VIVEKANAND EDUCATION SOCIETY'S

POLYTECHNIC

Chembur, Mumbai -71

SINCE 1962

SMART SCHOOL MANAGEMENT SYSTEM.

A PROJECT REPORT

Submitted By

Mr. ACHUTHA MENON

Mr. LAXMAN RATHOD

Mr. HASAN KHOT

Mr. ANOOP CHAUHAN

In partial fulfillment for the award of the degree

Of

DIPLOMA

IN

ELECTRONICS & COMMUNICATION ENGINEERING UNDER THE GUIDENCE OF Ms. NEELIMA PALASPAGAR

Maharashtra State Board of
Technical Education
(MSBTE)
Govt. of Maharashtra

VIVEKANAND EDUCATION SOCIETY'S

POLYTECHNIC

Chembur, Mumbai -71

SINCE 1962

SMART SCHOOL MANAGEMENT SYSTEM

A PROJECT REPORT

Submitted By

Mr. ANOOP CHAUHAN

In partial fulfillment for the award of the degree

Of

DIPLOMA

IN

ELECTRONICS & COMMUNICATION ENGINEERING

UNDER THE GUIDENCE OF

Mrs. NEELIMA PALASPAGAR

Maharashtra State Board of
Technical Education
(MSBTE)
Govt. of Maharashtra

2017-2018

CERTIFICATE

Certified: **Mr. ANOOP CHAUHAN** have carried out the project work presented in this report entitled **"SMART SCHOOL MANAGEMENT SYSTEM"** for the award of Diploma in Electronics & Communication from VIVEKANAND EDUCATION SOCIETY'S POLYTECHNIC, Chembur, Mumbai under my guidance. The report embodies results of original work, and studies are carried out by the students and the contents of the report do not form the basis for the award of any other degree to candidate or to anybody else from this or any other Institution.

Signature of the guide
(**Mrs. NEELIMA PALASPAGAR.)**

Department Electronic & Communication
Engineering

Date: ……..
Place: ……..

Department of Electronic & Communication Engineering

V.E.S. Polytechnic, Mumbai-400071

SINCE 1962

VIVEKANAND EDUCATION SOCIETY'S

POLYTECHNIC

CHEMBUR, MUMBAI-400071

SINCE 1962

CERTIFICATE OF APPROVAL

Project titled on "SMART SCHOOL MANAGEMENT SYSTEM" submitted by **Mr. ANOOP CHAUHAN** is approved for the diploma of Engineering.

_____

Guide

Mrs. NEELIMA PALASPAGAR.

_____

Examiner

_____

Head of the Department

Mrs. Deena Shah

_____

Principal

Mr. Vikrant Joshi

Department of Electronic & Communication Engineering

V.E.S. Polytechnic, Mumbai-400071

Acknowledgement

It is indeed a matter of great pleasure and proud privilege to be able to present this project on **"SMART SCHOOL MANAGEMENT SYSTEM".**

We would like to express our deep regards and gratitude to the principal **Mr. Vikrant Joshi** and Head of Department **Mrs. Deena Shah.**

The completion of this project work is a milestone in student life and its execution is inevitable in the hands of guide. We are highly indebted the project guides **Mrs. NEELIMA PALASPAGAR**. for her invaluable guidance and appreciation for giving form and substance to this report. It is due to her enduring efforts, patience and enthusiasm, which has given a sense of direction and purposefulness to this project and ultimately made it a success.

We would also like to tender our sincere thanks to the staff members for their co-operation.

We would wish to thank non-teaching staff and our friends who have helped us all the time in way or the other.

INDEX

INTRODUCTION

1.1 An Introduction to Smart School Management system

Let's consider a school in which the functioning is automated. Every classroom has:

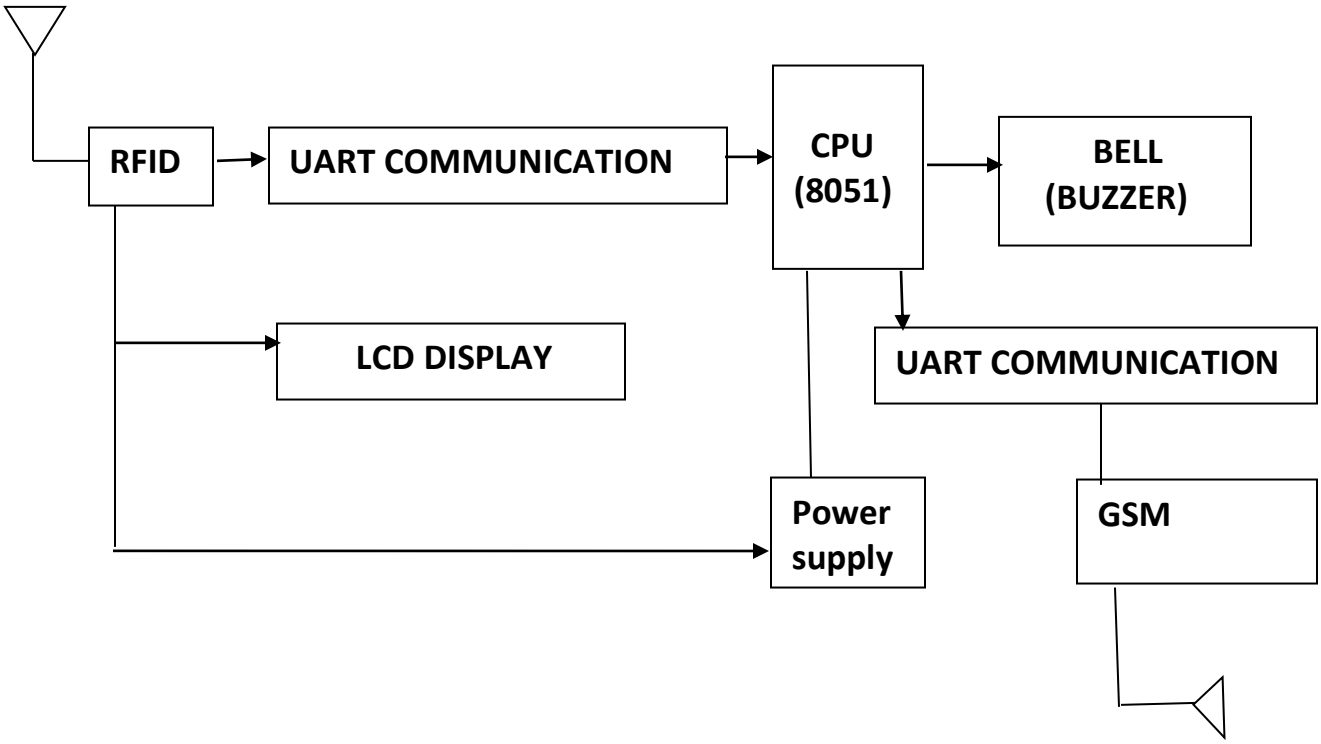
Attendance: As the student enters the class he will swipe the card on the RFID Reader. The card will be provided by the school. The card called as Tag will contain the name of student and respective enrollment number. The tag will also be provided for the teacher. This will register the log in and log off time of the students and teachers. The registration will be shown on LCD Display.

The alarm will be ranged automatically as the lecture gets over at intervals.
Using the IR sensor there is automatic switching on and off of the lights near the school at the night time.

If a student's bunks the school, he won't swipe his/her card and hence his/her log out time won't be registered that will automatically send a SMS to respective parent through GSM.

BLOCK DIAGRAM, & CIRCUIT DIAGRAM

2.1 BLOCK DIAGRAM



2.1.1 BLOCK DIAGRAM DESCRIPTION

Radio-frequency identification (RFID) uses electromagnetic fields to automatically identify and track tags attached to objects. The tags contain electronically stored information. Passive tags collect energy from a nearby RFID reader's interrogating radio waves. Active tags have a local power source (such as a battery) and may operate hundreds of meters from the RFID reader. Unlike a barcode, the tag need not be within the line of sight of the reader, so it may be embedded in the tracked object. RFID is one method for Automatic Identification and Data Capture (AIDC).

In ***UART communication***, two UARTs communicate directly with each other. The transmitting UART converts parallel data from a controlling device like a CPU into serial form, transmits it in serial to the receiving UART, which then converts the serial data back into parallel data for the receiving device. Only two wires are needed to transmit data between two UARTs. Data flows from the Tx pin of the transmitting UART to the Rx pin of the receiving UART.

The Intel ***8051*** is an 8-bit microcontroller which means that most available operations are limited to 8 bits. There are 3 basic "sizes" of the 8051: Short, Standard, and Extended. The Short and Standard chips are often available in DIP (dual in-line package) form, but the Extended 8051 models often have a different form factor, and are not "drop-in compatible". All these things are called 8051 because they can all be programmed using 8051 assembly language, and they all share certain features (although the different models all have their own special features).

Some of the features that have made the 8051 popular are:

- ✓ 4 KB on chip program memory.
  - ❖ 128 bytes on chip data memory(RAM)
  - ❖ 32 bytes devoted to register banks
  - ❖ 16 bytes of bit-addressable memory
  - ❖ 80 bytes of general-purpose memory
- ✓ 4 reg banks.
- ✓ 128 user defined software flags.
- ✓ 8-bit data bus
- ✓ 16-bit address bus
- ✓ 16 bit timers (usually 2, but may have more, or less).
- ✓ 3 internal and 2 external interrupts.
- ✓ Bit as well as byte addressable RAM area of 16 bytes.

- ✓ Four 8-bit ports, (short models have two 8-bit ports).
- ✓ 16-bit program counter and data pointer.
- ✓ 1 Microsecond instruction cycle with 12 MHz Crystal.

A ***16X2 LCD*** has two registers, namely, command and data. The register select is used to Switch from one register to other. RS=0 for command register, whereas RS=1 for data register.

Command Register: The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to d o a predefined task like initializing it, Clearing its screen, setting the cursor position, controlling display etc. Processing for commands happen in the command register.

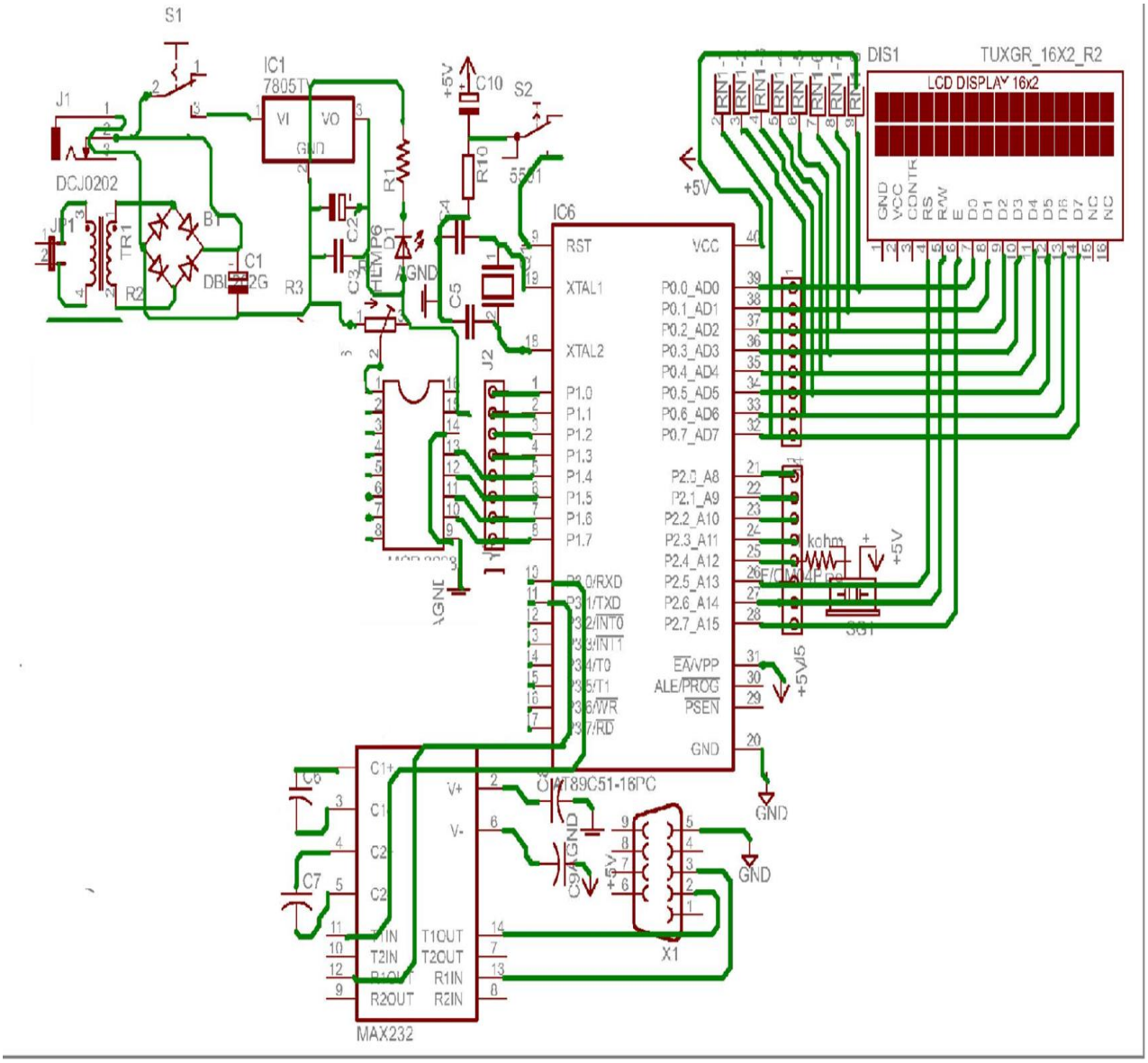
Data Register: The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD. When we send data to LCD it goes to the data register and is processed there. When RS=1, data register is selected.

A ***buzzer*** or beeper is an audio signalling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke.

A ***power supply*** is an electrical device that supplies electric power to an electrical load. The primary function of a power supply is to convert electric current from a source to the correct voltage, current, and frequency to power the load. As a result, power supplies are sometimes referred to as electric power converters. Some power supplies are separate standalone pieces of equipment, while others are built into the load appliances that they power. Examples of the latter include power supplies found in desktop computers and consumer electronics devices. Other functions that power supplies may perform include limiting the current drawn by the load to safe levels, shutting off the current in the event of an electrical fault, power conditioning to prevent electronic noise or voltage surges on the input from reaching the load, power-factor correction, and storing energy so it can continue to power the load in the event of a temporary interruption in the source power (uninterruptible power supply).

GSM (Global System for Mobile Communications), is a standard developed by the European Telecommunications Standards Institute (ETSI) to describe the protocols for second - generation digital cellular networks used by mobile devices such as tablets, first deployed in Finland in December 1991. As of 2014, it has become the global standard for mobile communications – with over 90% market share, operating in over 193 countries and territories.
2G networks developed as a replacement for first generation (1G) analog cellular networks, and the GSM standard originally described as a digital, circuit - switched network optimized for full duplex voice telephony. This expanded over time to include data communications, first by circuit-switched transport, then by packet data transport via GPRS (General Packet Radio Services) and EDGE (Enhanced Data rates for GSM Evolution, or EGPRS).

2.3.1 Circuit Diagram



COMPONENTS & HARDWARE

3.1 POWER SUPPLY

3.1.1 POWER SUPPLY BLOCK DIAGRAM

- **CONNECTOR**: 2-PIN Screw terminal is basically a connector used to interface power supply section with other circuits through wires i.e. wires are inserted in connectors.

- **BRIDGE RECTIFIER:** W10 full wave bridge rectifier is used to convert incoming AC signals into varying DC signals i.e. it consists of ripple at the output. As a result, varying DC signal is not suitable for electronic circuits unless they include a smoothing capacitor.

- **FILTER CAPACITOR:** Filter Capacitor of 2200uf at output bridge rectifier is used to provide less ripple at the output which is suitable for most electronic circuits. But their output is not regulated hence we are using regulator at the output side.

- **REGULATOR:** Regulator IC is used to provide constant DC or Smooth output with no ripple. Regulator IC mainly used are 7805, 7809, 7812. 78XX represents positive voltage regulator and last two digits represent output voltage.

