one or a few dedicated functions often with real-time computing constraints. It is embedded as part of a complete device often including hardware and mechanical parts. By contrast, a general-purpose

computer, such as a personal computer (PC), is designed to be flexible and to meet a wide range of end-user needs. Embedded systems control many devices in common use today.

Embedded systems are controlled by one or more main processing cores that are typically either microcontrollers or digital signal processors (DSP). The key characteristic, however, is being dedicated to handle a particular task, which may require very powerful processors. For example, air traffic control systems may usefully be viewed as embedded, even though they involve mainframe computers and dedicated regional and national networks between airports and radar sites. (Each radar probably includes one or more embedded systems of its own.)

Since the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

Physically embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic lights, factory controllers, or the systems controlling nuclear power plants. Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large chassis or enclosure.

In general, "embedded system" is not a strictly definable term, as most systems have some element of extensibility or programmability. For example, handheld computers share some elements with embedded systems such as the operating systems and microprocessors which power them, but they allow different applications to be loaded and peripherals to be connected. Moreover, even systems which don't expose programmability as a primary feature generally need to support software updates. On a continuum from "general purpose" to "embedded", large application systems will have subcomponents at most points even if the system as a whole is "designed to perform one or a few dedicated functions", and is thus appropriate to call "embedded". A modern example of embedded system is shown in fig: 2.1.

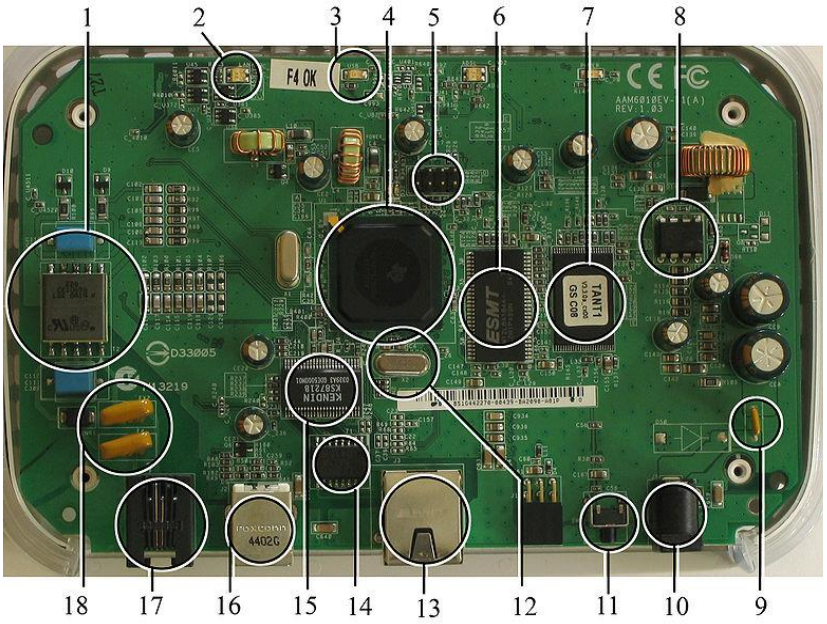


Fig 2.1: A modern example of embedded system

Labeled parts include microprocessor (4), RAM (6), flash memory (7).Embedded systems programming is not like normal PC programming. In many ways, programming for an embedded system is like programming PC 15 years ago. The hardware for the system is usually chosen to make the device as cheap as possible. Spending an extra dollar a unit in order to make things easier to program can cost millions. Hiring a programmer for an extra month is cheap in comparison. This means the programmer must make do with slow processors and low memory, while at the same time battling a need for efficiency not seen in most PC applications. Below is a list of issues specific to the embedded field.

2.1.1 History:

In the earliest years of computers in the 1930–40s, computers were sometimes dedicated to a single task, but were far too large and expensive for most kinds of tasks performed by embedded computers of today. Over time however, the concept of programmable controllers evolved from traditional electromechanical sequencers, via solid state devices, to the use of computer technology.

One of the first recognizably modern embedded systems was the Apollo Guidance Computer, developed by Charles Stark Draper at the MIT Instrumentation Laboratory. At the project's inception, the Apollo guidance computer was considered the riskiest item in the Apollo project as it employed the then newly developed monolithic integrated circuits to reduce the size and weight. An early mass-produced embedded system was the Auto notices D-17 guidance computer for the Minuteman missile, released in 1961. It was built from transistor logic and had a hard disk for main memory. When the Minuteman II went into production in 1966, the D-17 was replaced with a new computer that was the first high-volume use of integrated circuits.

2.1.2 Tools:

Embedded development makes up a small fraction of total programming. There's also a large number of embedded architectures, unlike the PC world where 1 instruction set rules, and the UNIX world where there's only 3 or 4 major ones. This means that the tools are more expensive. It also means that they're lowering featured, and less developed. On a major embedded project, at some point you will almost always find a compiler bug of some sort.

Debugging tools are another issue. Since you can't always run general programs on your embedded processor, you can't always run a debugger on it. This makes fixing your program difficult. Special hardware such as JTAG ports can overcome this issue in part. However, if you stop on a breakpoint when your system is controlling real world hardware (such as a motor), permanent equipment damage can occur. As a result, people doing embedded programming quickly become masters at using serial IO channels and error message style debugging.

2.1.3 Resources:

To save costs, embedded systems frequently have the cheapest processors that can do the job. This means your programs need to be written as efficiently as possible. When dealing with large data sets, issues like memory cache misses that never matter in PC programming can hurt you. Luckily, this won't happen too often- use reasonably efficient algorithms to start, and optimize only when necessary. Of course, normal profilers won't work well, due to the same reason debuggers don't work well.

Memory is also an issue. For the same cost savings reasons, embedded systems usually have the least memory they can get away with. That means their algorithms must be memory efficient (unlike in PC programs, you will frequently sacrifice processor time for memory, rather than the reverse). It also means you can't afford to leak memory. Embedded applications generally use deterministic memory techniques and avoid the default "new" and "malloc" functions, so that leaks can be found and eliminated more easily. Other resources programmers expect may not even exist. For example, most embedded processors do not have hardware FPUs (Floating-Point Processing Unit). These resources either need to be emulated in software, or avoided altogether.

2.1.4 Real Time Issues:

Embedded systems frequently control hardware, and must be able to respond to them in real time. Failure to do so could cause inaccuracy in measurements, or even damage hardware such as motors. This is made even more difficult by the lack of resources available. Almost all embedded systems need to be able to prioritize some tasks over others, and to be able to put off/skip low priority tasks such as UI in favor of high priority tasks like hardware control.

2.2 Need For Embedded Systems:

The uses of embedded systems are virtually limitless, because every day new products are introduced to the market that utilizes embedded computers in novel ways. In recent years, hardware such as microprocessors, microcontrollers, and FPGA chips have become much cheaper. So when implementing a new form of control, it's wiser to just buy the generic chip and write your own custom software for it. Producing a custom-made chip to handle a particular task or set of tasks costs far more time and money. Many embedded computers even come with extensive libraries, so that "writing your own software" becomes a very trivial task indeed. From an implementation viewpoint, there is a major difference between a computer and an embedded system. Embedded systems are often required to provide Real-Time response. The main elements that make embedded systems unique are its reliability and ease in debugging.

2.2.1 Debugging:

Embedded debugging may be performed at different levels, depending on the facilities available. From simplest to most sophisticated they can be roughly grouped into the following areas:

- Interactive resident debugging, using the simple shell provided by the embedded operating system (e.g. Forth and Basic)
- External debugging using logging or serial port output to trace operation using either a monitor in flash or using a debug server like the Remedy Debugger which even works for heterogeneous multi core systems.
- An in-circuit debugger (ICD), a hardware device that connects to the microprocessor via a JTAG or Nexus interface. This allows the operation of the microprocessor to be controlled externally, but is typically restricted to specific debugging capabilities in the processor.
- An in-circuit emulator replaces the microprocessor with a simulated equivalent, providing full control over all aspects of the microprocessor.
- A complete emulator provides a simulation of all aspects of the hardware, allowing all of it to be controlled and modified and allowing debugging on a normal PC.
- Unless restricted to external debugging, the programmer can typically load and run software through the tools, view the code running in the processor, and start or stop its operation. The view of the code may be as assembly code or source-code.

Because an embedded system is often composed of a wide variety of elements, the debugging strategy may vary. For instance, debugging a software (and microprocessor) centric embedded system is different from debugging an embedded system where most of the processing is performed by peripherals (DSP, FPGA, co-processor). An increasing number of embedded systems today use more than one single processor core. A common problem with multi-core development is the proper synchronization of software execution. In such a case, the embedded system design may wish to check the data traffic on the busses between the processor cores, which requires very low-level debugging, at signal/bus level, with a logic analyzer, for instance.

2.2.2 Reliability:

Embedded systems often reside in machines that are expected to run continuously for years without errors and in some cases recover by themselves if an error occurs. Therefore the software is usually developed and tested more carefully than that for personal computers, and unreliable mechanical moving parts such as disk drives, switches or buttons are avoided.

Specific reliability issues may include:

- The system cannot safely be shut down for repair, or it is too inaccessible to repair. Examples include space systems, undersea cables, navigational beacons, bore-hole systems, and automobiles.
- The system must be kept running for safety reasons. "Limp modes" are less tolerable. Often backup s is selected by an operator. Examples include aircraft navigation, reactor control systems, safety-critical chemical factory controls, train signals, engines on single-engine aircraft.
- The system will lose large amounts of money when shut down: Telephone switches, factory controls, bridge and elevator controls, funds transfer and market making, automated sales and service.

A variety of techniques are used, sometimes in combination, to recover from errors—both software bugs such as memory leaks, and also soft errors in the hardware:

- Watchdog timer that resets the computer unless the software periodically notifies the watchdog
- Subsystems with redundant spares that can be switched over to
- software "limp modes" that provide partial function

- Designing with a Trusted Computing Base (TCB) architecture[6] ensures a highly secure & reliable system environment
- An Embedded Hypervisor is able to provide secure encapsulation for any subsystem component, so that a compromised software component cannot interfere with other subsystems, or privileged-level system software. This encapsulation keeps faults from propagating from one subsystem to another, improving reliability. This may also allow a subsystem to be automatically shut down and restarted on fault detection.
- Immunity Aware Programming

2.3 Explanation of Embedded Systems:

2.3.1 Software Architecture:

There are several different types of software architecture in common use.

- Simple Control Loop:

In this design, the software simply has a loop. The loop calls subroutines, each of which manages a part of the hardware or software.

- Interrupt Controlled System:

Some embedded systems are predominantly interrupting controlled. This means that tasks performed by the system are triggered by different kinds of events. An interrupt could be generated for example by a timer in a predefined frequency, or by a serial port controller receiving a byte. These kinds of systems are used if event handlers need low latency and the event handlers are short and simple.

Usually these kinds of systems run a simple task in a main loop also, but this task is not very sensitive to unexpected delays. Sometimes the interrupt handler will add longer tasks to a queue structure. Later, after the interrupt handler has finished, these tasks are executed by the main loop. This method brings the system close to a multitasking kernel with discrete processes.

- Cooperative Multitasking:

A non-preemptive multitasking system is very similar to the simple control loop scheme, except that the loop is hidden in an API. The programmer defines a series of tasks, and each task gets its own environment to “run” in. When a task is idle, it calls an idle routine, usually called “pause”, “wait”, “yield”, “nop” (stands for no operation), etc. The advantages and disadvantages are very similar to the control loop, except that adding new software is easier, by simply writing a new task, or adding to the queue-interpreter.

- Primitive Multitasking:

In this type of system, a low-level piece of code switches between tasks or threads based on a timer (connected to an interrupt). This is the level at which the system is generally considered to have an "operating system" kernel. Depending on how much functionality is required, it introduces more or less of the complexities of managing multiple tasks running conceptually in parallel.

As any code can potentially damage the data of another task (except in larger systems using an MMU) programs must be carefully designed and tested, and access to shared data must be controlled by some synchronization strategy, such as message queues, semaphores or a non-blocking synchronization scheme.

Because of these complexities, it is common for organizations to buy a real-time operating system, allowing the application programmers to concentrate on device functionality rather than operating system services, at least for large systems; smaller systems often cannot afford the overhead associated with a generic real time system, due to limitations regarding memory size, performance, and/or battery life.

- Microkernels And Exokernels:

A microkernel is a logical step up from a real-time OS. The usual arrangement is that the operating system kernel allocates memory and switches the CPU to different threads of execution. User mode processes implement major functions such as file systems, network interfaces, etc.

In general, micro kernels succeed when the task switching and inter task communication is fast, and fail when they are slow. Exo kernels communicate efficiently by normal subroutine calls. The hardware and all the software in the system are available to, and extensible by application programmers. Based on performance, functionality, requirement the embedded systems are divided into three categories:

2.3.2 Stand Alone Embedded System:

These systems takes the input in the form of electrical signals from transducers or commands from human beings such as pressing of a button etc.., process them and produces desired output. This entire process of taking input, processing it and giving output is done in standalone mode. Such embedded systems comes under stand alone embedded systems

Eg: microwave oven, air conditioner etc.

2.3.3 Real-time embedded systems:

Embedded systems which are used to perform a specific task or operation in a specific time period those systems are called as real-time embedded systems. There are two types of real-time embedded systems.

- Hard Real-time embedded systems:

These embedded systems follow an absolute dead line time period i.e.., if the tasking is not done in a particular time period then there is a cause of damage to the entire equipment.

Eg: consider a system in which we have to open a valve within 30 milliseconds. If this valve is not opened in 30 ms this may cause damage to the entire equipment. So in such cases we use embedded systems for doing automatic operations.

- Soft Real Time embedded systems:

Eg: Consider a TV remote control system ,if the remote control takes a few milliseconds delay it will not cause damage either to the TV or to the remote control. These systems which will not cause damage when they are not operated at considerable time period those systems comes under soft real-time embedded systems.

2.3.4 Network communication embedded systems:

A wide range network interfacing communication is provided by using embedded systems.

Eg:

- Consider a web camera that is connected to the computer with internet can be used to spread communication like sending pictures, images, videos etc.., to another computer with internet connection throughout anywhere in the world.
- Consider a web camera that is connected at the door lock.

Whenever a person comes near the door, it captures the image of a person and sends to the desktop of your computer which is connected to internet. This gives an alerting message with image on to the desktop of your computer, and then you can open the door lock just by clicking the mouse. Fig: 2.2 show the network communications in embedded systems.

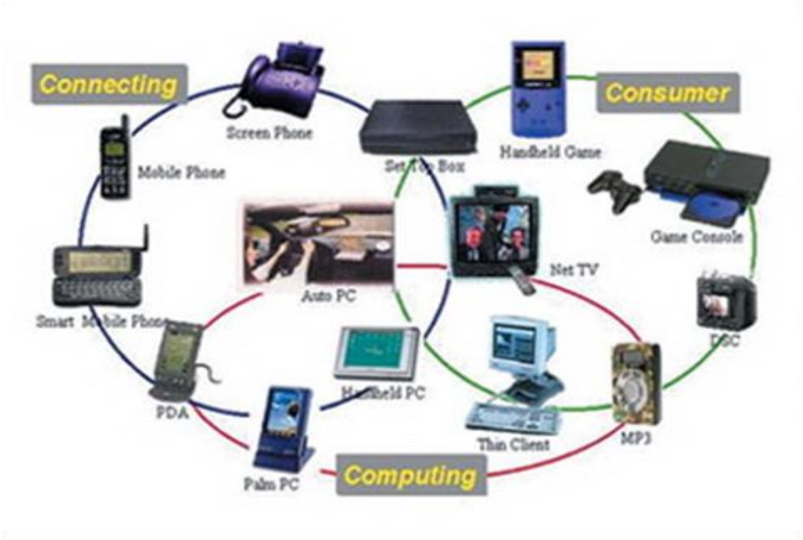


Fig 2.2: Network communication embedded systems

2.3.5 Different types of processing units:

The central processing unit (c.p.u) can be any one of the following microprocessor, microcontroller, digital signal processing.

- Among these Microcontroller is of low cost processor and one of the main advantage of microcontrollers is, the components such as memory, serial communication interfaces, analog to digital converters etc.., all these are built on a single chip. The numbers of external components that are connected to it are very less according to the application.

- Microprocessors are more powerful than microcontrollers. They are used in major applications with a number of tasking requirements. But the microprocessor requires many external components like memory, serial communication, hard disk, input output ports etc.., so the power consumption is also very high when compared to microcontrollers.

- Digital signal processing is used mainly for the applications that particularly involved with processing of signals

2.4 APPLICATIONS OF EMBEDDED SYSTEMS:

2.4.1 Consumer applications:

At home we use a number of embedded systems which include microwave oven, remote control, vcd players, dvd players, camera etc….

Fig2.3: Automatic coffee makes equipment

2.4.2 Office automation:

We use systems like fax machine, modem, printer etc…

Fig2.4: Fax machine **Fig2.5: Printing machine**

2.4.3. Industrial automation:

Today a lot of industries are using embedded systems for process control. In industries we design the embedded systems to perform a specific operation like monitoring temperature, pressure, humidity ,voltage, current etc.., and basing on these monitored levels we do control other devices, we can send information to a centralized monitoring station.

Fig 2.6: Robot

In critical industries where human presence is avoided there we can use robot**s** which are programmed to do a specific operation.

2.4.5 Computer networking:

Embedded systems are used as bridges routers etc...

Fig2.7: Computer networking

2.4.6 Tele communications:

Cell phones, web cameras etc.

Fig2.8: Cell Phone

Fig2.9: Web camera

CHAPTER-3

MICROCONTROLLER

3.1 Microcontrollers:

Microprocessors and microcontrollers are widely used in embedded systems products. Microcontroller is a programmable device. A microcontroller has a CPU in addition to a fixed amount of RAM, ROM, I/O ports and a timer embedded all on a single chip. The fixed amount of on-chip ROM, RAM and number of I/O ports in microcontrollers makes them ideal for many applications in which cost and space are critical

The Intel 8051 is Harvard architecture, single chip microcontroller (µC) which was developed by Intel in 1980 for use in embedded systems. It was popular in the 1980s and early 1990s, but today it has largely been superseded by a vast range of enhanced devices with 8051-compatible processor cores that are manufactured by more than 20 independent manufacturers including Atmel, Infineon Technologies and Maxim Integrated Products.8051 is an 8-bit processor, meaning that the CPU can work on only 8 bits of data at a time. Data larger than 8 bits has to be broken into 8-bit pieces to be processed by the CPU. 8051 is available in different memory types such as UV-EPROM, Flash and NV-RAM.

3.2 Features of AT89S52:

- 8K Bytes of Re-programmable Flash Memory.
- RAM is 256 bytes.
- 4.0V to 5.5V Operating Range.
- Fully Static Operation: 0 Hz to 33 MHz’s
- Three-level Program Memory Lock.
- 256 x 8-bit Internal RAM.
- 32 Programmable I/O Lines.
- Three 16-bit Timer/Counters.
- Eight Interrupt Sources.
- Full Duplex UART Serial Channel.
- Low-power Idle and Power-down Modes.

- Interrupt recovery from power down mode.
- Watchdog timer.
- Dual data pointer.
- Power-off flag.
- Fast programming time.
- Flexible ISP programming (byte and page mode).

