**RAILWAY LEVEL CROSSING GATE CONTROL BY ANDROID APPLICATION**

|  |  |  |
| --- | --- | --- |
| **INDEX** | | |
| **Chapter No.** | **TOPIC** | **Page No.** |
| **1**  1.1  1.2  1.3  1.4  1.5 | **INTRODUCTION**  **General**  Existing System  Disadvantage Of Existing System  Proposed System  Advantage Of Proposed System |  |
| **2**  2.1  2.2  2.3  2.3.1  2.3.2  2.3.3  2.4  2.4.1  2.4.2  2.4.3  2.4.4  2.4.5  2.4.6  2.4.7  2.4.8  2.5  2.5.1  2.5.2  2.5.3    2.6  2.6.1  2.6.2  2.6.3  2.6.4  2.6.5  2.6.6  2.7  2.7.1  2.7.2  2.7.3  2.7.4  2.8  2.8.1  2.9  2.9.1 | **PROJECT DESCRIPTION**  General Introduction To Embedded System  Block Diagram  Modules  **Power Supply**  Transformer  Rectifier  Voltage Regulator    **MICROCONTROLLER**  Microcontroller in embedded system  Basic structure of a microcontroller  i/o subsystem  Registers  Memory  Features  Pin Description  Ports  **LIQUID CRYSTAL DISPLAY(LCD)**  Introduction  Pin Diagram  List Of Commands  **BLUETOOTH**  Introduction  Specification  Difference  Communication  Bluetooth Host  Power Management  **MOTOR DRIVER**  Introduction  Description  Pin Diagram  Working    **MOTOR**  Uses  **BUZZER**  Uses |  |
| **3**  3.1  3.2  3.3 | **SOFTWARE SPECIFICATIONS**  **KEIL SOFTWARE**  Introduction  µVision3 Overview  STEPS |  |
| **4**  4.1 | **IMPLEMENTATION**  Schematic Design |  |
| **5**  5.1 | **SIMULATION AND DESIGN**  Output Screen Shots |  |
| **6** | **CONCLUSION** |  |
| **7** | **REFERENCES** |  |

**CHAPTER 1**

**INTRODUCTION**

**1.1GENERAL**

In everywhere at level crossing between railroad and highway there are many railway accidents happening due to the laziness and carelessness in manual operations or lack of workers. So, this paper describes the automatic railway gate control system using android app and Bluetooth module for saving human lives and avoiding major disasters in railway track. Railway gate may be saved for the road users to prevent accidents in terms of train at level crossing. This system uses Bluetooth module with the help of microcontroller. The microcontroller forms the main unit of the system. It receives input signal from the APP and sends information to the gate motor driver for opening and closing the gate. Besides, the input signal will active LCD display which shows the status of the railway gate was provided as additional part of this system. The gate is closed, when the train enters in the specified range and the gate is opened, when the train gone to the specified range. This system deals about one of the efficient methods to avoid train accidents. It is based on software programming to operate the hardware structure.

* 1. **EXISTING SYSTEM:**

The opening and closing of railway gate is traditionally operated through manual lever pulling method. This method leads to a lot of accidents due to the rational technique and lever jamming. Railway is a lifeline of India and it is being the cheapest modes of transportation are prepared over all other means of transportation. When we go through the daily newspapers we come across many accidents in rail road railings. Rail road related accidents are more dangerous than other transportation accidents in terms of severity and death rate etc. Therefore more effort are necessary for improving safety.

* 1. **PROPOSED SYSTEM**

In this project a smartphone mobile is used which contain Android app connect to railway. This Bluetooth module sends data to microcontroller. Due to APP, Now this signal goes to motor driver IC LM293D which has two input pins and two output pins and four ground pins. It require 12V supply. It drives the signal in specific direction. This signal goes to DC motor . DC Motor is used to open or close the gate. It works according to microcontroller. Whenever pin is set gate remains close and when pin reset becomes active, gate remains open.SO, Firstly gate remains open at railway gate as there is no any train passes from railway gate. Controller access signal from module and follows condition of set according to incoming signal and passes to DC motor through motor driver. Then DC motor close the gate .In this way DC motor rotates in clockwise and anticlockwise direction to open or close the gate. Whenever gate open or close , the status of gate candisplay on LCD and Alert through Buzzer.

**CHAPTER 2**

**PROJECT DESCRIPTION**

**2.1 GENERAL INTRODUCTION TO EMBEDDED SYSTEM**

Embedded systems are designed to do some specific task rather than be a general purpose computer for multiple tasks. Some also have real time performance constraints that must met, for reason such as safety and usability; others may have low or no performance requirements, allowing the system hardware to be simplified to reduce costs.

An embedded system is not always a separate block very often it is physically built in to the device it is controlling. The software written for embedded systems is often called firmware, and is stored in read only memory or flash convector chips rather than a disk drive. It often runs with limited computer hardware resources: small or no keyboard, screen and little memory.

To perform any application in the embedded system we require microprocessor and microcontroller. In the microprocessor an external memory is connected which increases the size of the microprocessor and multiple operations are being performed by the microprocessor but whereas in the microprocessor the memory is inbuilt and also we can use this controller only for the specific applications where the speed is increased so most probably microcontrollers are used in the different applications in the embedded systems rather than microprocessor.

An embedded system can be defined as a computing device that does a specific focused job. Appliances such as the air-conditioner, VCD player, DVD player, printer, fax machine, mobile phone etc. are examples of embedded systems. Each of these appliances will have a processor and special hardware to meet the specific requirement of the application along with the embedded software that is executed by the processor for meeting that specific requirement. The embedded software is also called “firm ware”. The desktop/laptop computer is a general purpose computer. You can use it for a variety of applications such as playing games, *word* processing, accounting, software development and so on. In contrast, the software in the embedded systems is always fixed listed below:

· Embedded systems do a very specific task; they cannot be programmed to do different things. . Embedded systems have very limited resources, particularly the memory. Generally, they do not have secondary storage devices such as the CDROM or the floppy disk. Embedded systems have to work against some deadlines. A specific job has to be completed within a specific time. In some embedded systems, called real-time systems, the deadlines are stringent. Missing a deadline may cause a catastrophe-loss of life or damage to property. Embedded systems are constrained for power. As many embedded systems operate through a battery, the power consumption has to be very low.

· Some embedded systems have to operate in extreme environmental conditions such as very high temperatures and humidity.

**Application Areas**

Nearly 99 per cent of the processors manufactured end up in embedded systems. The embedded system market is one of the highest growth areas as these systems are used in very market segment- consumer electronics, office automation, industrial automation, biomedical engineering, wireless communication, data communication, telecommunications, transportation, military and so on.

**Consumer appliances**:

At home we use a number of embedded systems which include digital camera, digital diary, DVD player, electronic toys, microwave oven, remote controls for TV and air-conditioner, VCO player, video game consoles, video recorders etc. Today’s high-tech car has about 20 embedded systems for transmission control, engine spark control, air-conditioning, navigation etc. Even wristwatches are now becoming embedded systems. The palmtops are powerful embedded systems using which we can carry out many general-purpose tasks such as playing games and word processing.

**Office Automation:**

The office automation products using embedded systems are copying machine, fax machine, key telephone, modem, printer, scanner etc.

**Industrial Automation**:

Today a lot of industries use embedded systems for process control. These include pharmaceutical, cement, sugar, oil exploration, nuclear energy, electricity generation and transmission. The embedded systems for industrial use are designed to carry out specific tasks such as monitoring the temperature, pressure, humidity, voltage, current etc., and then take appropriate action based on the monitored levels to control other devices or to send information to a centralized monitoring station. In hazardous industrial environment, where human presence has to be avoided, robots are used, which are programmed to do specific jobs. The robots are now becoming very powerful and carry out many interesting and complicated tasks such as hardware assembly.

**Medical Electronics**:

Almost every medical equipment in the hospital is an embedded system. This equipment include diagnostic aids such as ECG, EEG, blood pressure measuring devices, X-ray scanners; equipment used in blood analysis, radiation, colonoscopy, endoscopy etc. Developments in medical electronics have paved way for more accurate diagnosis of diseases.

**Computer Networking**:

Computer networking products such as bridges, routers, Integrated Services Digital Networks (ISDN), Asynchronous Transfer Mode (ATM), X.25 and frame relay switches are embedded systems which implement the necessary data communication protocols. For example, a router interconnects two networks. The two networks may be running different protocol stacks. The router’s function is to obtain the data packets from incoming pores, analyze the packets and send them towards the destination after doing necessary protocol conversion. Most networking equipment’s, other than the end systems (desktop computers) we use to access the networks, are embedded systems.

**Telecommunications**:

In the field of telecommunications, the embedded systems can be categorized as subscriber terminals and network equipment. The subscriber terminals such as key telephones, ISDN phones, terminal adapters, web cameras are embedded systems. The network equipment includes multiplexers, multiple access systems, Packet Assemblers Dissemblers (PADs), sate11ite modems etc. IP phone, IP gateway, IP gatekeeper etc. are the latest embedded systems that provide very low-cost voice communication over the Internet.

**Wireless Technologies**:

Advances in mobile communications are paving way for many interesting applications using embedded systems. The mobile phone is one of the marvels of the last decade of the 20’h century. It is a very powerful embedded system that provides voice communication while we are on the move. The Personal Digital Assistants and the palmtops can now be used to access multimedia service over the Internet. Mobile communication infrastructure such as base station controllers, mobile switching centers are also powerful embedded systems.

**Insemination:**

Testing and measurement are the fundamental requirements in all scientific and engineering activities. The measuring equipment we use in laboratories to measure parameters such as weight, temperature, pressure, humidity, voltage, current etc. are all embedded systems. Test equipment such as oscilloscope, spectrum analyzer, logic analyzer, protocol analyzer, radio communication test set etc. are embedded systems built around powerful processors. Thank to miniaturization, the test and measuring equipment are now becoming portable facilitating easy testing and measurement in the field by field-personnel.