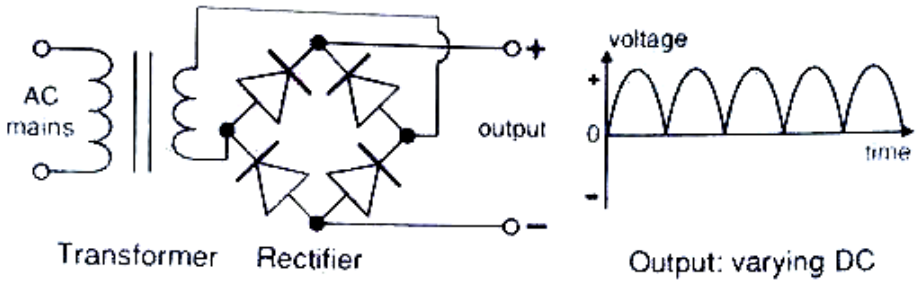
7805= +5V

7809= +9V

7812= +12V

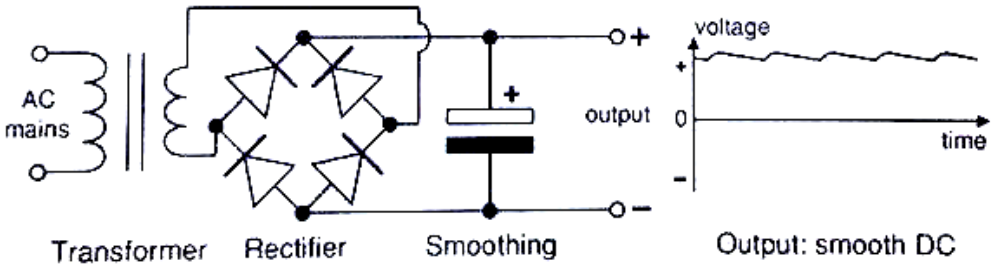
There are many types of power supply. Most are designed to convert high voltage AC mains electricity to a suitable low voltage supply for electronic circuits and other devices. A power supply can be broken down into a series of blocks, each of which performs a particular function.

- **TRANSFORMER + RECTIFIER**



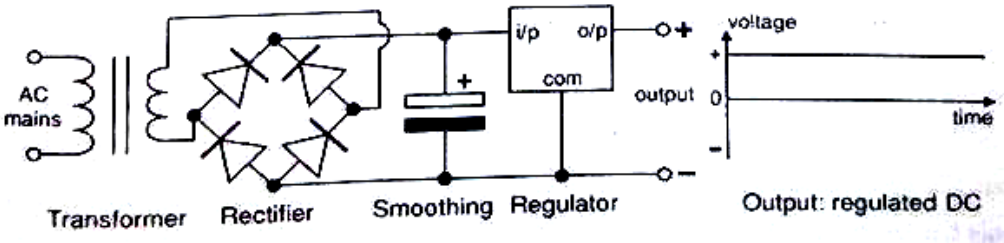
The varying DC output is suitable for lamps, heaters & standard motors. It is not suitable for electronic circuits unless they include a smoothing capacitor.

- **TRANSFORMER + RECTIFIER + SMOOTHING**



The smooth DC output has a small ripple. It is suitable for most electronic circuits.

- **TRANSFORMER + RECTIFIER + SMOOTHING + REGULATOR**



The regulated DC output is very smooth with no ripple. It is suitable for all electronic circuits.

The fig. above shows the circuit diagram of the power supply unit.

This block mainly consists of a two regulating IC 7805 and a bridge rectified and it provides a regulated supply) approximately 5V.

The transformer used in this circuit has secondary rating of 7.W.

The main function of the transformer is to step down the AC voltage available from the main. The main connections are given to its primary winding through a switch connected to a phase line.

The transformer provides a 7.5V AC output at its secondary terminals and the maximum current that can be drawn from the transformer is I Amp which is well above the required level for the circuit.

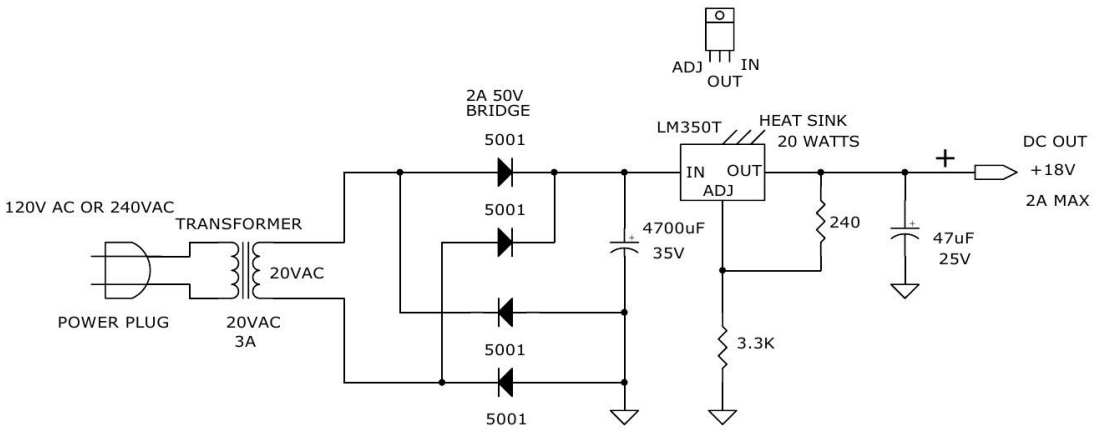
The bridge rectified the AC voltage available from the secondary of the transformer, i.e. the bridge rectifier converts the AC power available into DC power but this DC voltage available is not constant. It is a unidirectional voltage with varying amplitude.

To regulate the voltage from the bridge rectifier, capacitors are connected. Capacitors Cl filter the output voltage of the rectifier but their output is not regulated and hence 7805 is connected which is specially designed for this purpose.

Although voltage regulators can be designed using op -amps, it is quicker and easier to use IC voltage regulator. Furthermore, IC voltage regulators are available with features such as programmable output current/ voltage boosting, internal short circuit current limiting, thermal shut down and floating operation for high voltage applications.

The 78)*0C* series consists of three terminals via, input, output & ground. This is a group of fixed positive voltage regulator to give and output voltage ranging from 5V to 24V.

These IC's are designed as fixed voltage regulators and with adequate heat sinking, can delivery output current in excess of 1 Amp although these devices do not require external components and such components can be used to obtain adjustable voltage and current limiting. In addition, the difference between the input and output voltages (V in VO) called the dropout voltage must be typically 2V even from a power supply filter. Capacitors C2, C3, C4, and C5 are small filters which are used for extra filtering. LEDI& LED2 are used for Power ON indicator for ICI and IC2, current-limiting resistors R2&R4, which prevents the LED's from getting heated and thus damaged.



18V 1 AMP POWER SUPPLY

DESIGNED BY DAVE JOHNSON
DISCOVERCIRCUITS.COM

DRAWN BY: DAVE JOHNSON

DAVID JOHNSON AND ASSOCIATES		
Title: 18V 1 AMP POWER SUPPLY		
Size: A	Document Number: 18VPS1.DSN	Rev: A
Date: Sunday, July 09, 2006	Sheet 1 of 1	

3.2.1 RFID READER

Description: 125 KHz RFID reader with serial and Wiegand26 output format. Features: 1. Easy interface to computer serial terminal through DB9 connector or direct interface to microcontroller via onboard connectors. 2. Onboard buzzer and led for indicating card detection. 3. Onboard switch (SEL) for selecting Serial Wiegand26 output format or RS232 output
4. TTL RS232 signals are available on Male header & 10 pin FRC Box header. Technical Specifications: 1. Supply Voltage - 9V to 12V DC 2. Operating current - 50mA 3. Operating Frequency - 125 KHz 4. Read Distance – 5 to 10cm. Output Data format: 1. Wiegand26 (format)

Bit 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 Note P D D D D D D D D D D D D D D D D D D D D D D D P P E E E E E E E E E E E E O O O O O O O O O O O O P

Note: E: Summed for even parity O: Summed for odd parity P: Parity (even or odd) D: Data code for card: the data will use the last 24 data bits of card 2. RS232 interface format:
10 ASCII DATA (card no.) + 2 ASCII DATA (XOR result)

1. Data baud rate - 9600 bps 2. Data bit - 8bits
3. Parity Check- None 4. Stop bit - 1 Testing of RFID card reader. Connect RFID card reader to the serial port and to the supply. In Hyper terminal use settings as Baud rate “9600”, Parity “None”, RTS & CTS check on none for that. Then bring tag to sensor. You will get Tag ID. reading in Hyper terminal window.

3.2.2 RFID TAG

Description: This is a basic RFID Tag Card/Keyfob used for presence sensing, Aceess Control etc. Works in the 125kHz RF range. These tags come with a unique 32-bit ID and are not re-programmable. Card/Keyfob is blank, smooth, and mildly flexible.

Features:

1. EM4001 ISO based RFID IC.
2. 125kHz carrier
3. 2kbps ASK
4. Manchester encoding
5. 32-bit unique ID 6. 64-bit data stream [Header +ID + Data + parity]

3.3 Microcontroller

Definition: -

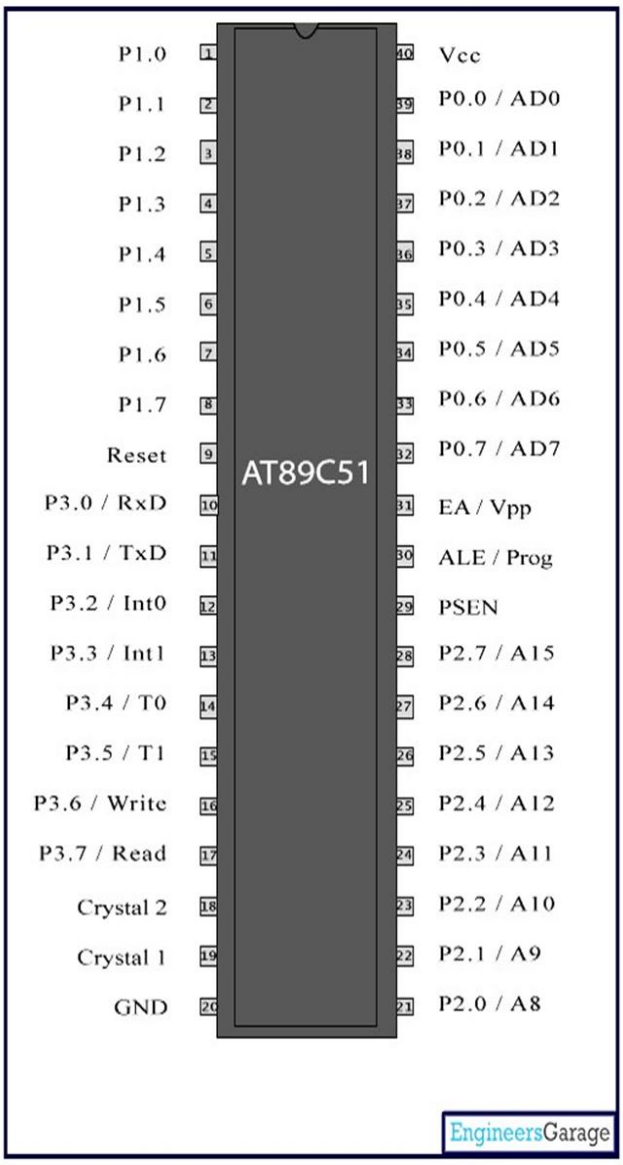
An embedded microcontroller is a chip which has a computer processor with all support functions (clock & reset). Memory (both program and data) and **I/O** (including bus interface) built into the device. These built in functions minimize the need for external circuits and devices to be designed in the final application.

3.3.1 The 89C51 microcontroller

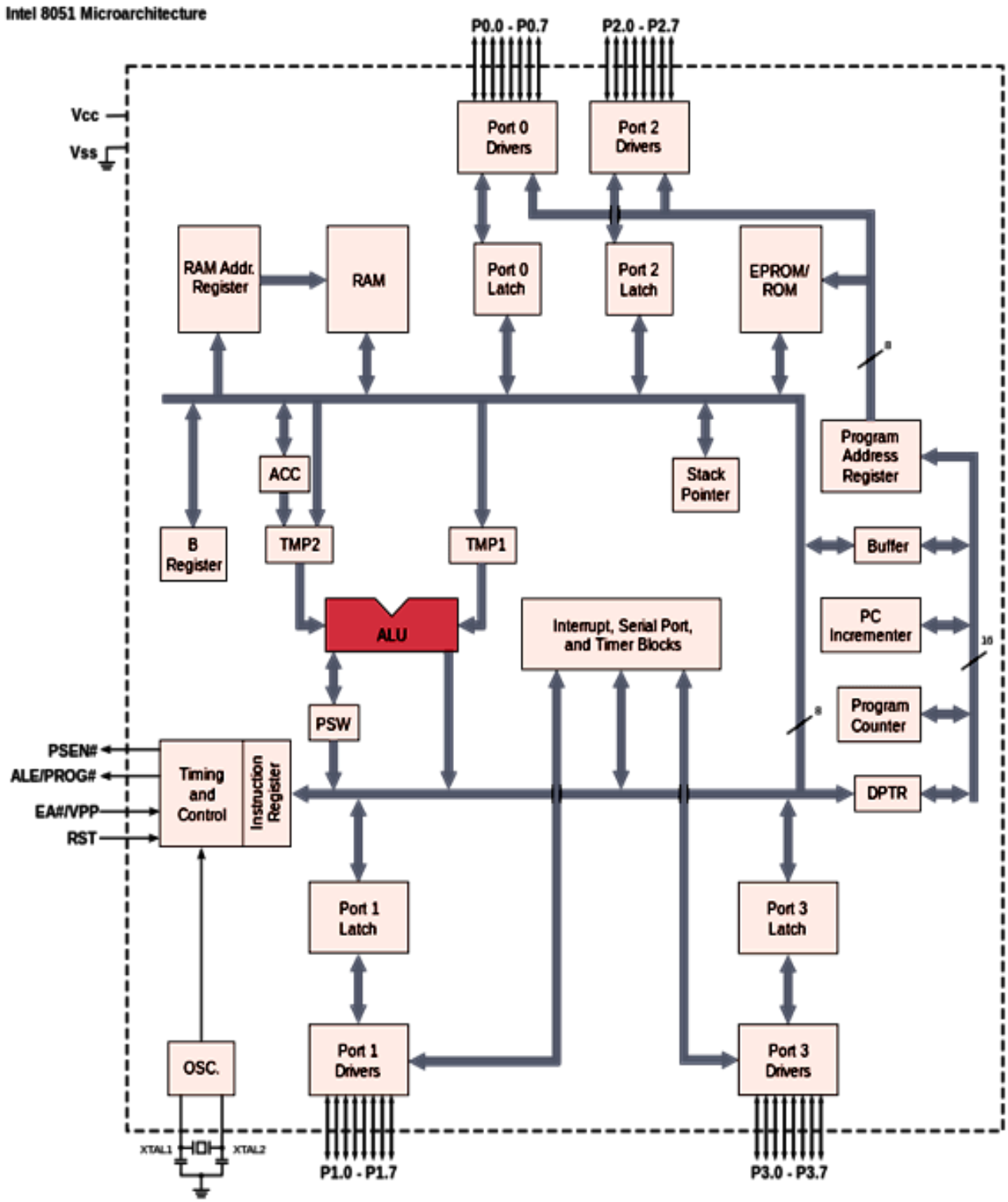
The AT89C51 is a low -power, high-performance CMOS 8 -bit microcomputer with 4K bytes of Flash programmable and erasable read only memory (PEROM). The device is manufactured using Phillips's high -density nonvolatile memory technology and is compatible with the industry - standard MCS-5 I instruction set and pin out. The on -chip Flash allows the program memory to be reprogrammed in -system or by a conventional nonvolatile memory programmer. By combining a versatile 8 -bit CPU with Flash on a monolithic chip, the Phillips AT89C51 is a powerful microcomputer which provides a highly-flexible and cost-effective solution to many embedded control applications.

The AT89C51 is designed with static logic for operation down to zero frequency and supports two Software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port and interrupt system to continue functioning. The Power -down Mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next Hardware reset.