3.2.1 Description:

The AT89s52 is a low-voltage, high-performance CMOS 8-bit microcomputer with 8K bytes of Flash programmable memory. The device is manufactured using Atmel’s high density nonvolatile memory technology and is compatible with the industry-standard MCS-51 instruction set. The on chip flash allows the program memory to be reprogrammed in system or by a conventional non volatile memory programmer. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89s52 is a powerful microcomputer, which provides a highly flexible and cost-effective solution to many embedded control applications.

In addition, the AT89s52 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port and interrupt system to continue functioning. The power-down mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next hardware reset.

3.3 Pin Diagram

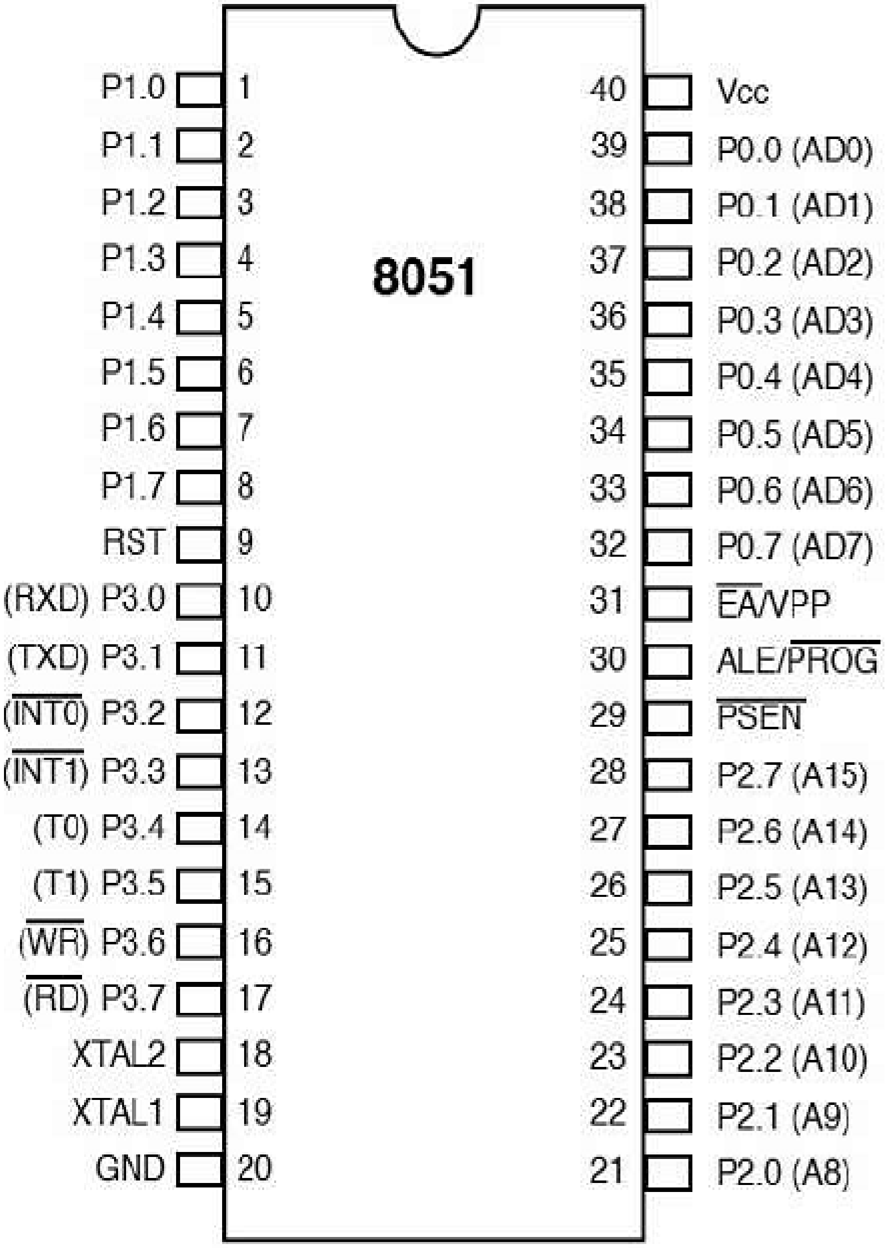


Fig: 3.1 Pin Diagram of 8051

2.4 Block Diagram

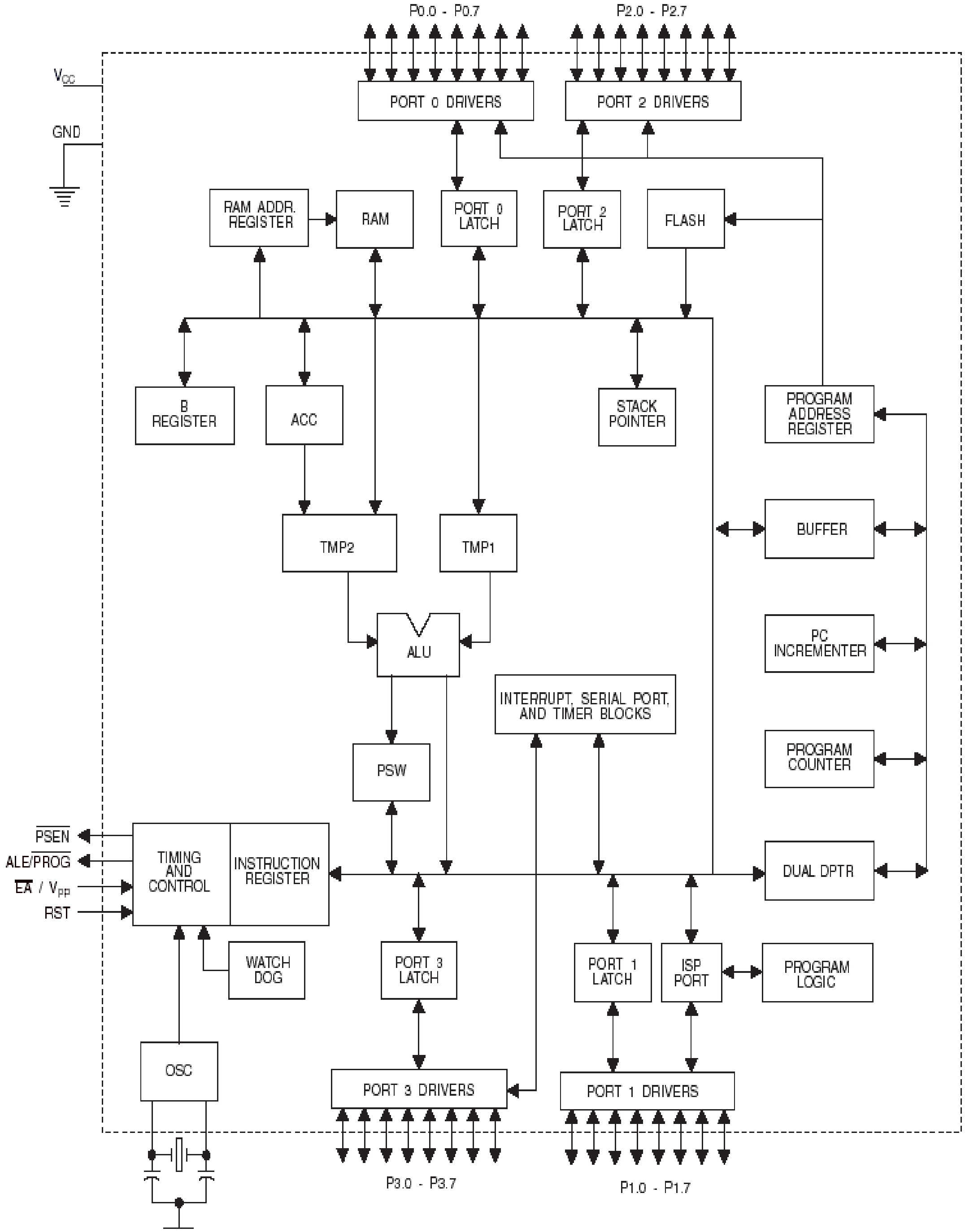


Fig: Block diagram

Fig: 3.2 Block diagram of 8051

3.5 Pin Description:

V_{cc}: Pin 40 provides supply voltage to the chip. The voltage source is +5V.

GND: Pin 20 is the ground.

Port 0

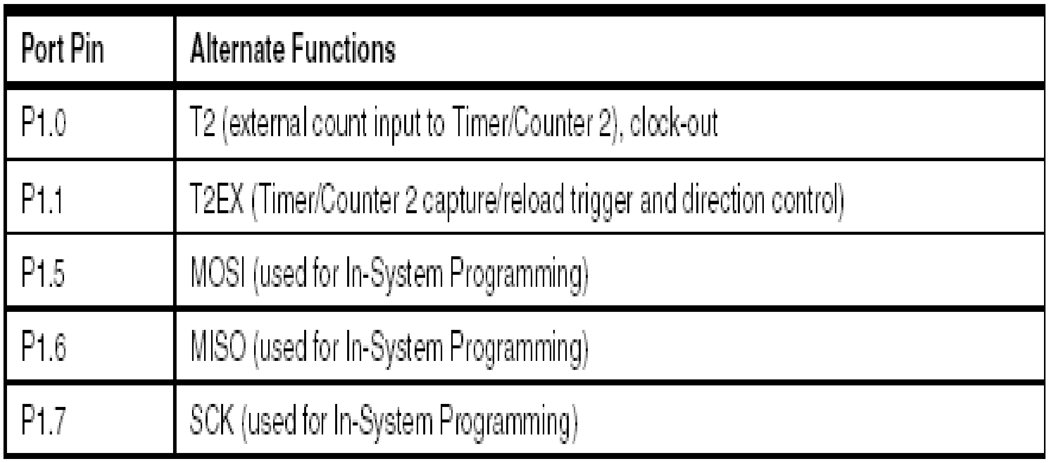
Port 0 is an 8-bit open drain bidirectional I/O port. As an output port, each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high impedance inputs. Port 0 can also be configured to be the multiplexed low-order address/data bus during accesses to external program and data memory. In this mode, P0 has internal pull-ups.

Port 0 also receives the code bytes during Flash programming and outputs the code bytes during Program verification. External pull-ups are required during program verification.

Port 1

Port 1 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 1 output buffers can sink/source four TTL inputs. When 1s are written to Port 1 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current (IIL) because of the internal pull-ups. In addition, P1.0 and P1.1 can be configured to be the timer/counter 2 external count input (P1.0/T2) and the timer/counter 2 trigger input (P1.1/T2EX), respectively, as shown in the following table.

Port 1 also receives the low-order address bytes during Flash programming and verification.



Port Pin	Alternate Functions
P1.0	T2 (external count input to Timer/Counter 2), clock-out
P1.1	T2EX (Timer/Counter 2 capture/reload trigger and direction control)
P1.5	MOSI (used for In-System Programming)
P1.6	MISO (used for In-System Programming)
P1.7	SCK (used for In-System Programming)

Table:3.1 Functions of Port1

Port 2

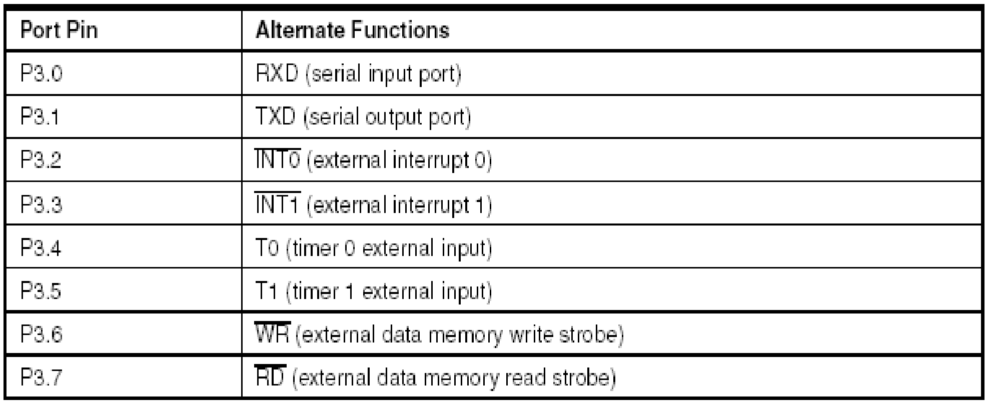
Port 2 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 2 output buffers can sink/source four TTL inputs. When 1s are written to Port 2 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current (IIL) because of the internal pull-ups.

Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that uses 16-bit addresses (MOVX @ DPTR). In this application, Port 2 uses strong internal pull-ups when emitting 1s. During accesses to external data memory that uses 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register. The port also receives the high-order address bits and some control signals during Flash programming and verification.

Port 3

Port 3 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (IIL) because of the pull-ups. Port 3 receives some control signals for Flash programming and verification.

Port 3 also serves the functions of various special features of the AT89S52, as shown in the following table.



Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{\text{INT0}}$ (external interrupt 0)
P3.3	$\overline{\text{INT1}}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{\text{WR}}$ (external data memory write strobe)
P3.7	$\overline{\text{RD}}$ (external data memory read strobe)

Table: 3.5 Functions of Port3

RST

Reset input A high on this pin for two machine cycles while the oscillator is running resets the device. This pin drives high for 98 oscillator periods after the Watchdog times out. The DISRTO bit in SFR AUXR (address 8EH) can be used to disable this feature. In the default state of bit DISRTO, the RESET HIGH out feature is enabled.

ALE/PROG

Address Latch Enable (ALE) is an output pulse for latching the low byte of the address during accesses to external memory. This pin is also the program pulse input (PROG) during Flash programming.

In normal operation, ALE is emitted at a constant rate of 1/6 the oscillator frequency and may be used for external timing or clocking purposes. Note, however, that one ALE pulse is skipped during each access to external data memory.

If desired, ALE operation can be disabled by setting bit 0 of SFR location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction. Otherwise, the pin is weakly pulled high. Setting the ALE-disable bit has no effect if the microcontroller is in external execution mode.

PSEN

Program Store Enable (PSEN) is the read strobe to external program memory. When the AT89S52 is executing code from external program memory, PSEN is activated twice each machine cycle, except that two PSEN activations are skipped during each access to external data memory.

EA/VPP

External Access Enable EA must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH. Note, however, that if lock bit 1 is programmed, EA will be internally latched on reset.

EA should be strapped to VCC for internal program executions. This pin also receives the 12-volt programming enable voltage (VPP) during Flash programming.

XTAL1

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting oscillator amplifier.

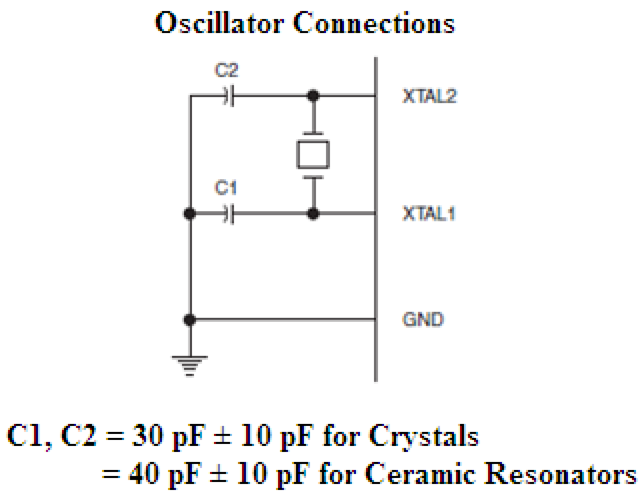

Oscillator Connections
C2
XTAL2
C1
XTAL1
GND
C1, C2 = 30 pF ± 10 pF for Crystals
= 40 pF ± 10 pF for Ceramic Resonators

Fig: 3.3 Oscillator Circuit

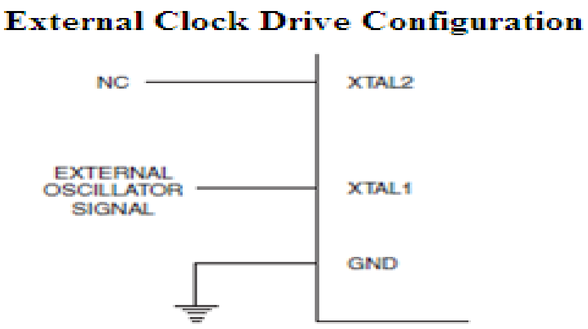

External Clock Drive Configuration
NC
XTAL2
EXTERNAL OSCILLATOR SIGNAL
XTAL1
GND

Fig: 3.4 External Clock Drive configuration

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Special Function Registers

A map of the on-chip memory area called the Special Function Register (SFR) space is shown in the following table. It should be noted that not all of the addresses are occupied and unoccupied addresses may not be implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have an indeterminate effect.

User software should not write 1s to these unlisted locations, since they may be used in future products to invoke new features. In that case, the reset or inactive values of the new bits will always be 0.

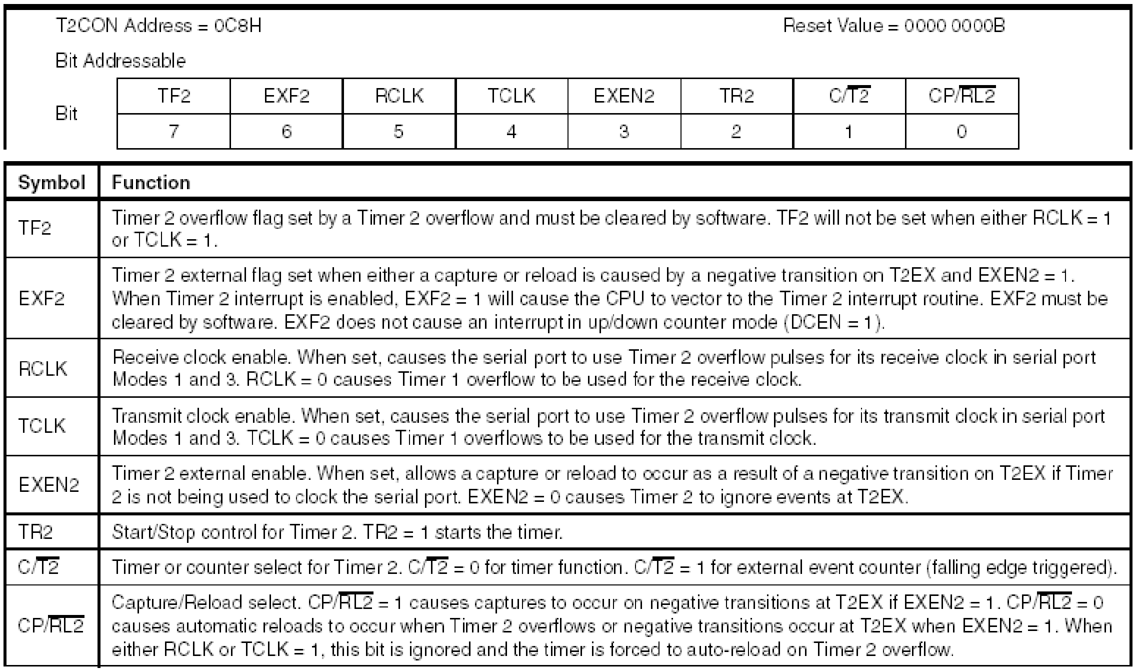
Timer 2 Registers:

Control and status bits are contained in registers T2CON and T2MOD for Timer 2. The register pair (RCAP2H, RCAP2L) is the Capture/Reload register for Timer 2 in 16-bit capture mode or 16-bit auto-reload mode.

Interrupt Registers:

The individual interrupt enable bits are in the IE register. Two priorities can be set for each of the six interrupt sources in the IP register.