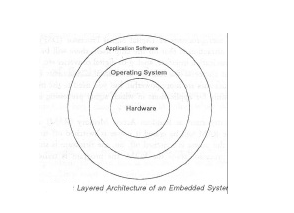
**Security:**

Security of persons and information has always been a major issue. We need to protect our homes and offices; and also the information we transmit and store. Developing embedded systems for security applications is one of the most lucrative businesses nowadays. Security devices at homes, offices, airports etc. for authentication and verification are embedded systems. Encryption devices are nearly 99 per cent of the processors that are manufactured end up in~ embedded systems. Embedded systems find applications in every industrial segment- consumer electronics, transportation, avionics, biomedical engineering, manufacturing, process control and industrial automation, data communication, telecommunication, defense, security etc. Used to encrypt the data/voice being transmitted on communication links such as telephone lines. Biometric systems using fingerprint and face recognition are now being extensively used for user authentication in banking applications as well as for access control in high security buildings.

**Finance:**

Financial dealing through cash and cheques are now slowly paving way for transactions using smart cards and ATM (Automatic Teller Machine, also expanded as Any Time Money) machines. Smart card, of the size of a credit card, has a small micro-controller and memory; and it interacts with the smart card reader! ATM machine and acts as an electronic wallet. Smart card technology has the capability of ushering in a cashless society. Well, the list goes on. It is no exaggeration to say that eyes wherever you go, you can see, or at least feel, the work of an embedded system.

**Overview of Embedded System Architecture**

Every embedded system consists of custom-built hardware built around a Central Processing Unit (CPU). This hardware also contains memory chips onto which the software is loaded. The software residing on the memory chip is also called the ‘firmware’. The embedded system architecture can be represented as a layered architecture as shown in Fig. The operating system runs above the hardware, and the application software runs above the operating system. The same architecture is applicable to any computer including a desktop computer. However, there are significant differences. It is not compulsory to have an operating system in every embedded system. For small appliances such as remote control units, air conditioners, toys etc., there is no need foran operating system and you can write only the software specific to that application. For applications involving complex processing, it is advisable to have an operating system. In such a case, you need to integrate the application software with the operating system and then transfer the entire software on to the memory chip. Once the software is transferred to the memory chip, the software will continue to run fora long time you don’t need to reload new software.

Now, let us see the details of the various building blocks of the hardware of an embedded system. As shown in Fig. the building blocks are;

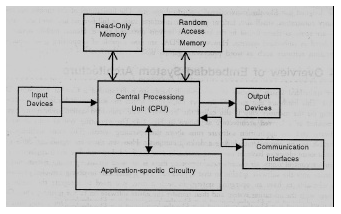
· Central Processing Unit (CPU)

· Memory (Read-only Memory and Random Access Memory)

· Input Devices

· Output devices

· Communication interfaces

· Application-specific circuitry

**Central Processing Unit (CPU):**

The Central Processing Unit (processor, in short) can be any of the following: microcontroller, microprocessor or Digital Signal Processor (DSP). A micro-controller is a low-cost processor. Its main attraction is that on the chip itself, there will be many other components such as memory, serial communication interface, analog-to digital converter etc. So, for small applications, a micro-controller is the best choice as the number of external components required will be very less. On the other hand, microprocessors are more powerful, but you need to use many external components with them. D5P is used mainly for applications in which signal processing is involved such as audio and video processing.

**Memory:**

The memory is categorized as Random Access Memory (RAM) and Read Only Memory (ROM). The contents of the RAM will be erased if power is switched off to the chip, whereas ROM retains the contents even if the power is switched off. So, the firmware is stored in the ROM. When power is switched on, the processor reads the ROM; the program is program is executed.

**Input Devices**:

Unlike the desktops, the input devices to an embedded system have very limited capability. There will be no keyboard or a mouse, and hence interacting with the embedded system is no easy task. Many embedded systems will have a small keypad-you press one key to give a specific command. A keypad may be used to input only the digits. Many embedded systems used in process control do not have any input device *for* user interaction; they take inputs fromsensors or transducers 1’fnd produce electrical signals that are in turn fed to other systems.

**Output Devices**:

The output devices of the embedded systems also have very limited capability. Some embedded systems will have a fewLight Emitting Diodes (LEDs) toindicate the health status of the system modules, or forvisual indication of alarms. A small Liquid Crystal Display (LCD) may also be used to display some important parameters.

**2.2 BLOCK DIAGRAM**

**Micro**

**Controller**

**MOTOR**

**Power Supply**

**Reset**

**L293D**

**LCD**

**BLUETOOTH**

**Crystal**

**Buzzer**

**2.3 MODULES**

**1. POWER SUPPLY**

The power supply section is the section which provide +5V for the components to work. IC LM7805 is used for providing a constant power of +5V.

The ac voltage, typically 220V, is connected to a transformer, which steps down that ac voltage down to the level of the desired dc output. A diode rectifier then provides a full-wave rectified voltage that is initially filtered by a simple capacitor filter to produce a dc voltage. This resulting dc voltage usually has some ripple or ac voltage variation.

A regulator circuit removes the ripples and also retains the same dc value even if the input dc voltage varies, or the load connected to the output dc voltage changes. This voltage regulation is usually obtained using one of the popular voltage regulator IC units.

****

Figure: Block Diagram of Power Supply

**Transformer**

Transformers convert AC electricity from one voltage to another with little loss of power. Transformers work only with AC and this is one of the reasons why mains electricity is AC.

Step-up transformers increase voltage, step-down transformers reduce voltage. Most power supplies use a step-down transformer to reduce the dangerously high mains voltage (230V in India) to a safer low voltage.

The input coil is called the primary and the output coil is called the secondary. There is no electrical connection between the two coils; instead they are linked by an alternating magnetic field created in the soft-iron core of the transformer. Transformers waste very little power so the power out is (almost) equal to the power in. Note that as voltage is stepped down current is stepped up. The transformer will step down the power supply voltage (0-230V) to (0- 6V) level. Then the secondary of the potential transformer will be connected to the bridge rectifier, which is constructed with the help of PN junction diodes. The advantages of using bridge rectifier are it will give peak voltage output as DC.

**Rectifier**

There are several ways of connecting diodes to make a rectifier to convert AC to DC. The bridge rectifier is the most important and it produces full-wave varying DC. A full-wave rectifier can also be made from just two diodes if a center-tap transformer is used, but this method is rarely used now that diodes are cheaper. A single diode can be used as a rectifier but it only uses the positive (+) parts of the AC wave to produce half-wave varying DC

**Bridge Rectifier**

When four diodes are connected as shown in figure, the circuit is called as bridge rectifier. The input to the circuit is applied to the diagonally opposite corners of the network, and the output is taken from the remaining two corners. Let us assume that the transformer is working properly and there is a positive potential, at point A and a negative potential at point B. the positive potential at point A will forward bias D3 and reverse bias D4.



Figure: Bridge Rectifier

The negative potential at point B will forward bias D1 and reverse D2. At this time D3 and D1 are forward biased and will allow current flow to pass through them; D4 and D2 are reverse biased and will block current flow.

One advantage of a bridge rectifier over a conventional full-wave rectifier is that with a given transformer the bridge rectifier produces a voltage output that is nearly twice that of the conventional full-wave circuit.

i. The main advantage of this bridge circuit is that it does not require a special centre tapped transformer, thereby reducing its size and cost.

ii. The single secondary winding is connected to one side of the diode bridge network and the load to the other side as shown below.

iii. The result is still a pulsating direct current but with double the frequency.

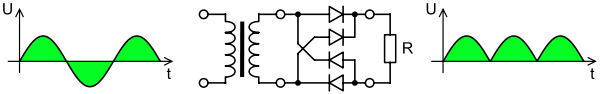


Figure: Output Waveform of DC

**Smoothing**

Smoothing is performed by a large value electrolytic capacitor connected across the DC supply to act as a reservoir, supplying current to the output when the varying DC voltage from the rectifier is falling. The capacitor charges quickly near the peak of the varying DC, and then discharges as it supplies current to the output.

**Voltage Regulators**

Voltage regulators comprise a class of widely used ICs. Regulator IC units contain the circuitry for reference source, comparator amplifier, control device, and overload protection all in a single IC. IC units provide regulation of either a fixed positive voltage, a fixed negative voltage, or an adjustable set voltage. The regulators can be selected for operation with load currents from hundreds of milli amperes to tens of amperes, corresponding to power ratings from milli watts to Tens of watts.

A fixed three-terminal voltage regulator has an unregulated dc input voltage, Vi, applied to one input terminal, a regulated dc output voltage, Vo, from a second terminal, with the third terminal connected to ground. The series 78 regulators provide fixed positive regulated voltages from 5 to 24 volts. Similarly, the series 79 regulators provide fixed negative regulated voltages from 5 to 24 volts. Voltage regulator ICs are available with fixed (typically 5, 12 and 15V) or variable output voltages. They are also rated by the maximum current they can pass. Negative voltage regulators are available, mainly for use in dual supplies. Most regulators include some automatic protection from excessive current ('overload protection') and overheating ('thermal protection').

Many of the fixed voltage regulator ICs has 3 leads and look like power transistors, such as the 7805 +5V 1Amp regulator. They include a hole for attaching a heat sink if necessary.

****

Figure: Regulator

****

Figure: Circuit Diagram of Power Supply

**2.4. MICROCONTROLLER**

A Microcontroller (or MCU) is a [computer](file:///D:\\wiki\\Computer)-on-a-[chip](file:///D:\\wiki\\Integrated_circuit) used to control [electronic](file:///D:\\wiki\\Electronics) [devices](file:///D:\\wiki\\Devices). It is a type of [microprocessor](file:///D:\\wiki\\Microprocessor) emphasizing self-sufficiency and cost-effectiveness, in contrast to a general-purpose microprocessor (the kind used in a [PC](file:///D:\\wiki\\Personal_computer)). A typical microcontroller contains all the [memory](file:///D:\\wiki\\Memory) and [interfaces](file:///D:\\wiki\\Interface_(computer_science)) needed for a simple application, whereas a general purpose microprocessor requires additional chips to provide these functions.