3.3.2 Pin Diagram of 89C51



3.3.3 Architecture of 89C51



3.3.4 Features of 89C51

Following is the features of 89C51 microcontroller as per the datasheet given by Phillips-

- Compatible with MCS-51^{TM} Products
- 4K Bytes of In -System Reprogrammable Flash Memory Endurance: 1,000 Write/Erase Cycles
- Fully Static Operation: 0 Hz to 24 MHz
- Three -level Program Memory Lock
- 128 x 8 -bit Internal RAM
- 32 Programmable I/O Lines
- Two I6 -bit Timer/Counters
- Six Interrupt Sources
- Programmable Serial Channel
- Low -power Idle and Power -down Modes

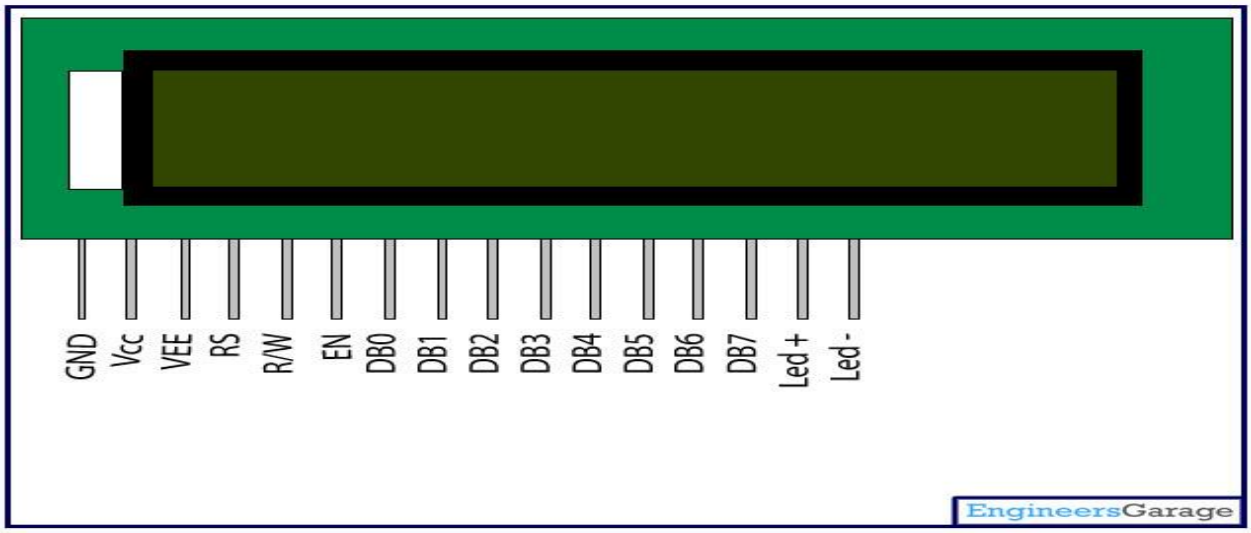
3.4 LCD 16 * 2:

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 and other multi segment LEDs. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even custom characters (unlike in seven segments), animations 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. Click to learn more about internal structure of a LCD.

Pin Diagram:



Pin Discription:

Pin No	Function	Name
1	Ground (0V)	Ground
2	Supply voltage; 5V (4.7V – 5.3V)	Vcc
3	Contrast adjustment; through a variable resistor	V_{EE}
4	Selects command register when low; and data register when high	Register Select
5	Low to write to the register; High to read from the register	Read/write
6	Sends data to data pins when a high to low pulse is given	Enable
7	8-bit data pins	DB0
8		DB1
9		DB2
10		DB3
11		DB4
12		DB5
13		DB6
14		DB7
15	Backlight V_{CC} (5V)	Led+
16	Backlight Ground (0V)	Led-

3.5 LED (Light Emitting Diode)

A Light emitting diode (LED) is essentially a pn junction diode. When carriers are injected across a forward-biased junction, it emits incoherent light. Most of the commercial LEDs are realized using a highly doped n and a p Junction.

To understand the principle, let's consider an unbiased pn+ junction. The depletion region extends mainly into the p-side. There is a potential barrier from Ec on the n-side to the Ec on the p-side, called the built-in voltage, V0. This potential barrier prevents the excess free electrons on the n+ side from diffusing into the p side.

When a Voltage V is applied across the junction, the built-in potential is reduced from V0 to V0 – V. This allows the electrons from the n+ side to get injected into the p-side. Since electrons are the minority carriers in the p-side, this process is called minority carrier injection. But the whole injection from the p side to n+ side is very less and so the current is primarily due to the flow of electrons into the p-side.

These electrons injected into the p-side recombine with the holes. This recombination (see Appendix 1) results in spontaneous emission of photons (light). This effect is called injection electroluminescence. These photons should be allowed to escape from the device without being reabsorbed.

The recombination can be classified into the following two kinds

- Direct recombination
- Indirect recombination

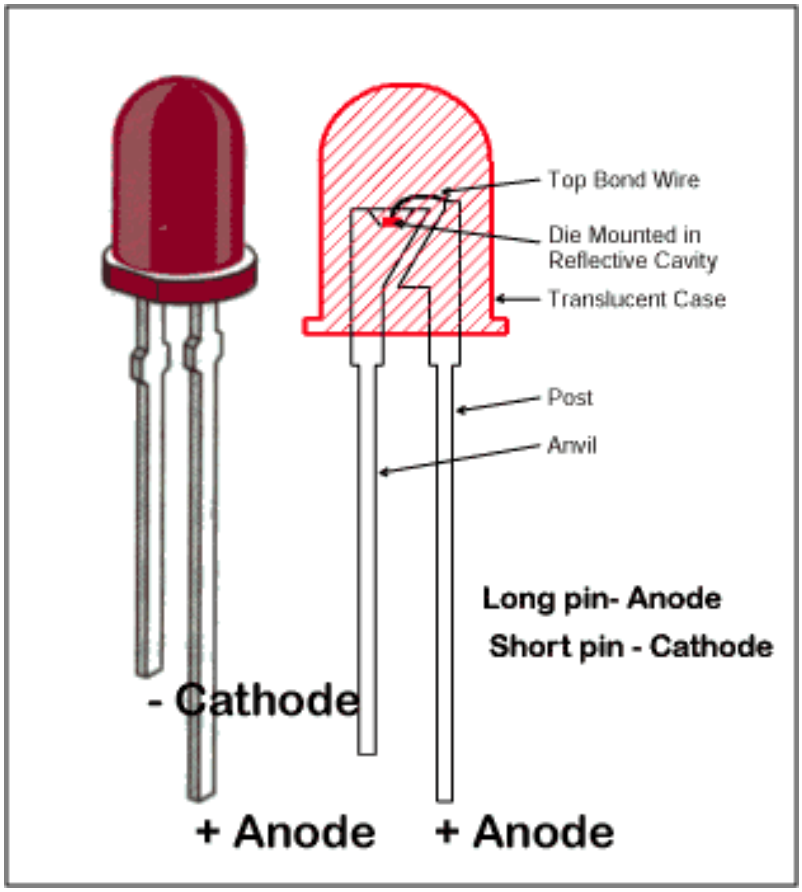
Direct Recombination:

In direct band gap materials, the minimum energy of the conduction band lies directly above the maximum energy of the valence band in momentum space energy. In this material, free electrons at the bottom of the conduction band can recombine directly with free holes at the top of the valence band, as the momentum of the two particles is the same. This transition from conduction band to valence band involves photon emission (takes care of the principle of energy conservation). This is known as direct recombination. Direct recombination occurs spontaneously. GA As is an example of a direct band-gap material.

Indirect Recombination:

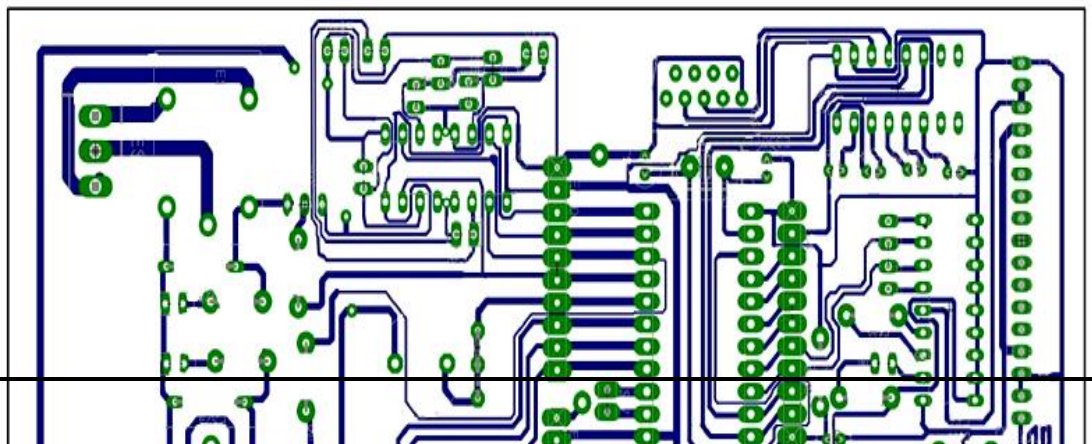
In the indirect band gap materials, the minimum energy in the conduction band is shifted by a k-vector relative to the valence band. The k-vector difference represents a difference in momentum. Due to this difference in momentum, the probability of direct electron hole recombination is less.

In these materials, additional dopants (impurities) are added which form very shallow donor states. These donor states capture the free electrons locally; provides the necessary momentum shift for recombination. These donor states serve as the recombination centers. This is called indirect (non-radiative) Recombination.



PCB Layout

4.1 PCB Layout

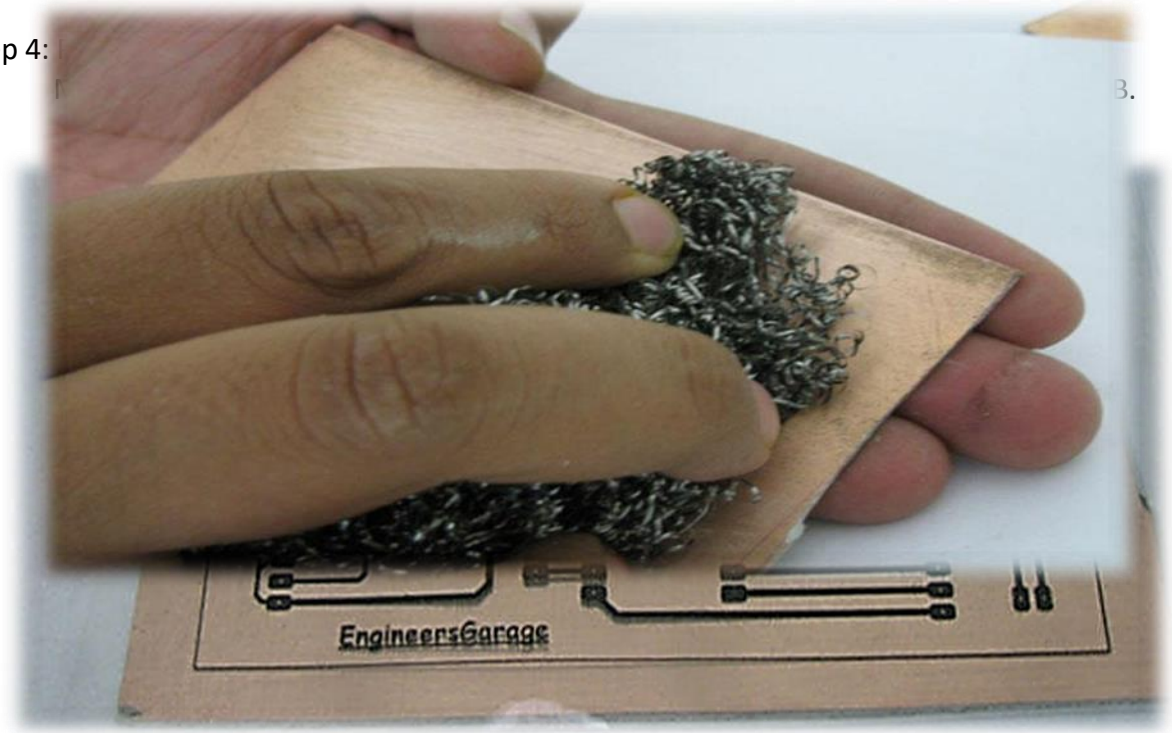


Step 1: Prepare a layout of the circuit on any commonly used PCB designing software. A layout is a design which interconnects the components according to the schematic Diagram (circuit diagram). Take a mirror image print of the layout on the OHP sheet using a laser printer. Make sure that the design is correct with proper placement of the components.

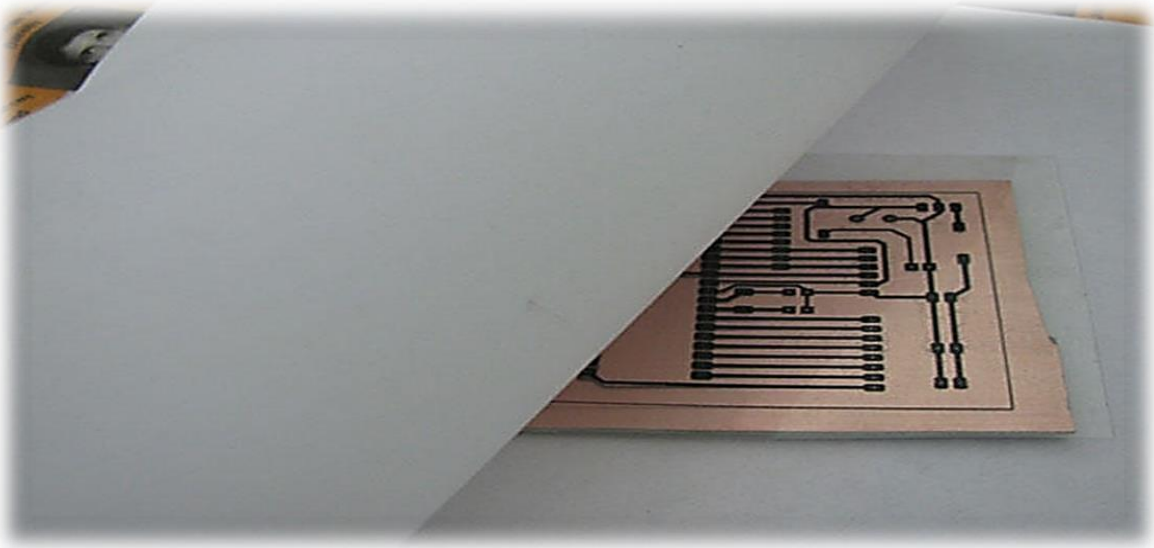
Step 2: Cut the copper board according to the size of layout. A copper board is the base of a PCB, it can be single layer, double layer or multi layer board. Single layer copper board has copper on one side of the PCB, they are used to make single layer PCBs, it is widely used by hobbyist or in the small circuits. A double layer copper board consists of copper on both the sides of the PCB. These boards are generally used by the industries. A multilayer board has multiple layers of copper; they are quite costly and mainly used for complex circuitries like mother board of PC.

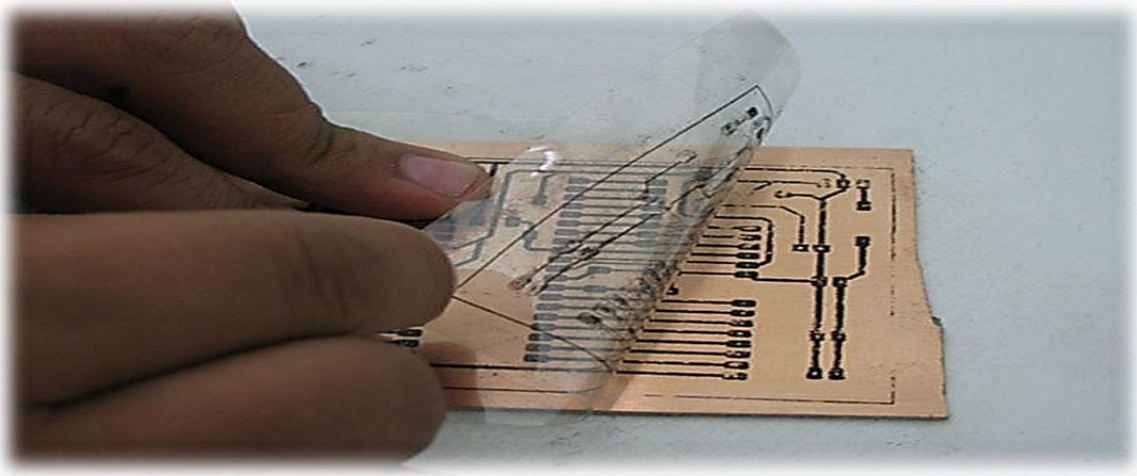
Step 3: Rub the copper side of PCB using steel wool. This removes the top oxide layer of copper as well as the photo resists layer if any.