T2CON – Timer/Counter 2 Control Register



T2CON Address = 0C8H Reset Value = 0000 0000B

Bit Addressable

Bit	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	$C/\overline{T2}$	$CP/\overline{RL2}$
	7	6	5	4	3	2	1	0

Symbol	Function
TF2	Timer 2 overflow flag set by a Timer 2 overflow and must be cleared by software. TF2 will not be set when either RCLK = 1 or TCLK = 1.
EXF2	Timer 2 external flag set when either a capture or reload is caused by a negative transition on T2EX and EXEN2 = 1. When Timer 2 interrupt is enabled, EXF2 = 1 will cause the CPU to vector to the Timer 2 interrupt routine. EXF2 must be cleared by software. EXF2 does not cause an interrupt in up/down counter mode (DCEN = 1).
RCLK	Receive clock enable. When set, causes the serial port to use Timer 2 overflow pulses for its receive clock in serial port Modes 1 and 3. RCLK = 0 causes Timer 1 overflow to be used for the receive clock.
TCLK	Transmit clock enable. When set, causes the serial port to use Timer 2 overflow pulses for its transmit clock in serial port Modes 1 and 3. TCLK = 0 causes Timer 1 overflows to be used for the transmit clock.
EXEN2	Timer 2 external enable. When set, allows a capture or reload to occur as a result of a negative transition on T2EX if Timer 2 is not being used to clock the serial port. EXEN2 = 0 causes Timer 2 to ignore events at T2EX.
TR2	Start/Stop control for Timer 2. TR2 = 1 starts the timer.
$C/\overline{T2}$	Timer or counter select for Timer 2. $C/\overline{T2}$ = 0 for timer function. $C/\overline{T2}$ = 1 for external event counter (falling edge triggered).
$CP/\overline{RL2}$	Capture/Reload select. $CP/\overline{RL2}$ = 1 causes captures to occur on negative transitions at T2EX if EXEN2 = 1. $CP/\overline{RL2}$ = 0 causes automatic reloads to occur when Timer 2 overflows or negative transitions occur at T2EX when EXEN2 = 1. When either RCLK or TCLK = 1, this bit is ignored and the timer is forced to auto-reload on Timer 2 overflow.

Table 3.3 TCON Register Description

Dual Data Pointer Registers:

To facilitate accessing both internal and external data memory, two banks of 16-bit Data Pointer Registers are provided: DP0 at SFR address locations 82H-83H and DP1 at 84H and 85H. Bit DPS = 0 in SFR AUXR1 selects DP0 and DPS = 1 selects DP1. The user should ALWAYS initialize the DPS bit to the appropriate value before accessing the respective Data Pointer Register.

Power off Flag:

The Power off Flag (POF) is located at bit 4 (PCON.4) in the PCON SFR. POF is set to "1" during power up. It can be set and rest under software control and is not affected by reset.

Memory Organization

MCS-51 devices have a separate address space for Program and Data Memory. Up to 64K bytes each of external Program and Data Memory can be addressed.

Program Memory

If the EA pin is connected to GND, all program fetches are directed to external memory. On the AT89S52, if EA is connected to VCC, program fetches to addresses 0000H through 1FFFH are directed to internal memory and fetches to addresses 2000H through FFFFH are to external memory.

Data Memory

The AT89S52 implements 256 bytes of on-chip RAM. The upper 128 bytes occupy a parallel address space to the Special Function Registers. This means that the upper 128 bytes have the same addresses as the SFR space but are physically separate from SFR space.

When an instruction accesses an internal location above address 7FH, the address mode used in the instruction specifies whether the CPU accesses the upper 128 bytes of RAM or the SFR space. Instructions which use direct addressing access the SFR space.

For example, the following direct addressing instruction accesses the SFR at location 0A0H (which is P2).

```
MOV 0A0H, #data
```

The instructions that use indirect addressing access the upper 128 bytes of RAM. For example, the following indirect addressing instruction, where R0 contains 0A0H, accesses the data byte at address 0A0H, rather than P2 (whose address is 0A0H).

```
MOV @R0, #data
```

It should be noted that stack operations are examples of indirect addressing, so the upper 128 bytes of data RAM are available as stack space.

Watchdog Timer (One-time Enabled with Reset-out)

The WDT is intended as a recovery method in situations where the CPU may be subjected to software upsets. The WDT consists of a 14-bit counter and the Watchdog Timer Reset (WDTRST) SFR. The WDT is defaulted to disable from exiting reset. To enable the WDT, a user must write 01EH and 0E1H in sequence to the WDTRST register (SFR location 0A6H). When the WDT is enabled, it will increment every machine cycle while the oscillator is running. The WDT timeout period is dependent on the external clock frequency. There is no way to disable the WDT except through reset (either hardware reset or WDT overflow reset). When WDT overflows, it will drive an output RESET HIGH pulse at the RST pin.

Using the WDT

To enable the WDT, a user must write 01EH and 0E1H in sequence to the WDTRST register (SFR location 0A6H). When the WDT is enabled, the user needs to service it regularly by writing 01EH and 0E1H to WDTRST to avoid a WDT overflow. The 14-bit counter overflows when it reaches 16383 (3FFFH) and this will reset the device. When the WDT is enabled, it will increment every machine cycle while the oscillator is running. This means the user must reset the WDT at least for every 16383 machine cycles.

To reset the WDT, the user must write 01EH and 0E1H to WDTRST. WDTRST is a write-only register. The WDT counter cannot be read or written. When WDT overflows, it will generate an output RESET pulse at the RST pin. The RESET pulse duration is 98xTOSC, where TOSC = 1/FOSC. To make the best use of the WDT, it should be serviced in those sections of code that will periodically be executed within the time required to prevent a WDT reset.

WDT during Power-down and Idle:

In Power down mode the oscillator stops, which means the WDT also stops. Thus the user does not need to service the WDT in Power down mode.

There are two methods of exiting Power down mode:

1. By a hardware reset or
2. By a level-activated external interrupt which is enabled prior to entering Power down mode.

When Power-down is exited with hardware reset, servicing the WDT should occur as it normally does whenever the AT89S52 is reset. Exiting Power down with an interrupt is significantly different.

The interrupt is held low long enough for the oscillator to stabilize. When the interrupt is brought high, the interrupt is serviced. To prevent the WDT from resetting the device while the interrupt pin is held low, the WDT is not started until the interrupt is pulled high. It is suggested that the WDT be reset during the interrupt service for the interrupt used to exit Power down mode.

To ensure that the WDT does not overflow within a few states of exiting Power down, it is best to reset the WDT just before entering Power down mode.

Before going into the IDLE mode, the WDIDLE bit in SFR AUXR is used to determine whether the WDT continues to count if enabled. The WDT keeps counting during IDLE (WDIDLE bit = 0) as the default state. To prevent the WDT from resetting the AT89S52 while in IDLE mode, the user should always set up a timer that will periodically exit IDLE, service the WDT and reenter IDLE mode. With WDIDLE bit enabled, the WDT will stop to count in IDLE mode and resumes the count upon exit from IDLE.

3.6 INTERRUPTS

An interrupt is some event which interrupts normal program execution. Program flow is always sequential, being altered only by those instructions, which expressly cause program flow to deviate in some way. However, interrupt give us a mechanism to “put on hold” the normal program flow, execute a subroutine, and then resume normal program flow as if we had never left it. This subroutine, called an interrupt handler, is only executed when a certain event (interrupt) occurs. Interrupts may be generated by internal chip operations or provided by external sources. Any interrupt can cause the 8051 to perform a hardware call to an interrupt-handling subroutine that is located at a predetermined (by the 8051 designers) absolute address in program memory. 8051 has 5 interrupt sources: 2 external interrupts, 2 timer interrupts and one serial port interrupt. The External interrupts INT0 and INT1 can each be either level activated or transition activated, depending on bits IT0 and IT1 in register TCON. The flags that actually generate these interrupts are bits IE0 and IE1 in TCON. The Timer 0 and Timer 1 Interrupts are generated by TF0 and TF1, which are set by a rollover in their respective timer/counter registers. When a timer interrupt is generated, the flag that generated it is cleared by the on-chip hardware when the service routine is vectored to. The Serial Port Interrupt is generated by the logical OR of RI and TI. The service routine will have to determine whether it was RI or TI that generated the interrupt and the bit will have to be cleared in software. All the bits that generate interrupts can be set or cleared by software, with the same result as though it had been set or cleared by hardware. That is, interrupts can be generated or pending interrupts can be cancelled by the software.

Each of those interrupt sources can be individually enabled or disabled by setting or clearing a bit in the Special Function Register IE. Note that IE also contains a global disable bit, in which disables all interrupts at once.

3.6.1 The Interrupt Enable (IE) SFR:

Bit	Symbol	Function
7	EA	Enable interrupt bit. Cleared to 0 by program to disable all interrupts; set to 1 to permit individual interrupts to be enabled by their enable bits.
6	---	Not implemented
5	ET2	Reserved for future use.

4	ES	Enable serial port interrupt. Set to 1 by program to enable Serial port interrupts; cleared to 0 to disable serial port interrupt
3	ET1	Enable timer 1 overflow interrupt. Set to 1 by program to enable timer 1 overflow interrupt; cleared to 0 to disable timer 1 overflow interrupt.
2	EX1	Enable external interrupt 1. Set to 1 by program to enable INTI interrupt; cleared to 0 to disable INTI interrupt
1	ETO	Enable timer 0 overflow interrupt. Set to 1 by program to enable timer 0 overflow interrupt; cleared to 0 to disable timer 0 overflow interrupt.
0	EX0	Enable external interrupt 0. Set to 1 by program to enable INT0 interrupt; cleared to 0 to disable INT0 interrupt.

3.6.2 The Interrupt Priority (IP) SFR:

Bit	**Symbol**	**Function**
7	-	Not implemented.
6	-	Not implemented.
5	PT2	Reserved for future use
4	PS	Priority of serial port interrupts. Set/cleared by program.
3	PT1	Priority of timer 1 overflow interrupt. Set/cleared by program.
2	PX1	Priority of external interrupts 1. Set/cleared by program
1	PT0	Priority of timer 0 overflow interrupts. Set/cleared by program.
0	PX0	Priority of external interrupt 0. Set/cleared by program.

3.6.3 TIMER FLAG INTERRUPT:

When a timer/counter overflows, the corresponding Timer flag, TF0 or TF1, is set to 1. The flag is cleared to 0 when the resulting interrupt generates a program call to the appropriate timer subroutine in memory.

3.6.4 SERIAL PORT INTERRUPT

When a data byte is received, an interrupt bit RI is set in the SCON register, whereas when a data byte is transmitted, the interrupt bit TI is set. These are ORed together to provide a single interrupt to the processor. These bits are not cleared when the processor makes the interrupt-generated program call. The program that handles serial data communication must reset R1 or TI to 0 to enable the next data communication operation.

3.6.5 EXTERNAL INTERRUPTS

Pins INT0 and INT1 are used by external circuitry. Inputs on these pins can set the Interrupt flags IE0 and IE1 in the TCON register to 1 by two different methods. The IEX flags may be set when the INTX pin signal reaches a low level, or the flags may be set when a high-to-low transition takes place on the INTX pin. Bits IT0 and IT1 in TCON program the INTX pins for low-level interrupt when set to 0 and program the INTX pins for transition interrupt when set to 1.

Flags IEX will be reset when a transition-generated interrupt is accepted by the processor and the interrupt subroutine is accessed. It is the responsibility of the system designer and programmer to reset any level-generated external interrupts when they are serviced by the program.

3.6.6 RESET

A reset can be considered to be the ultimate interrupt because the program may not block the action of the voltage on the RST pin. This interrupt is a non-maskable one, because no combination of bits in any register can stop, or mask, the reset action. Unlike other interrupts, the

PC is not stored for later program resumption; a reset is an absolute command to jump to program address 0000h and commence execution from the start. Internal RAM contents may change during reset; also, the states of the internal RAM bytes when power is first applied to the 8051 are random. Register bank 0 is selected on reset as all bits in PSW are 0.

The program must be able, at critical times, to inhibit the action of some or all of the interrupts so that

3.6.7 INTERRUPT CONTROL

crucial operations can be finished. The IE register holds the programmable bits that can enable or disable all the interrupts as a group, or if the group is enabled, each individual interrupt source can be enabled or disabled.

Often, it is desirable to be able to set priorities among competing interrupts that may conceivably occur simultaneously. The IP register bits may be set by the program to assign priorities among the various interrupt sources so that more important interrupts can be serviced first when two or more interrupts occur at the same time.

3.6.8 INTERRUPT ENABLE / DISABLE

Bits in the IE register are set to 1 if the corresponding interrupt source is to be enabled and set to 0 to disable the interrupt source. Bit EA is a master bit that enables or disables all of the interrupts.

3.6.9 INTERRUPT PRIORITY

If two requests of different priority levels are received simultaneously, the request of higher priority level will be serviced first. If requests of the same priority level are received simultaneously, an internal polling sequence determines which request has to be serviced. Thus, within each priority level, there is a second priority structure determined by the polling sequence, as follows:

- External 0 Interrupt
- Timer 0 Interrupt
- External 1 Interrupt
- Timer 1 Interrupt
- Serial Interrupt

3.10 COUNTERS AND TIMERS

Many micro controller applications require the counting of external events, such as the frequency of a pulse train, or the generation of precise internal time delays between computer actions. Both of these tasks can be accomplished using software techniques, but software loops for counting or timing keep the processor occupied so that other, perhaps more important, functions are not done. To relieve the processor of this burden, two 16-bit up counters, named T0 and T1, are provided for the general use of the programmer. Each counter may be programmed to count internal clock pulses, acting as a timer, or programmed to count external pulses as a counter. The counters are divided into two 8-bit registers called the timer low (TL0, TL1) and high (TH0, TH1). All counter action is controlled by bit states in the timer mode control register (TMOD), the timer/counter control register (TCON), and certain program instructions.

TCON register is used to control the entire operation of the timer/counters of 8051. This register can also be bit addressed.

TF1	TCON7	Timer 1 over flow flag. Set by hardware when the Timer/ Counter Overflows. Cleared by hardware as the processor vectors to the Interrupt service routine.
TR1	TCON6	Timer 1 run control bit. Set/Cleared by software to turn Timer/Counter 1 ON/OFF.
TF0	TCON5	Timer 0 overflows flag. Set by hardware when the Timer/Counter 0 overflows. Cleared by hardware as the

		Processor vectors to the service routine
TR0	TCON4	Timer 0 run control bit. Set/cleared by software to turn Timer/Counter 0 ON/OFF
IE1	TCON3	External Interrupt 1 edge flag. Set by hardware when External Interrupt edge is detected. Cleared by hardware when interrupt is processed
IT1	TCON2	Interrupt 1 type control bit. Set/Cleared by software to specify falling edge/low level triggered External Interrupt.
IE0	TCON1	External Interrupt 0 edge flag. Set by hardware when External Interrupt edge detected. Cleared by hardware when interrupt is processed.
IT0	TCON0	Interrupt 0 type control bit. Set/cleared by software to specify falling edge/low level triggered External Interrupt.

3.10.1 TIMER COUNTER INTERRUPTS

The counters have been included on the chip to relieve the processor of timing and counting processes. When the program is to count a certain number of internal pulses or external events, a number is placed in one of the counters. The number represents the maximum count lessthe desired count, plus 1. The counter increment from the initial number to the maximum and then rolls over to 0 on the final pulse and also sets a timer flag. The program may check the flag status.

If a counter is programmed to be a timer, it will count the internal clock frequency of the 8051 oscillator divided by 12d. As an example, if the crystal frequency is 6.0 megahertz, then the timer clock will have a frequency of 500 kilohertz.

3.10.2 TIMER MODES OF OPERATION:

TIMER MODE O

Setting timer X mode bits to 00b in the TMOD register results in using the THX register as an 8-bit counter and TLX as a 5-bit counter; the pulse input is divided by 32d in TL so that TH counts the original oscillator frequency reduced by a total 384d.

TIMER MODE 1

Mode 1 is similar to mode 0 except TLX is configured as a full 8-bit counter when the mode bits are set to 01b in TMOD. The pulse input is divided by 256d at TLX and further divided by 256d at THX. It is a complete 16 bit timer.

TIMER MODE 2

Setting the mode bits to 10b in TMOD configures the timer to use only the TLX counter as an 8-bit counter. THX is used to hold a value that is loaded into TLX every time TLX overflows from FFh to 00h. The Timer flag is also set when TLX overflows. This is the auto reload mode.

TIMER MODE 3

Timer 0 in mode 3 becomes two completely separate 8-bit counters. TL0 is controlled by the gate arrangement and sets while timer flag TF0 whenever it overflows from FFh to 00h. TH0 receives the timer clock (the oscillator divided by 12) under the control of TR1 only and sets the TF1 flag when it overflows. Timer 1 cannot be used in mode 3.

Timer 1 may still be used in modes 0, 1, and 2, timer 0 is in mode 3 with one exception: No interrupts will be generated by timer 1 while timer 0 is using the TF1 overflow flag.

3.10.3 COUNTING

The only difference between counting and timing is the source of the clock pulses to the counters. When used as a timer, the clock pulses are sourced from the oscillator through the divide-by-12d circuit. When used as a counter, pin T0 (P3.4) supplies pulses to counter 0, and pin T1 (P3.5) to counter 1. The C/□T bit in TMOD must be set to 1 to enable pulses from the TX pin to reach the control circuit.

3.10.4 SERIAL DATA INPUT/OUTPUT

Computers must be able to communicate with each other in modern multiprocessor distributed systems. One cost-effective way to communicate is to send and receive data bits serially. The 8051 has a serial data communication circuit that uses register SBUF to hold data. Register SCON controls data communication, register PCON controls data rates, and pins RXD (P3.0) and TXD (P3.1) connect to the serial data network.

SBUF is physically two registers. One is ‘write’ only and is used to hold data to be transmitted out of the 8051 via TXD. The other is ‘read’ only and holds received data from external sources via RXD. Both mutually exclusive registers use address 99h. There are four programmable modes for serial data communication that are chosen by setting the SMX bits in SCON. Baud rates are determined by the mode chosen.

3.10.5 SERIAL PORT CONTROL (SCON) Special Function Register:

Bit	Symbol	Function
7	SM0	Serial port mode bit 0. Set/cleared by program to select mode.
6	SM1	Serial port mode bit 1. Set/cleared by program to select mode.

SM0	SM1	Mode	Description
0	0	0	Shift register; baud = f/12
0	1	1	8-bit UART; baud = variable

1	0	2	9-bit UART; baud= f/32 or f/64
1	1	3	9-bit UART; baud = variable

5	SM2	Multiprocessor communications bit. Set/cleared by program to enable multiprocessor communication in modes 2 and 3. When set to 1 an interrupt is generated if bit 9 of the received data is a 1; no interrupt is generated if bit 9 is a 0. If set to 1 for mode 1, no interrupt will be generated unless a valid stop bit is received. Clear to 0 if mode 0 is in use.
	REN	Receive enable bit. Set to 1 to enable reception; cleared to 0 to disable reception.
3	TB8	Transmitted bit 8. Set/cleared by program in modes 2 and 3.
2	RB8	Received bit 8. Bit 8 of received data in modes 2 and 3; stop bit in mode 1. Not used in mode 0.
1	T1	Transmit Interrupt flag. Set to one at the end of bit 7 times in mode and the beginning of the stop bit for other modes. Must be cleared by the program.
0	R1	Receive Interrupt flag. Set to one at the end of bit 7 time in mode, and halfway through the stop bit for other modes. Must be cleared by the program.