A microcontroller is a single [integrated circuit](file:///D:\\wiki\\Integrated_circuit) with the following key features:

1. [central processing unit](file:///D:\\wiki\\Central_processing_unit) - ranging from small and simple 8-[bit](file:///D:\\wiki\\Bit) processors to sophisticated 32- or 64-bit processors
2. [input/output](file:///D:\\wiki\\Input\\output) [interfaces](file:///D:\\wiki\\Network_interface) such as [serial ports](file:///D:\\wiki\\Serial_port)
3. [RAM](file:///D:\\wiki\\RAM) for data storage
4. [ROM](file:///D:\\wiki\\Read-only_Memory), [EEPROM](file:///D:\\wiki\\EEPROM) or [Flash memory](file:///D:\\wiki\\Flash_memory) for [program](file:///D:\\wiki\\Computer_program) storage
5. [clock generator](file:///D:\\wiki\\Clock_generator) - often an oscillator for a quartz timing crystal, resonator or [RC](file:///D:\\wiki\\RC_circuit) circuit

Microcontrollers are inside many kinds of [electronic equipment](file:///D:\\wiki\\Electronic_equipment) (see [embedded system](file:///D:\\wiki\\Embedded_system)). They are the vast majority of all processor chips sold. Over 50% are "simple" controllers, and another 20% are more specialized [digital signal processors (DSPs)](file:///D:\\wiki\\Digital_signal_processor) (ref?). A typical home in a [developed country](file:///D:\\wiki\\Developed_country) is likely to have only one or two general-purpose microprocessors but somewhere between one and two dozen microcontrollers. A typical mid range vehicle has as many as 50 or more microcontrollers. They can also be found in almost any [electrical](file:///D:\\wiki\\Electrical) [device](file:///D:\\wiki\\Tool): [washing machines](file:///D:\\wiki\\Washing_machine), [microwave ovens](file:///D:\\wiki\\Microwave_oven), [telephones](file:///D:\\wiki\\Telephone) etc.

**FEATURES**

• Compatible with MCU®-51 Products

• 8K Bytes of In-System Programmable (ISP) Flash Memory

– Endurance: 10,000 Write/Erase Cycles

• 4.0V to 5.5V Operating Range

• Fully Static Operation: 0 Hz to 33 MHz

• 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

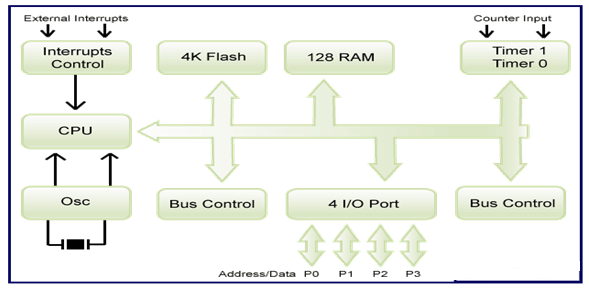
• Fast Programming Time

• Flexible ISP Programming (Byte and Page Mode)

**Description**

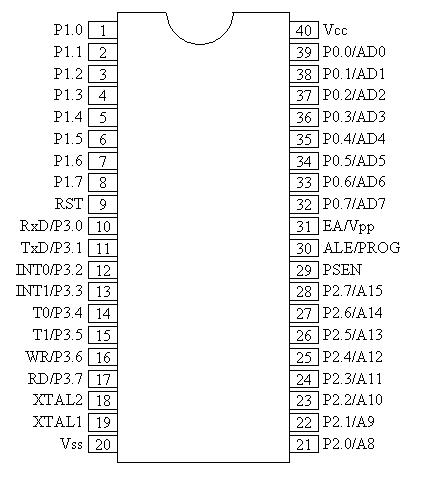
The AT89S52 is a low-power, high-performance CMOS 8-bit microcontroller with 8K bytes of in-system programmable Flash memory. The device is manufactured using Atmel’s high-density nonvolatile memory technology and is compatible with the industry-standard 80C51 instruction set and pin-out. The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with in-system programmable Flash on a monolithic chip, the Atmel AT89S52 is a powerful microcontroller which provides a highly-flexible and cost-effective solution to many embedded control applications.

The AT89S52 provides the following standard features: 8K bytes of Flash, 256 bytes of RAM, 32 I/O lines, three 16-bit timer/counters, a six-vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator, and clock circuitry. 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 con-tents but freezes the oscillator, disabling all other chip functions until the next interrupt or hardware reset.



**Fig. BLOCK DIAGRAM OF 8052**

**PIN DIAGRAM**



**Fig. PIN DIAGRAM OF 8052**

**PIN DESCRIPTION**

40 pin VCC Supply voltage.

20 pin GND 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. 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.

**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 because of the internal pull-ups. In this application, Port 2 uses strong internal pull-ups when emitting 1s. During accesses to external data memory that use 8-bit addresses, Port 2 emits the contents of the P2 Special Function Register. Port 2 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 because of the pull-ups. Port 3 also serves the functions of various special features of the AT89S52, as shown in the following table.



**RST**

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device.

**WR**

It is active low write O/P control signal. During External RAM (Data memory). Write to external RAM.

**RD**

It is active low read O/P control signal. During External RAM (Data memory). Read from External RAM.

**XTAL1** Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

**XTAL2** Output from the inverting oscillator amplifier.

**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. 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.

**EA**

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.

**ARCHITECTURE OF 8052**



**Fig. INTERNAL ARCHITECTURE OF 8052**

**REGISTERS**

Registers and accumulators serve as temporary memory locations during CPU operations. The exchange of information among them takes place through one or more internal buses. The length of each register is equal to the width of the internal data bus.

Registers are categorized as general-purpose registers and special-purpose registers.

A general-purpose register may be used as an accumulator or as a data register for (arithmetic) and logic operations. An accumulator is a register for storing the results of an arithmetic operation. A general-purpose register may also be used as an address register.

A special-purpose register is dedicated to a specific function. Some of these registers are the program counter (PC), the instruction register (IR), the address register (AR), the status register, and the stack pointer (SP).

The program counter holds the address of the next instruction to be executed. When the CPU executes a branch instruction telling it to branch to another part of the program, the new address is loaded into the program counter and the sequential order resumes.

The instruction register extracts the operation code (op code) from an instruction. An instruction consists of op code and operand(s). The contents of the instruction register are decoded by the control unit. The status register or condition code register consists of status or flag bits and control bits.

The flag bits are: carry/borrow (C), overflow (V), negative (N), zero (Z), and half-carry (H). The logic states of the flag bits are used by the CPU’s branch instructions to make decisions. The program counter sends its contents (address) to the address register. The address register then sends this address to the address bus.

A stack is a specially reserved area in memory where information is stored or removed in a last-in-first-out (LIFO) fashion. The stack pointer points to the next free location on the stack. That is, it holds the address of the top of the stack. Each time data is stored in the stack, the stack pointer is automatically decremented and each time data is retrieved from the stack, the stack pointer is automatically incremented.

**Memory Organization**

MCU-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.

The instructions to be executed by the microcontroller CPU and the data to be operated on by these instructions are stored in memory. When the CPU accesses the information stored in memory, it is performing a read operation. When the CPU sends information to be stored in the memory, it is performing a write operation. Memory is classified as internal and external memory. Internal memory is on-chip memory and is a semiconductor type with low capacity and high speeds. External memory is outside the chip and includes the semiconductor type and serial memory such as magnetic disks, magnetic tapes, and bubble memory. Semiconductor memory may be volatile or nonvolatile. Volatile memory loses its contents after the power is removed from the memory chip. Nonvolatile memory does not lose its contents when power is removed. The nonvolatile memory can store information permanently or at least semi-permanently (ten years or more). Volatile memory includes RAM while nonvolatile memory includes ROM, EPROM, EEPROM, and battery-backed RAM.

RAM (random access memory) comprises DRAM (Dynamic RAM) and SRAM (Static RAM). Each storage cell of DRAM consists of a capacitor and a MOSFET (Metal Oxide Semi conductor Field Effect Transistor). If the capacitor is charged, a logic 1 is stored in the cell. If the capacitor has no charge, a logic 0 is stored in the cell. The charge stored in the capacitors dissipates fast because of leakage. Therefore, a DRAM has to be periodically refreshed otherwise its contents will fade away after some time even in the presence of the power to memory chip.

The storage cell of a ROM (Read-only memory) consists of a BJT (Bipolar Junction Transistor). A ROM may be programmed at the manufacturing stage using a mask process. Such a ROM is called mask-programmed ROM. The contents of a mask-programmed ROM cannot be changed once it is programmed.

**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.

**UART (Universal Asynchronous Receiver and Transmitter)**

One of the microcontroller features making it so powerful is an integrated UART, better known as a serial port. It is a full-duplex port, thus being able to transmit and receive data simultaneously and at different baud rates. Without it, serial data send and receive would be an enormously complicated part of the program in which the pin state is constantly changed and checked at regular intervals. When using UART, all the programmer has to do is to simply select serial port mode and baud rate. When it's done, serial data transmit is nothing but writing to the SBUF register, while data receive represents reading the same register. The microcontroller takes care of not making any error during data transmission.

**TIMERS**

Timer is use to generate time delay, a timer always counts up. It doesn’t matter whether the timer is being used as a timer, a counter, or a baud rate generator: A timer is always incremented by the microcontroller. There are 3 timers i.e., Timer 0, Timer 1 and Timer 2.

**Timer 0 and 1**

Timer 0 and Timer 1 in the AT89S52 operate the same way as Timer 0 and Timer 1 in the 8051.

Timer T0 is a 16 bit timer. The 16-bit Timer 0 is accessed as low byte and high byte. the low byte is called TL0(Timer 0 low byte)and the high byte is referred to as TH0(Timer 0 high byte).

As like Timer 1 is also 16-bit timer is split into two bytes, referred to as TL1 (Timer 1 low byte) and TH1(Timer 1 high byte). These timer are accessible in the same way as the Timer 0.

**Timer 2**

Timer 2 is a 16-bit Timer/Counter that can operate as either a timer or an event counter. The type of operation is selected by bit C/T2 in the SFR T2CON (shown in Table 5-2). Timer 2 has three operating modes: capture, auto-reload (up or down counting), and baud rate generator. Timer 2 consists of two 8-bit registers, TH2 and TL2. In the Timer function, the TL2 register is incremented every machine cycle. Since a machine cycle consists of 12 oscillator periods, the count rate is 1/12 of the oscillator frequency.