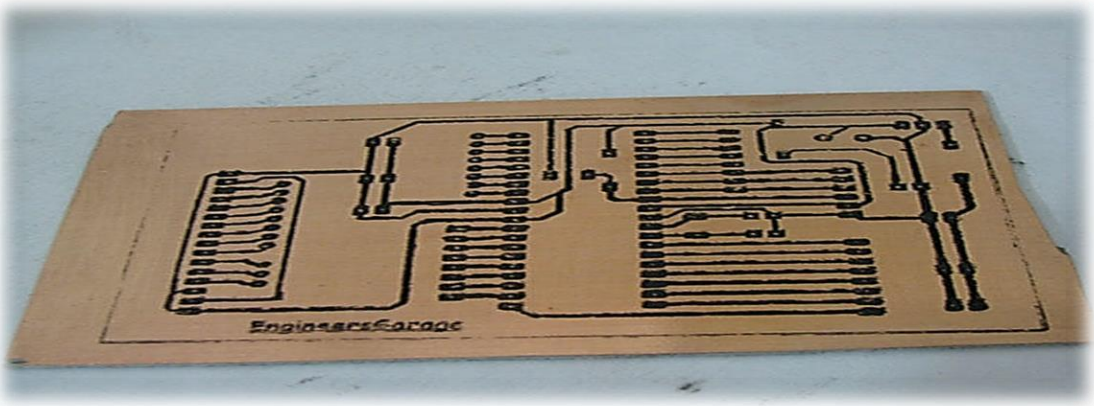
Step 4:



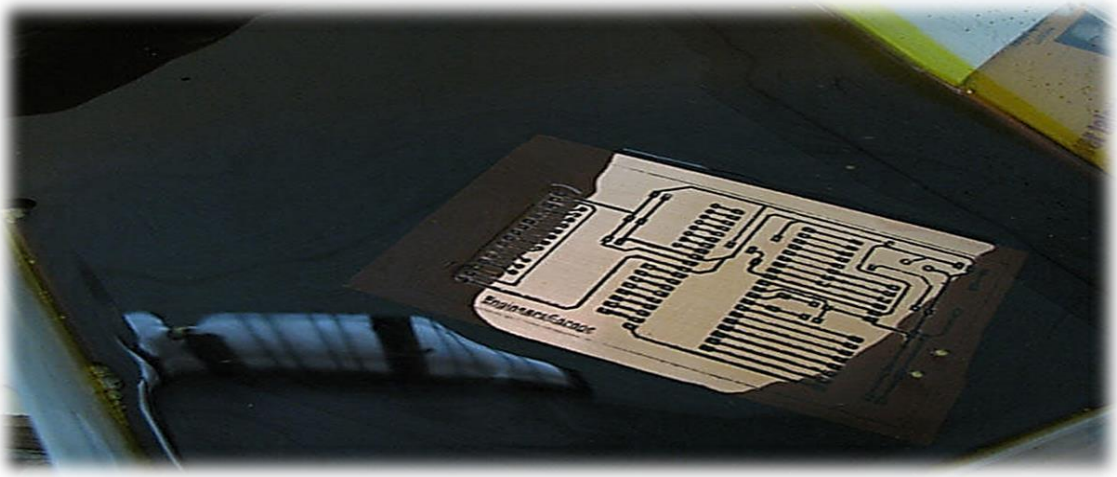
Step 5: Put a white paper on the OHP sheet and start ironing. The heat applied by the electric iron causes the ink of the traces on the OHP sheet to stick on the copper plate exactly in the same way it is printed on the OHP sheet. This means that the copper sheet will now have the layout of the PCB printed on it. Allow the PCB plate to cool down and slowly remove the OHP sheet. Since it is manual process it may happen that the layout doesn't comes properly on PCB or some of the tracks are broken in between. Use the permanent marker and complete the tracks properly.



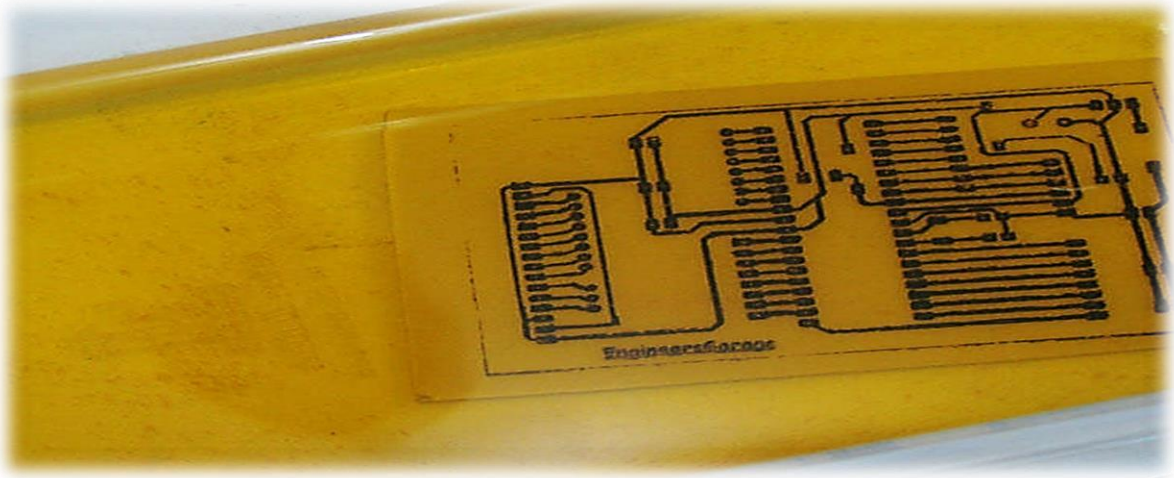


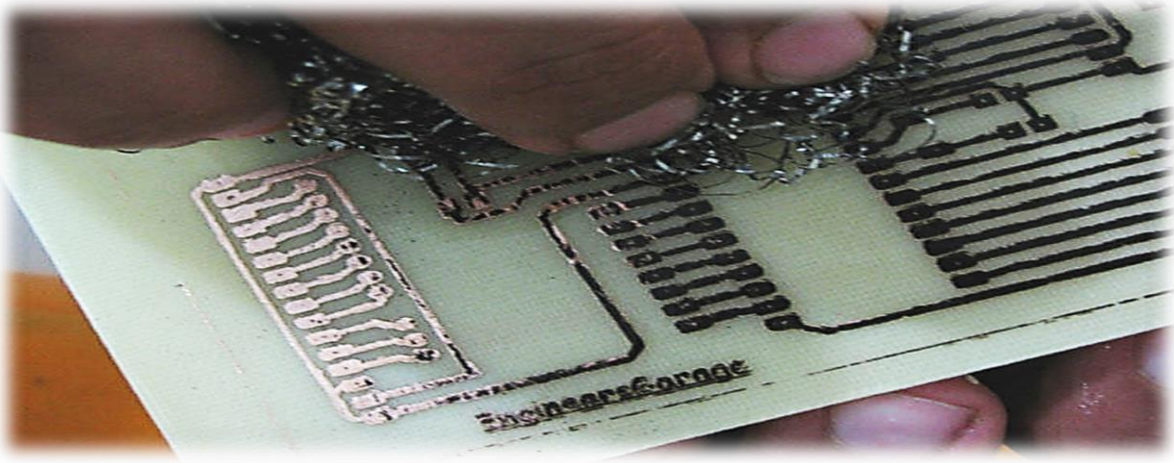


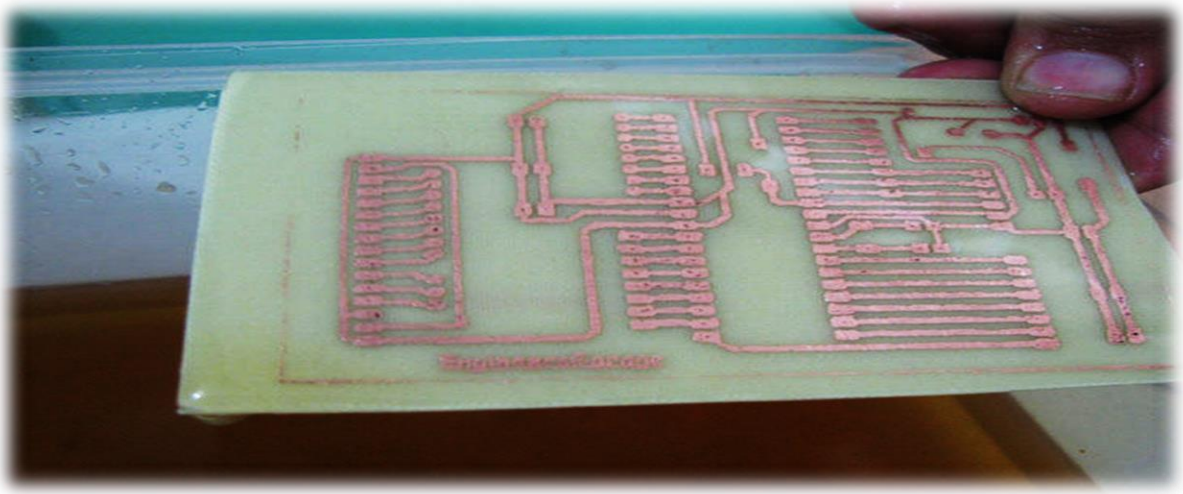
Step 6: Now the layout is printed on PCB. The area covered by ink is known as the masked area and the unwanted copper, not covered by the ink is known as unmasked area. Now make a solution of ferric chloride. Take a plastic box and fill it up with some water. Dissolve 2-3 tea spoon of ferric chloride power in the water. Dip the PCB into the Etching solution (Ferric chloride solution, Fecl3) for approximately 30 mins. The Fecl3 reacts with the unmasked copper and removes the unwanted copper from the PCB. This process is called as Etching. Use pliers to take out the PCB and check if the entire unmasked area has been etched or not. In case it is not etched leave it for some more time in the solution.



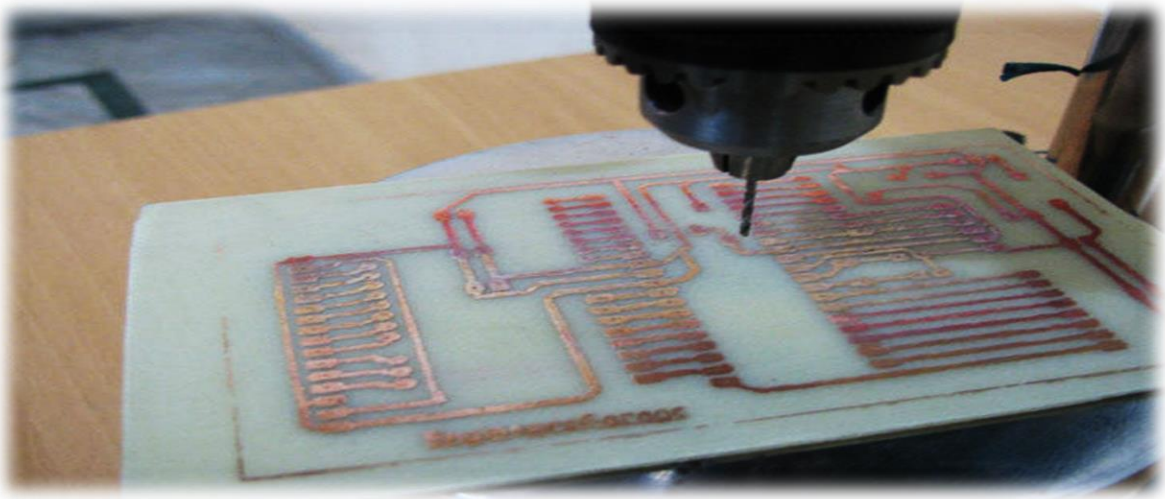
Step 7: Take out the PCB wash it in cold water and remove the ink by rubbing it with steel wool. The remaining area which has not been etched is the conductive copper tracks which connect the components as per the circuit diagram.







Step 8: Now carefully drill the PCB using a drilling machine on the pads.



Step 9: Put the components in the correct holes and solder them. This completes your PCB fabrication now put the components on mounting side and solder them. Make sure that you properly dispose of the FeCl3 solution, clean your tools and wash your hands after this exercise. You can also store the solution in a plastic box for future use but not for too long.

SOFTWARE DESCRIPTION

5.1 C language basic

C Language Introduction

C is a procedural programming language. It was initially developed by Dennis Ritchie between 1969 and 1973. It was mainly developed as a system programming language to write operating system. The main features of C language include low-level access to memory, simple set of keywords, and clean style, these features make C language suitable for system programming like operating system or compiler development. Many later languages have borrowed syntax/features directly or indirectly from C language. Like syntax of Java, PHP, JavaScript and many other languages is mainly based on C language. C++ is nearly a superset of C language (There are few programs that may compile in C, but not in C++).

Beginning with C programming:

1) Finding a Compiler:
Before we start C programming, we need to have a compiler to compile and run our programs. There are certain online compilers like https://ide.geeksforgeeks.org/, http://ideone.com/ or http://codepad.org/ that can be used to start C without installing a compiler.

Windows: There are many compilers available freely for compilation of C programs like Code Blocks and Dev-CPP. We strongly recommend Code Blocks.

Linux: For Linux, gcc comes bundled with the linux, Code Blocks can also be used with Linux.

2) Writing first program:

Following is first program in C

```
#include <stdio.h>
int main(void)
{
   printf("GeeksQuiz");
   return 0;
}
```

Output:

```
GeeksQuiz
```

Let us analyze the program line by line.
Line 1: [#include <stdio.h>] In a C program, all lines that start with **#** are processed by preprocessor which is a program invoked by the compiler. In a very basic term, preprocessor takes a C program and produces another C program. The produced program has no lines starting with #, all such lines are processed by the preprocessor. In the above example, preprocessor copies the preprocessed code of stdio.h to our file. The .h files are called header files in C. These header files generally contain declaration of functions. We need stdio.h for the function printf() used in the program.

Line 2 [int main(void)] There must to be starting point from where execution of compiled C program begins. In C, the execution typically begins with first line of main(). The void written in brackets indicates that the main doesn't take any parameter (See this for more details). main() can be written to take parameters also. We will be covering that in future posts. The int written before main indicates return type of main(). The value returned by main indicates status of program termination. See this post for more details on return type.

Line 3 and 6: [{ and }] In C language, a pair of curly brackets define a scope and mainly used in functions and control statements like if, else, loops. All functions must start and end with curly brackets.

Line 4 [printf("GeeksQuiz");] printf() is a standard library function to print something on standard output. The semicolon at the end of printf indicates line termination. In C, semicolon is always used to indicate end of statement.

Line 5 [return 0;] The return statement returns the value from main(). The returned value may be used by operating system to know termination status of your program. The value 0 typically means successful termination.

5.2 Keil compiler

C51 C Compiler

In 2004, MATLAB had around one million users across industry and academia. MATLAB users come from various backgrounds of engineering, science, and economics.

The Keil C51 C Compiler for the 8051 microcontroller is the most popular 8051 C compiler in the world. It provides more features than any other 8051 C compiler available today.

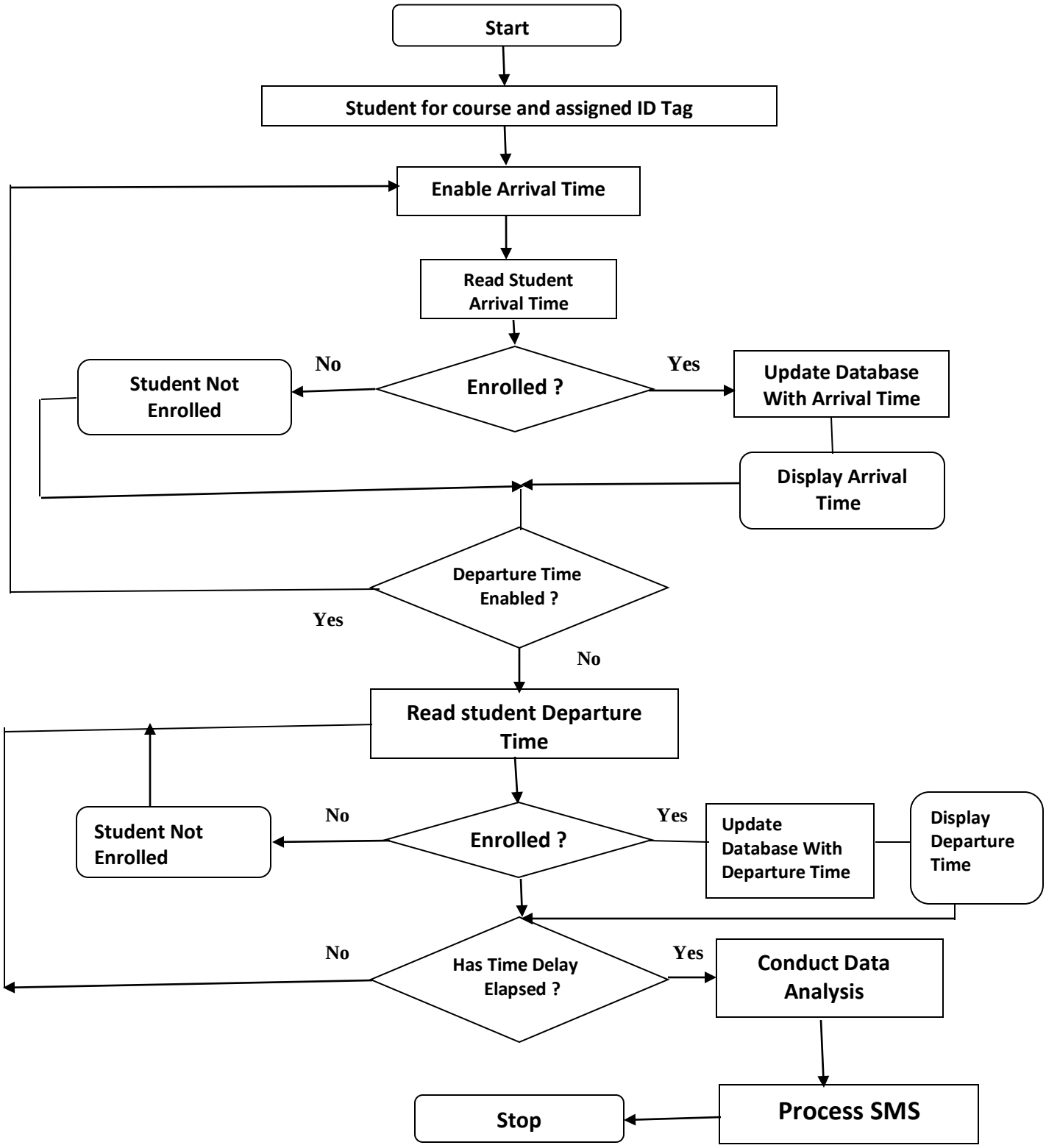
The C51 Compiler allows you to write 8051 microcontroller applications in C that, once compiled, have the efficiency and speed of assembly language. Language extensions in the C51 Compiler give you full access to all resources of the 8051.