3.10.6 PCON Special Function Registers

Bit	**Symbol**	**Function**
7	SMOD	Serial baud rate modify bit. Set to 1 by program to double baud rate using timer 1 for modes 1, 2, and 3. Cleared to 0 by program to use timer 1

		baud rate.
6-4	---	Not implemented.
3	GF1	General purpose user flag bit 1. Set/cleared by program.
2	GFO	General purpose user flag bit 0. Set/cleared by program
1	PD	Power down bit. Set to 1 by program to enter power down configuration for CHMOS processors
0	IDL	Idle mode bit. Set to 1 by program to enter idle mode configuration for CHMOS processors. PCON is not bit addressable.

3.10.7 SERIAL DATA INTERRUPTS

Serial data communication is relatively a slow process. Serial Data flags are included in SCON to aid in efficient data transmission and reception. The serial data flags in SCON, T1 and R1, are set whenever a data byte is transmitted (TI) or received (RI). These flags are ORed together to produce an interrupt to the program. These flags need to be checked through the program.

3.10.8 DATA TRANSMISSION AND RECEPTION

Transmission of serial data bits begins whenever data is written to SBUF. TI is set to ‘1’ when the data has been transmitted and signifies that SBUF is empty (for transmission purposes) and that the next data byte can be sent.

Reception of serial data will begin if they receive enable bit (REN) in SCON is set to ‘1’ for all modes. In addition, for mode 0, R1 must be cleared to 0. Receiver Interrupt flag R1 is set after data has been received in all modes. Setting REN is the only direct program control that limits the reception of unexpected data; the requirement that R1 also be 0 for mode 0 prevents the reception of new data until the program has dealt with the old data and reset R1.

Reception can begin in modes 1, 2, and 3 if RI is set when the serial stream of bits begins. The program must have reset RI before the last bit is received or the incoming data will be lost. Incoming data is not transferred to

SBUF until the last data bit has been received so that the previous transmission can be read from SBUF while new data is being received.

3.11 SERIAL DATA TRANSMISSION MODES

The 8051 designers have included four modes of serial data transmission that enable data communication to be done in a variety of ways and multiple baud rates. The programmer selects modes, by setting the mode bits SM0 and SM1 in SCON. Baud rate is fixed for mode 0 and for modes 1, 2, 3 baud rate can be modified by altering the bit (SMOD) in PCON register.

SERIAL DATA MODE 0 – SHIFT REGISTER MODE

Setting bits SM0 and SM1 in SCON to 00b configure SBUF to receive or transmit eight data bits using pin RXD for both functions. Pin TXD is connected to the internal shift frequency pulse source to supply shift pulses to external circuits. The shift frequency, or baud rate, is fixed at 1/12 of the oscillator frequency.

When transmitting, data is shifted out of RXD, the data changes on the falling edge of S6P2 (machine cycle), or one clock pulse after the rising edge of the output TXD shift clock.

Received data comes in on pin RXD and should be synchronized with the shift clock produced at TXD. Data is sampled on the falling edge of S5P2 and shifted in to SBUF on the rising edge of the shift clock.

3.11.1 SERIAL DATA MODE 1 – STANDARD UART

When SM0 and SM1 are set to 01b, SBUF becomes a 10-bit full-duplex receiver/transmitter that may receive and transmit data at the same time. Pin RXD receives data; and pin TXD transmits data.

MODE 1-BAUD RATES

Timer 1 is used to generate the baud rate for mode 1 by using the overflow flag of the timer to determine the baud frequency. Typically, timer 1 is used in timer mode 2 as an auto load 8-bit timer that generates the baud frequency:

2SMOD x Oscillator frequency
32d 12d x [256d – (THI)]

If timer 1 is not run in timer mode 2, then the baud rate is

$$\frac{\text{2SMOD x}}{\text{32d}}$$

And timer 1 can be run using the internal clock or as a counter that receives clock pulses from any external source via pin TI.

3.11.2 SERIAL DATA MODE 2 – MULTIPROCESSOR MODE

Mode 2 is similar to mode 1 except 11 bits are transmitted: a start bit, nine data bits, and a stop bit. The ninth data bit is copied from bit TB8 in SCON during transmit and stored in bit RB8 of SCON when data is received. Both the start and stop bits are discarded. The baud rate is programmed as follows:

$$\frac{\text{2SMOD}}{\text{64d}}$$

Here, as in the case for mode 0, the baud rate is much higher than standard communication rates. This high data rate is needed in many multiprocessor applications. Data can be collected quickly from an extensive network of communicating micro controllers if high baud rates are employed.

The conditions for setting RI for mode 2 are similar to mode 1. Setting RI based on the state of SM2 in the receiving 8051 and the state of bit 9 in the transmitted message makes multiprocessing possible by enabling some receivers to be interrupted by certain messages, while other receivers ignore those messages. Only those 8051s that have SM2 set to 0 will be interrupted by received data that has the ninth data bit set to 0; those with SM2 set to 1 will not be interrupted by messages with data bit 9 at 0. All receivers will be interrupted by data words that have the ninth data bit set to 1. The state of SM2 will not block reception of such messages.

Receiver Samples Data in Center of Bit Time

Bit Time = t

3.11.3 SERIAL DATA MODE 3

Mode 3 is identical to mode 2 except that the baud rate is determined exactly as in mode 1, using timer 1 to generate communication frequencies.

CHAPTER 4

HARDWARE IMPLEMENTATION OF THE PROJECT

4.1 Introduction

This chapter briefly explains about the Hardware Implementation of the project. It discusses the design and working of the design with the help of block diagram and circuit diagram and explanation of circuit diagram in detail. It explains the features, timer programming, serial communication, interrupts of AT89S52 microcontroller. It also explains the various modules used in this project.

4.2 Project Design

The implementation of the project design can be divided in two parts.

- Hardware implementation
- Firmware implementation

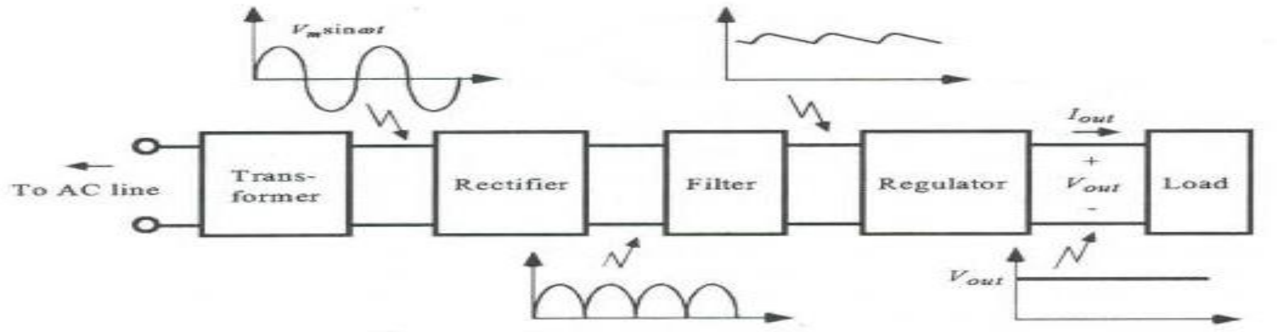
Hardware implementation deals in drawing the schematic on the plane paper according to the application, testing the schematic design over the breadboard using the various IC's to find if the design meets the objective, carrying out the PCB layout of the schematic tested on breadboard, finally preparing the board and testing the designed hardware.

The firmware part deals in programming the microcontroller so that it can control the operation of the IC's used in the implementation. In the present work, we have used the Orcad design software for PCB circuit design, the Keil µv3 software development tool to write and compile the source code, which has been written in the C language. The Proload programmer has been used to write this compile code into the microcontroller. The firmware implementation is explained in the next chapter.

The project design and principle are explained in this chapter using the block diagram and circuit diagram. The block diagram discusses about the required components of the design and wo2rking condition is explained using circuit diagram and system wiring diagram.

4.3 Power Supply

The input to the circuit is applied from the regulated power supply. The a.c. input i.e., 230V from the mains supply is step down by the transformer to 12V and is fed to a rectifier. The output obtained from the rectifier is a pulsating d.c voltage. So in order to get a pure d.c voltage, the output voltage from the rectifier is fed to a filter to remove any a.c components present even after rectification. Now, this voltage is given to a voltage regulator to obtain a pure constant dc voltage.



Components of a regulated power supply

Fig: 4.1

Transformer:

Usually, DC voltages are required to operate various electronic equipment and these voltages are 5V, 9V or 12V. But these voltages cannot be obtained directly. Thus the a.c input available at the mains supply i.e., 230V is to be brought down to the required voltage level. This is done by a transformer. Thus, a step down transformer is employed to decrease the voltage to a required level.

Rectifier:

The output from the transformer is fed to the rectifier. It converts A.C. into pulsating D.C. The rectifier may be a half wave or a full wave rectifier. In this project, a bridge rectifier is used because of its merits like good stability and full wave rectification.

Filter:

Capacitive filter is used in this project. It removes the ripples from the output of rectifier and smoothens the D.C. Output received from this filter is constant until the mains voltage and load is maintained constant. However, if either of the two is varied, D.C. voltage received at this point changes. Therefore a regulator is applied at the output stage.

Voltage Regulator:

As the name itself implies, it regulates the input applied to it. A voltage regulator is an electrical regulator designed to automatically maintain a constant voltage level. In this project, power supply of 5V and 12V are required. In order to obtain these voltage levels, 7805 and 7812 voltage regulators are to be used. The first number 78 represents positive supply and the numbers 05, 12 represent the required output voltage levels

4.4 LIQUID CRYSTAL DISPLAY:

LCD stands for **L**iquid **C**rystal **D**isplay. LCD is finding wide spread use replacing LEDs (seven segment LEDs or other multi segment LEDs) because of the following reasons:

1. The declining prices of LCDs.
2. The ability to display numbers, characters and graphics. This is in contrast to LEDs, which are limited to numbers and a few characters.
3. Incorporation of a refreshing controller into the LCD, thereby relieving the CPU of the task of refreshing the LCD. In contrast, the LED must be refreshed by the CPU to keep displaying the data.
4. Ease of programming for characters and graphics.

These components are “specialized” for being used with the microcontrollers, which means that they cannot be activated by standard IC circuits. They are used for writing different messages on a miniature LCD.

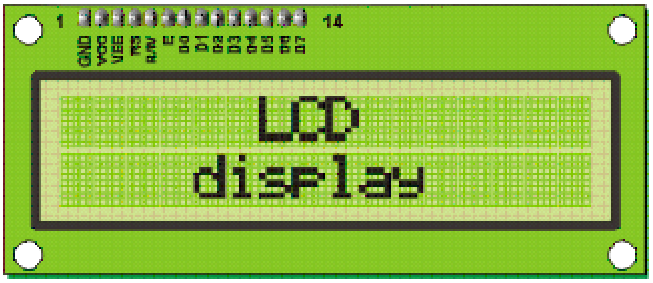


Fig: 4.2 LCD Display

A model described here is for its low price and great possibilities most frequently used in practice. It is based on the HD44780 microcontroller (Hitachi) and can display messages in two lines with 16 characters each. It displays all the alphabets, Greek letters, punctuation marks, mathematical symbols etc. In addition, it is possible to display symbols that user makes up on its own.

Automatic shifting message on display (shift left and right), appearance of the pointer, backlight etc. are considered as useful characteristics.

4.4.1 Pins Functions

There are pins along one side of the small printed board used for connection to the microcontroller. There are total of 14 pins marked with numbers (16 in case the background light is built in). Their function is described in the table below:

Function	**Pin Number**	**Name**	**Logic State**	**Description**
Ground	1	Vss	-	0V
Power supply	2	Vdd	-	+5V
Contrast	3	Vee	-	0 – Vdd
Control of operating	4	RS	0 1	D0 – D7 are interpreted as commands

				D0 – D7 are interpreted as data
	5	R/W	0 1	Write data (from controller to LCD) Read data (from LCD to controller)
	6	E	0 1 From 1 to 0	Access to LCD disabled Normal operating Data/commands are transferred to LCD
Data / commands	7	D0	0/1	Bit 0 LSB
	8	D1	0/1	Bit 1
	9	D2	0/1	Bit 2
	10	D3	0/1	Bit 3
	11	D4	0/1	Bit 4
	12	D5	0/1	Bit 5
	13	D6	0/1	Bit 6
	14	D7	0/1	Bit 7 MSB

4.4.2 LCD screen:

LCD screen consists of two lines with 16 characters each. Each character consists of 5x7 dot matrix. Contrast on display depends on the power supply voltage and whether messages are displayed in one or two lines. For that reason, variable voltage 0-Vdd is applied on pin marked as Vee. Trimmer potentiometer is usually used for that purpose. Some versions of displays have built in backlight (blue or green diodes). When used during operating, a resistor for current limitation should be used (like with any LE diode).

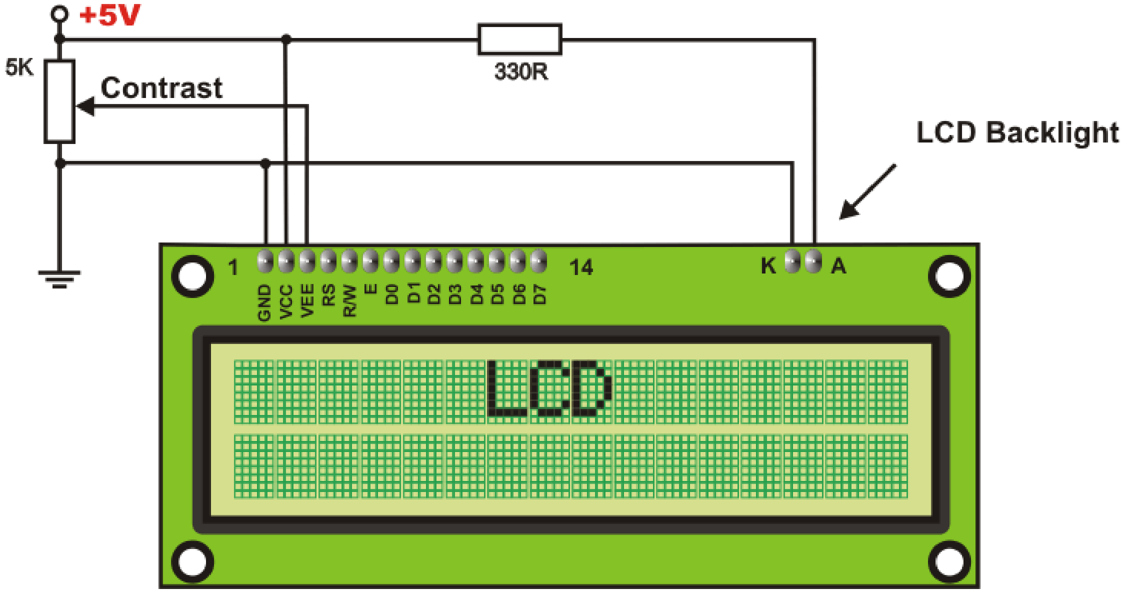


Fig: 4.3 LCD Screen

4.4.3 LCD Basic Commands

All data transferred to LCD through outputs D0-D7 will be interpreted as commands or as data, which depends on logic state on pin RS:

RS = 1 - Bits D0 - D7 are addresses of characters that should be displayed. Built in processor addresses built in "map of characters" and displays corresponding symbols. Displaying position is determined by DDRAM address. This address is either previously defined or the address of previously transferred character is automatically incremented.

RS = 0 - Bits D0 - D7 are commands which determine display mode. List of commands which LCD recognizes are given in the table below**:**

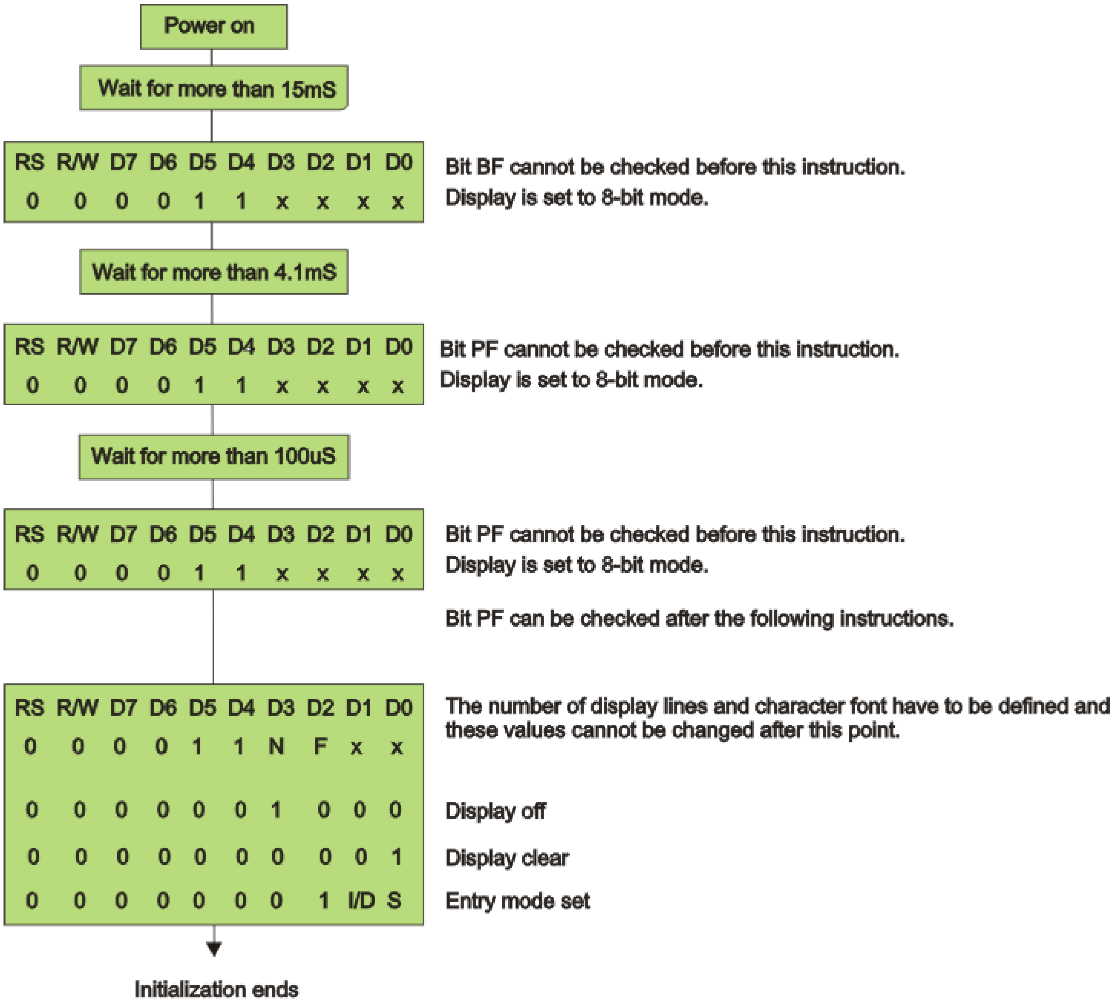
4.4.4 LCD Connection

Depending on how many lines are used for connection to the microcontroller, there are 8-bit and 4-bit LCD modes. The appropriate mode is determined at the beginning of the process in a phase called “initialization”. In the first case, the data are transferred through outputs D0-D7 as it has been already explained. In case of 4-bit LED mode, for the sake of saving valuable I/O pins of the microcontroller, there are only 4 higher bits (D4-D7) used for communication, while other may be left unconnected.