In the Counter function, the register is incremented in response to a 1-to-0 transition at its corresponding external input pin, T2. In this function, the external input is sampled during S5P2 of every machine cycle. When the samples show a high in one cycle and a low in the next cycle, the count is incremented. The new count value appears in the register during S3P1 of the cycle following the one in which the transition was detected. Since two machine cycles (24 oscillator periods) are required to recognize a 1-to-0 transition, the maximum count rate is 1/24 of the oscillator frequency. To ensure that a given level is sampled at least once before it changes, the level should be held for at least one full machine cycle.

**INTERRUPTS**

The AT89S52 has a total of six interrupt vectors: two external interrupts (INT0 and INT1), three timer interrupts (Timers 0, 1, and 2), and the serial port interrupt.

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

Timer 2 interrupt is generated by the logical OR of bits TF2 and EXF2 in register T2CON. Neither of these flags is cleared by hardware when the service routine is vectored to. In fact, the service routine may have to determine whether it was TF2 or EXF2 that generated the interrupt, and that bit will have to be cleared in software.

The Timer 0 and Timer 1 flags, TF0 and TF1, uses the cycle in which the timers overflow. The values are then polled by the circuitry in the next cycle. However, the Timer 2 flag, TF2, is set at S2P2 and is polled in the same cycle in which the timer overflows.

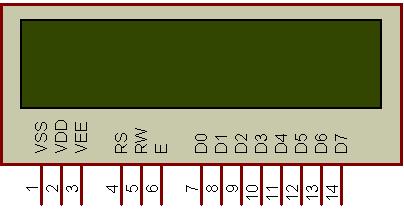
**2.5 LIQUID CRYSTAL DISPLAY**

**2.5.1 INTRODUCTION**

LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over [seven segments](http://www.engineersgarage.com/content/seven-segment-display) and other multi segment [LED](http://www.engineersgarage.com/content/led)s. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even [custom characters](http://www.engineersgarage.com/microcontroller/8051projects/create-custom-characters-LCD-AT89C51) (unlike in seven segments), [animations](http://www.engineersgarage.com/microcontroller/8051projects/display-custom-animations-LCD-AT89C51) and so on.

A **16x2 LCD** means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data.

The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD.



**Fig: 16x2 LCD**

**Introduction**

The most commonly used Character based LCDs are based on Hitachi's HD44780 controller or other which are compatible with HD44580.

**2.5.2 Pin Description**

Most LCDs with 1 controller has 14 Pins and LCDs with 2 controller has 16 Pins (two pins are extra in both for back-light LED connections). Pin description is shown in the table below.

**Pin Configuration table for a 16X2 LCD character display:-**

|  |  |  |
| --- | --- | --- |
| **Pin Number** | **Symbol** | **Function** |
| **1** | Vss | Ground Terminal |
| **2** | Vcc | Positive Supply |
| **3** | Vdd | Contrast adjustment |
| **4** | RS | Register Select; 0→Instruction Register, 1→Data Register |
| **5** | R/W | Read/write Signal; 1→Read, 0→ Write |
| **6** | E | Enable; Falling edge |
| **7** | DB0 | Bi-directional data bus, data transfer is performed once, thru DB0 to DB7, in the case of interface data length is 8-bits; and twice, through DB4 to DB7 in the case of interface data length is 4-bits. Upper four bits first then lower four bits. |
| **8** | DB1 |
| **9** | DB2 |
| **10** | DB3 |
| **11** | DB4 |
| **12** | DB5 |
| **13** | DB6 |
| **14** | DB7 |
| **15** | LED-(K) | Back light LED cathode terminal |
| **16** | LED+(A) | Back Light LED anode terminal |

Fig: Pin Configuration of LCD

**Data/Signals/Execution of LCD**

LCD accepts two types of signals, one is data, and another is control. These signals are recognized by the LCD module from status of the RS pin. Now data can be read also from the LCD display, by pulling the R/W pin high. As soon as the E pin is pulsed, LCD display reads data at the falling edge of the pulse and executes it, same for the case of transmission.

          LCD display takes a time of 39-43µS to place a character or execute a command. Except for clearing display and to seek cursor to home position it takes 1.53ms to 1.64ms. Any attempt to send any data before this interval may lead to failure to read data or execution of the current data in some devices. Some devices compensate the speed by storing the incoming data to some temporary registers.

**Instruction Register (IR) and Data Register (DR)**

There are two 8-bit registers in HD44780 controller Instruction and Data register. Instruction register corresponds to the register where you send commands to LCD e.g LCD shift command, LCD clear, LCD address etc. and Data register is used for storing data which is to be displayed on LCD. When send the enable signal of the LCD is asserted, the data on the pins is latched in to the data register and data is then moved automatically to the DDRAM and hence is displayed on the LCD. Data Register is not only used for sending data to DDRAM but also for CGRAM, the address where you want to send the data, is decided by the instruction you send to LCD. We will discuss more on LCD instruction set further in this tutorial.

**2.5.3 Commands and Instruction set**

Only the instruction register (IR) and the data register (DR) of the LCD can be controlled by the MCU. Before starting the internal operation of the LCD, control information is temporarily stored into these registers to allow interfacing with various MCUs, which operate at different speeds, or various peripheral control devices. The internal operation of the LCD is determined by signals sent from the MCU. These signals, which include register selection signal (RS), read/write signal (R/W), and the data bus (DB0 to DB7), make up the LCD instructions (Table 3). There are four categories of instructions that:

* Designate LCD functions, such as display format, data length, etc.
* Set internal RAM addresses
* Perform data transfer with internal RAM
* Perform miscellaneous functions

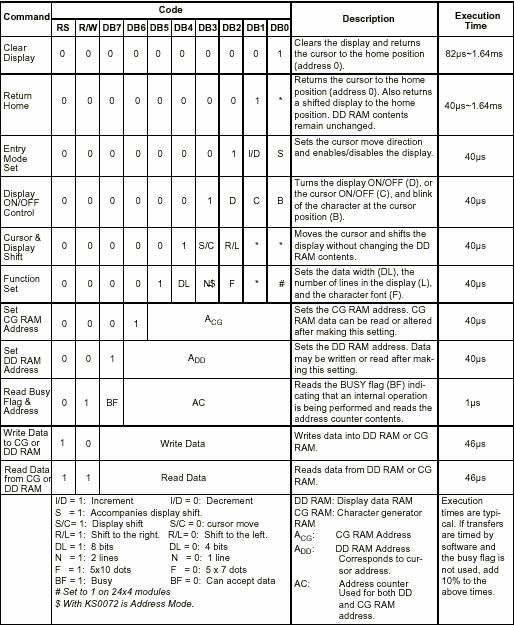


Fig: Code for LCD

Although looking at the table you can make your own commands and test them. Below is a brief list of useful commands which are used frequently while working on the LCD.

**List of Command**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Instruction** | **Hex** | **Decimal** |
| 1 | Function Set: 8-bit, 1 Line, 5x7 Dots | 0x30 | 48 |
| 2 | Function Set: 8-bit, 2 Line, 5x7 Dots | 0x38 | 56 |
| 3 | Function Set: 4-bit, 1 Line, 5x7 Dots | 0x20 | 32 |
| 4 | Function Set: 4-bit, 2 Line, 5x7 Dots | 0x28 | 40 |
| 5 | Entry Mode | 0x06 | 6 |
| 6 | Display off Cursor off (clearing display without clearing DDRAM content) | 0x08 | 8 |
| 7 | Display on Cursor on | 0x0E | 14 |
| 8 | Display on Cursor off | 0x0C | 12 |
| 9 | Display on Cursor blinking | 0x0F | 15 |
| 10 | Shift entire display left | 0x18 | 24 |
| 12 | Shift entire display right | 0x1C | 30 |
| 13 | Move cursor left by one character | 0x10 | 16 |
| 14 | Move cursor right by one character | 0x14 | 20 |
| 15 | Clear Display (also clear DDRAM content) | 0x01 | 1 |
| 16 | Set DDRAM address or cursor position on display | 0x80+add\* | 128+add\* |
| 17 | Set CGRAM address or set pointer to CGRAM location | 0x40+add\*\* | 64+add\*\* |

**Table: Frequently used commands and instructions for LCD**

\* DDRAM address given in LCD basics section see Figure 2,3,4  
\*\* CGRAM address from 0x00 to 0x3F, 0x00 to 0x07 for char1 and so on.

**Liquid crystal displays interfacing with Controller**

The LCD standard requires 3 control lines and 8 I/O lines for the data bus.

• **8 data pins D7:D0**

Bi-directional data/command pins.  
Alphanumeric characters are sent in ASCII format.

• **RS:  Register Select**

RS = 0 -> Command Register is selected  
RS = 1 -> Data Register is selected

• **R/W: Read or Write**

0 -> Write, 1 -> Read

• **E: Enable (Latch data)**

Used to latch the data present on the data pins.  
A high-to-low edge is needed to latch the data.

**2.6 BLUETOOTH**

**Bluetooth** is a wireless technology standard for exchanging data over short distances (using short-wavelength UHF radio waves in the ISM band from 2.4 to 2.485 GHz) from fixed and mobile devices and building personal area networks (PANs). In 1994 a group of engineers at Ericsson, a Swedish company, invented a wireless communication technology, later called Bluetooth. In 1998, the original group of Promoter companies—Ericsson, Intel, Nokia, Toshiba and IBM—came together to form the Bluetooth Special Interest Group (SIG).

Bluetooth networking transmits data via low-power radio waves. It communicates on a frequency of 2.45 gigahertz (actually between 2.402 GHz and 2.480 GHz, to be exact). This frequency band has been set aside by international agreement for the use of industrial, scientific and medical devices (ISM). By comparison, the most powerful cell phones can transmit a signal of 3 watts. The low power limits the range of a Bluetooth device to about 10 meters (32 feet), cutting the chances of interference between your computer system and your portable telephone or television.

**Some specification of Bluetooth**

• Operates in the 2.4 GHZ band which is globally available

• It has 79 channels

• Uses FHSS ,GFSK modulation

• 1600 hops per second

• Can support up to 8 devices in a piconet

• Omni-directional, non-line of sight transmission through walls

• 10m to 100m range

• Low cost, $20

• 1mW power

• Extended range with external power amplifier (100 meters)

**Difference**

There are many other wireless communication techniques are available in the market like IEEE 802.11 (Wi-Fi), IrDA, GSM, GPRS etc. But still Bluetooth is very popular among them. There are many reasons like some techniques required huge infrastructure like GSM, GPRS etc. and also they are not suitable for Adhoc networks like Bluetooth. But some wireless communications techniques are there that are the competitors of Bluetooth. Some main competitors are IEEE 802.11 and IrDA (Infrared Data Association). In this section we will discuss the differences b/w these and Bluetooth.