The C51 Compiler translates C source files into relocatable object modules which contain full symbolic information for debugging with the µVision Debugger or an in-circuit emulator. In addition to the object file, the compiler generates a listing file which may optionally include symbol table and cross reference information.

Features

- Nine basic data types, including 32-bit IEEE floating-point,
- Flexible variable allocation with **bit**, **data**, **bdata**, **idata**, **xdata**, and **pdata** memory types,
- Interrupt functions may be written in C,
- Full use of the 8051 register banks,
- Complete symbol and type information for source-level debugging,
- Use of **AJMP** and **ACALL** instructions,
- Bit-addressable data objects,
- Built-in interface for the RTX51 Real-Time Kernel,
- Support for dual data pointers on Atmel, AMD, Cypress, Dallas Semiconductor, Infineon, Philips, and Triscend microcontrollers,
- Support for the Philips 8*x*C750, 8*x*C751, and 8*x*C752 limited instruction sets,
- Support for the Infineon 80C517 arithmetic unit.

5.3.Flowchart



A careful observation of the trend of usage of RFID tags leads one to consider the possibility of its utilization for monitoring the attendance of students in educational institutions, with the aid of program driven computers. While every student given a specific RFID tag attends the lecture through entrance door, a serial number (related to each student's matriculation number) of tag is associated with the student database entry. So every time a student uses his/her card, the entries will be entered into the database with the time stamp. The use of webcam might be optionally necessary to take a snap of the person using the card. Webcam reduces proxy attendance attempts. This is used to cross-verify in the event of an undesirable event or dispute. Consequently, the attendance data then can be used to create many types of reports like daily attendance details, monthly, weekly and real time feedback to parents. The attendance score calculation can be automated using the collected data. After setting up the student attendance RFID system from the mode of operation depicted in the circuit.

The tag is activated when it passes through a radio frequency (RF) field (125 kHz in this case), which is generated by the antenna embedded within the reader box. The program checks whether the tag is valid or not. If the tag is valid, it will continue to the database program and registers the student's attendance for the course. If the tag is invalid, the program gives a notification that the tag has not been registered to any student and requires the user to either supply a valid tag.

Due to the reason of cost and flexibility of implementation, this RFID attendance design application uses a passive tag and thus for every class, students would have to bring their tags close to the reader (about 10 cm from the reader). On doing this, the reader reads the tag and the application program records the student's arrival time and when leaving the class, students will also have to bring their tags close to the reader again. With records of arrival and exit time, appropriate short message service is forwarded to the parents' mobile phone number in real time or as a weekly sms/email digest through the SMS/EMAIL gateway as shown in Figure 1.0. Each course lecturer has RFID tag that serves as the control for the beginning and end of classroom lecture with additional time delay for end of class activation to allow every student to record exit time on the reader. The lecturer/instructor can call for information over any student by using queries provided by the application. More flexibility and unconscious interaction of students to the developed system can be achieved by using active tags. This will increase the overall cost of the system.

At the end of the semester, the lecturer can grade students attendance scores in a particular course based on some specific metrics provided in the application. The selected metrics could be frequency of presence in class, duration of stay in class, punctuality, etc. A privileged user can de-assign students from their specific tag, and reassign the tag to other students if need be as shown in the Graphical User Interfaces (GUI's) of the RFID system application control program