Consequently, each data is sent to LCD in two steps: four higher bits are sent first (that normally would be sent through lines D4-D7), four lower bits are sent afterwards. With the help of initialization, LCD will correctly connect and interpret each data received.

Besides, with regards to the fact that data are rarely read from LCD (data mainly are transferred from microcontroller to LCD) one more I/O pin may be saved by simple connecting R/W pin to the Ground. Such saving has its price.

Even though message displaying will be normally performed, it will not be possible to read from busy flag since it is not possible to read from display.



Power on

Wait for more than 15mS

RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	1	1	x	x	x	x

Bit BF cannot be checked before this instruction.
Display is set to 8-bit mode.

Wait for more than 4.1mS

RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	1	1	x	x	x	x

Bit PF cannot be checked before this instruction.
Display is set to 8-bit mode.

Wait for more than 100uS

RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	1	1	x	x	x	x

Bit PF cannot be checked before this instruction.
Display is set to 8-bit mode.

Bit PF can be checked after the following instructions.

RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0	
0	0	0	0	1	1	N	F	x	x	The number of display lines and character font have to be defined and these values cannot be changed after this point.
0	0	0	0	0	0	1	0	0	0	Display off
0	0	0	0	0	0	0	0	0	1	Display clear
0	0	0	0	0	0	0	1	I/D	S	Entry mode set

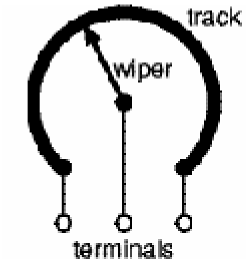
Initialization ends

Fig: Procedure on 8-bit initialization

Fig: 4.4 Procedure on 8-bit intilization

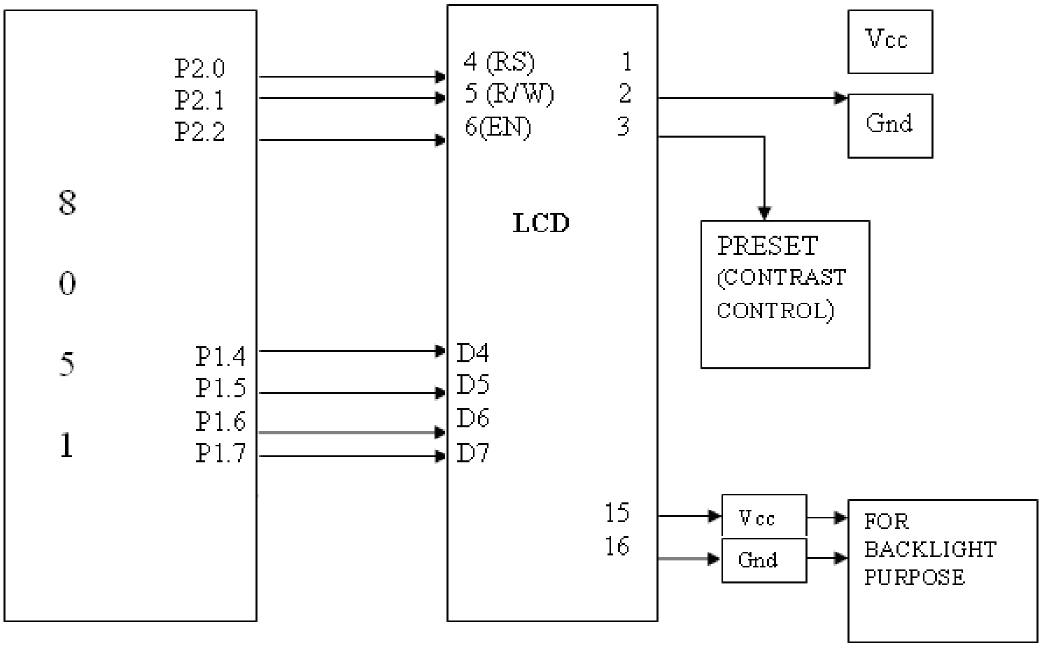
4.4.5 Contrast control:

To have a clear view of the characters on the LCD, contrast should be adjusted. To adjust the contrast, the voltage should be varied. For this, a preset is used which can behave like a variable voltage device. As the voltage of this preset is varied, the contrast of the LCD can be adjusted.



Variable resistor

4.4.6 LCD interface with the microcontroller (4-bit mode):



1

Fig: 4.4.6 LCD interface with the microcontroller (4-bit mode):

4.5 GSM (Global System for Mobile communications)

4.5.1 Introduction:

GSM (Global System for Mobile communications) is a cellular network, which means that mobile phones connect to it by searching for cells in the immediate vicinity. GSM networks operate in four different frequency ranges. Most GSM networks operate in the 900 MHz or 1800 MHz bands. Some countries in the Americas use the 850 MHz and 1900 MHz bands because the 900 and 1800 MHz frequency bands were already allocated.

The rarer 400 and 450 MHz frequency bands are assigned in some countries, where these frequencies were previously used for first-generation systems.

GSM-900 uses 890–915 MHz to send information from the mobile station to the base station (uplink) and 935–960 MHz for the other direction (downlink), providing 124 RF channels (channel numbers 1 to 124) spaced at 200 kHz. Duplex spacing of 45 MHz is used. In some countries the GSM-900 band has been extended to cover a larger frequency range. This 'extended GSM', E-GSM, uses 880–915 MHz (uplink) and 925–960 MHz (downlink), adding 50 channels (channel numbers 975 to 1023 and 0) to the original GSM-900 band. Time division multiplexing is used to allow eight full-rate or sixteen half-rate speech channels per radio frequency channel. There are eight radio timeslots (giving eight burst periods) grouped into what is called a TDMA frame. Half rate channels use alternate frames in the same timeslot. The channel data rate is 270.833 kbit/s, and the frame duration is 4.615 ms.

4.5.2 GSM Advantages:

GSM also pioneered a low-cost, to the network carrier, alternative to voice calls, the Short t message service (SMS, also called "text messaging"), which is now supported on other mobile standards as well. Another advantage is that the standard includes one worldwide Emergency telephone number, 112. This makes it easier for international travelers to connect to emergency services without knowing the local emergency number.

4.5.3 The GSM Network:

GSM provides recommendations, not requirements. The GSM specifications define the functions and interface requirements in detail but do not address the hardware. The GSM network is divided into three major systems: the switching system (SS), the base station system (BSS), and the operation and support system (OSS).

4.5.4 The Switching System:

The switching system (SS) is responsible for performing call processing and subscriber-related functions. The switching system includes the following functional units.

- **Home location register (HLR)**: The HLR is a database used for storage and management of subscriptions. The HLR is considered the most important database, as it stores permanent data about subscribers, including a subscriber's service profile, location information, and activity status. When an individual buys a subscription from one of the PCS operators, he or she is registered in the HLR of that operator.
- **Mobile services switching center (MSC):** The MSC performs the telephony switching functions of the system. It controls calls to and from other telephone and data systems. It also performs such functions as toll ticketing, network interfacing, common channel signaling, and others.
- **Visitor location register (VLR)**: The VLR is a database that contains temporary information about subscribers that is needed by the MSC in order to service visiting subscribers. The VLR is always integrated with the MSC. When a mobile station roams into a new MSC area, the VLR connected to that MSC will request data about the mobile station from the HLR. Later, if the mobile station makes a call, the VLR will have the information needed for call setup without having to interrogate the HLR each time.
- **Authentication center (AUC)**: A unit called the AUC provides authentication and encryption parameters that verify the user's identity and ensure the confidentiality of each call. The AUC protects network operators from different types of fraud found in today's cellular world.
- **Equipment identity register (EIR)**: The EIR is a database that contains information about the identity of mobile equipment that prevents calls from stolen, unauthorized, or defective mobile stations. The AUC and EIR are implemented as stand-alone nodes or as a combined AUC/EIR node.

4.5.5 The Base Station System (BSS):

All radio-related functions are performed in the BSS, which consists of base station controllers (BSCs) and the base transceiver stations (BTSs).

- **BSC**: The BSC provides all the control functions and physical links between the MSC and BTS. It is a high-capacity switch that provides functions such as handover, cell configuration data, and control of radio frequency (RF) power levels in base transceiver stations. A number of BSCs are served by an MSC.
- **BTS**: The BTS handles the radio interface to the mobile station. The BTS is the radio equipment (transceivers and antennas) needed to service each cell in the network. A group of BTSs are controlled by a BSC.

4.5.6 The Operation and Support System

The operations and maintenance center (OMC) is connected to all equipment in the switching system and to the BSC. The implementation of OMC is called the operation and support system (OSS). The OSS is the functional entity from which the network operator monitors and controls the system. The purpose of OSS is to offer the customer cost-effective support for centralized, regional and local operational and maintenance activities that are required for a GSM network. An important function of OSS is to provide a network overview and support the maintenance activities of different operation and maintenance organizations.

- **Message center (MXE)**: The MXE is a node that provides integrated voice, fax, and data messaging. Specifically, the MXE handles short message service, cell broadcast, voice mail, fax mail, e-mail, and notification.
- **Mobile service node (MSN)**: The MSN is the node that handles the mobile intelligent network (IN) services.

- **Gateway mobile services switching center (GMSC)**: A gateway is a node used to interconnect two networks. The gateway is often implemented in an MSC. The MSC is then referred to as the GMSC.
- **GSM inter-working unit (GIWU)**: The GIWU consists of both hardware and software that provides an interface to various networks for data communications. Through the GIWU, users can alternate between speech and data during the same call. The GIWU hardware equipment is physically located at the MSC/VLR.

4.5.8 GSM Network Areas:

The GSM network is made up of geographic areas. As shown in bellow figure, these areas include cells, location areas (LAs), MSC/VLR service areas, and public land mobile network (PLMN) areas.

4.5.9 Location Areas:

The cell is the area given radio coverage by one base transceiver station. The GSM network identifies each cell via the cell global identity (CGI) number assigned to each cell. The location area is a group of cells. It is the area in which the subscriber is paged. Each LA is served by one or more base station controllers, yet only by a single MSC Each LA is assigned a location area identity (LAI) number.

4.5.10 MSC/VLR service areas:

An MSC/VLR service area represents the part of the GSM network that is covered by one MSC and which is reachable, as it is registered in the VLR of the MSC.

4.5.11 PLMN service areas:

The PLMN service area is an area served by one network operator.

4.5.12 GSM Specifications:

Specifications for different personal communication services (PCS) systems vary among the different PCS networks. Listed below is a description of the specifications and characteristics for GSM.

- **Frequency band**: The frequency range specified for GSM is 1,850 to 1,990 MHz (mobile station to base station).
- **Duplex distance**: The duplex distance is 80 MHz. Duplex distance is the distance between the uplink and downlink frequencies. A channel has two frequencies, 80 MHz apart.
- **Channel separation**: The separation between adjacent carrier frequencies. In GSM, this is 200 kHz.
- **Modulation**: Modulation is the process of sending a signal by changing the characteristics of a carrier frequency. This is done in GSM via Gaussian minimum shift keying (GMSK).

- **Transmission rate**: GSM is a digital system with an over-the-air bit rate of 270 kbps.
- **Access method**: GSM utilizes the time division multiple access (TDMA) concept. TDMA is a technique in which several different calls may share the same carrier. Each call is assigned a particular time slot.
- **Speech coder**: GSM uses linear predictive coding (LPC). The purpose of LPC is to reduce the bit rate. The LPC provides parameters for a filter that mimics the vocal tract. The signal passes through this filter, leaving behind a residual signal. Speech is encoded at 13 kbps.

4.5.13 GSM Subscriber Services:

Dual-tone multifrequency (DTMF): DTMF is a tone signaling scheme often used for various control purposes via the telephone network, such as remote control of an answering machine. GSM supports full-originating DTMF.

Facsimile group III: GSM supports CCITT Group 3 facsimile. As standard fax machines are designed to be connected to a telephone using analog signals, a special fax converter connected to the exchange is used in the GSM system. This enables a GSM–connected fax to communicate with any analog fax in the network.

Short message services: A convenient facility of the GSM network is the short message service. A message consisting of a maximum of 160 alphanumeric characters can be sent to or from a mobile station. This service can be viewed as an advanced form of alphanumeric paging with a number of advantages. If the subscriber's mobile unit is powered off or has left the coverage area, the message is stored and offered back to the subscriber when the mobile is powered on or has reentered the coverage area of the network. This function ensures that the message will be received.

Cell broadcast: A variation of the short message service is the cell broadcast facility. A message of a maximum of 93 characters can be broadcast to all mobile subscribers in a certain geographic area. Typical applications include traffic congestion warnings and reports on accidents.

Voice mail: This service is actually an answering machine within the network, which is controlled by the subscriber. Calls can be forwarded to the subscriber's voice-mail box and the subscriber checks for messages via a personal security code.

Fax mail: With this service, the subscriber can receive fax messages at any fax machine. The messages are stored in a service center from which they can be retrieved by the subscriber via a personal security code to the desired fax number

4.5.14 Supplementary Services:

GSM supports a comprehensive set of supplementary services that can complement and support both telephony and data services.

Call forwarding: This service gives the subscriber the ability to forward incoming calls to another number if the called mobile unit is not reachable, if it is busy, if there is no reply, or if call forwarding is allowed unconditionally.

Barring of outgoing calls: This service makes it possible for a mobile subscriber to prevent all outgoing calls.

Barring of incoming calls: This function allows the subscriber to prevent incoming calls. The following two conditions for incoming call barring exist: baring of all incoming calls and barring of incoming calls when roaming outside the home PLMN.

Advice of charge (AoC): The AoC service provides the mobile subscriber with an estimate of the call charges. There are two types of AoC information: one that provides the subscriber with an estimate of the bill and one that can be used for immediate charging purposes. AoC for data calls is provided on the basis of time measurements.

Call hold: This service enables the subscriber to interrupt an ongoing call and then subsequently reestablish the call. The call hold service is only applicable to normal telephony.

Call waiting: This service enables the mobile subscriber to be notified of an incoming call during a conversation. The subscriber can answer, reject, or ignore the incoming call. Call waiting is applicable to all GSM telecommunications services using a circuit-switched connection.

Multiparty service: The multiparty service enables a mobile subscriber to establish a multiparty conversation—that is, a simultaneous conversation between three and six subscribers. This service is only applicable to normal telephony.

Calling line identification presentation/restriction: These services supply the called party with the integrated services digital network (ISDN) number of the calling party. The restriction service enables the calling party to restrict the presentation. The restriction overrides the presentation.

Closed user groups (CUGs): CUGs are generally comparable to a PBX. They are a group of subscribers who are capable of only calling themselves and certain numbers

4.5.15 Main AT commands:

"AT command set for GSM Mobile Equipment" describes the Main AT commands to communicate via a serial interface with the GSM subsystem of the phone.

AT commands are instructions used to control a modem. AT is the abbreviation of Attention. Every command line starts with "AT" or "at". That's why modem commands are called AT commands. Many of the commands that are used to control wired dial-up modems, such as ATD (Dial), ATA (Answer), ATH (Hook control) and ATO (Return to online data state), are also supported by GSM/GPRS modems and mobile phones. Besides this common AT command set, GSM/GPRS modems and mobile phones support an AT command set that is specific to the GSM technology, which includes SMS-related commands like AT+CMGS (Send SMS message), AT+CMSS (Send SMS message from storage), AT+CMGL (List SMS messages) and AT+CMGR (Read SMS messages).

Note that the starting "AT" is the prefix that informs the modem about the start of a command line. It is not part of the AT command name. For example, D is the actual AT command name in ATD and +CMGS is the actual AT command name in AT+CMGS. However, some books and web sites use them interchangeably as the name of an AT command.

Here are some of the tasks that can be done using AT commands with a GSM/GPRS modem or mobile phone:

- Get basic information about the mobile phone or GSM/GPRS modem. For example, name of manufacturer (AT+CGMI), model number (AT+CGMM), IMEI number (International Mobile Equipment Identity) (AT+CGSN) and software version (AT+CGMR).
- Get basic information about the subscriber. For example, MSISDN (AT+CNUM) and IMSI number (International Mobile Subscriber Identity) (AT+CIMI).
- Get the current status of the mobile phone or GSM/GPRS modem. For example, mobile phone activity status (AT+CPAS), mobile network registration status (AT+CREG), radio signal strength (AT+CSQ), battery charge level and battery charging status (AT+CBC).
- Establish a data connection or voice connection to a remote modem (ATD, ATA, etc).
- Send and receive fax (ATD, ATA, AT+F*).
- Send (AT+CMGS, AT+CMSS), read (AT+CMGR, AT+CMGL), write (AT+CMGW) or delete (AT+CMGD) SMS messages and obtain notifications of newly received SMS messages (AT+CNMI).
- Read (AT+CPBR), write (AT+CPBW) or search (AT+CPBF) phonebook entries.
- Perform security-related tasks, such as opening or closing facility locks (AT+CLCK), checking whether a facility is locked (AT+CLCK) and changing passwords (AT+CPWD).

  (Facility lock examples: SIM lock [a password must be given to the SIM card every time the mobile phone is switched on] and PH-SIM lock [a certain SIM card is associated with the mobile phone. To use other SIM cards with the mobile phone, a password must be entered.])

- Control the presentation of result codes / error messages of AT commands. For example, you can control whether to enable certain error messages (AT+CMEE) and whether error messages should be displayed in numeric format or verbose format (AT+CMEE=1 or AT+CMEE=2).
- Get or change the configurations of the mobile phone or GSM/GPRS modem. For example, change the GSM network (AT+COPS), bearer service type (AT+CBST), radio link protocol parameters (AT+CRLP), SMS center address (AT+CSCA) and storage of SMS messages (AT+CPMS).
- Save and restore configurations of the mobile phone or GSM/GPRS modem. For example, save (AT+CSAS) and restore (AT+CRES) settings related to SMS messaging such as the SMS center address.

4.6 RADIO FREQUENCY:

Radio frequency (**RF**): is a frequency or rate of oscillation within the range of about 3 Hz to 300 GHz. This range corresponds to frequency of alternating current electrical signals used to produce and detect radio waves. Since most of this range is beyond the vibration rate that most mechanical systems can respond to, RF usually refers to oscillations in electrical circuits or electromagnetic radiation.

4.6.1 Properties of RF:

Electrical currents that oscillate at RF have special properties not shared by direct current signals. One such property is the ease with which it can ionize air to create a conductive path through air. This property is exploited by 'high frequency' units used in electric arc welding. Another special property is an electromagnetic force that drives the RF current to the surface of conductors, known as the skin effect. Another property is the ability to appear to flow through paths that contain insulating material, like the dielectric insulator of a capacitor. The degree of effect of these properties depends on the frequency of the signals.

4.6.2 DIFFERENT RANGES PRESENT IN RF AND APPLICATIONS IN THEIR RANGES

Extremely low frequency:

ELF 3 to 30 Hz

10,000 km to 100,000 km

Directly audible when converted to sound, communication with submarines

Super low frequency:

SLF 30 to 300 Hz

1,000 km to 10,000 km

Directly audible when converted to sound, AC power grids (50 hertz and 60 hertz)

Ultra low frequency:

ULF 300 to 3000 Hz

100 km to 1,000 km

Directly audible when converted to sound, communication with mines

Very low frequency:

VLF 3 to 30 kHz

10 km to 100 km

Directly audible when converted to sound (below ca. 18-20 kHz; or "ultrasound" 20-30+ kHz)

Low frequency:

LF 30 to 300 kHz

1 km to 10 km

AM broadcasting, navigational beacons, low FER.