• **IEEE 802.11**

It is a family of IEEE standards for wireless LANs that were designed to extend 802.3 (wired Ethernet) into the wireless domain. The 802.11 standard is more widely known as "Wi-Fi". Based on the Bluetooth Specification, is now an IEEE standard under the denomination of 802.15 WPANs

The first 802.11 specifications were introduced in 1997 and included two spread spectrum methods for transmission in the unlicensed 2.4GHz band: 1 Mbps frequency hopping (FHSS) and 1 and 2 Mbps direct sequence (DSSS).

**11b**

In 1999, 802.11b boosted speed to 11 Mbps using DSSS. The 1 and 2 Mbps DSSS modes are still valid, and devices can throttle down to the lower speeds to maintain a connection when signals are weak.

An 802.11 system works in two modes. In "infrastructure" mode, wireless devices communicate to a wired LAN via base stations known as "access points." Each access point and its wireless devices are known as a Basic Service Set (BSS). An Extended Service Set (ESS) is two or more BSSs in the same subnet.

In "ad hoc" mode, also known as "peer-to-peer" mode, wireless devices communicate with each other directly without an access point. This is an Independent BSS (IBSS).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Peak Data Rate** | **Range** | **Relative Cost** | **Voice network support** | **Data network support** |
| **IEEE 802.11** | 2 Mbps | 50m | Medium | Via IP | TCP / IP |
| **IrDA** | 16 Mbps | < 2m | Low | Via IP | Via PPP |
| **Bluetooth** | 1 Mbps | < 10m | Medium | Via IP and cellular | Via PPP |
| **HomeRF** | 1.6 Mbps | 50m | Medium | Via IP and PSTN | TCP / IP |

**Communication**

A Bluetooth transceiver is a frequency hopping spread-spectrum (FHSS) device that uses the unlicensed (worldwide) 2.4 GHz ISM (Industrial, Scientific, Medical) frequency band. In most countries, there are 79 channels available. The nominal bandwidth for each channel is 1MHz. When connected to other Bluetooth devices, a Bluetooth device hops (changes frequencies) at the rate of 1600 times per second for typical use, with a residence time of 625 µsec.

When in inquiry or page mode, it hops at 3200 hops per second with a residence time of 312.5 µsec. A Bluetooth transceiver uses all 79 channels, and hops pseudo-randomly across all channels at a rate of 1600 hops per second for standard transmissions. It has a range of approximately10 meters, although ranges up to 100 meters can be achieved with amplifiers. Because the transceiver has an extremely small footprint, it is easily embedded into physical devices, making it a truly ubiquitous radio link. The Bluetooth specification uses time division duplexing (TDD) and time division multiple access (TDMA) for device communication.

A single time slot is 625 µ sec in length, representing the length of a single-slot packet. At the Baseband layer, a packet consists of an access code, a header, and the payload, as shown in Fig. 3. The access code contains the piconet address (to filter out messages from other piconets) and is usually 72 bits in length. The header contains link control data, encoded with a forward error-correcting code (FEC) with a 1/3 rate for high reliability. Such code is a repetition code and thus every bit in the header is transmitted three times. The header is usually 18 bits in length, and includes the active member address for a currently active slave.

The payload can contain from 0 to 2745 bits of data, and may be protected by a 1/3 rate FEC (simple bit repetition, for SCO packets only), a 2/3 rate FEC (which is a (15,10) shortened Hamming code capable of correcting all one-bit errors and detecting all two-bit errors), or a 3/3/ rate (no FEC). For SCO connections, packets must be exactly one time-slot in length. For ACL links, packets may be 1, 3, or 5 time slots in length. Bluetooth uses polling-based packet transmission. All communication between devices takes place between a master and a slave, using time-division duplex (TDD), with no direct slave-to slave communication.

The master will poll each active slave to determine if it has data to transmit. The slave may only transmit data when it has been polled. Also, it must send its data in the time slot immediately following the one in which it was polled. The master transmits only in even numbered time slots, while the slaves transmit only in odd-numbered time slots. In each time slot, a different frequency channel f is used (a hop in the hopping sequence).

### Bluetooth Module

**Baseband** - There are two basic types of physical links that can be established between a master and a slave:

* Synchronous Connection Oriented (SCO)
* Asynchronous Connection-Less (ACL)

An SCO link provides a symmetric link between the master and the slave, with regular periodic exchange of data in the form of reserved slots. Thus, the SCO link provides a circuit-switched connection where data are regularly exchanged, and as such it is intended for use with time-bounded information as audio. A master can support up to three SCO links to the same or to different slaves. A slave can support up to three SCO links from the same master.

An ACI link is a point-to-multipoint link between the master and all the slaves on the piconet. It can use all of the remaining slots on the channel not used for SCO links. The ACL link provides a packet-switched connection where data are exchanged sporadically, as they become available from higher layers of the stack. The traffic over the ACL link is completely scheduled by the master.

Each Bluetooth device has a 48 bit IEEE MAC address that is used for the derivation of the *access code*. The access code has pseudo-random properties and includes the identity of the piconet master. All the packets exchanged on the channel are identified by this master identity. That prevents packets sent in one piconet to be falsely accepted by devices in another piconet that happens to use the same hopping frequency in the certain time slot. . All packets have the same format, starting with an access code, followed by a packet header and ending with the user payload.

Access Code

Header

Payload

0 - 2745 bits

54 bits

68 or 72 bits

**Figure:** Bluetooth packet structure

The access code is used to address the packet to a specific device. The header contains all the control information associated with the packet and the link. The payload contains the actual message information. The Bluetooth packets can be 1, 3, or 5 slots long, but the multi slot packets are always sent on a single-hop carrier.

**The Link Controller** - The link control layer is responsible for managing device discoverability, establishing connections and maintaining them. In Bluetooth, three elements have been defined to support connection establishment: scan, page and inquiry.

Inquiry is a process in which a device attempts to discover all the Bluetooth enabled devices in its local area. A unit that wants to make a connection broadcasts an inquiry message that induces the recipients to return their addresses. Units that receive the inquiry message return an FHS (FH-synchronization) packet which includes, among other things, their identity and clock information. The identity of the recipient is required to determine the page message and wake-up sequence. For the return of FHS packets, a random back off mechanism is used to prevent collisions.

**Figure:** Discovering a Bluetooth device

Laptop

Computer

Mobile

Phone

Inquiry

Inquiry

FHS

**o**

**o**

**o**

A unit in idle mode wants to sleep most of the time to save power, but, from time to time, it also has to listen whether other units want to connect (page scan). In truly ad hoc system, there is no common control channel a unit can lock to in order to listen for page messages. So, every time the unit wakes up, it scans at a different hop carrier for an extended time. A trade-off has to be made between idle mode power consumption and response time: increasing the sleep time reduces power consumption but prolongs the time before an access can be made. The unit that wants to connect has to solve the frequency-time uncertainty: it doesn't know when the idle unit will wake up and on which frequency. For that reason, the paging unit transmits the access code repeatedly at different frequencies: every 1.25ms the paging unit transmits two access codes and listens twice for a response. In 10ms period, 16 different hop carriers are visited. If the idle unit wakes up in any of these 16 frequencies, it will receive the access code and start with a connection setup procedure. First, it will notify the paging unit by returning a message, and then it will transmit a FHS packet which contains all of the pager's information. This information is then used by both units to establish the piconet. Once a baseband link is established, the master and slave can exchange roles if they wish, so that slave becomes master and master becomes slave.

It should be noted that the control of links rests completely with the local device. If it doesn't make itself discoverable by page scanning it cannot be found, if it does not make itself connectable by page scanning it cannot be linked with, and once in a connection it is free to disconnect without warning at any time.

**Audio** - Audio data is carried via SCO (Synchronous Connection Oriented) channels. These SCO channels use pre-reserved slots to maintain temporal consistency of the audio carried on them. This allows us to build devices such as wireless headsets, microphones and headphones using Bluetooth for many consumer products such as cellular phones, call centre switchboards, or even personal musical playback.

There are two routes for audio to pass through a Bluetooth system: through the HCI as data in HCI packets, and via direct PCM connection to the baseband CODECs.

Radio

Baseband

Link Manager

Host Controller Interface

Control

L2CAP

Audio

Higher Layers and Applications

Audio

Data

**Figure:** Position of audio in the Bluetooth stack

The HCI route has some deficiencies in carrying audio data, i.e. packets crossing the HCL are subject to flow control and therefore to variable latency due to microcontroller executing the HCI and LM (Link Manager) tasks. The direct PCM route is not well specified in the Bluetooth specifications, but is very common in commercial implementations.

**The Link Manager** - The host drives a Bluetooth device through Host Controller Interface (HCI) commands, but it is the link manager that translates those commands into operations at the baseband level. Its main functions are to control piconet management (establishing and destruction of the links and role change), link configuration, and security and QoS functions.

Link manager communicates with its peers on other devices using the Link Management Protocol (LMP). Every LMP message begins with a flag bit which is 0 if a master initiated the transaction and 1 if the slave initiated the transaction. That bit is followed by a 7-bit Operation Code, and by the message's parameters.

When a link is first set up, it uses single-slot packets by default. Multi-slot packets make more efficient use of the band, but there are some occasions when they can't be used, for example on noisy links or if SCO links don't leave sufficient space between their slots for multi-slot packets.

LMP also provides a mechanism for negotiating encryption modes and coordinating encryption keys used by devices on both ends of the link. In addition, LMP supports messages for configuration of the quality of service on a connection. Packet types can automatically change according to the channel quality, so that the data can be transferred at a higher rate when the channel quality is good, and on lower rates with more error protection if the channel quality deteriorates.