SOFTWARE IMPLEMENTATION

6.1 Keil uVision PROGRAMMING

6.1.1 Keil uVision Programming for Smart School Management System

```
#include<headers.h>
#include "string.h"

uchar code card_id1[] = {"4500413D2E17"};  //Student card 1
uchar code card_id2[] = {"4500416782E1"};  //2
uchar code card_id3[] = {"45005CEF47B1"};  //3
uchar code card_id4[] = {"4500416AA3CD"};  //Achutha4
//uchar code card_id5[] = {"5200B0A48FC1"};
//uchar code card_id6[] = {"5200B0E45252"};
//uchar code card_id7[] = {"5200B0AB3173"};
//uchar code card_id8[] = {"5200B1518C34"};
//uchar code card_id9[] = {"5200B0A48FC5"};
//uchar code card_id10[]= {"5200B0E45256"};

uchar code card_id11[] = {"4500413C625A"};             // Teacher card  HOD
uchar code card_id12[] = {"45004150E6B2"};             //Anoop


uchar CardNoDigit = 0,Teacher1 = 0,Teacher2 = 0;
uchar receive_buffer[12] = {0};
bit card_detected  = 0,object_flag=0;
uchar card_number = 0;
bit card1 = 0, card2 = 0,card3 = 0, card4 = 0,card5 = 0,card6 = 0,card7 =
0,card8 = 0,card9 = 0,card10 = 0,card11 = 0,card12 = 0;
sbit object_sensor = P3^7;
sbit garbage_level_detector   =P1^1;
sbit Motion_sensor= P1^0;
sbit light = P2^3;
sbit tap= P2^2;

//sbit waterpump = P2^2;

sbit buzzer = P2^4;

//sbit street_light = P1^7;
//sbit moisture_sensor = P1^3;

void delay_ms1(unsigned int c1)     //for 11.059Mhz
{
unsigned c2;
for(;c1>0;--c1)
for(c2=0;c2<115;++c2)

//for(c3=0;c3<10;++c3)
}
void initialise_signal();
void auto_parking();
void serial_interrupt(); interrupt 4
```

```
{
if(RI)
{
receive_buffer[CardNoDigit] = SBUF;
CardNoDigit++;
if(CardNoDigit>=12)
{
CardNoDigit=0;
card_detected = 1;
            }
RI = 0;
      }
}
uchar check_card()               //according to recieving card no by serial
(uart) card no is generated
{
uchar j = 0,k = 0;
bit a = 0;
for(j=0;j<12;j++)     // From 1st to 10 loop is for to get student card
{
if(receive_buffer[j] != card_id1[j]) //not equal to
{
a = 0;                                  //break with a=0;
break;
       }
else
      {
a = 1;                          //if both card no same a=1
            }
      }
if(a)
      {
k = 1;                   //k is actual indicate the card no's
      }
else                     //after card 1 is not match jump to this routing
      {
for(j=0;j<12;j++)        // Loop to check the ID of the second card
            {
if(receive_buffer[j] != card_id2[j])
      {
a = 0;
break;
            }
else                     //if both card no same a=1
{
a = 1;

            }
            }
if(a)                           //if both card no same a=1
            {
k = 2;                          //k is actual indicate the card no's
            }

      }
```

```
for(j=0;j<12;j++)     // Loop to check the ID of the first card
      {
if(receive_buffer[j] != card_id3[j]) //not equal to
            {
a = 0;                                          //break with a=0;
break;
            }
else
            {
a = 1;                        //if both card no same a=1
            }
      }
if(a)
      {
k = 3;                              //k is actual indicate the card no's
      }
else                        //after card 1 is not match jump to this routing
      {
for(j=0;j<12;j++)       // Loop to check the ID of the second card
            {
if(receive_buffer[j] != card_id4[j])
            {
a = 0;
break;
            }
else                        //if both card no same a=1
            {
a = 1;
            }
            }
if(a)                           //if both card no same a=1
            {
k = 4;                          //k is actual indicate the card no's
            }
            }
//    for(j=0;j<12;j++)     // Loop to check the ID of the first card
//    {
//    if(receive_buffer[j] != card_id5[j]) //not equal to
//          {
//    a = 0;       //break with a=0;
//    break;
//          }
//    else
//          {
//    a = 1;      //if both card no same a=1
//          }
//    }
//    if(a)
//    {
//          k = 5; //k is actual indicate the card no's
//    }
//    else     //after card 1 is not match jump to this
             routing
//    {
//    for(j=0;j<12;j++)       // Loop to check the ID of the second card
```

```
//    {
//    if(receive_buffer[j] != card_id6[j])
//    {
//    a = 0;
//    break;
//          }
//    else  //if both card no same a=1
//          {
//    a = 1;
//          }
//          }
//    if(a) //if both card no same a=1
//          {
//    k = 6;       //k is actual indicate the card no's
//          }
//    }
//
//    for(j=0;j<12;j++)     // Loop to check the ID of the first card
//    {
//          if(receive_buffer[j] != card_id7[j]) //not equal to
//          {
//    a = 0;                                          //break with a=0;
//                break;
//          }
//          else
//          {
//                a = 1;                        //if both card no same a=1
//          }
//    }
//    if(a)
//    {
//    k = 7;                   //k is actual indicate the card no's
//    }
//    else              //after card 1 is not match jump to this routing
//    {
//    for(j=0;j<12;j++)       // Loop to check the ID of the second card
//          {
//    if(receive_buffer[j] != card_id8[j])
//          {
//    a = 0;
//    break;
//          }
//    else                    //if both card no same a=1
//          {
//    a = 1;
//          }
//          }
//    if(a)                   //if both card no same a=1
//          {
//    k = 8;                  //k is actual indicate the card no's
//          }
//    }
//                break;
//          }
//          else
```

```
//          {
//    a = 1;                        //if both card no same a=1
//          }
//    }
//    if(a)
//    {
//    k = 9;                    //k is actual indicate the card no's
//    }
//    else             //after card 1 is not match jump to this routing
//    {
//    for(j=0;j<12;j++)       // Loop to check the ID of the second card
//          {
//    if(receive_buffer[j] != card_id10[j])
//                {
//    a = 0;
//    break;
//                }
//    else                        //if both card no same a=1
//                {
//    a = 1;
//                }
//                }
//    if(a)                         //if both card no same a=1
//          {
//    k = 10;                       //k is actual indicate the card no's
//          }
//    }
for(j=0;j<12;j++)     // For Teacher card recived
      {
if(receive_buffer[j] != card_id11[j]) //not equal to
            {
a = 0;                    //break with a=0;
break;
            }
else
            {
a = 1;                    //if both card no same a=1
            }
      }
      if(a)
      {
k = 11;                   //k is actual indicate the card no's
      }
else                      //after card 1 is not match jump to this routing
      {
for(j=0;j<12;j++)         // For Teacher card recived
            {
if(receive_buffer[j] != card_id12[j])
      {
a = 0;
break;
```

```
}
else                    //if both card no same a=1
{
a = 1;
      }
      }
if(a)                   //if both card no same a=1
      {
k = 12;                 //k is actual indicate the card no's
      }
      }
return k;

}
void beep()
{
      buzzer=0;
      delayl();
            delayl();
      delayl();
      delayl();

      buzzer=1;
}
void update_data( uchar no )  // update bit filds for recived card no
      {
      switch(no)
            {
            case 1:
                  card1 = 1;                //card no 1 scan
            break;

            case 2:
                  card2 = 1;                //card no 2 scan
            break;

            case 3:
                  card3 = 1;                //card no 3 scan
            break;

            case 4:
                  card4 = 1;                //card no 4 scan
            break;
//          case 5:
//                card5 = 1;                //card no 4 scan
//          break;
//
```

```
//          case 6:
//                card6 = 1;                //card no 4 scan
//          break;
//
//          case 7:
//                card7 = 1;                //card no 4 scan
//          break;
//
//          case 8:
//                card8 = 1;                //card no 4 scan
//          break;
//
//          case 9:
//                card9 = 1;                //card no 4 scan
//          break;
//
//          case 10:
//                card10 = 1;               //card no 4 scan
//          break;

            case 11:
                  card11 = 1;               //card no 4 scan
            break;

            case 12:
                  card12 = 1;               //card no 4 scan
            break;

             default:
card1 =card2 =card3 =card4 =card5 =card6 =card7 =card8 =card9 =card10 =card11
=card12 = 0;
            break;
            }
      }
//void send_sms(unsigned char *str)
//{
//
//unsigned  char buff[30],i;
//for(i=0;i<29;i++)
//    buff[i] = ' ';
//strcpy(buff,str);
//    transmit_string("AT+CMGS=\"09594263347\"\r\n");
//    delayl();
//    transmit_string(&buff);
//    txdata(0x1a);
//    transmit_string("\r\n");
//}
//
//void send_sms_2(unsigned char *str)
//{
```

```
//
//unsigned  char buff[30],i;
//for(i=0;i<29;i++)
//    buff[i] = ' ';
//strcpy(buff,str);
//    transmit_string("AT+CMGS=\"09768456038\"\r\n");
//    delayl();
//    transmit_string(&buff);
//    txdata(0x1a);
//    transmit_string("\r\n");
//}
//void send_sms_3(unsigned char *str)
//{
//
//unsigned  char buff[30],i;
//for(i=0;i<29;i++)
//    buff[i] = ' ';
//strcpy(buff,str);
//    transmit_string("AT+CMGS=\"09867109360\"\r\n");
//    delayl();
//    transmit_string(&buff);
//    txdata(0x1a);
//    transmit_string("\r\n");
//}

//void send_sms_4(unsigned char *str)
//{
//
//unsigned  char buff[30],i;
//for(i=0;i<29;i++)
//    buff[i] = ' ';
//strcpy(buff,str);
//    transmit_string("AT+CMGS=\"09594263347\"\r\n");
//    delayl();
//    transmit_string(&buff);
//    txdata(0x1a);
//    transmit_string("\r\n");
//}
// void display_bus_information()
// {
//    lcdcmd(0x01);
//    delayl();
//    card_detected = 0;
//    lcdstr("Card detected: ");
// }
void InitialiseTimer()
{
TMOD |= 0x01;
      TL0 = 0x00;
      TH0 = 0xDC;
      TR0 = 1;
```

```
}
unsigned char
TenMs,HundredMS,F_OneSec,OneSec,F_FiveSec,F_TenSec,F_TeacherPresent,NoOfLectu
res;
uchar
temp1,temp2,temp3,temp4,temp5,temp6,temp7,temp8,temp9,temp10,temp11,temp12,
logintime;
void Timer0() interrupt 1

      {
      TF0 = 0 ;
      TL0 = 0x00;
      TH0 = 0xDC;
      TR0 = 1;

      TenMs++;
      if(TenMs>=10)
      {
       HundredMS++;
       TenMs = 0;

       if(HundredMS>=10)
            {
                  F_OneSec= 1;
                  HundredMS = 0;
            }
      }

                   }

                   void delay22(unsigned char kk)
                  {
      unsigned int ll,mm;
      for(ll = 0;ll<=kk;ll++)
            for(mm=0;mm<1111;mm++)
            {}

                   }
      void main()
                  {
      init_serial();
      InitialiseTimer();
      init_lcd();
      IE = 0x92;
      delay22(200);
```

```
transmit_string("AT+CMGF=1"); // set the SMS mode to text
 txdata('\r');
 txdata('\n');
 delay22(200);
 delay22(200);
 delay22(200);
 transmit_string("AT+CMGS=\"+917738520827\"");  //AT+CMGS=”xxxxxxxxxx”
  // xxxxxxxxxx is the phone number. after this
 txdata('\r');
 txdata('\n');
 delay22(200);
 delay22(200);
 delay22(200);
 transmit_string("Laxman Today Absent");
 txdata(0x1A);
 delay22(200);
 txdata(0x1A);
 }
 if(temp2 == 0)
            {
 lcdcmd(0x01);
 lcdstr("Anoop Absent");                    //send_sms();
 transmit_string("AT");
 txdata('\r');
 txdata('\n');
 delay22(200);
 delay22(200);
 delay22(200);
 delay22(200);
 transmit_string("AT+CMGF=1"); // set the SMS mode to text
 txdata('\r');
 txdata('\n');
 delay22(200);
 delay22(200);
 delay22(200);
 delay22(200);
 transmit_string("AT+CMGS=\"+917738520827\"");  //AT+CMGS=”xxxxxxxxxx”
 // xxxxxxxxxx is the phone number. after this
 txdata('\r');
 txdata('\n');
 delay22(200);
 delay22(200);
 transmit_string("Anoop Today Absent");
 txdata(0x1A);
 delay22(200);
 txdata(0x1A);
                  }
 if(temp3 == 0)
            {
```

```
lcdcmd(0x01);
lcdstr("Achuta Absent");    //send_sms();
transmit_string("AT");
txdata('\r');
txdata('\n');
delay22(200);
delay22(200);
delay22(200);
delay22(200);
delay22(200);
delay22(200);
transmit_string("AT+CMGF=1"); // set the SMS mode to text
txdata('\r');
txdata('\n');
delay22(200);
delay22(200);
delay22(200);
delay22(200);
transmit_string("AT+CMGS=\"+917738520827\"");  //AT+CGS="xxxxxxxxxx"
// xxxxxxxxxx is the phone number. after this
txdata('\r');
txdata('\n');
delay22(200);
delay22(200);
delay22(200);
delay22(200);
transmit_string("Achuta Today Absent");
txdata(0x1A);
delay22(200);
delay22(200);
delay22(200);
txdata(0x1A);
             }
if(temp4== 0)
          {
lcdcmd(0x01);
lcdstr("Hasan Absent");                 //send_sms();
transmit_string("AT");
txdata('\r');
txdata('\n');
delay22(200);
delay22(200);
delay22(200);
delay22(200);
transmit_string("AT+CMGF=1"); // set the SMS mode to text
txdata('\r');
txdata('\n');
delay22(200);
delay22(200);
delay22(200);
delay22(200);
```

```
transmit_string("AT+CMGS=\"+917738520827\"");  //AT+CMGS=”xxxxxxxxxx”
// xxxxxxxxxx is the phone number. after this
txdata('\r');
txdata('\n');
delay22(200);
delay22(200);
delay22(200);
delay22(200);
transmit_string("Hasan Today Absent");
txdata(0x1A);
delay22(200);
txdata(0x1A);
          }
          }
F_FiveSec = 1;
          }
OneSec = 0;

          }
if(NoOfLectures>=5)
          {
F_TenSec = F_TenSec+1;
F_TeacherPresent = 0;
F_FiveSec = 0;
OneSec = 0;
             }
if(F_TenSec>=15)
          {
lcdcmd(0x01);

F_TenSec  = 0;
if(temp1%2 != 0 )
                {
lcdstr("Laxman no Logout");       //send_sms();
//send_sms("Archit Today Not Loged Out");
transmit_string("AT");
                                    txdata('\r');
                                    txdata('\n');
                                    delay22(200);
                                    delay22(200);
                              delay22(200);
                                    delay22(200);
transmit_string("AT+CMGF=1"); // set the SMS mode to text
                                    txdata('\r');
                                    txdata('\n');
                                    delay22(200);
                                    delay22(200);
transmit_string("AT+CMGS=\"+917773971399\"");  //AT+CMGS=”xxxxxxxxxx”
// xxxxxxxxxx is the phone number. after this
                                     txdata('\r');
```

```
txdata('\n');
delay22(200);
delay22(200);

transmit_string("Laxman no Logout");
                              txdata(0x1A);
                              delay22(200);
                              txdata(0x1A);
                  }
                  if(temp2%2 != 0)
                  {
lcdcmd(0x01);
lcdstr("Anoop no Logt");            //send_sms();
                              transmit_string("AT");
                              txdata('\r');
                              txdata('\n');
                              delay22(200);
                              delay22(200);
                         delay22(200);
                              delay22(200);
transmit_string("AT+CMGF=1"); // set the SMS mode to text
                              txdata('\r');
                              txdata('\n');
                              delay22(200);
                              delay22(200);
transmit_string("AT+CMGS=\"+917773971399\"");  //AT+CMGS="xxxxxxxxxx"
// xxxxxxxxxx is the phone number. after this
                               txdata('\r');
                              txdata('\n');
                           delay22(200);
                              delay22(200);
                              delay22(200);
                              delay22(200);
 transmit_string("Anoop no Logout");
                              txdata(0x1A);
                              delay22(200);
                    txdata(0x1A);
                  }
                  if(temp3%2 != 0)
                  {
                         lcdcmd(0x01)
  lcdstr("Achuta no Logt");         //send_sms();
                              transmit_string("AT");
                              txdata('\r');
                              txdata('\n');
                              delay22(200);
                              delay22(200);
```

```
delay22(200);
                              delay22(200);
transmit_string("AT+CMGF=1"); // set the SMS mode to text

txdata('\r');
                              txdata('\n');
                              delay22(200);
                              delay22(200);
                        delay22(200);
                              delay22(200);

transmit_string("AT+CMGS=\"+917773971399\"");  //AT+CMGS="xxxxxxxxxx"
// xxxxxxxxxx is the phone number. after this
                               txdata('\r');
                              txdata('\n');
                             delay22(200);
                              delay22(200);
                              delay22(200);
                              delay22(200);
transmit_string("Achuta no Logout");
                              txdata(0x1A);
                              delay22(200);
                        txdata(0x1A);
                  }
                  if(temp4%2 != 0)
                  {
                        lcdcmd(0x01);
lcdstr("Hasan no Logout");          //send_sms();
                              transmit_string("AT");
                              txdata('\r');
                              txdata('\n');
                              delay22(200);
                              delay22(200);
                        delay22(200);
                              delay22(200);
 transmit_string("AT+CMGF=1"); // set the SMS mode to text
                              txdata('\r');
                              txdata('\n');
                              delay22(200);
                              delay22(200);
                        delay22(200);
                              delay22(200);
 transmit_string("AT+CMGS=\"+917773971399\"");  //AT+CMGS="xxxxxxxxxx"
 // xxxxxxxxxx is the phone number. after this
                               txdata('\r');
```

```
txdata('\n');
delay22(200);
                                    delay22(200);
                                    delay22(200);
                                    delay22(200);
                                    transmit_string("Hasan no Logout");
                                    txdata(0x1A);
                                    delay22(200);
                                    txdata(0x1A);
                          }
//if(temp5%2 != 0)// check var if even no uncomment for more than 4 card no
//                        {
//    lcdcmd(0x01);
//    lcdstr("A3 Problem");         //send_sms();
//                              delay22(100);
//                        }
//                        if(temp6%2 != 0)
//                        {
//                              lcdcmd(0x01);
//                              lcdstr("A4 Problem");
      //send_sms();
//                              delay22(100);
//                        }
//                        if(temp7%2 != 0)
//                        {
//                              lcdcmd(0x01);
//                              lcdstr("A5 Problem");
      //send_sms();
//                              delay22(100);
//                        }
//                        if(temp8%2 != 0)
//                        {
//                              lcdcmd(0x01);
//                              lcdstr("A6 Problem");
      //send_sms();
//                              delay22(100);
//                        }
//                        if(temp9%2 != 0)
//                        {
//                              lcdcmd(0x01);
//    lcdstr("A7 Problem");         //send_sms();
//                              delay22(100);
//                        }
//                        if(temp10%2 != 0)
//                        {
```

```
//    lcdcmd(0x01);
//    lcdstr("A8 Problem");         //send_sms();
//                              delay22(100);
//                        }
                        while(1);
                  }
 F_OneSec = 0;
            }
            if(card_detected == 1)
            {
                  delayl();
                  delayl();
                  delayl();
                  lcdcmd(0x01);
                  delayl();
                  card_detected = 0;

card_number = check_card();      ////according to recieving card no by serial
(uart) card no is generated
update_data(card_number);       //the value of  card_number is passes to the
card by using switch case

                  if(card1 == 1 && card_number == 1)
                  {
                        lcdcmd(0x01);
                        delayl();
                        lcdstr("Class:EXTC Div:A");
                        lcdcmd(0xc0);
lcdstr("Laxman RL No:10");              //send_sms();
                        delayl();
                  beep();
                        temp1++;
                  }
                  else if(card2 == 1 && card_number == 2)
                  {
                        lcdcmd(0x01);
                        delayl();
                        lcdstr("Class:EXTC Div:A");
                        lcdcmd(0xc0);
lcdstr("Anoop RL No:11");                  //send_sms();
                        delayl();
                        delayl();
                        beep();
                        temp2++;
                  }
else if(card3 == 1 && card_number == 3)
                  {

            lcdcmd(0x01);
```

```
delayl();
                        lcdstr("Class:EXTC Div:A");
                        lcdcmd(0xc0);
lcdstr("Achuta RL No:12");                //send_sms();
                        delayl();
                        delayl();
                        beep();
                        temp3++;
                  }
else if(card4 == 1 && card_number == 4)
                  {
                        lcdcmd(0x01);
                        delayl();
                        lcdstr("Class:EXTC Div:A");
                        lcdcmd(0xc0);
lcdstr("Hasan RL No:13");                 //send_sms();
                        delayl();
                        delayl();
                        beep();
                        temp4++;
                  }
//    else if(card5 == 1 && card_number == 5)
//                  {
//                        lcdcmd(0x01);
//                        delayl();
//                        lcdstr("Class:X Div:A");
//                        lcdcmd(0xc0);
//   lcdstr("A3 RL No:11");                 //send_sms();
//                        delayl();
//                        delayl();
//                        beep();
//                        temp5++;
//
//                  }
//    else if(card6 == 1 && card_number == 6)
//                  {
//                        lcdcmd(0x01);
//                        delayl();
//                        lcdstr("Class:X Div:A");
//                        lcdcmd(0xc0);
//    lcdstr("A4 RL No:11");                //send_sms();
//                        delayl();
//                        delayl();
//                        beep();
//                        temp6++;
//
//                  }
//                  else if(card7 == 1 && card_number == 7)
//                  {
//                        lcdcmd(0x01);
```

```
//                      delayl();
//                      lcdstr("Class:X Div:A");
//                      lcdcmd(0xc0);
//    lcdstr("A5 RL No:11");                    //send_sms();
//                      delayl();
//                      delayl();
//                      beep();
//                      temp7++;
//
//                }
//                else if(card8 == 1 && card_number == 8)
//                {
//                      lcdcmd(0x01);
//                      delayl();
//                      lcdstr("Class:X Div:A");
//                      lcdcmd(0xc0);
//     lcdstr("A6 RL No:11");              //send_sms();
//                      delayl();
//                      delayl();
//                      beep();
//                      temp8++;
//
//                }
//    else if(card9 == 1 && card_number == 9)
//                {
//                      lcdcmd(0x01);
//                      delayl();
//                      lcdstr("Class:X Div:A");
//                      lcdcmd(0xc0);
//    lcdstr("A7");                         //send_sms();
//                      delayl();
//                beep();
//                      F_TeacherPresent = 1;
//                      OneSec = 0;
//                      temp9++;
//                }
//    else if(card10 == 1 && card_number == 10)
//                {
//                      lcdcmd(0x01);
//                      delayl();
//                      lcdstr("Class:X Div:A");
//                      lcdcmd(0xc0);
//    lcdstr("A8");                         //send_sms();
//                      delayl();
//                      beep();
//                      F_TeacherPresent  = 2;
//                      OneSec=0;
//                      temp10++;
//
//
      }
```

```
lcdcmd(0x01);
delayl();
else if(card11 == 1 && card_number == 11)
                  {

lcdstr("Class:EXTC Div:A");
                        lcdcmd(0xc0);
lcdstr("Nilima Mam");                     //send_sms();
                        delayl();
                        beep();
                        F_TeacherPresent  = 1;
                        OneSec=0;
                        temp11++;
                  }
else if(card12 == 1 && card_number == 12)
                  {
                        lcdcmd(0x01);
                        delayl();
                        lcdstr("Class:EXTC Div:A");
                        lcdcmd(0xc0);
lcdstr("Deena Mam");                      //send_sms();
                        delayl();
                        beep();
                        F_TeacherPresent  = 2;
                        OneSec=0;
                        temp12++;
                  }
                        delayl();
                        delayl();
                        card_number = 0;
card1 = card2 =card3 = card4 =
card5=card6=card7=card8=card9=card10=card11=card12= 0;
                        buzzer=1;
            //txdata(0x30|card_number);
          }
      }
}
// void auto_parking()
// {
//    if(slot_1_magnetic_sensor==0)
//    {
//          slot_green=0;
//          lcdcmd(0x01);
//          lcdcmd(0x80);
//          lcdstr("PARKING PLACE");
//          lcdcmd(0xc0);
//          lcdstr("AT SLOT:1");
//          delay_ms1(500);
//          delay_ms1(500);
//    }
```

```
//    else
//    {
//          slot_green=1;
//    }
//    if(slot_2_magnetic_sensor==0)
//    {
//          slot_yellow=0;
//          lcdcmd(0x01);
//          lcdcmd(0x80);
//          lcdstr("PARKING PLACE");
//          lcdcmd(0xc0);
//          lcdstr("AT SLOT:2");
//          delay_ms1(500);
//          delay_ms1(500);
//    }
//    else
//    {
//          slot_yellow=1;
//    }
//    if(slot_3_magnetic_sensor==0)
//    {
//          slot_red=0;
//          lcdcmd(0x01);
//          lcdcmd(0x80);
//          lcdstr("PARKING PLACE");
//          lcdcmd(0xc0);
//          lcdstr("AT SLOT:3");
//          delay_ms1(500);
//          delay_ms1(500);
//    }
//    else
//    {
//          slot_red=1;
//    }
// //       if(slot_4_magnetic_sensor==0)
// //       {
// //             slot_white=0;
// // //          lcdcmd(0x01);
// // //          lcdcmd(0x80);
// // //          display_msg("PARKING PLACE");
// // //          lcdcmd(0xc0);
// // //          display_msg("AT SLOT:4");
// // //          delay_ms1(500);
// // //          delay_ms1(500);
// // //    }
// // //    else
// // //    {
// // //          slot_white=1;
// // //    }
```

```
//  }
// void initialise_signal()
// {
//    slot_green  =1;
//    slot_yellow =1;
//    slot_red    =1;
//    //slot_white  =1;

//    slot_1_magnetic_sensor        =1;
//    slot_2_magnetic_sensor        =1;
//    slot_3_magnetic_sensor        =1;
//    //slot_4_magnetic_sensor      =1;
//
// }
```

ADVANTAGES
& DISADVANTAGES

7.1 ADVANTAGES

1. Man power reduced
2. One time investment
3. Low Cost
4. Providing security system based on super position level.
5. Better future for student due to Discipline followed.

7.2 DISADVANTAGES

1. The location of the student is not found where he goes exactly if he bunks the school.
2. Future Scope: We will send parents the location where the student has gone via GSM.

Future Scope, Application, Conclusion

FUTURE SCOPE OF SMART SCHOOL MANAGEMENT SYSTEMS

- Through GSM will be able to find out where the student has gone.
- Rather using card, Biometric System would be started in schools.

APPLICATIONS

Smart School Management System can be accessed remotely. With the help of this application, parents can know their children's attendance-records, grades, etc. The application is also a great help for teachers as the attendance in each class is automatically updated in his/her tablet or the PC. One can obtain the application at competitive rates.

CONCLUSION

We conclude that with the growing times we need to upgrade our skills/systems with ever-growing Technology. Those institutions/individuals who do not have value addition time to time are nowhere in the market. We have to be a part of fast moving technology.We have to polish our skills time and again.

DATASHEETS

Features

- Compatible with MCS®-51Products
- 2K Bytes of Reprogrammable Flash Memory
  - Endurance: 10,000 Write/Erase Cycles
- 2.7V to 6V Operating Range
- Fully Static Operation: 0 Hz to 24 MHz
- Two-level Program Memory Lock
- 128 x 8-bit Internal RAM
- 15 Programmable I/O Lines
- Two 16-bit Timer/Counters
- Six Interrupt Sources
- Programmable Serial UART Channel
- Direct LED Drive Outputs
- On-chip Analog Comparator
- Low-power Idle and Power-down Modes
- Green (Pb/Halide-free) Packaging Option

1. Description

The AT89C2051 is a low-voltage, high-performance CMOS 8-bit microcomputer with 2K bytes of Flash programmable and erasable read-only memory (PEROM). The device is manufactured using Atmel's high-density nonvolatile memory technology and is compatible with the industry-standard MCS-51 instruction set. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C2051 is a powerful microcomputer which provides a highly-flexible and cost-effective solution to many embedded control applications.

The AT89C2051 provides the following standard features: 2K bytes of Flash, 128 bytes of RAM, 15 I/O lines, two 16-bit timer/counters, a five vector two-level interrupt architecture, a full duplex serial port, a precision analog comparator, on-chip oscillator and clock circuitry. In addition, the AT89C2051 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port and interrupt system to continue functioning. The power-down mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next hardware reset.