Medium frequency:

MF 300 to 3000 kHz

100 m to 1 km

Navigational beacons, AM broadcasting, maritime and aviation communication

High frequency:

HF 3 to 30 MHz

10 m to 100 m

Shortwave, amateur radio, citizens' band radio

Very high frequency:

VHF 30 to 300 MHz

1 m to 10 m

FM broadcasting broadcast television, aviation, GPR

Ultra high frequency:

UHF 300 to 3000 MHz

10 cm to 100 cm

Broadcast television, mobile telephones, cordless telephones, wireless networking, remote keyless entry for automobiles, microwave ovens, GPR

Super high frequency:

SHF 3 to 30 GHz

1 cm to 10 cm

Wireless networking, satellite links, microwave links, Satellite television, door openers.

Extremely high frequency:

EHF 30 to 300 GHz

1 mm to 10 mm

Microwave data links, radio astronomy, remote sensing, advanced weapons systems, advanced security scanning

4.6.3 Brief description of RF:

Radio frequency (abbreviated RF) is a term that refers to alternating current (AC) having characteristics such that, if the current is input to an antenna, an electromagnetic (EM) field is generated suitable for wireless broadcasting and/or communications. These frequencies cover a significant portion of the electromagnetic radiation spectrum, extending from nine kilohertz (9 kHz),the lowest allocated wireless communications frequency (it's within the range of human hearing), to thousands of gigahertz(GHz).

When an RF current is supplied to an antenna, it gives rise to an electromagnetic field that propagates through space. This field is sometimes called an RF field; in less technical jargon it is a "radio wave." Any RF field has a wavelength that is inversely proportional to the frequency. In the atmosphere or in outer space, if *f* is the frequency in megahertz and *s*is the wavelength in meters, then

$$s = 300/f$$

The frequency of an RF signal is inversely proportional to the wavelength of the EM field to which it corresponds. At 9 kHz, the free-space wavelength is approximately 33 kilometers (km) or 21 miles (mi). At the highest radio frequencies, the EM wavelengths measure approximately one millimeter (1 mm). As the frequency is increased beyond that of the RF spectrum, EM energy takes the form of infrared (IR), visible, ultraviolet (UV), X rays, and gamma rays.

Many types of wireless devices make use of RF fields. Cordless and cellular telephone, radio and television broadcast stations, satellite communications systems, and two-way radio services all operate in the RF spectrum. Some wireless devices operate at IR or visible-light frequencies, whose electromagnetic wavelengths are shorter than those of RF fields. Examples include most television-set remote-control boxes some cordless computer keyboards and mice and a few wireless hi-fi stereo headsets.

The RF spectrum is divided into several ranges, or bands. With the exception of the lowest-frequency segment, each band represents an increase of frequency corresponding to an order of magnitude (power of 10). The table depicts the eight bands in the RF spectrum, showing frequency and bandwidth ranges. The SHF and EHF bands are often referred to as the microwave spectrum.

4.6.4 ADVANTAGES AND DISADVANTAGES OF RF

RF Advantages:

1. No line of sight is needed.
2. Not blocked by common materials: It can penetrate most solids and pass through walls.
3. Longer range.
4. It is not sensitive to the light;
5. It is not much sensitive to the environmental changes and weather conditions.

RF Disadvantages:

1. Interference: communication devices using similar frequencies - wireless phones, scanners, wrist radios and personal locators can interfere with transmission
2. Lack of security: easier to "eavesdrop" on transmissions since signals are spread out in space rather than confined to a wire
3. Higher cost than infrared
4. Federal Communications Commission(FCC) licenses required for some products
5. Lower speed: data rate transmission is lower than wired and infrared transmission

WHAT ARE THE MAIN REQUIREMENTS FOR THE COMMUNICATION USING RF?

- **RF Transmitter**
- **RF Receiver**
- **Encoder and Decoder**

4.6.5 RF TRANSMITTER STT-433MHz:

STT-433 MHz TRANSMITTER

4.6.6 FACTORS INFLUENCED TO CHOOSE STT-433MHz

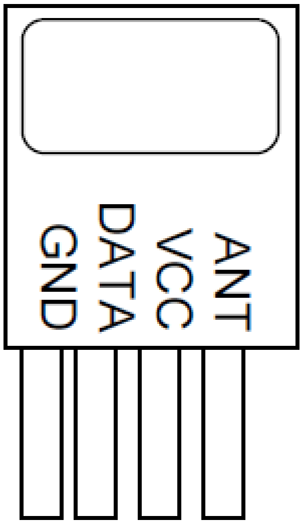
ABOUT THE TRANSMITTER:

- The STT-433 is ideal for remote control applications where low cost and longer range is required.
- The transmitter operates from a1.5-12V supply, making it ideal for battery-powered applications.
- The transmitter employs a SAW-stabilized oscillator, ensuring accurate frequency control for best range performance.
- The manufacturing-friendly SIP style package and low-cost make the STT-433 suitable for high volume applications.

4.6.7 Features:

- 433.92 MHz Frequency
- Low Cost
- 1.5-12V operation
- Small size

4.6.8 PIN DESCRIPTION:



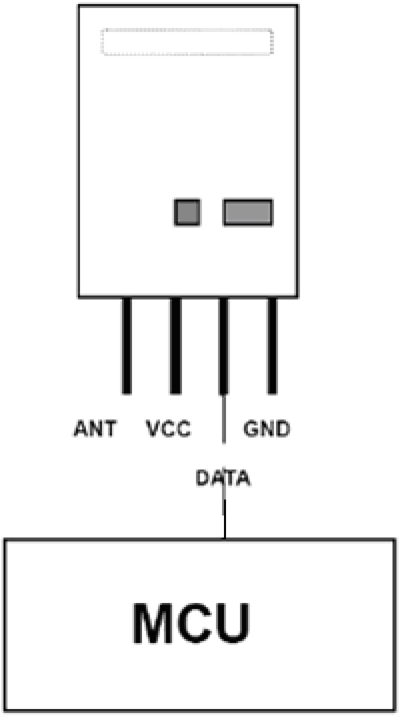
GND: Transmitter ground. Connect to ground plane

DATA: Digital data input. This input is CMOS compatible and should be driven with CMOS level inputs.

VCC: Operating voltage for the transmitter. VCC should be bypassed with a .01uF ceramic capacitor and filtered with a 4.7uF tantalum capacitor. Noise on the power supply will degrade transmitter noise performance.

ANT: 50 ohm antenna output. The antenna port impedance affects output power and harmonic emissions. Antenna can be single core wire of approximately 17cm length or PCB trace antenna.

4.6.9 APPLICATION:



The typical connection shown in the above figure cannot work exactly at all times because there will be no proper synchronization between the transmitter and the microcontroller unit. i.e., whatever the microcontroller sends the data to the transmitter, the transmitter is not able to accept this data as this will be not in the radio frequency range. Thus, we need an intermediate device which can accept the input from the microcontroller, process it in the range of radio frequency range and then send it to the transmitter. Thus, an encoder is used.

The encoder used here is **HT640** from **HOLTEK SEMICONDUCTORS INC.**

4.7 HT12E

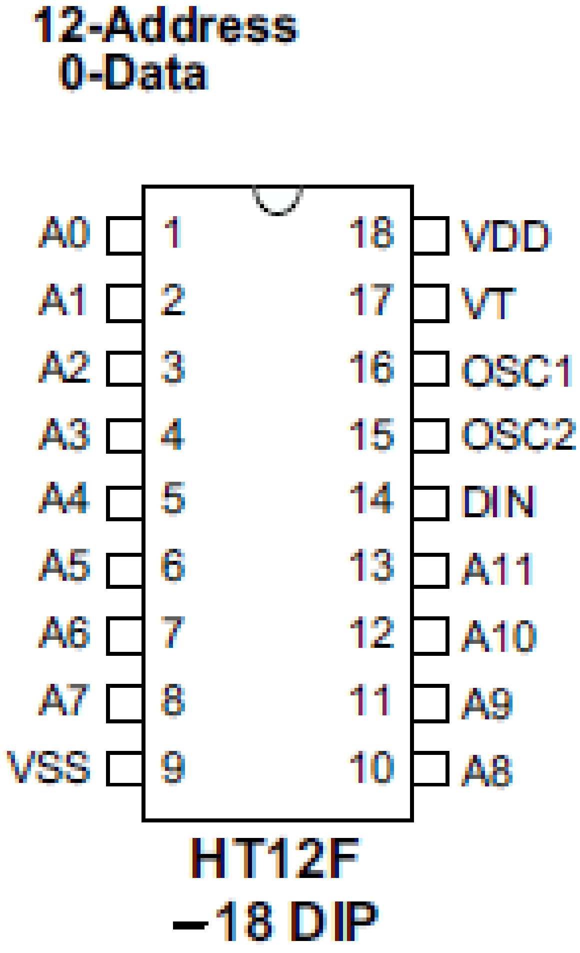
Introduction:

The 2^12 encoders are a series of CMOS LSIs (large scale integrated circuits) for remote control system applications. They are capable of encoding information which consists of N address bits and 12_N data bits. Each address/data input can be set to one of the two logic states. The programmed addresses/data are transmitted together with the header bits via an RF or an infrared transmission medium upon receipt of a trigger signal. The capability to select a TE trigger on the HT12E.

Features:

- Operating voltage
- 2.4V~12V for the HT12E
- Low power and high noise immunity CMOS technology
- Low standby current: 0.1_A at VDD=5V
- Minimum transmission word
- Four words for the HT12E
- Built-in oscillator needs only 5% resistor
- Data code has positive polarity
- Minimal external components
- HT12E: 18-pin DIP/20-pin SOP package

4.7.1 PIN Diagram:



PIN Description:

A0~A7 : (NMOS Transmission Gate Protection Diode). Input pins for address A0~A7 setting. These pins can be externally set to VSS or left open.

AD8~AD11: (NMOS Transmission Gate protection Diode) Input pins for address/data AD8~AD11 setting. These pins can be externally set to VSS or left open

D8~D1 : (CMOS IN) Pull-high Input pins for data D8~D11 setting and transmission enable, active low. These pins should be externally set to VSS or left open

DOUT: (CMOS OUT) Encoder data serial transmission output

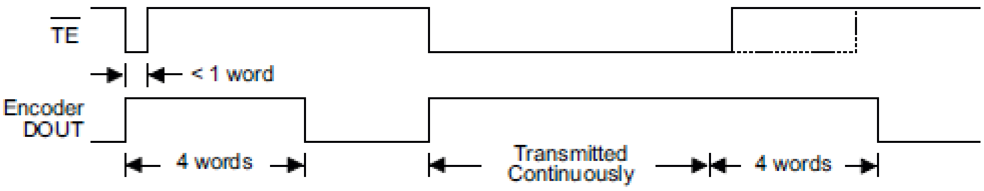
L/MB: (CMOS IN) Pull-high Latch/Momentary transmission format selection pin:

Latch: Floating or VDD

Momentary: VSS

4.7.2 Operation:

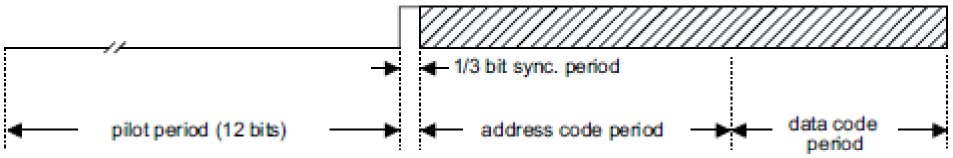
The 2^12 series of encoders begin a 4-word transmission cycle upon receipt of a transmission enable (TE for the HT12E active low). This cycle will repeat itself as long as the transmission enable (TE or D8~D11) is held low. Once the transmission enable returns high the encoder output completes its final cycle and then stops.



Transmission timing for the HT12E

4.7.3 Information word:

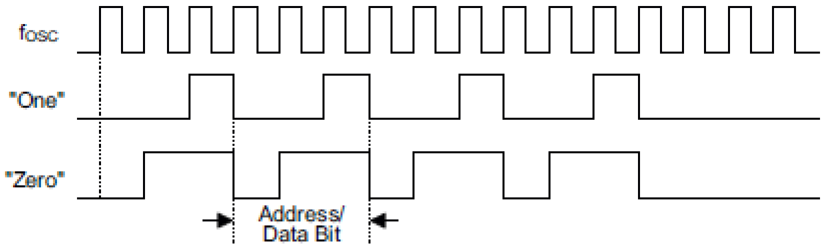
If L/MB=1 the device is in the latch mode (for use with the latch type of data decoders). When the transmission enable is removed during a transmission, the DOUT pin outputs a complete word and then stops. On the other hand, if L/MB=0 the device is in the momentary mode (for use with the momentary type of data decoders). When the transmission enable is removed during a transmission, the DOUT outputs a complete word and then adds 7 words all with the _1_ data code. An information word consists of 4 periods.



Composition of information

4.7.4 Address/data waveform:

Each programmable address/data pin can be externally set to one of the two logic states.

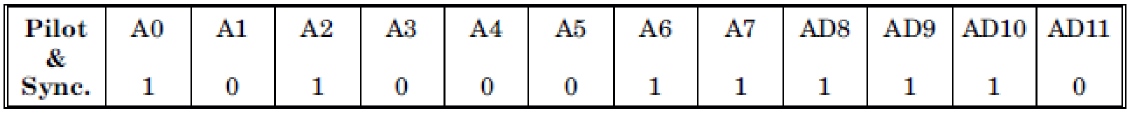


Address/Data bit waveform for the HT12E

4.7.5 Address/data programming (preset):

The status of each address/data pin can be individually pre-set to logic _high_ or _low_. If a transmission enable signal is applied, the encoder scans and transmits the status of the 12 bits of address/data serially in the order A0 to AD11 for the HT12E. During information transmission these bits are transmitted with a preceding synchronization bit. If the trigger signal is not applied, the chip enters the standby mode and consumes a reduced current of less than 1A for a supply voltage of 5V. Usual applications preset the address pins with individual security codes using DIP switches or PCB wiring, while the data is selected by push buttons or electronic switches.

The transmitted information is as shown:



Pilot & Sync.	A0	A1	A2	A3	A4	A5	A6	A7	AD8	AD9	AD10	AD11
	1	0	1	0	0	0	1	1	1	1	1	0

Address/Data sequence:

The following provides the address/data sequence table for various models of the 212 series of encoders. The correct device should be selected according to the individual address and data requirements

Part No. :	0	1	2	3	4	5	6	7	8	9	10	11
Address/Data Bits:	A0	A1	A2	A3	A4	A5	A6	A7	AD8	AD9	AD10	AD11

Transmissions enable:

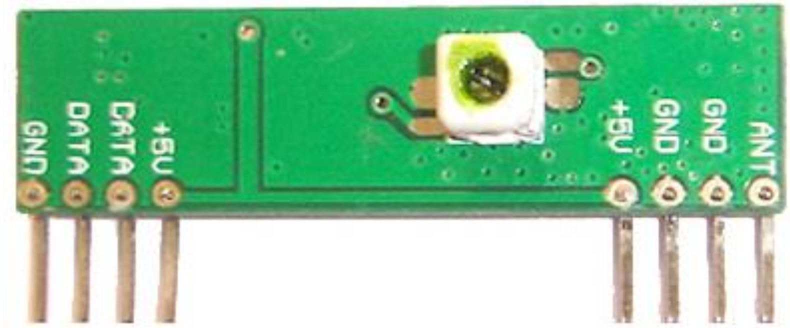
For the HT12E encoders, transmission is enabled by applying a low signal to the TE pin.

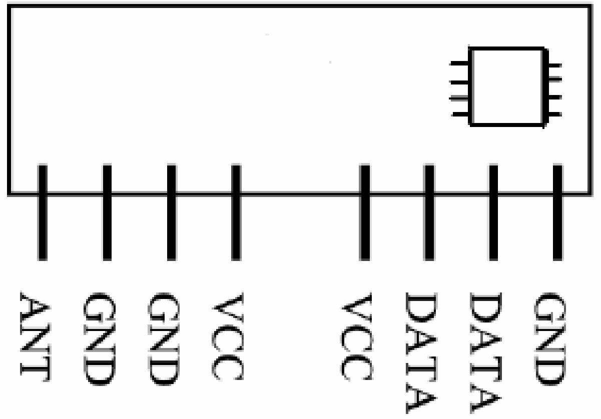
4.7.6 Applications:

- Burglar alarm system
- Smoke and fire alarm system
- Garage door controllers
- Car door controllers
- Car alarm system
- Security system
- Cordless telephones

4.7.8 FACTOR INFLUENCED TO CHOOSE STR-433MHz

4.7.8.1 RF RECEIVER STR-433 MHz:





The data is received by the RF receiver from the antenna pin and this data is available on the data pins. Two Data pins are provided in the receiver module. Thus, this data can be used for further applications.

PINOUT:

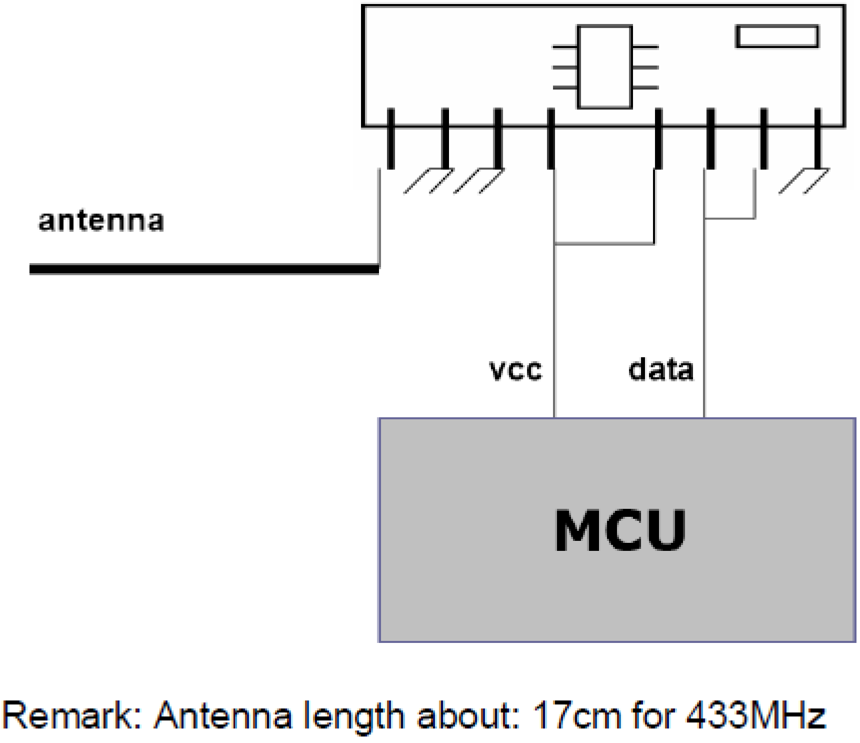
ANT: Antenna input.

GND: Receiver Ground. Connect to ground plane.

VCC (5V): VCC pins are electrically connected and provide operating voltage for the receiver. VCC can be applied to either or both. VCC should be bypassed with a .1µF ceramic capacitor. Noise on the power supply will degrade receiver sensitivity.