**Bluetooth Host**

**Logical Link Control and Adaptation Protocol (L2CAP)** - Logical Link Control and Adaptation Protocol takes data from higher layers of the Bluetooth stack and from applications and sends them over the lower layers of the stack. It passes packets either to the HCI, or in a host-less system directly to the Link Manager. The major functions of the L2CAP are:

* Multiplexing between different higher layer protocols to allow several higher layer links to share a single ACL connection. L2CAP uses channel numbers to label packets so that, when they are received, they can be routed to the correct place.
* Segmentation and reassembly to allow transfer of larger packets than lower layers support.
* Quality of service management for higher layer protocols.

All applications must use L2CAP to send data. It is also used by Bluetooth's higher layers such as RFCOMM and SDP, so L2CAP is a compulsory part of every Bluetooth system.

**RFCOMM** - RFCOMM is a simple, reliable transport protocol that provides emulation of the serial cable line settings and status of an RS-232 serial port. It provides connections to multiple devices by relying on L2CAP to handle multiplexing over single connection. RFCOMM supports two types of devices:

* Type 1 - Internal emulated serial port. These devices usually are the end of a communication path, for example a PC or printer.
* Type 2 - Intermediate device with physical serial port. These are devices that sit in the middle of a communication path, for example a modem.

Up to 30 data channels can be set up, so RFCOMM can theoretically support 30 different services at once. RFCOMM is based on GSM TS 07.10 standard, which is an asymmetric protocol used by GSM cellular phones to multiplex several streams of data onto one physical serial cable.

**The Service Discovery Protocol** - One of the most important members of the Bluetooth protocol stack is Service Discovery Protocol (SDP). It provides a means for an SDP client to access information about services offered by SDP servers. An SDP server is any Bluetooth device which offers services to other Bluetooth devices. Information about services is maintained in SDP databases. There is no centralized database, so each SDP server maintains its own database. The SDP database is simply a set of records describing all the services which a Bluetooth device can offer to another Bluetooth device, and service discovery protocol provides a means for another device to look at these records. To make it easier to find the service you want, services are arranged in a hierarchy structure as a tree which can be browsed. Clients begin by examining the root of the tree, then follow the hierarchy out to the leaf nodes where individual services are described.

To browse service classes, or get information about a specific service, SDP clients and servers exchange messages which are carried in SDP Protocol Data Units (PDUs). The first byte of PDU is an ID, identifying the message in the PDU. Services have Universally Unique Identifiers (UUIDs) that describe them. The services defined by the Bluetooth profiles have UUIDs assigned by the standard, but service providers can define their own services and assign their own UUIDs to those services.

SDP relies on L2CAP links being established between SDP client and server, before retrieving SDP information. Stages in setting up an SDP connection are shown on a figure.

Inquiry

Paging

LMP\_host connection\_req

LMP\_accepted

LMP\_name\_req

LMP\_name\_res

Authentication

LMP\_Setup\_complete

LMP\_Setup\_complete

L2CAP\_connection\_req

L2CAP\_connection\_res

SDP\_inquires

SDP\_responses

Terminate Connection

Link Controller

Connection Setup

Link Manager

Connection Setup

L2CAP

Connection Setup

SDP Session

Disconnect

Local Device

(SDP Client)

Remote Device

(SDP Server)

**Figure:** Stages in setting up an SDP session

**Supported Protocols** - As mentioned at the beginning of this paper, one of the most important characteristics of the Bluetooth specification is that it should allow devices from lots of different manufacturers to work with one another. For that reason, Bluetooth is designed in such a way to allow many different protocols to be run on top of it. Some of these protocols are:

**The Wireless Access Protocol (WAP) -** WAP provides a protocol stack similar to the IP stack, but it is tailored for the needs of mobile devices. It supports the limited display size and resolution typically found on mobile devices by providing special formats for Web pages which suit their capabilities. It also provides for the low bandwidth of mobile devices by defining a method for WAP content to be compressed before it is transmitted across a wireless link. WAP can use Bluetooth as a bearer layer in the same way as it can use GSM, CDMA and other wireless services. The WAP stack is joined to the Bluetooth stack using User Datagram Protocol (UDP), Internet Protocol (IP) and Point to Point Protocol (PPP).

**Object Exchange Protocol (OBEX)** - OBEX is a protocol designed to allow a variety of devices to exchange data simply and spontaneously. Bluetooth has adopted this protocol from the Infrared Data Association (IrDA) specifications. OBEX has a client/server architecture and allows a client to push data to a server or pull data from the server. For example, a PDA might pull a file from a laptop, or a phone synchronizing an address book might push it to a PDA. The similarities between the two communications protocols' lower layers mean that IrDA's OBEX protocol is ideally suited to transferring objects between Bluetooth devices.

**The Telephony Control Protocol -** Bluetooth's Telephony Control protocol Specification (TCS) defines how telephone calls should be sent across a Bluetooth link. It gives guidelines for the signaling needed to set up both point to point and point to multipoint calls. By use of TCS, calls from an external network can be directed to other Bluetooth devices. For instance, a cellular phone could receive a call and use TCS to redirect the call to a laptop, allowing the laptop to be used as a hands-free phone. TCS is driven by a telephony application which provides the user interface, and provides the source of voice or data transferred across the connection set up by TCS.

**Applications: The Bluetooth Profiles** - In addition to protocols which guarantee that two units speak the same language, Bluetooth specification defines the profiles. They are associated with applications. The profiles specify which protocol elements are mandatory in certain applications. This concept prevents devices with little memory and processing power implementing the entire Bluetooth stack when they only require a small fraction of it. Simple devices like a headset or mouse can thus be implemented with a strongly reduced protocol stack.

The Bluetooth profiles are organized into groups, with each profile building upon the one beneath and inheriting features from below. For developers, this means that key features of one Bluetooth solution can be reused in other solutions, bringing down development costs and speeding up the development cycle.

Cordless

Telephony Profile

Intercom

Profile

Telephony Control Protocol Specification

File Transfer

Profile

Object Push

Profile

Synchronization

Profile

Generic Object Exchange Profile

Dial-Up Networking

Profile

FAX

Profile

Headset

Profile

LAN Access

Profile

Serial Port Profile

Service Discovery

Application Profile

Generic Access Profile

**Figure:** Bluetooth profiles

The profiles implemented by Bluetooth version 1.0 are:

* Generic Access - It defines the basic rules for using the protocol stack.
* Serial Port - How to use RFCOMM's serial port emulation capabilities in Bluetooth products.
* Dial-up Networking - A Bluetooth link to a modem.
* FAX - How to transfer a fax over Bluetooth.
* Headset - A duplex link to a headset, controlled by an audio gateway such as cellular phone.
* LAN Access Point - A link to LAN via Bluetooth.
* Generic Object Exchange - A set of rules for using OBEX, which supports file transfer, object push and synchronization profiles.
* File Transfer - Transferring files between Bluetooth devices.
* Object Push - Pushing objects from a Bluetooth enabled server to a client.
* Synchronization - Synchronizing objects between Bluetooth devices.
* Cordless Telephony - Forwarding telephone calls to Bluetooth devices.
* Intercom - Short range voice connections between Bluetooth devices.

### SECURITY

Basic security elements need to be considered to prevent unauthorized usage and eavesdropping in Bluetooth system though it is mainly intended for short-range connectivity between personal devices. Security features are included at the link level and are based on a secret link key that is shared by a pair of devices. To generate this key a pairing procedure is used when the two devices communication for the first time.

At connection establishment, an authentication process is carried out to verify the identities of the units involved. The authentication process uses a conventional challenge-response routine. The verifier compares signed response (SRES) produced by the claimant with its own SRES and decides if the challenger may continue with connection establishment. To prevent eavesdropping on the link, which is a danger inherent to radio communications, the payload of each packet is encrypted. Encryption is based on stream ciphering; the payload bits are modulo-2 added to a binary key stream.

The central element in the security process is the 128-bit link key. This link key is a secret key residing in the Bluetooth hardware and is not accessible by the user. The link key is generated during an initialization phase. Once the initialization has been carried out, the 128-bit link keys reside in the devices and can from then on be used for automatic authentication without user interaction. In addition, methods are available to use the same encryption keys for all slaves in a single piconet.

Bluetooth provides limited number of security elements at the lowest level. More advanced security procedures can be implemented at higher layers.

**POWER MANAGEMENT**

As many Bluetooth devices are operated by batteries, special attention has been paid to the reduction of power consumption in the design. And many tests have been done to prove that Bluetooth devices are too low in power to have any negative impact on health. Three low-power modes, which extend battery life by reducing activity on a connection, have been defined. These modes are called Park, Hold, and Sniff.

Park mode provides the greatest opportunities for power saving. The device only wakes up in periodic beacon slots when it listens for unpark transmission from the Master. If it is not unparked, it goes back to sleep, switching off its receiver. Devices that are parked give up their active member addresses, so one Master can have more devices in Park mode at once. In Sniff mode, the slave does not scan at every master-to-slave slot, but has a larger interval between scans. Devices in Sniff mode keep their active member address. Typically, sniffing devices will be active more often than parked devices. Both Park and Sniff modes involve putting devices into a state where they wake up periodically while Hold mode just puts a connection in a low-power state for a single period. So a Master needs to perform an inquiry to be able to service the connections again.

In the connection state, current consumption is minimized and wasteful interference prevented by only transmitting when data is available. In longer periods of silence, the master needs to send a packet on the channel once in a while such that all slaves can resynchronize their clocks and compensate for drift. During continuous TX/RX operations, a unit starts to scan for the access code at the beginning of the RX slot. If the access code is not found, or even if it is found but the slave address does not match the recipient, the unit goes to sleep until the next slot. The header indicates what type of packet it is and how long the packet will last; therefore, the non-addressed recipients can determine how long they can sleep.

The nominal transmit power used by most Bluetooth applications for short-range connectivity is 0 dBm. This restricts current consumption and keeps interference to other systems to a minimum. However, the Bluetooth radio specifications allow TX power up to 20 dBm. Above 0 dBm, closed-loop received signal strength indication power control is mandatory. This power control can compensate for propagation losses and slow fading.

In low-power modes many layers of the Bluetooth protocol stack are involved: as after periods of inactivity, the device may lose synchronization and need to listen for transmissions over a wider window than usual, the baseband layer alters correlator properties. The link manager provides a variety of messages to configure and negotiate the low-power modes between ends of a connection. HCI provides a set of commands that may be used by a host to configure and control the power-saving capabilities of a module. L2CAP must be aware of low-power modes for its quality of service commitments.