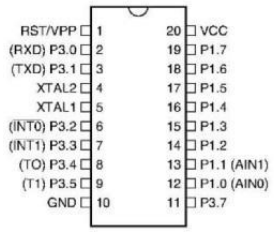
8-bit Microcontroller with 2K Bytes Flash

AT89C2051

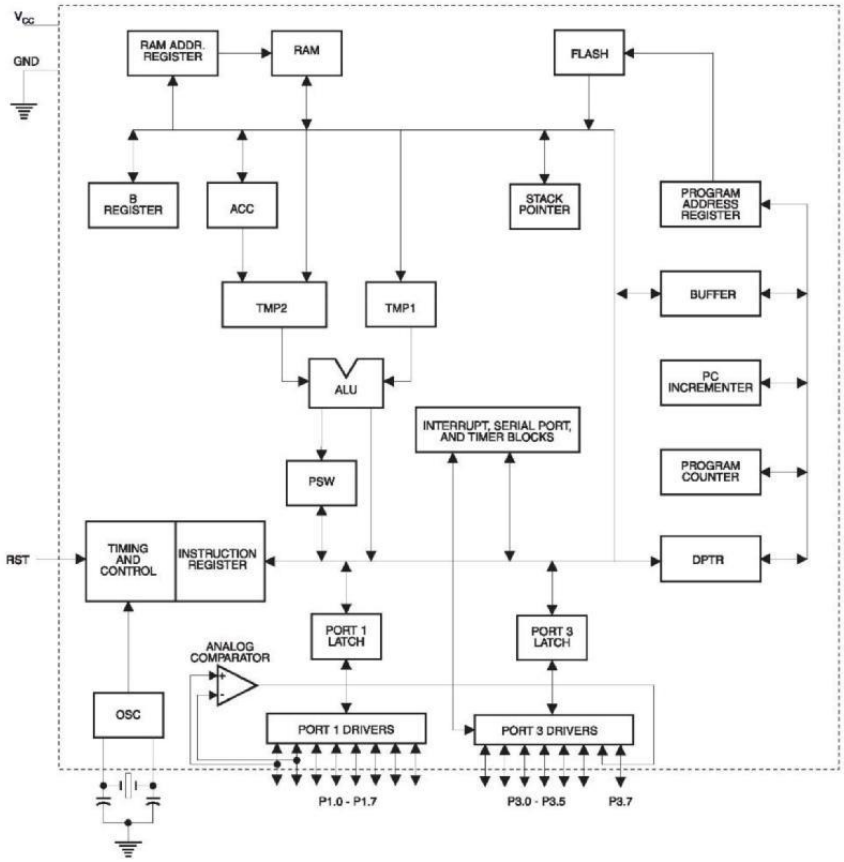
0368H–MICRO–6/08

2. Pin Configuration

2.1 20-lead PDIP/SOIC



3. Block Diagram



4. Pin Description

4.1 VCC

Supply voltage.

4.2 GND

Ground.

4.3 Port 1

The Port 1 is an 8-bit bi-directional I/O port. Port pins P1.2 to P1.7 provide internal pull-ups. P1.0 and P1.1 require external pull-ups. P1.0 and P1.1 also serve as the positive input (AIN0) and the negative input (AIN1), respectively, of the on-chip precision analog comparator. The Port 1 output buffers can sink 20 mA and can drive LED displays directly. When 1s are written to Port 1 pins, they can be used as inputs. When pins P1.2 to P1.7 are used as inputs and are externally pulled low, they will source current (I_{IL}) because of the internal pull-ups.

Port 1 also receives code data during Flash programming and verification.

4.4 Port 3

Port 3 pins P3.0 to P3.5, P3.7 are seven bi-directional I/O pins with internal pull-ups. P3.6 is hard-wired as an input to the output of the on-chip comparator and is not accessible as a general-purpose I/O pin. The Port 3 output buffers can sink 20 mA. 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 (I_{IL}) because of the pull-ups.

Port 3 also serves the functions of various special features of the AT89C2051 as listed below:

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{INT0}$ (external interrupt 0)
P3.3	$\overline{INT1}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)

Port 3 also receives some control signals for Flash programming and verification.

4.5 RST

Reset input. All I/O pins are reset to 1s as soon as RST goes high. Holding the RST pin high for two machine cycles while the oscillator is running resets the device.

Each machine cycle takes 12 oscillator or clock cycles.

4.6 XTAL1

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

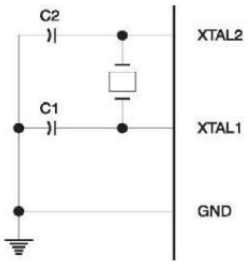
4.7 XTAL2

Output from the inverting oscillator amplifier.

5. Oscillator Characteristics

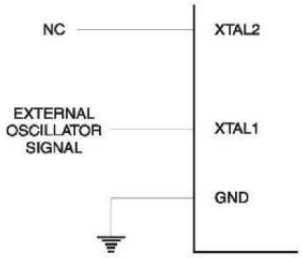
The XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator, as shown in Figure 5-1. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven as shown in Figure 5-2. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Figure 5-1. Oscillator Connections



Note: C1, C2 = 30 pF ± 10 pF for Crystals
= 40 pF ± 10 pF for Ceramic Resonators

Figure 5-2. External Clock Drive Configuration



6. Special Function Registers

A map of the on-chip memory area called the Special Function Register (SFR) space is shown in the table below.

Note that not all of the addresses are occupied, and unoccupied addresses may not be implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have an indeterminate effect.

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

Table 6-1. AT89C2051 SFR Map and Reset Values

0F8H									0FFH
0F0H	B 00000000								0F7H
0E8H									0EFH
0E0H	ACC 00000000								0E7H
0D8H									0DFH
0D0H	PSW 00000000								0D7H
0C8H									0CFH
0C0H									0C7H
0B8H	IP XXX00000								0BFH
0B0H	P3 11111111								0B7H
0A8H	IE 0XX00000								0AFH
0A0H									0A7H
98H	SCON 00000000	SBUF XXXXXXXX							9FH
90H	P1 11111111								97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000			8FH
80H		SP 00000111	DPL 00000000	DPH 00000000				PCON 0XXX0000	87H

7. Restrictions on Certain Instructions

The AT89C2051 and is an economical and cost-effective member of Atmel's growing family of microcontrollers. It contains 2K bytes of Flash program memory. It is fully compatible with the MCS-51 architecture, and can be programmed using the MCS-51 instruction set. However, there are a few considerations one must keep in mind when utilizing certain instructions to program this device.

All the instructions related to jumping or branching should be restricted such that the destination address falls within the physical program memory space of the device, which is 2K for the AT89C2051. This should be the responsibility of the software programmer. For example, LJMP 7E0H would be a valid instruction for the AT89C2051 (with 2K of memory), whereas LJMP 900H would not.

7.1 Branching Instructions

LCALL, LJMP, ACALL, AJMP, SJMP, JMP @A+DPTR – These unconditional branching instructions will execute correctly as long as the programmer keeps in mind that the destination branching address must fall within the physical boundaries of the program memory size (locations 00H to 7FFH for the 89C2051). Violating the physical space limits may cause unknown program behavior.

CJNE [...], DJNZ [...], JB, JNB, JC, JNC, JBC, JZ, JNZ – With these conditional branching instructions the same rule above applies. Again, violating the memory boundaries may cause erratic execution.

For applications involving interrupts the normal interrupt service routine address locations of the 80C51 family architecture have been preserved.

7.2 MOVX-related Instructions, Data Memory

The AT89C2051 contains 128 bytes of internal data memory. Thus, in the AT89C2051 the stack depth is limited to 128 bytes, the amount of available RAM. External DATA memory access is not supported in this device, nor is external PROGRAM memory execution. Therefore, no MOVX [...] instructions should be included in the program.

A typical 80C51 assembler will still assemble instructions, even if they are written in violation of the restrictions mentioned above. It is the responsibility of the controller user to know the physical features and limitations of the device being used and adjust the instructions used correspondingly.

8. Program Memory Lock Bits

On the chip are two lock bits which can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in the Table 8-1.

Table 8-1. Lock Bit Protection Modes[1]

	Program Lock Bits		
	LB1	LB2	Protection Type
1	U	U	No program lock features
2	P	U	Further programming of the Flash is disabled
3	P	P	Same as mode 2, also verify is disabled

Note: 1. The Lock Bits can only be erased with the Chip Erase operation.

9. Idle Mode

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special functions registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset.

The P1.0 and P1.1 should be set to "0" if no external pull-ups are used, or set to "1" if external pull-ups are used.

It should be noted that when idle is terminated by a hardware reset, the device normally resumes program execution, from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write to a port pin when Idle is terminated by reset, the instruction following the one that invokes Idle should not be one that writes to a port pin or to external memory.

10. Power-down Mode

In the power-down mode the oscillator is stopped, and the instruction that invokes power-down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the power-down mode is terminated. The only exit from power-down is a hardware reset. Reset redefines the SFRs but does not change the on-chip RAM. The reset should not be activated before V_{CC} is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize.

The P1.0 and P1.1 should be set to "0" if no external pull-ups are used, or set to "1" if external pull-ups are used.

11. Programming The Flash

The AT89C2051 is shipped with the 2K bytes of on-chip PEROM code memory array in the erased state (i.e., contents = FFH) and ready to be programmed. The code memory array is programmed one byte at a time. *Once the array is programmed, to re-program any non-blank byte, the entire memory array needs to be erased electrically.*

Internal Address Counter: The AT89C2051 contains an internal PEROM address counter which is always reset to 000H on the rising edge of RST and is advanced by applying a positive going pulse to pin XTAL1.

Programming Algorithm: To program the AT89C2051, the following sequence is recommended.

1. Power-up sequence:
   Apply power between V_{CC} and GND pins
   Set RST and XTAL1 to GND
2. Set pin RST to "H"
   Set pin P3.2 to "H"
3. Apply the appropriate combination of "H" or "L" logic
   levels to pins P3.3, P3.4, P3.5, P3.7 to select one of the programming operations
   shown in the PEROM Programming Modes table.

To Program and Verify the Array:

4. Apply data for Code byte at location 000H to P1.0 to P1.7.
5. Raise RST to 12V to enable programming.
6. Pulse P3.2 once to program a byte in the PEROM array or the lock bits. The byte-write cycle is self-timed and typically takes 1.2 ms.
7. To verify the programmed data, lower RST from 12V to logic "H" level and set pins P3.3 to P3.7 to the appropriate levels. Output data can be read at the port P1 pins.
8. To program a byte at the next address location, pulse XTAL1 pin once to advance the internal address counter. Apply new data to the port P1 pins.
9. Repeat steps 6 through 8, changing data and advancing the address counter for the entire 2K bytes array or until the end of the object file is reached.
10. Power-off sequence:
    set XTAL1 to "L"
    set RST to "L"
    Turn V_{CC} power off

$\overline{\text{Data}}$ Polling: The AT89C2051 features $\overline{\text{Data}}$ Polling to indicate the end of a write cycle. During a write cycle, an attempted read of the last byte written will result in the complement of the written data on P1.7. Once the write cycle has been completed, true data is valid on all outputs, and the next cycle may begin. $\overline{\text{Data}}$ Polling may begin any time after a write cycle has been initiated.

Ready/$\overline{\text{Busy}}$: The Progress of byte programming can also be monitored by the RDY/$\overline{\text{BSY}}$ output signal. Pin P3.1 is pulled low after P3.2 goes High during programming to indicate BUSY. P3.1 is pulled High again when programming is done to indicate READY.

Program Verify: If lock bits LB1 and LB2 have not been programmed code data can be read back via the data lines for verification:

1. Reset the internal address counter to 000H by bringing RST from "L" to "H".
2. Apply the appropriate control signals for Read Code data and read the output data at the port P1 pins.
3. Pulse pin XTAL1 once to advance the internal address counter.
4. Read the next code data byte at the port P1 pins.
5. Repeat steps 3 and 4 until the entire array is read.

The lock bits cannot be verified directly. Verification of the lock bits is achieved by observing that their features are enabled.

Chip Erase: The entire PEROM array (2K bytes) and the two Lock Bits are erased electrically by using the proper combination of control signals and by holding P3.2 low for 10 ms. The code array is written with all "1"s in the Chip Erase operation and must be executed before any non-blank memory byte can be re-programmed.

Reading the Signature Bytes: The signature bytes are read by the same procedure as a normal verification of locations 000H, 001H, and 002H, except that P3.5 and P3.7 must be pulled to a logic low. The values returned are as follows.

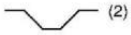
(000H) = 1EH indicates manufactured by Atmel
(001H) = 21H indicates 89C2051

12. Programming Interface

Every code byte in the Flash array can be written and the entire array can be erased by using the appropriate combination of control signals. The write operation cycle is self-timed and once initiated, will automatically time itself to completion.

Most major worldwide programming vendors offer support for the Atmel AT89 microcontroller series. Please contact your local programming vendor for the appropriate software revision.

13. Flash Programming Modes

Mode		RST/VPP	P3.2/$\overline{PROG}$	P3.3	P3.4	P3.5	P3.7
Write Code Data[1][3]		12V		L	H	H	H
Read Code Data[1]		H	H	L	L	H	H
Write Lock	Bit - 1	12V		H	H	H	H
	Bit - 2	12V		H	H	L	L
Chip Erase		12V	 (2)	H	L	L	L
Read Signature Byte		H	H	L	L	L	L

Notes:
1. The internal PEROM address counter is reset to 000H on the rising edge of RST and is advanced by a positive pulse at XTAL1 pin.
2. Chip Erase requires a 10 ms $\overline{PROG}$ pulse.
3. P3.1 is pulled Low during programming to indicate RDY/$\overline{BSY}$.

Figure 13-1. Programming the Flash Memory

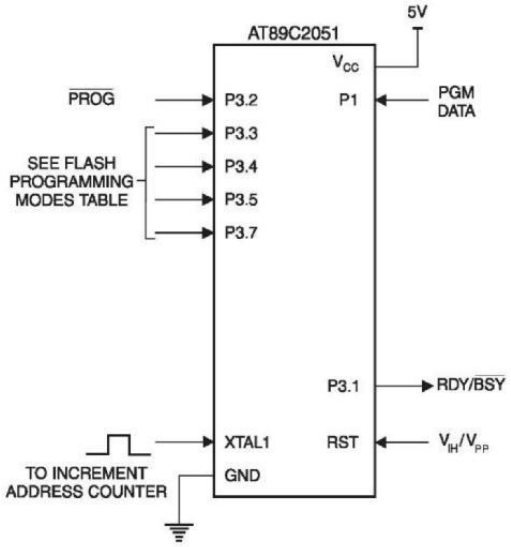
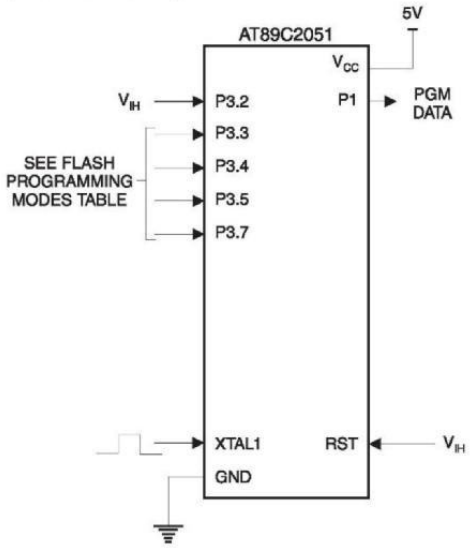


Figure 13-2. Verifying the Flash Memory



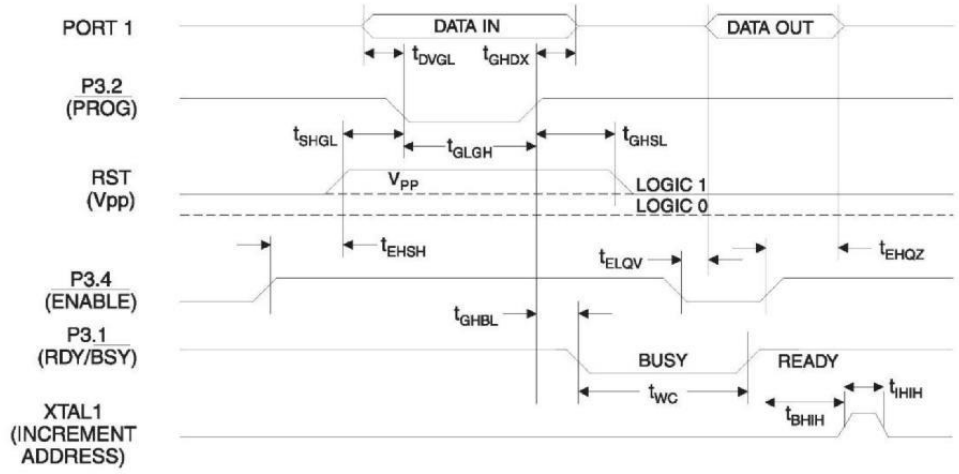
14. Flash Programming and Verification Characteristics

T_A = 0°C to 70°C, V_{CC} = 5.0 ± 10%

Symbol	Parameter	Min	Max	Units
V_{PP}	Programming Enable Voltage	11.5	12.5	V
I_{PP}	Programming Enable Current		250	µA
t_{DVGL}	Data Setup to $\overline{PROG}$ Low	1.0		µs
t_{GHDX}	Data Hold after $\overline{PROG}$	1.0		µs
t_{EHSH}	P3.4 ($\overline{ENABLE}$) High to V_{PP}	1.0		µs
t_{SHGL}	V_{PP} Setup to $\overline{PROG}$ Low	10		µs
t_{GHSL}	V_{PP} Hold after $\overline{PROG}$	10		µs
t_{GLGH}	$\overline{PROG}$ Width	1	110	µs
t_{ELQV}	$\overline{ENABLE}$ Low to Data Valid		1.0	µs
t_{EHQZ}	Data Float after $\overline{ENABLE}$	0	1.0	µs
t_{GHBL}	$\overline{PROG}$ High to $\overline{BUSY}$ Low		50	ns
t_{WC}	Byte Write Cycle Time		2.0	ms
t_{BHIH}	RDY/$\overline{BSY}$\ to Increment Clock Delay	1.0		µs
t_{IHIL}	Increment Clock High	200		ns

Note: 1. Only used in 12-volt programming mode.