DATA: Digital data output. This output is capable of driving one TTL or CMOS load. It is a CMOS compatible output

4.7.9 APPLICATIONS:



Similarly, as the transmitter requires an encoder, the receiver module requires a decoder.

The decoder used is **HT-12E** HOLTEK **SEMICONDUCTOR INC.**

4.8 HT12D

4.8.1 Introduction:

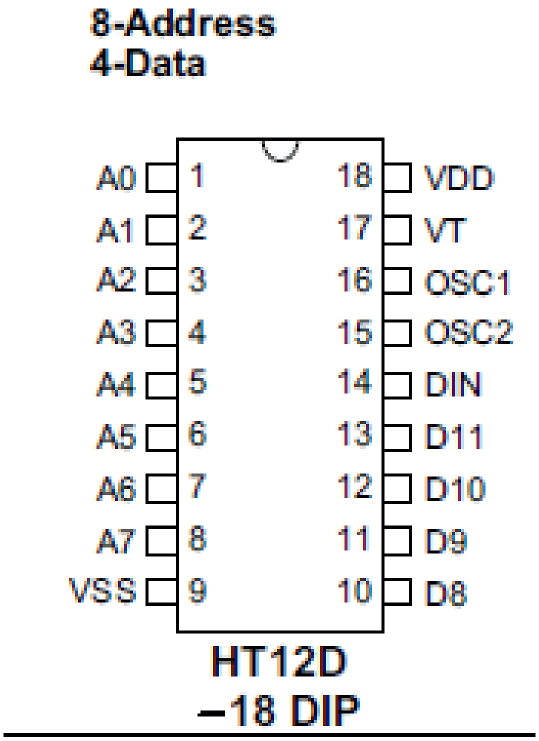
HT12D is having 2^12 decoders are a series of CMOS LSIs (large scale integrated circuits) for remote control system applications. It is an 18 Pin DIP/20 SOP package. They are paired with Hotlink's 2 12 series of encoders (refer to the encoder/decoder cross reference table). For proper operation, a pair of encoder/decoder with the same number of addresses and data format should be chosen. The decoders receive serial addresses and data from a programmed 2 12 series of encoders that are transmitted by a carrier using an RF or an IR transmission medium. They compare the serial input data three times continuously with their local addresses. If no error or unmatched codes are found, the input data codes are decoded and then transferred to the output pins. The VT pin also goes high to indicate a valid transmission. The 2^12 series of decoders are capable of decoding information that consist of N bits of address and 12_N bits of data. Of this series, the HT12D is arranged to provide 8 address bits and 4 data bits, and HT12F is used to decode 12 bits of address information.

Features:

- Operating voltage: 2.4V~12V
- Low power and high noise immunity CMOS technology
- Low standby current
- Capable of decoding 12 bits of information
- Pair with Hotlink's 2^12 series of encoders
- Binary address setting
- Received codes are checked 3 times
- Address/Data number combination
- HT12D: 8 address bits and 4 data bits
- Built-in oscillator needs only 5% resistor
- Valid transmission indicator

- Easy interface with an RF or an infrared transmission medium
- Minimal external components

4.8.2 PIN Diagram:



4.8.3 PIN Description:

A0~A11 : (NMOS TRANSMISSION GATE) Input pins for address A0~A11 setting.

They can be externally set to VDD or VSS.

D8~D11: (CMOS OUT) Output data pins

DIN : (CMOS IN) Serial data input pin

VT : (CMOS OUT) Valid transmission, active high

OSC : (OSCILLATOR) oscillator input pin

OSC2: (OSCILLATOR) oscillator output pin

VSS: Negative power supply (GND)

VDD: Positive power supply.

4.8.4 Maximum ratings:

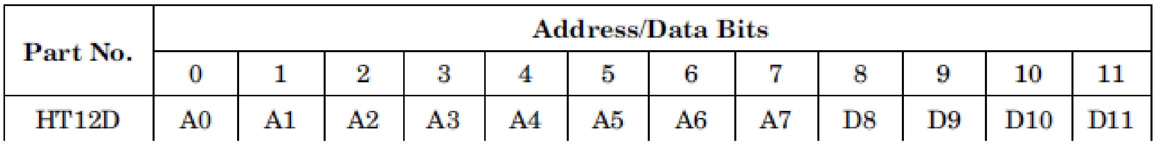
Supply Voltage	-0.3V to 13V
Storage Temperature	-50 degree Centigrade to 125 degree Centigrade
Input Voltage	VSS-0.3 to VDD+0.3V
Operating Temperature	-20degree Centigrade to 75degre Centigrade

4.8.5 Operation:

The 2^12 series of decoders provides various combinations of addresses and data pins in different packages so as to pair with the 2^12 series of encoders. The decoders receive data that are transmitted by an encoder and interpret the first N bits of code period as addresses and the last 12_N bits as data, where N is the address code number. A signal on the DIN pin activates the oscillator which in turn decodes the incoming address and data. The decoders will then check the received address three times continuously. If the received address codes all match the contents of the decoder's local address, the 12_N bits of data are decoded to activate the output pins and the VT pin is set high to indicate a valid transmission. This will last unless the address code is incorrect or no signal is received. The output of the VT pin is high only when the transmission is valid. Otherwise it is always low. Output type is latch (4 latch type data pins).

4.8.6 Address/Data sequence:

The following table provides address/data sequence for various models of the 2^12 series of decoders. A correct device should be chosen according to the requirements of the individual addresses and data.



Part No.	Address/Data Bits											
	0	1	2	3	4	5	6	7	8	9	10	11
HT12D	A0	A1	A2	A3	A4	A5	A6	A7	D8	D9	D10	D11

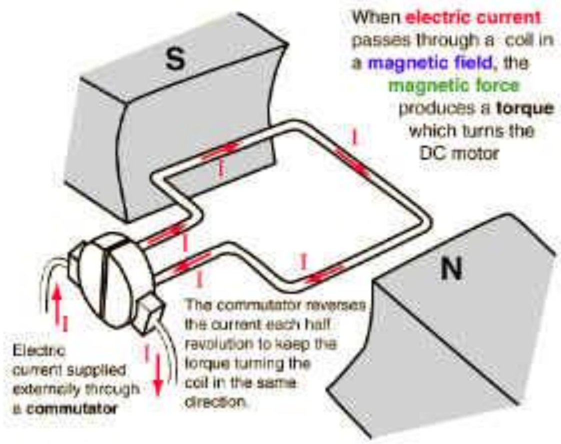
4.8.7 Applications:

- Burglar alarm system
- Smoke and fire alarm system
- Garage door controllers
- Car door controllers
- Car alarm system
- Security system
- Cordless telephones

- Other remote control systems

4.9 DC MOTOR

A DC motor is designed to run on DC electric power. Two examples of pure DC designs are Michael Faraday's homopolar motor (which is uncommon), and the ball bearin motor, which is (so far) a novelty. By far the most common DC motor types are the brushed and brushless types, which use internal and external commutation respectively to create an oscillating AC current from the DC source -- so they are not purely DC machines in a strict sense.


S
N
When electric current passes through a coil in a magnetic field, the magnetic force produces a torque which turns the DC motor
I
I
I
I
I
I
The commutator reverses the current each half revolution to keep the torque turning the coil in the same direction.
I
Electric current supplied externally through a commutator
I

4.9.1 Types of dcmotors:

1. Brushed DC Motors
2. Brushless DC motors
3. Coreless DC motors

4.9.2 Brushed DC motors:

The classic DC motor design generates an oscillating current in a wound rotor with a split ring commutator, and either a wound or permanent magnet stator. A rotor consists of a coil wound around a rotor which is then powered by any type of battery.

Many of the limitations of the classic commutator DC motor are due to the need for brushes to press against the commutator. This creates friction. At higher speeds, brushes have increasing difficulty in maintaining contact. Brushes may bounce off the irregularities in the commutator surface, creating sparks. This limits the maximum speed of the machine. The current density per unit area of the brushes limits the output of the motor. The imperfect electric contact also causes electrical noise. Brushes eventually wear out and require replacement, and the commutator itself is subject to wear and maintenance. The commutator assembly on a large machine is a costly element, requiring precision assembly of many parts. there are three types of dc motor 1. dc series motor 2. dc shunt motor 3. dc compound motor - these are also two type a. cummulative compound b. deffercial compounnd

4.9.3 Brushless DC motors:

Some of the problems of the brushed DC motor are eliminated in the brushless design. In this motor, the mechanical "rotating switch" or commutator/brushgear assembly is replaced by an external electronic switch synchronised to the rotor's position. Brushless motors are typically 85-90% efficient, whereas DC motors with brush gear are typically 75-80% efficient.

Midway between ordinary DC motors and stepper motors lies the realm of the brushless DC motor. Built in a fashion very similar to stepper motors, these often use a permanent magnet external rotor, three phases of driving coils, one or more Hall effect sensors to sense the position of the rotor, and the associated drive electronics. The coils are activated, one phase after the other, by the drive electronics as cued by the signals from the Hall effect sensors. In effect, they act as three-phase synchronous motors containing their own variable-frequency drive electronics. A specialized class of brushless DC motor controllers utilize EMF feedback through the main phase connections instead of Hall effect sensors to determine position and velocity. These motors are used extensively in electric radio-controlled vehicles. When configured with the magnets on the outside, these are referred to by mode lists as out runner motors.

Brushless DC motors are commonly used where precise speed control is necessary, as in computer disk drives or in video cassette recorders, the spindles within CD, CD-ROM (etc.)

drives, and mechanisms within office products such as fans, laser printers and photocopiers. They have several advantages over conventional motors:

- Compared to AC fans using shaded-pole motors, they are very efficient, running much cooler than the equivalent AC motors. This cool operation leads to much-improved life of the fan's bearings.
- Without a commutator to wear out, the life of a DC brushless motor can be significantly longer compared to a DC motor using brushes and a commutator. Commutation also tends to cause a great deal of electrical and RF noise; without a commutator or brushes, a brushless motor may be used in electrically sensitive devices like audio equipment or computers.
- The same Hall effect sensors that provide the commutation can also provide a convenient tachometer signal for closed-loop control (servo-controlled) applications. In fans, the tachometer signal can be used to derive a "fan OK" signal.
- The motor can be easily synchronized to an internal or external clock, leading to precise speed control.
- Brushless motors have no chance of sparking, unlike brushed motors, making them better suited to environments with volatile chemicals and fuels. Also, sparking generates ozone which can accumulate in poorly ventilated buildings risking harm to occupants' health.
- Brushless motors are usually used in small equipment such as computers and are generally used to get rid of unwanted heat.
- They are also very quiet motors which is an advantage if being used in equipment that is affected by vibrations.

Modern DC brushless motors range in power from a fraction of a watt to many kilowatts. Larger brushless motors up to about 100 kW rating are used in electric vehicles. They also find significant use in high-performance electric model aircraft.

4.9.4 Coreless DC motors:

Nothing in the design of any of the motors described above requires that the iron (steel) portions of the rotor actually rotate; torque is exerted only on the windings of the electromagnets. Taking advantage of this fact is the coreless DC motor, a specialized form of a brush or brushless DC motor. Optimized for rapid acceleration, these motors have a rotor that is constructed without any iron core. The rotor can take the form of a winding-filled cylinder inside the stator magnets, a basket surrounding the stator magnets, or a flat *pancake* (possibly formed on a printed wiring board) running between upper and lower stator magnets. The windings are typically stabilized by being impregnated with Electrical epoxy potting systems. Filled epoxies that have moderate mixed viscosity and a long gel time. These systems are highlighted by low shrinkage and low exotherm. Typically UL 1446 recognized as a potting compound for use up to 180C (Class H) UL File No. E 210549.

Because the rotor is much lighter in weight (mass) than a conventional rotor formed from copper windings on steel laminations, the rotor can accelerate much more rapidly, often achieving a mechanical time constant under 1 ms. This is especially true if the windings use aluminum rather than the heavier copper. But because there is no metal mass in the rotor to act as a heat sink, even small coreless motors must often be cooled by forced air.

These motors were commonly used to drive the capstan(s) of magnetic tape drives and are still widely used in high-performance servo-controlled systems, like radio-controlled vehicles/aircraft, humanoid robotic systems, industrial automation, medical devices, etc

4.10 SMTP (Simple Mail Transfer Protocol)

The basic internet mail protocols provide mail and message exchange between TCP/IP hosts. SMTP is an internet mail transfer protocol.

- The term Simple Mail Transfer Protocol (SMTP) is frequently used to refer to the combined set of protocols.

- Data sent via SMTP is 7-bit ASCII data.

- RFC 1652 is titled SMTP Service Extension for 8-bit-MIMEtransport, since the MIME standard allows messages to be declared as consisting of 8-bit data rather than 7-bit data. Such messages cannot be transmitted

- Whenever a client SMTP attempts to send 8-bit data to a server that does not support this extension, the client SMTP must either encode the message contents into a 7-bit representation compliant with the MIME (Multipurpose Internet Mail Extensions) standard or return a permanent error to the user.

- A protocol for message size declaration (RFC 1870) that allows a server to inform a client of the maximum size message it can accept.

- Each of these SMTP Service Extensions is a draft standard protocol and each has a status of elective.

SMTP works:

- SMTP is based on end-to-end delivery; an SMTP client will contact the destination host's SMTP server directly to deliver the mail.

- There is a possibility to exchange mail between the TCP/IP SMTP mailing system and the locally used mailing systems. These applications are called gateways or mail bridges.

- The SMTP end-to-end transmission is host-to-gateway,

- SMTP end-to-end transmission is host to gateway, gateway to host or gateway to gateway.

- The client SMTP uses the Domain Name System to determine the IP address of the destination mailbox.
- The message can contained headers and contents i.e. data.

Transaction flow:

- Sender SMTP establishes a TCP connection with the destination SMTP.
- Sending message (hello) is sent, to which the receiver will identify himself or herself by sending back its domain name. The sender SMTP can use this to verify if it contacted the Wright destination.
- Sending MAIL command to receiver (start of mail) it can report errors. If accepted, the receiver replies with 250 OK.

CHAPTER 5

FIRMWARE IMPLEMENTATION OF THE PROJECT DESIGN

KEIL COMPILER

5.1 KEILµVISION

Keil compiler is software used where the machine language code is written and compiled. After compilation, the machine source code is converted into hex code which is to be dumped into the microcontroller for further processing. Keil compiler also supports C language code.

5.1.1 Steps To Write a C Program InKeil and Compile It

- Install the Keil Software in the PC in any of the drives.
- After installation, an icon will be created with the name "Keil µVision3". Just drag this icon onto the desktop so that it becomes easy whenever you try to write programs in keil.
- Double click on this icon to start the keil compiler.
- A page opens with different options in it showing the project workspace at the leftmost corner side, output window in the bottom and an ash coloured space for the program to be written.
- Now to start using the keil, click on the option "project".
- A small window opens showing the options like new project, import project, open project etc. Click on "New project".
- A small window with the title bar "Create new project" opens. The window asks the user to give the project name with which it should be created and the destination location. The

project can be created in any of the drives available. You can create a new folder and then a new file or can create directly a new file.

- After the file is saved in the given destination location, a window opens where a list of vendors will be displayed and you have to select the device for the target you have created.
- The most widely used vendor is Atmel. So click on Atmel and now the family of microcontrollers manufactured by Atmel opens. You can select any one of the microcontrollers according to the requirement.
- When you click on any one of the microcontrollers, the features of that particular microcontroller will be displayed on the right side of the page. The most appropriate microcontroller with which most of the projects can be implemented is the AT89C51. Click on this microcontroller and have a look at its features. Now click on “OK” to select this microcontroller.
- A small window opens asking whether to copy the startup code into the file you have created just now. Just click on “No” to proceed further.
- Now you can see the TARGET and SOURCE GROUP created in the project workspace.
- Now click on “File” and in that “New”. A new page opens and you can start writing program in it.
- After the program is completed, save it with any name but with the .asm extension. Save the program in the file you have created earlier.
- You can notice that after you save the program, the predefined keywords will be highlighted in bold letters.
- Now add this file to the target by giving a right click on the source group. A list of options open and in that select “Add files to the source group”. Check for this file where you have saved and add it.

- Right click on the target and select the first option “Options for target”. A window opens with different options like device, target, output etc. First click on “target”.

- Since the set frequency of the microcontroller is 11.0592 MHz to interface with the PC, just enter this frequency value in the Xtal (MHz) text area and put a tick on the Use on-chip ROM. This is because the program what we write here in the keil will later be dumped into the microcontroller and will be stored in the inbuilt ROM in the microcontroller.

- Now click the option “Output” and give any name to the hex file to be created in the “Name of executable” text area and put a tick to the “Create HEX file” option present in the same window. The hex file can be created in any of the drives. You can change the folder by clicking on “Select folder for Objects”.

- Now to check whether the program you have written is errorless or not, click on the icon exactly below the “Open file” icon which is nothing but Build Target icon. You can even use the shortcut key F7 to compile the program written.

- To check for the output, there are several windows like serial window, memory window, project window etc. Depending on the program you have written, select the appropriate window to see the output by entering into debug mode.

- The icon with the letter “d” indicates the debug mode.

- Click on this icon and now click on the option “View” and select the appropriate window to check for the output.

- After this is done, click the icon “debug” again to come out of the debug mode.

- The hex file created as shown earlier will be burned into the microcontroller ROM with the help of software called µFLASH.

5.1.2 Steps

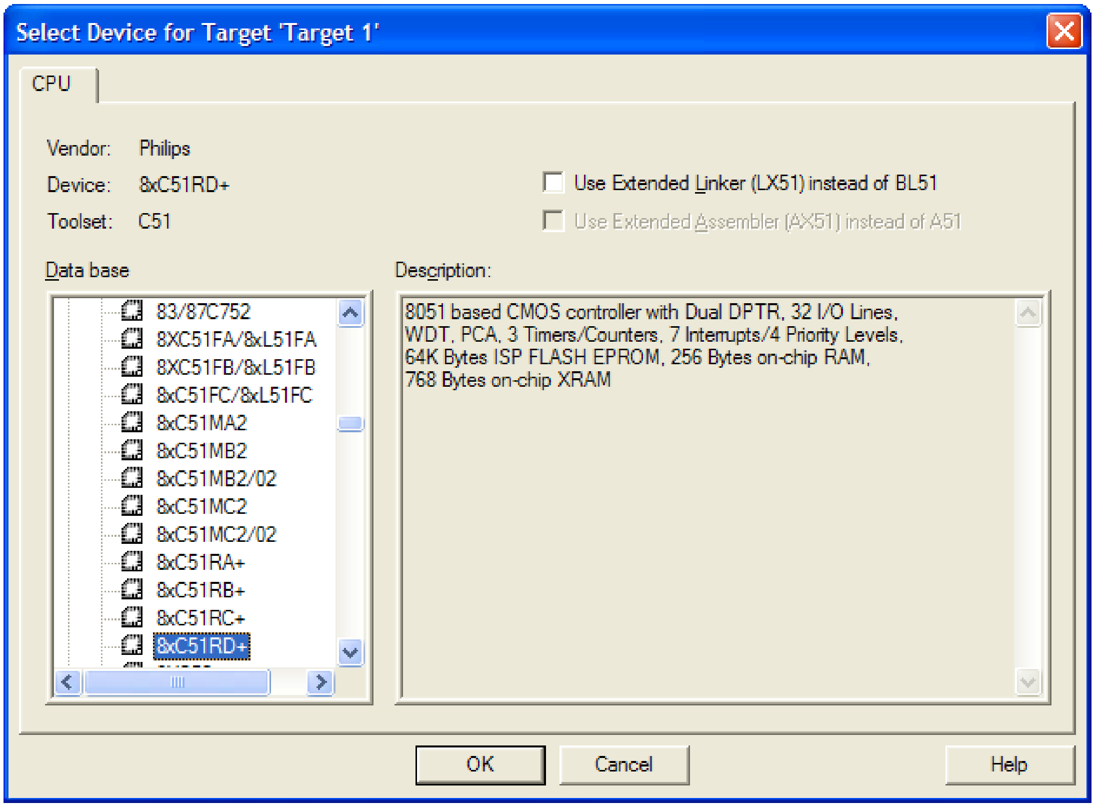
Create a Project File

To create a new project file select from the µVision menu Project – New Project…. This opens a standard Windows dialog that asks you for the new project file name.