Different Bluetooth devices may have different requirements for data rate, delay variance, and reliability. The specification provides Quality Of Service (QOS) configuration for the properties of links according to the requirements of higher layer applications or protocols. These properties include the type of QOS, token rate, token rate bucket size, peak bandwidth, latency and delay variation.

Host Controller Interface

Logical Link Control and Adaptation

QOS Requirements

QOS Config. Success or FAil

QOS Setup

QOS Violations

QOS Setup

QOS Violations

Link Manager

Link Control

Link Manager

Link Information

QOS Violations

High Layer Protocols and Applications

Host Controller Interface

Logical Link Control and Adaptation

QOS Requirements

QOS Config. Success or FAil

QOS Setup

QOS Violations

QOS Setup

QOS Violations

Link Manager

Link Control

Link Manager

Link Information

QOS Violations

High Layer Protocols and Applications

L2CAP QOS Negotiation

LMP QOS Negotiation

#### Figure: QoS Messaging

Figure shows how to use message throughout the Bluetooth protocol stack to control QOS. Messages configuring and setting up QOS flow vertically up and down the layers of the stack, while Link manager and Logical Link control and Adaptation layer (L2CAP) configure QOS in peer to peer negotiations. Link Manager actually implements QOS policies for it configures and controls the baseband links and has various means to try to meet the QOS which L2CAP requests.

When a link is first set up, QOS is requested from the higher layer to L2CAP. Then the negotiation packets of QOS configuration are sent between local and remote L2CAP. The link manager provides QOS capabilities according to the requests from L2CAP. On systems with an HCI, this interaction between L2CAP and Link Manager is accomplished through a series of HCI commands and events. LMP commands can be used to configure the poll interval, the maximum interval between packets sent from Master to Slave, and the broadcast packet repeat times. QOS setup completion is generated when LMP has finished setting. If failed, message will be sent back to higher layer to decide whether to try again or to give up. If succeeded, the channel will then open for transferring data at the desired QOS.

Even a channel has been configured, it is important that applications are aware whether their QOS is not as requested, as they may wish to either shut down the link rather than run it at an inappropriate quality, or shut down other links to improve this link. In such case, lower layers send QOS violation events to tell the higher layers and let them decide what to do about it.

### WHAT IS BLUETOOTH FOR?

Although originally thought of simply as a replacement for the nest of wires that connects PCs to keyboards and printers, Bluetooth quickly evolved into a system that will allow people to detect and communicate with each other through a variety of mainly portable devices without their users' intervention. Bluetooth devices will be able to talk to each other as they come into range, which is about 10 meters, although this can be extended to more than 100 meters by increasing the transmit power from a nominal 1mW to as much as 100mW. Bluetooth technology is expected to make its debut in cell phones and Palm-type personal digital assistants (PDAs), but then will move quickly into notebook and laptop computers, printers, scanners, digital cameras, household appliances, games, toys, and more.

With Bluetooth technology, one can send e-mail from the computer on his lap to the cellular phone in his briefcase. Bluetooth-linked cell phone or similarly equipped PDA can automatically synchronize with desktop PC whenever the cell phone passes it within the Bluetooth range. Or, one can have hands-free communication between a Bluetooth enabled headset and a cell phone. Or download images from a digital camera to a PC or a cell phone...

Presently, Nokia and Fujifilm are working on a mobile imaging technology that should enable Nokia to add a Bluetooth chip to its clamshell-shaped 9110 Communicator so that it can receive images taken on a Bluetooth-equipped Fujifilm digital camera.

Finnish telecom operator Sonera has even demonstrated a Bluetooth enabled vending machine - consumers buy products out of the machine by simply signaling an account code from a Bluetooth cell phone or PDA.

Many other applications can be also thought of Bluetooth can serve as a means for connecting laptop computers or other devices to the public Internet in airport lounges and conference centres through permanent access points. It can also enable its users to exchange business cards with everyone who passed on a street through a Bluetooth enabled Palm - but not unless it has been given permission to identify the user to anyone. Maybe it would be neat to have a system that would automatically reset all the digital clocks in a house following the power outage. Or, to have a Bluetooth link between the roller blades and a speedometer in a digital watch. But all these applications will have to wait for some more time before they hit the market, since there is still a lot of work to be done, mostly regarding interoperability issues and final test procedures for Bluetooth products.

**2.7 DRIVER CIRCUIT (L293D)**

**INTRODUCTION**

L293D IC generally comes as a standard 16-pin DIP (dual-in line package). This motor driver IC can simultaneously control two small motors in either direction; forward and reverse with just 4 microcontroller pins (if you do not use enable pins).

**Some of the features (and drawbacks) of this IC are:**

1. Output current capability is limited to 600mA per channel with peak output current limited to 1.2A (non-repetitive). This means you cannot drive bigger motors with this IC. However, most small motors used in hobby robotics should work. If you are unsure whether the IC can handle a particular motor, connect the IC to its circuit and run the motor with your finger on the IC. If it gets really hot, then beware... Also note the words "non-repetitive"; if the current output repeatedly reaches 1.2A, it might destroy the drive transistors.
2. Supply voltage can be as large as 36 Volts. This means you do not have to worry much about voltage regulation.
3. L293D has an enable facility which helps you enable the IC output pins. If an enable pin is set to logic high, then state of the inputs match the state of the outputs. If you pull this low, then the outputs will be turned off regardless of the input states
4. The datasheet also mentions an "over temperature protection" built into the IC. This means an internal sensor senses its internal temperature and stops driving the motors if the temperature crosses a set point
5. Another major feature of **L293D** is its internal clamp diodes. This flyback diode helps protect the driver IC from voltage spikes that occur when the motor coil is turned on and off (mostly when turned off)
6. The logical low in the IC is set to 1.5V. This means the pin is set high only if the voltage across the pin crosses 1.5V which makes it suitable for use in high frequency applications like switching applications (upto 5KHz)
7. Lastly, this integrated circuit not only drives DC motors, but can also be used to drive relay solenoids, stepper motors etc.

**Description**

It works on the concept of H-bridge. H-bridge is a circuit which allows the voltage to be flown in either direction. As you know voltage need to change its direction for being able to rotate the motor in clockwise or anticlockwise direction, Hence H-bridge IC are ideal for driving a DC motor.

In a single l293d chip there two h-Bridge circuit inside the IC which can rotate two dc motor independently. Due its size it is very much used in robotic application for controlling DC motors. Given below is the pin diagram of a L293D motor controller.

There are two Enable pins on l293d. Pin 1 and pin 9, for being able to drive the motor, the pin 1 and 9 need to be high. For driving the motor with left H-bridge you need to enable pin 1 to high. And for right H-Bridge you need to make the pin 9 to high. If anyone of the either pin1 or pin9 goes low then the motor in the corresponding section will suspend working. It’s like a switch.

**Pin Diagram**

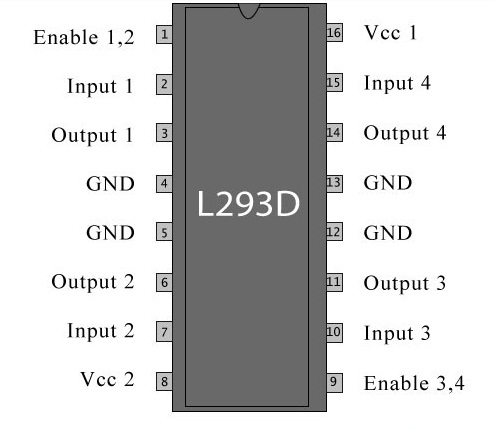


Fig showing pin diagram of L293D

**Pin Description**

|  |  |  |
| --- | --- | --- |
| **Pin No** | **Function** | **Name** |
| 1 | Enable pin for Motor 1; active high | Enable 1,2 |
| 2 | Input 1 for Motor 1 | Input 1 |
| 3 | Output 1 for Motor 1 | Output 1 |
| 4 | Ground (0V) | Ground |
| 5 | Ground (0V) | Ground |
| 6 | Output 2 for Motor 1 | Output 2 |
| 7 | Input 2 for Motor 1 | Input 2 |
| 8 | Supply voltage for Motors; 9-12V (up to 36V) | Vcc 2 |
| 9 | Enable pin for Motor 2; active high | Enable 3,4 |
| 10 | Input 1 for Motor 1 | Input 3 |
| 11 | Output 1 for Motor 1 | Output 3 |
| 12 | Ground (0V) | Ground |
| 13 | Ground (0V) | Ground |
| 14 | Output 2 for Motor 1 | Output 4 |
| 15 | Input2 for Motor 1 | Input 4 |
| 16 | Supply voltage; 5V (up to 36V) | Vcc 1 |

**Table Pin Description of L293D**

**Working of L293D**

The 4 input pins for this l293d, pin 2,7 on the left and pin 15 ,10 on the right as shown on the pin diagram. Left input pins will regulate the rotation of motor connected across left side and right input for motor on the right hand side. The motors are rotated on the basis of the inputs provided across the input pins as LOGIC 0 or LOGIC 1.

In simple you need to provide Logic 0 or 1 across the input pins for rotating the motor.

**Voltage Specification**

VCC is the voltage that it needs for its own internal operation 5v; L293D will not use this voltage for driving the motor. For driving the motor it has a separate provision to provide motor supply VSS (V supply).  L293d will use this to drive the motor. It means if you want to operate a motor at 9V then you need to provide a Supply of 9V across VSS Motor supply.

The maximum voltage for VSS motor supply is 36V. It can supply a max current of 600mA per channel. Since it can drive motors Up to 36v hence you can drive pretty big motors with this l293d.

VCC pin 16 is the voltage for its own internal Operation. The maximum voltage ranges from 5v and up to 36v.

**2.8 DC MOTOR**

A DC motor  in simple words is a device that converts direct current(electrical energy) into mechanical energy. It’s of vital importance for the industry today.

A DC motor is designed to run on DC electric power. Two examples of pure DC designs are Michael Faraday's homo-polar motor (which is uncommon), and the ball bearing 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.

We in our project are using brushed DC Motor, which will operate in the ratings of 12v DC 0.6A.