15. Flash Programming and Verification Waveforms



16. Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on Any Pin with Respect to Ground	-1.0V to +7.0V
Maximum Operating Voltage	6.6V
DC Output Current	25.0 mA

*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

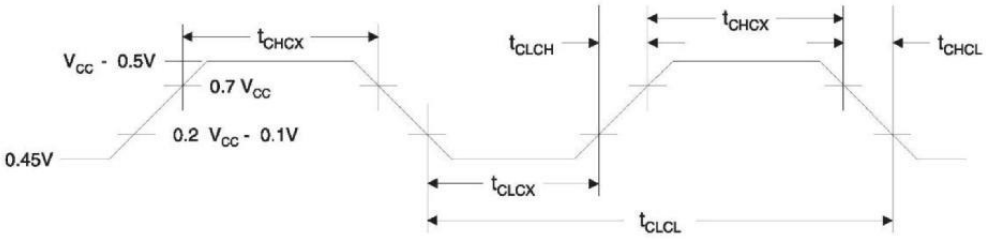
17. DC Characteristics

T_A = -40°C to 85°C, V_{CC} = 2.7V to 6.0V (unless otherwise noted)

Symbol	Parameter	Condition	Min	Max	Units
V_{IL}	Input Low-voltage		-0.5	0.2 V_{CC} - 0.1	V
V_{IH}	Input High-voltage	(Except XTAL1, RST)	0.2 V_{CC} + 0.9	V_{CC} + 0.5	V
V_{IH1}	Input High-voltage	(XTAL1, RST)	0.7 V_{CC}	V_{CC} + 0.5	V
V_{OL}	Output Low-voltage[1] (Ports 1, 3)	I_{OL} = 20 mA, V_{CC} = 5V I_{OL} = 10 mA, V_{CC} = 2.7V		0.5	V
V_{OH}	Output High-voltage (Ports 1, 3)	I_{OH} = -80 µA, V_{CC} = 5V ±10%	2.4		V
		I_{OH} = -30 µA	0.75 V_{CC}		V
		I_{OH} = -12 µA	0.9 V_{CC}		V
I_{IL}	Logical 0 Input Current (Ports 1, 3)	V_{IN} = 0.45V		-50	µA
I_{TL}	Logical 1 to 0 Transition Current (Ports 1, 3)	V_{IN} = 2V, V_{CC} = 5V ±10%		-750	µA
I_{LI}	Input Leakage Current (Port P1.0, P1.1)	0 < V_{IN} < V_{CC}		±10	µA
V_{OS}	Comparator Input Offset Voltage	V_{CC} = 5V		20	mV
V_{CM}	Comparator Input Common Mode Voltage		0	V_{CC}	V
RRST	Reset Pull-down Resistor		50	300	kΩ
C_{IO}	Pin Capacitance	Test Freq. = 1 MHz, T_A = 25°C		10	pF
I_{CC}	Power Supply Current	Active Mode, 12 MHz, V_{CC} = 6V/3V		15/5.5	mA
		Idle Mode, 12 MHz, V_{CC} = 6V/3V P1.0 & P1.1 = 0V or V_{CC}		5/1	mA
	Power-down Mode[2]	V_{CC} = 6V, P1.0 & P1.1 = 0V or V_{CC}		100	µA
		V_{CC} = 3V, P1.0 & P1.1 = 0V or V_{CC}		20	µA

Notes:
1. Under steady state (non-transient) conditions, I_{OL} must be externally limited as follows:
   Maximum I_{OL} per port pin: 20 mA
   Maximum total I_{OL} for all output pins: 80 mA
   If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.
2. Minimum V_{CC} for Power-down is 2V.

18. External Clock Drive Waveforms



19. External Clock Drive

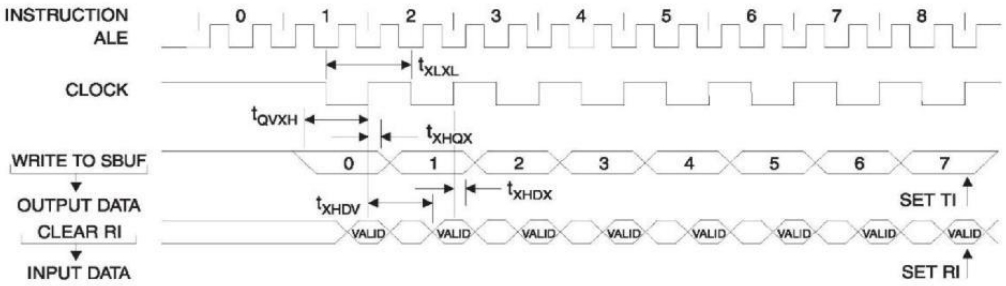
Symbol	Parameter	V_{CC} = 2.7V to 6.0V		V_{CC} = 4.0V to 6.0V		Units
		Min	Max	Min	Max	
$1/t_{CLCL}$	Oscillator Frequency	0	12	0	24	MHz
t_{CLCL}	Clock Period	83.3		41.6		ns
t_{CHCX}	High Time	30		15		ns
t_{CLCX}	Low Time	30		15		ns
t_{CLCH}	Rise Time		20		20	ns
t_{CHCL}	Fall Time		20		20	ns

20. Serial Port Timing: Shift Register Mode Test Conditions

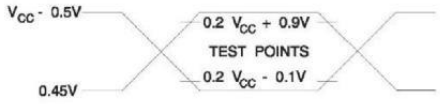
V_{CC} = 5.0V ±20%; Load Capacitance = 80 pF

Symbol	Parameter	12 MHz Osc		Variable Oscillator		Units
		Min	Max	Min	Max	
t_{XLXL}	Serial Port Clock Cycle Time	1.0		12 t_{CLCL}		µs
t_{QVXH}	Output Data Setup to Clock Rising Edge	700		10 t_{CLCL}-133		ns
t_{XHQX}	Output Data Hold after Clock Rising Edge	50		2 t_{CLCL}-117		ns
t_{XHDX}	Input Data Hold after Clock Rising Edge	0		0		ns
t_{XHDV}	Clock Rising Edge to Input Data Valid		700		10 t_{CLCL}-133	ns

21. Shift Register Mode Timing Waveforms

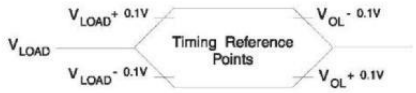


22. AC Testing Input/Output Waveforms[1]



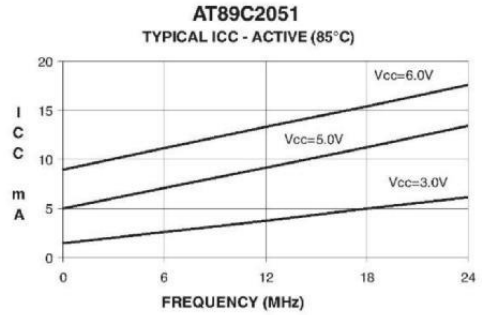
Note: 1. AC Inputs during testing are driven at V_{CC} - 0.5V for a logic 1 and 0.45V for a logic 0. Timing measurements are made at V_{IH} min. for a logic 1 and V_{IL} max. for a logic 0.

23. Float Waveforms[1]

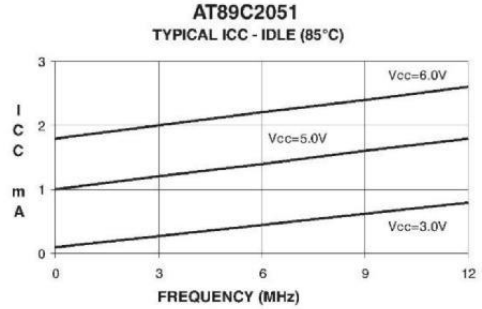


Note: 1. For timing purposes, a port pin is no longer floating when a 100 mV change from load voltage occurs. A port pin begins to float when 100 mV change from the loaded V_{OH}/V_{OL} level occurs.

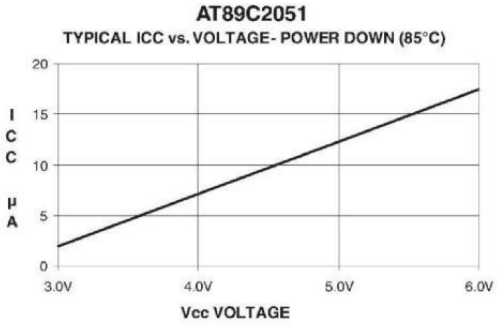
24. I_{CC} (Active Mode) Measurements



25. I_{CC} (Idle Mode) Measurements



26. I_{CC} (Power Down Mode) Measurements



Notes: 1. XTAL1 tied to GND
2. P.1.0 and P1.1 = V_{CC} or GND
3. Lock bits programmed

27. Ordering Information

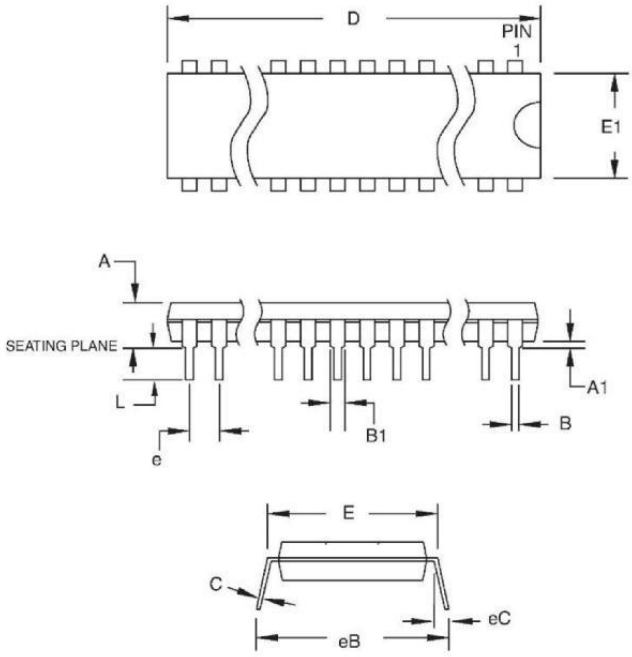
27.1 Green Package Option (Pb/Halide-free)

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
12	2.7V to 6.0V	AT89C2051-12PU AT89C2051-12SU	20P3 20S	Industrial (-40° C to 85° C)
24	4.0V to 6.0V	AT89C2051-24PU AT89C2051-24SU	20P3 20S	Industrial (-40° C to 85° C)

Package Type	
20P3	20-lead, 0.300" Wide, Plastic Dual In-line Package (PDIP)
20S	20-lead, 0.300" Wide, Plastic Gull Wing Small Outline (SOIC)

28. Package Information

28.1 20P3 – PDIP



COMMON DIMENSIONS
(Unit of Measure = mm)

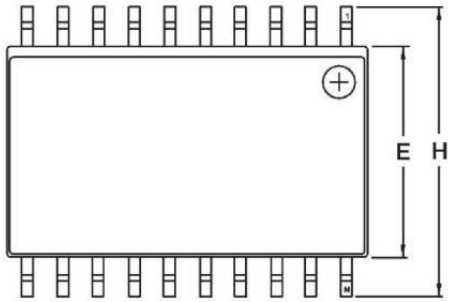
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	5.334	
A1	0.381	–	–	
D	24.892	–	26.924	Note 2
E	7.620	–	8.255	
E1	6.096	–	7.112	Note 2
B	0.356	–	0.559	
B1	1.270	–	1.551	
L	2.921	–	3.810	
C	0.203	–	0.356	
eB	–	–	10.922	
eC	0.000	–	1.524	
e	2.540 TYP			

Notes:
1. This package conforms to JEDEC reference MS-001, Variation AD.
2. Dimensions D and E1 do not include mold Flash or Protrusion. Mold Flash or Protrusion shall not exceed 0.25 mm (0.010").

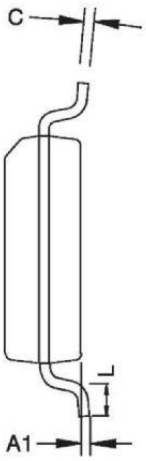
1/23/04

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**20P3**, 20-lead (0.300"/7.62 mm Wide) Plastic Dual Inline Package (PDIP)	20P3	D

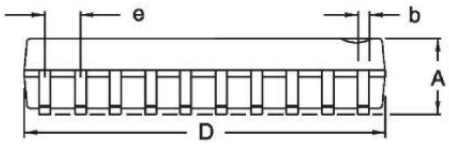
28.2 20S – SOIC



Top View



End View



Side View

COMMON DIMENSIONS
(Unit of Measure – mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	2.35		2.65	
A1	0.10		0.30	
b	0.33		0.51	4
C	0.23		0.32	
D	12.60		13.00	1
E	7.40		7.60	2
H	10.00		10.65	
L	0.40		1.27	3
e	1.27 BSC			

Notes.
1. This drawing is for general information only; refer to JEDEC Drawing MS-013, Variation AC for additional information.
2. Dimension 'D' does not include mold Flash, protrusions or gate burrs. Mold Flash, protrusions and gate burrs shall not exceed 0.15 mm (0.006') per side.
3. Dimension 'E' does not include inter-lead Flash or protrusion. Inter-lead Flash and protrusions shall not exceed 0.25 mm (0.010') per side.
4. 'L' is the length of the terminal for soldering to a substrate.
5. The lead width 'b', as measured 0.36 mm (0.014') or greater above the seating plane, shall not exceed a maximum value of 0.61 mm (0.024') per side.

11/6/06

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**20S2**, 20-lead, 0.300' Wide Body, Plastic Gull Wing Small Outline Package (SOIC)	20S2	B

Headquarters

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

International

Atmel Asia
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

Atmel Europe
Le Krebs
8, Rue Jean-Pierre Timbaud
BP 309
78054 Saint-Quentin-en-
Yvelines Cedex
France
Tel: (33) 1-30-60-70-00
Fax: (33) 1-30-60-71-11

Atmel Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Product Contact

Web Site
www.atmel.com

Technical Support
mcu@atmel.com

Sales Contact
www.atmel.com/contacts

Literature Requests
www.atmel.com/literature

0368H–MICRO–6/08

REFERENCES

- https://en.wikipedia.org/wiki/Atmel
- www.microchip.com/
- start.atmel.com/
- 2007.Nature Photonics-Dominic O'Brien, Gareth Parry & Paul Stavrinou.
- Lopez-Hernandez-Fj, Poves-E, Perez-Jimenez-R, and Rabadan-J: 'Low cost diffuse wireless optical communication system based on white LED'. Proc. 2006
- IEEE Tenth International Symposium on Consumer Electronics. St. Petersburg, Russia. 28 June 1 July 2006., pp.
- P. Amirshahi and M. Kavehrad , (2006). Broadband Access over Medium & Low Voltage Power-lines and use of White LEDs for Indoor Communications. In IEEE CCNC 2006 proceedings
- Amirshahi, P. and Kavehrad, M. 2006. Broadband access over medium and low voltage power-lines and use of white light emitting diodes for indoor communications. In IEEE Consumer Communications & Networking Conference, Las Vegas, Nevada. Citeseer.
- J. Carruthers and J. Kahn, "Multiple-Subcarrier Modulation for Nondirected
- Wireless Infrared Communication," IEEE Journal on Selected Areas in
- Communication, vol. 14, pp. 538–546, April 1996