We suggest that you use a separate folder for each project. You can simply use the icon Create New Folder in this dialog to get a new empty folder. Then select this folder and enter the file name for the new project, i.e. Project1. µVision creates a new project file with the name PROJECT1.UV2 which contains a default target and file group name. You can see these names in the Project Workspace – Files.

Select a Device

When you create a new project µVision asks you to select a CPU for your project. The Select Device dialog box shows the µVision device database. Just select the microcontroller you use. We are using for our examples the Philips 80C51RD+ controller. This selection sets necessary tool options for the 80C51RD+ device and simplifies in this way the tool configuration.



Select Device for Target 'Target 1'

Vendor: Philips

Device: 8xC51RD+

Toolset: C51

☐ Use Extended Linker (LX51) instead of BL51

☐ Use Extended Assembler (AX51) instead of A51

Description:

8051 based CMOS controller with Dual DPTR, 32 I/O Lines, WDT, PCA, 3 Timers/Counters, 7 Interrupts/4 Priority Levels, 64K Bytes ISP FLASH EPROM, 256 Bytes on-chip RAM, 768 Bytes on-chip XRAM

Fig: 5.1 select device

Once you have selected a CPU from the device database you can open the user manuals for that device in the Project Workspace – Books page. These user manuals are part of the Keil Development Tools CD-ROM that should be present in your CD drive.

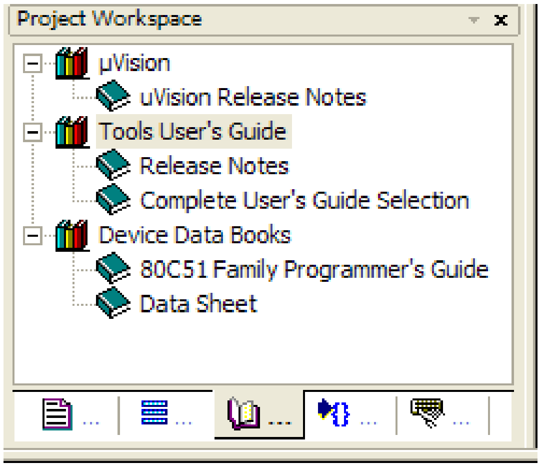
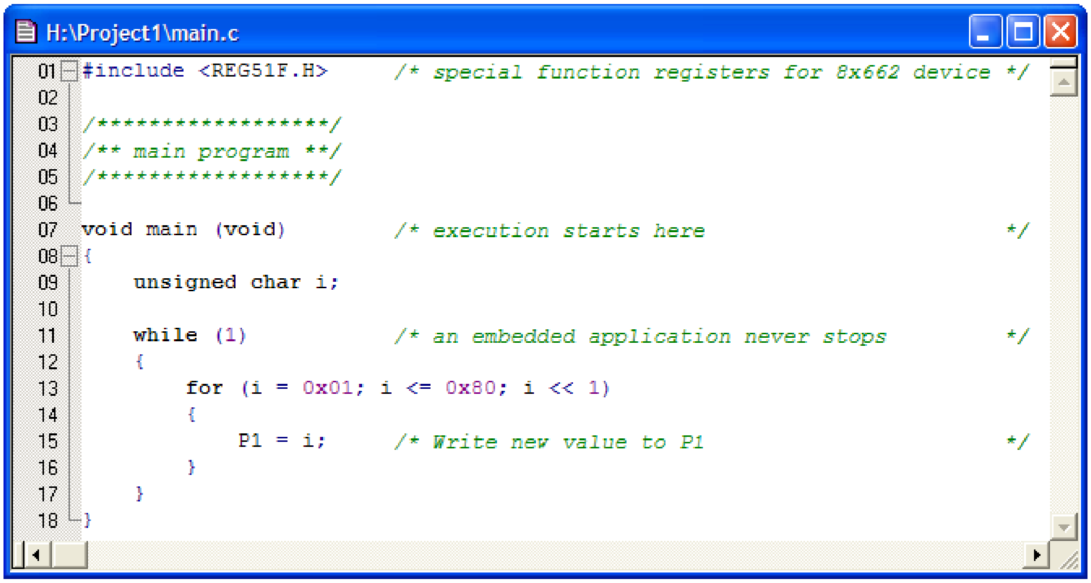


Fig: 5.2 select cd drive

Create New Source Files

You may create a new source file with the menu option File – New. This opens an empty editor window where you can enter your source code. µVision enables the C color syntax highlighting when you save your file with the dialog File – Save As… under a filename with the extension *.C. We are saving our example file under the name MAIN.C.



```
#include <REG51F.H>     /* special function registers for 8x662 device */

/******************/
/** main program **/
/******************/

void main (void)        /* execution starts here                        */
{
    unsigned char i;

    while (1)           /* an embedded application never stops          */
    {
        for (i = 0x01; i <= 0x80; i << 1)
        {
            P1 = i;     /* Write new value to P1                        */
        }
    }
}
```

Fig: 5.3 Create New Source Files

Add and Configure the Startup Code

The STARTUP.A51 file is the startup code for the most 8051 CPU variants. The startup code clears the data memory and initializes hardware and reentrant stack pointers. In addition, some 8051 derivatives require a CPU initialization code that needs to match the configuration of your hardware design. For example, the Philips 8051RD+ offers you on-chip xdata RAM that should be enabled in the startup code. Since you need to modify that file to match your target hardware, you should copy the STARTUP.A51 file from the folder C:\KEIL\C51\LIB to your project folder.

Group Project Files

File group allow you to organize large projects. For the CPU startup code and other system configuration files you may create a own file group in the Project – Components, Environment, Books… dialog box. Use the New (Insert) button to create a file group named System Files. In the project window you may drag and drop the STARTUP.A51 file to this new file group.

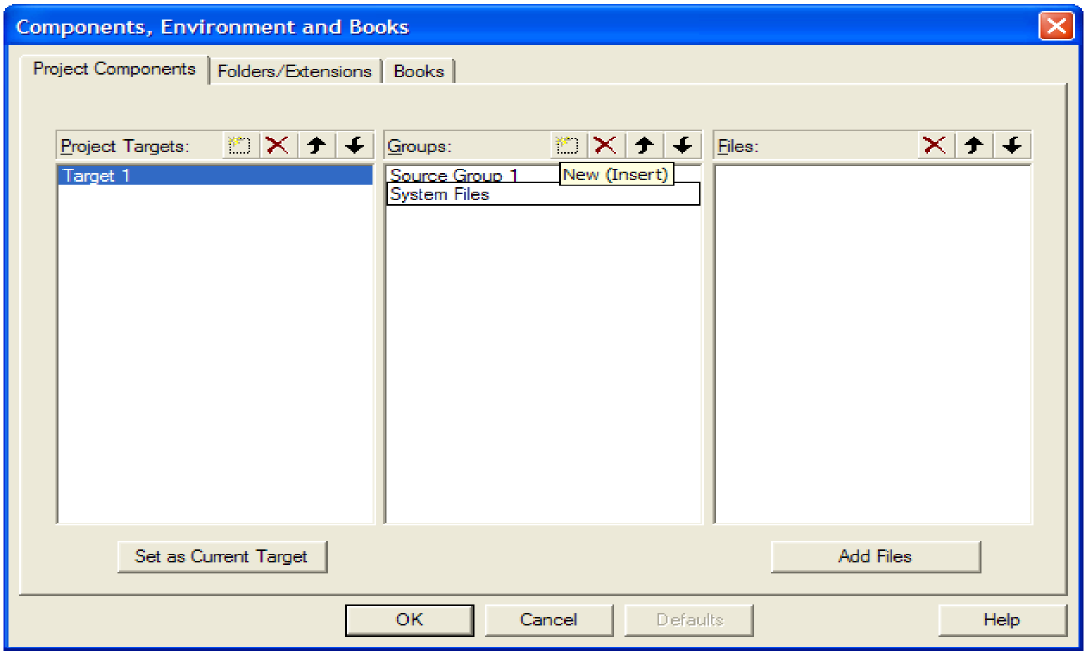
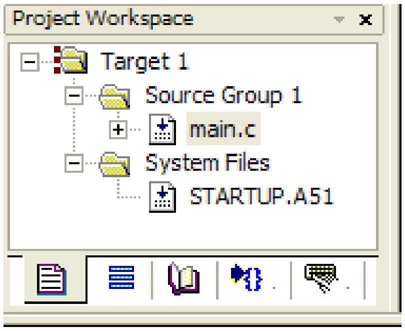


Fig: 5.4 Group Project Files

Now, the Project Workspace – Files lists all items of your project. To open a file for editing, double click on the file name in the Project Workspace. You may need to configure the startup STARTUP.A51 in the editor.

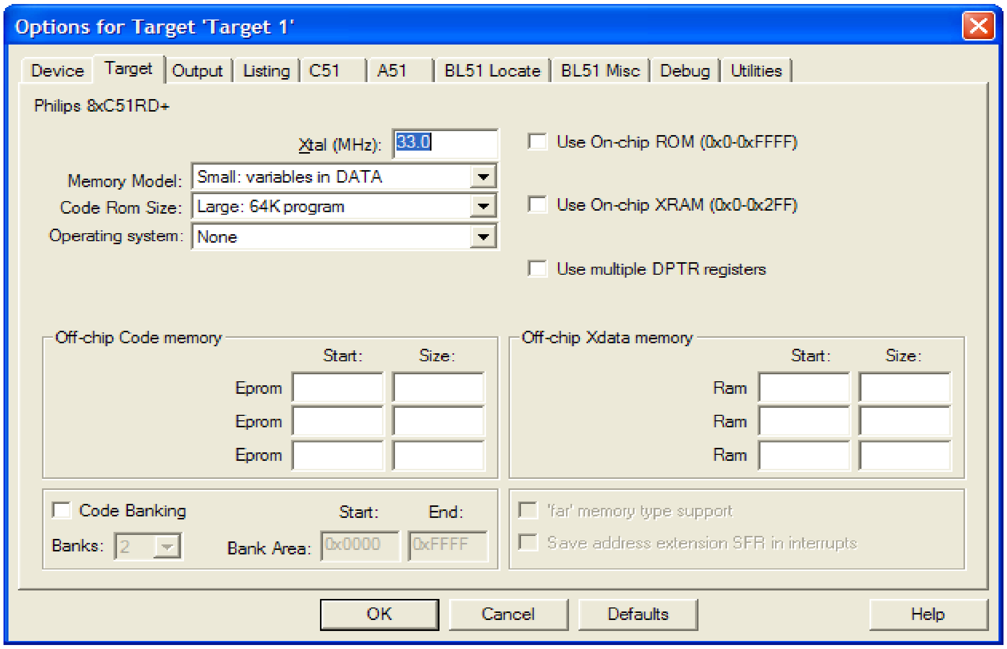


Project Workspace

- Target 1
  - Source Group 1
    - main.c
  - System Files
    - STARTUP.A51

Set Tool Options for Target

µVision lets you set options for your target hardware. The dialog Options for Target opens via the toolbar icon or via the Project - Options for Target menu item. In the Target tab you specify all relevant parameters of your target hardware and the on-chip components of the device you have selected. The following the settings for our example are shown.



Options for Target 'Target 1'

Philips 8xC51RD+

Xtal (MHz): 33.0

Memory Model: Small: variables in DATA

Code Rom Size: Large: 64K program

Operating system: None

□ Use On-chip ROM (0x0-0xFFFF)

□ Use On-chip XRAM (0x0-0x2FF)

□ Use multiple DPTR registers

Off-chip Code memory

	Start:	Size:
Eprom		
Eprom		
Eprom		

Off-chip Xdata memory

	Start:	Size:
Ram		
Ram		
Ram		

□ Code Banking Start: End:

Banks: 2 Bank Area: 0x0000 0xFFFF

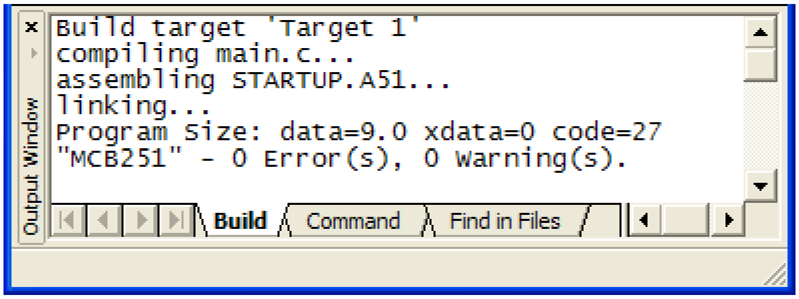
□ 'far' memory type support

□ Save address extension SFR in interrupts

Fig: 5.5 Set Tool Options for Target

Build Project and Create a HEX File

Typical, the tool settings under Options – Target are all you need to start a new application. You may translate all source files and line the application with a click on the Build Target toolbar icon. When you build an application with syntax errors, µVision will display errors and warning messages in the Output Window – Build page. A double click on a message line opens the source file on the correct location in a µVision editor window.



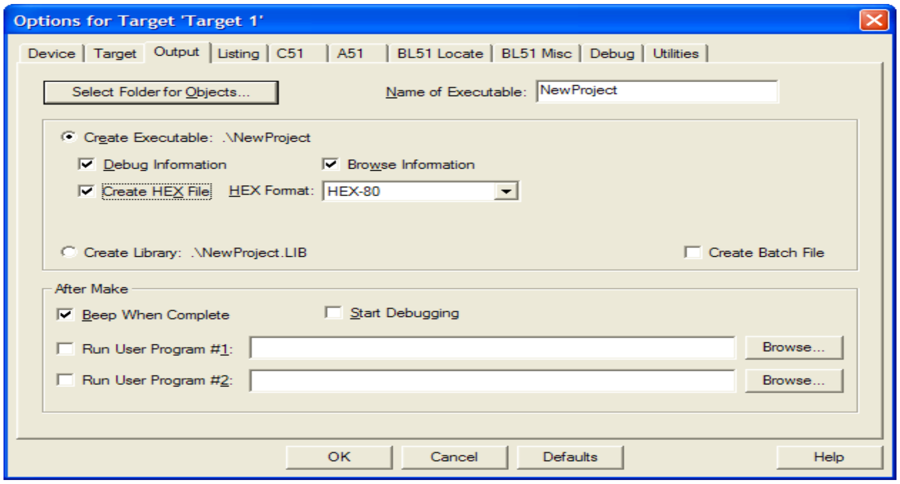
```
Build target 'Target 1'
compiling main.c...
assembling STARTUP.A51...
linking...
Program Size: data=9.0 xdata=0 code=27
"MCB251" - 0 Error(s), 0 Warning(s).
```

Fig: 5.6(a) Build Project and Create a HEX File

Once you have successfully generated your application you can start debugging as described under Testing Programs with the µVision Debugger.

Now you may modify existing source code or add new source files to the project. The Build Target toolbar button translates only modified or new source files and generates the executable file. µVision maintains a file dependency list and knows all include files used within a source file. Even the tool options are saved in the file dependency list, so that µVision rebuilds files only when needed. With the Rebuild Target command, all source files are translated, regardless of modifications.

After you have tested your application, it might be required to create an Intel HEX file and to download the application software into the physical device using a Flash programming utility. µVision creates HEX files with each build process when Create HEX files under Options for Target – Output is enabled. The Merge32K Hex file option is available for Code Banking Applications when you have selected the Extended Linker LX51. You may start your Flash programming utility after the make process when you specify the program under the option Run User Program #1.


Options for Target 'Target 1'
Device | Target | Output | Listing | C51 | A51 | BL51 Locate | BL51 Misc | Debug | Utilities
Select Folder for Objects...
After Make
Browse...
Browse...
OK
Cancel
Defaults
Help

Name of Executable: NewProject

● Create Executable: .\NewProject

☑ Debug Information ☑ Browse Information

☑ Create HEX File HEX Format: HEX-80

○ Create Library: .\NewProject.LIB □ Create Batch File

☑ Beep When Complete □ Start Debugging

□ Run User Program #1:

□ Run User Program #2:

FIG: 5.6(b) BUILD PROJECT AND CREATE A HEX FILE

CHAPTER-6

ADVANTAGES AND APPLICATIONS

6.1 ADVANTAGES:

- Low cost
- To help the passengers to cross the road safely
- By using this system we can control the speed
- This system help to avoiding the rash driving of the driver
- The vehicle will move only with the designed speed corresponding to the zone

6.2 APPLICATIONS:

- Used in industrial areas
- Used in hospitals
- Used in school zones
- Near parks
- Public areas

CHAPTER-7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

In this paper we developed a new design to control the speed of the automobiles. In normal driving mode, we can expect other vehicles interfering nearby and possibly blocking or attenuating RF signals. In this aspect, we are going to use gps location for restricted areas. REF

7.2 FUTURE SCOPE

- We can modify the system with the help of gps to identify the zone
- We can also modify the system with efficient breaking system in association with air flow control to the carburetor
- This system can be more efficiency used for any kind of automobiles such as buses,cars,lorries,bikes.

REFERENCES

1] Ankita Mishra, Jyoti Solanki "Design of RF based speed control system for vehicles," International Journal of Advanced Research in Computer and Communication Engineering, Vol. 1, No 8, 2012.

[2] Vinod Rao, Saketh Kuma, "Smart Zone Based Vehicle Speed Control Using RF and Obstacle Detection and Accident Prevention," International Journal of Emerging Technology and Advanced Engineering, Vol.4, No.3, 2014.

[3] Gummarekula Sattibabu, Satyanarayan , "Automatic Vehicle Speed Control With Wireless In-Vehicle Road Sign Delivery System Using ARM 7," International Journal Of Technology Enhancements And Emerging Engineering Research, Vol 2, No. 8, 2014

. [4] Deepa B Chavan, Abdul Rahim Makandar , "Automatic Vehicle Speed Reduction System Using Rf Technology," International Journal of Engineering Research and Applications, Vol.4, No.4, 2014.

[5] Jyothi Kameswari, Satwik , "A Design Model for Automatic Vehicle Speed Controller," International Journal of Computer Applications, Vol.35, No.9, 2011.

BIBLIOGRAPHY:

- Kenneth .J. Ayala, The 8051 Microcontroller and its applications, prenticehall, new Edition, 2006.
- Frank Vahid, Embedded system design, Tata McGraw hill, 3 Edition, 1995.
- Raj Kamal, Embedded Systems, JWE, 4 Editions, 2000.
- Jonathan Clark, Applications of Ultrasonic, Tata McGraw hill, new Edition, 2002.

WEB SITES

www.books.google.com

www.howstuffworks.com

www.epanorama.net

www.wikipedia.org