The speed of a DC motor can be controlled by changing the voltage applied to the armature or by changing the field current. The introduction of variable resistance in the armature circuit or field circuit allowed speed control. Modern DC motors are often controlled by power electronics  systems called DC drives.



Fig. Motor

**Usage**

The DC motor or Direct Current Motor to give it its full title, is the most commonly used actuator for producing continuous movement and whose speed of rotation can easily be controlled, making them ideal for use in applications were speed control, servo type control, and/or positioning is required. A DC motor consists of two parts, a "Stator" which is the stationary part and a "Rotor" which is the rotating part. The result is that there are basically three types of DC Motor available.

**2.9** **BUZZER**

A [buzzer](http://www.microbuzzer.com/) or beeper is a signaling device, usually electronic, typically used in automobiles, house hold appliances such as a microwave oven, or game shows.

It most commonly consists of a number of switches or sensors connected to a control unit that determines if and which button was pushed or a preset time has lapsed, and usually illuminates a light on the appropriate button or control panel, and sounds a warning in the form of a continuous or intermittent buzzing or beeping sound. Initially this device was based on an electromechanical system which was identical to an electric bell without the metal gong (which makes the ringing noise). Often these units were anchored to a wall or ceiling and used the ceiling or wall as a sounding board. Another implementation with some AC-connected devices was to implement a circuit to make the AC current into a noise loud enough to drive a loudspeaker and hook this circuit up to a cheap 8-ohm speaker. Nowadays, it is more popular to use a ceramic-based piezoelectric sounder like a Sonalert which makes a high-pitched tone. Usually these were hooked up to “driver” circuits which varied the pitch of the sound or pulsed the sound on and off.

In game shows it is also known as a “lockout system,” because when one person signals (“buzzes in”), all others are locked out from signalling. Several game shows have large buzzer buttons which are identified as “plungers”.



Fig. Buzzer

**USES**

* [Annunciator panels](http://en.wikipedia.org/wiki/Annunciator_panel" \o "Annunciator panel)
* Electronic [metronomes](http://en.wikipedia.org/wiki/Metronome" \o "Metronome)
* [Game shows](http://en.wikipedia.org/wiki/Game_show" \o "Game show)
* [Microwave ovens](http://en.wikipedia.org/wiki/Microwave_oven" \o "Microwave oven) and other [household appliances](http://en.wikipedia.org/wiki/Major_appliance" \o "Major appliance)
* [Sporting](http://en.wikipedia.org/wiki/Sport" \o "Sport) events such as [basketball](http://en.wikipedia.org/wiki/Basketball" \o "Basketball) games
* Electrical [alarms](http://en.wikipedia.org/wiki/Alarms" \o "Alarms)

**CHAPTER 3**

**SOFTWARE SPECIFICATION**

**KEIL SOFTWARE**

Keil compiler is a 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.

**General Introduction**

Keil Software is the leading vendor for 8/16-bit development tools (ranked at first position in the 2004 Embedded Market Study of the Embedded Systems and EE Times magazine). Keil Software is represented world-wide in more than 40 countries. Since the market introduction in 1988, the Keil C51 Compiler is the de facto industry standard and supports more than 500 current 8051 device variants. Now, Keil Software offers development tools for ARM.

Keil Software makes C compilers, macro assemblers, real-time kernels, debuggers, simulators, integrated environments, and evaluation boards for the 8051, 251, ARM, and XC16x/C16x/ST10 microcontroller families.

Keil Software is pleased to announce simulation support for the Atmel AT91 ARM family of microcontrollers. The Keil µVision Debugger simulates the complete ARM instruction-set as well as the on-chip peripherals for each device in the AT91 ARM/Thumb microcontroller family. The integrated simulator provides complete peripheral simulation. Other new features in the µVision Debugger include:

* An integrated Software Logic Analyzer that measures I/O signals as well as program variables and helps developers create complex signal processing algorithms.
* An Execution Profiler that measures time spent in each function, source line, and assembler instruction. Now developers can find exactly where programs spend the most time.

"Using nothing more than the provided simulation support and debug scripts, developers can create a high-fidelity simulation of their actual target hardware and environment. No extra hardware or test equipment is required. The Logic Analyzer and Execution Profiler will help developers when it comes time to develop and tune signaling algorithms." said Jon Ward, President of Keil Software USA, Inc.

**µVision3 Overview**

The µVision3 IDE is a Windows-based software development platform that combines a robust editor, project manager, and makes facility. µVision3 integrates all tools including the C compiler, macro assembler, linker/locator, and HEX file generator. µVision3 helps expedite the development process of your embedded applications by providing the following:

* Full-featured source code editor,
* Device database for configuring the development tool setting,
* Project manager for creating and maintaining your projects,
* Integrated make facility for assembling, compiling, and linking your embedded applications,
* Dialogs for all development tool settings,
* True integrated source-level Debugger with high-speed CPU and peripheral simulator,
* Advanced GDI interface for software debugging in the target hardware and for connection to Keil ULINK,
* Flash programming utility for downloading the application program into Flash ROM,
* Links to development tools manuals, device datasheets & user’s guides.

The µVision3 IDE offers numerous features and advantages that help you quickly and successfully develop embedded applications. They are easy to use and are guaranteed to help you achieve your design goals.

The µVision3 IDE and Debugger is the central part of the Keil development tool chain. µVision3 offers a Build Mode and a Debug Mode.

In the µVision3 [Build Mode](ms-its:C:\\Keil\\C51\\HLP\\uv3.chm::/uv3_creating_apps.htm) you maintain the project files and generate the application.

In the µVision3 [Debug Mode](ms-its:C:\\Keil\\C51\\HLP\\uv3.chm::/uv3_debugging.htm) you verify your program either with a powerful CPU and peripheral simulator or with the [Keil ULINK USB-JTAG Adapter](http://www.keil.com/ulink/" \t "_blank) (or other AGDI drivers) that connect the debugger to the target system. The ULINK allows you also to download your application into Flash ROM of your target system.

**STEPS TO WRITE AN ASSEMBLY LANGUAGE PROGRAM IN KEIL AND HOW TO COMPILE IT:**

1. Install the Keil Software in the PC in any of the drives.
2. After installation, an icon will be created with the name “Keil uVision3”. Just drag this icon onto the desktop so that it becomes easy whenever you try to write programs in keil.
3. Double click on this icon to start the keil compiler.
4. 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.
5. Now to start using the keil, click on the option “project”.
6. A small window opens showing the options like new project, import project, open project etc. Click on “New project”.
7. 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.
8. 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.
9. 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.
10. 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.
11. 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.
12. Now you can see the TARGET and SOURCE GROUP created in the project workspace.
13. Now click on “File” and in that “New”. A new page opens and you can start writing program in it.
14. 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.
15. You can notice that after you save the program, the predefined keywords will be highlighted in bold letters.
16. 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.
17. 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”.
18. 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.
19. 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”.
20. 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.
21. 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.
22. The icon with the letter “d” indicates the debug mode.
23. Click on this icon and now click on the option “View” and select the appropriate window to check for the output.
24. After this is done, click the icon “debug” again to come out of the debug mode.
25. The hex file created as shown earlier will be dumped into the microcontroller with the help of another software called Proload/Topwin.

**CHAPTER 4**

**IMPLEMENTATION**

**4.1 SCHEMATIC DIAGRAM:**

**CHAPTER 5**

**SIMULATION AND DESIGN**

**5.1 OUTPUT SCREEN SHOTS**

**CHAPTER 6**

**CONCLUSION**

From the above information of this system up to now, surely comes to know that it is highly reliable effective and economical at dense traffic area, sub urban area and the route where frequency of trains is more. As it saves some auxiliary structure as well as the expenditure on attendant it is more economical than traditional railway crossing gate system. We know that though it is very beneficial but it is also impossible to install such system at each and every place, but it gives certainly a considerable benefit to us, thereby to our nation.

**CHAPTER 7**

**REFERENCES:**

[1] en.wikipedia.org/wiki/Rail\_inspection

[2] en.wikipedia.org/wiki/Nondestructive\_testing

[3] en.wikipedia.org/wiki/Level\_crossing

[4] en.wikipedia.org/wiki/Laser\_scanning

[5] B. Brailson Mansingh, “Automation in unmanned railway level crossing”, in Intelligent Systems and Control (ISCO), pg. 1 – 4, 2015

[6] Onder Halis Bettemir, “Detection of railway track from image by heuristic method”, in Signal Processing and Communications Applications Conference (SIU), pg. 1366 – 1369, 2015

[7] Selvamraju Somalraju, Vigneshwar Murali, Gourav Saha, Dr.V.Vaidehi, “Robust Railway Crack Detection Scheme (RRCDS) Using LEDLDR Assembly,” in Networking, Sensing and Control, vol. 6, iss. 3, pg. 453-460 , May2012

[8] Qiao Jian-hua; Li Lin-sheng; Zhang Jing-gang; “Design of Rail Surface Crack- detecting System Based on Linear CCD Sensor,” IEEE Int.Conf. on Networking, Sensing and Control, vol. 14, no. 4, pp. 961-970, April 2008

[9] K. Vijayakumar, S.R. Wylie, J. D. Cullen, C.C. Wright, A.I. Shammaa, “ Non invasive rail track detection system using Microwave sensor,” Journal of App. Phy., vol. 9, iss. 11, pg. 1743-1749, June 2009

[10] Richard J. Greene, John R. Yates and Eann A. Patterson, “Crack detection in rail using infrared methods,” Opt. Eng. 46, 051013, May 2007

[11]. R.J. Greene, J.R. Yates, E.A. Patterson, “Rail Crack Detection: An Infrared Approach to In-service Track Monitoring,” SEM Annual Conference & Exposition on Experimental and Applied Mechanics, vol. 112, nos. 23, pp. 291301, May 2006

[12] Giovanni M. Carlomagno, Carosena Meola: HComparison between thermographic techniques for frescoes NDTH NDT & E International, Volume 35, Issue 8, December 2002, pg. 559-565.

[13] Hartless, P.C.M.; Whalley, D.C. Automated thermographic inspection of surface mount solder joints. Automotive Electronics, 1991., Eighth International Conference on Volume , Issue , 28-31 Oct 1991 Pg.178 – 181.