**LIFE CYCLE TESING OF ELECTRICAL LOADS BY DOWN COUNTER**

**CONTENTS PAGE NO.**

1. INTRODUCTION TO EMBEDDED SYSTEMS 02
2. PROJECT BLOCK DIAGRAM 06
3. HARDWARE REQUIREMENTS 07

4.1 TRANSFORMERS 07

4.2 VOLTAGE REGULATOR (LM7805) 10

4.3 RECTIFIER 12

4.4 FILTER 12

4.5 MICROCONTROLLER (AT89S52/C51) 13

4.6 SEVEN SEGMENT DISPLAY 20

* 1. KEY PAD 23
  2. RELAY 24
  3. LOAD
  4. BC547 28
  5. 1N4007 29
  6. LED 31
  7. RESISTORS 34
  8. CAPACITORS 36

5. SOFTWARE REQUIREMENTS 41

5.1 IDE 41

5.2 CONCEPT OF COMPILER 41

5.3 CONCEPT OF CROSS COMPILER 41

5.4 KEIL C CROSS COMPILER 42

5.5 BUILDING AN APPLICATION IN UVISION2 42

5.6 CREATING YOUR OWN APPLICATION IN UVISION2 42

5.7 DEBUGGING AN APPLICATION IN UVISION2 43

5.8 STARTING UVISION2 & CREATING A PROJECT 43

5.9 WINDOWS\_ FILES 44

5.10 BUILDING PROJECTS & CREATING HEX FILES 44

5.11 CPU SIMULATION 44

5.12 DATABASE SELECTION 44

5.13 START DEBUGGING 45

5.14 DISASSEMBLY WINDOW 46

5.15 EMBEDDED C 46

6. SCHEMATIC DIAGRAM 47

6.1 DESCRIPTION 48

7. LAYOUT DIAGRAM 52

8. HARDWARE NEEDS 53

9. CODING

9.1 PROGRAM CODE 56

9.2 COMPILER 64

10. HARDWARE TESTING 72

10.1 CONTINUITY TEST 72

10.2 POWER ON TEST 72

11. RESULTS 73

12.CONCLUSION 73

13. BIBLIOGRAPHY 73

**LIST OF FIGURES PAGE NO.**

2(a) EMBEDDED DESIGN CALLS 03

2(b) EMBEDDED DESIGN CYCLE 04

3 BLOCK DIAGRAM OF THE PROJECT 06

4.1 A TYPICAL TRANSFORMER 08

4.2(a) BLOCK DIAGRAM OF VOLTAGE

REGULATOR 10

4.3(b) PERFORMANCE CHARACTERISTICS

OF VOLTAGE REGULATOR 11

4.4(a) OSCILLATOR CONNECTIONS 19

4.4(b) EXTERNAL CLOCK DRIVE CONFIG. 19

4.5 BLOCK DIAGRAM OF AT89S52 15

4.5(b) PIN DIAGRAM OF AT89S52 16

6.0 SCHEMATIC DIAGRAM 47

7.0 LAYOUT DIAGRAM 52

**1. ABSTRACT**

. A desired number is entered through a keypad interfaced to a microcontroller of 8051 family. Upon activation, the system counts down one in each second till the set number reaches zero. A relay switches the load ON & OFF for every count thus testing the life cycle of the product.

The working life of many products such as lamps depends on the number of ON/OFF cycles it encounters. This project is designed to be used in industries for testing the life cycle of such electrical loads (lamps, motors, etc;) using a down counter.

This proposed system uses microcontroller of 8051 family with a keypad interfaced to it to enter a required number. 7-segment displays are used for displaying the number /count. Once the circuit is powered, the counter is set using the keypad to count down anywhere from 999 to 0. The counter starts decrementing from the set number to 0 per each count and simultaneously making the load (i.e. bulb in this circuit) turn ON and OFF by a relay. Reaching the zero count it finally remains OFF.

The project can be further enhanced by using mono stable triggering device to stop the counter in the event of failure of the load before reaching the zero. Thus the exact life cycle of the load can be estimated.

**2.INTRODUCTION TO EMBEDDED SYSTEMS**

**What is Embedded system?**

An Embedded System is a combination of computer hardware and software, and perhaps additional mechanical or other parts, designed to perform a specific function. An embedded system is a microcontroller-based, software driven, reliable, real-time control system, autonomous, or human or network interactive, operating on diverse physical variables and in diverse environments and sold into a competitive and cost conscious market.

An embedded system is not a computer system that is used primarily for processing, not a software system on PC or UNIX, not a traditional business or scientific application. **High-end embedded & lower end embedded systems**: High-end embedded system - Generally 32, 64 Bit Controllers used with OS. Examples Personal Digital Assistant and Mobile phones etc .Lower end embedded systems - Generally 8,16 Bit Controllers used with an minimal operating systems and hardware layout designed for the specific purpose. Examples Small controllers and devices in our everyday life like Washing Machine, Microwave Ovens, where they are embedded in.

**SYSTEM DESIGN CALLS:**

Figure 3.1(a) System Design Call

**EMBEDDED SYSTEM DESIGN CYCLE**

Figuren 3.1(b) “V Diagram”

**Characteristics of Embedded System**

* An embedded system is any computer system hidden inside a product other than a computer.
* They will encounter a number of difficulties when writing embedded system software in addition to those we encounter when we write applications
  + Throughput – Our system may need to handle a lot of data in a short period of time.
  + Response–Our system may need to react to events quickly
  + Testability–Setting up equipment to test embedded software can be difficult
  + Debugability–Without a screen or a keyboard, finding out what the software is doing wrong (other than not working) is a troublesome problem
  + Reliability – embedded systems must be able to handle any situation without human intervention
  + Memory space – Memory is limited on embedded systems, and you must make the software and the data fit into whatever memory exists
  + Program installation – you will need special tools to get your software into embedded systems
  + Power consumption – Portable systems must run on battery power, and the software in these systems must conserve power
  + Processor hogs – computing that requires large amounts of CPU time can complicate the response problem
  + Cost – Reducing the cost of the hardware is a concern in many embedded system projects; software often operates on hardware that is barely adequate for the job.
* Embedded systems have a microprocessor/ microcontroller and a memory. Some have a serial port or a network connection. They usually do not have keyboards, screens or disk drives.

**APPLICATIONS**

1. Military and aerospace embedded software applications
2. Communication Applications
3. Industrial automation and process control software
4. Mastering the complexity of applications.
5. Reduction of product design time.
6. Real time processing of ever increasing amounts of data.
7. Intelligent, autonomous sensors.

**CLASSIFICATION**

* Real Time Systems.
* RTS is one which has to respond to events within a specified deadline.
* A right answer after the dead line is a wrong answer

**RTS CLASSIFICATION**

* Hard Real Time Systems
* Soft Real Time System

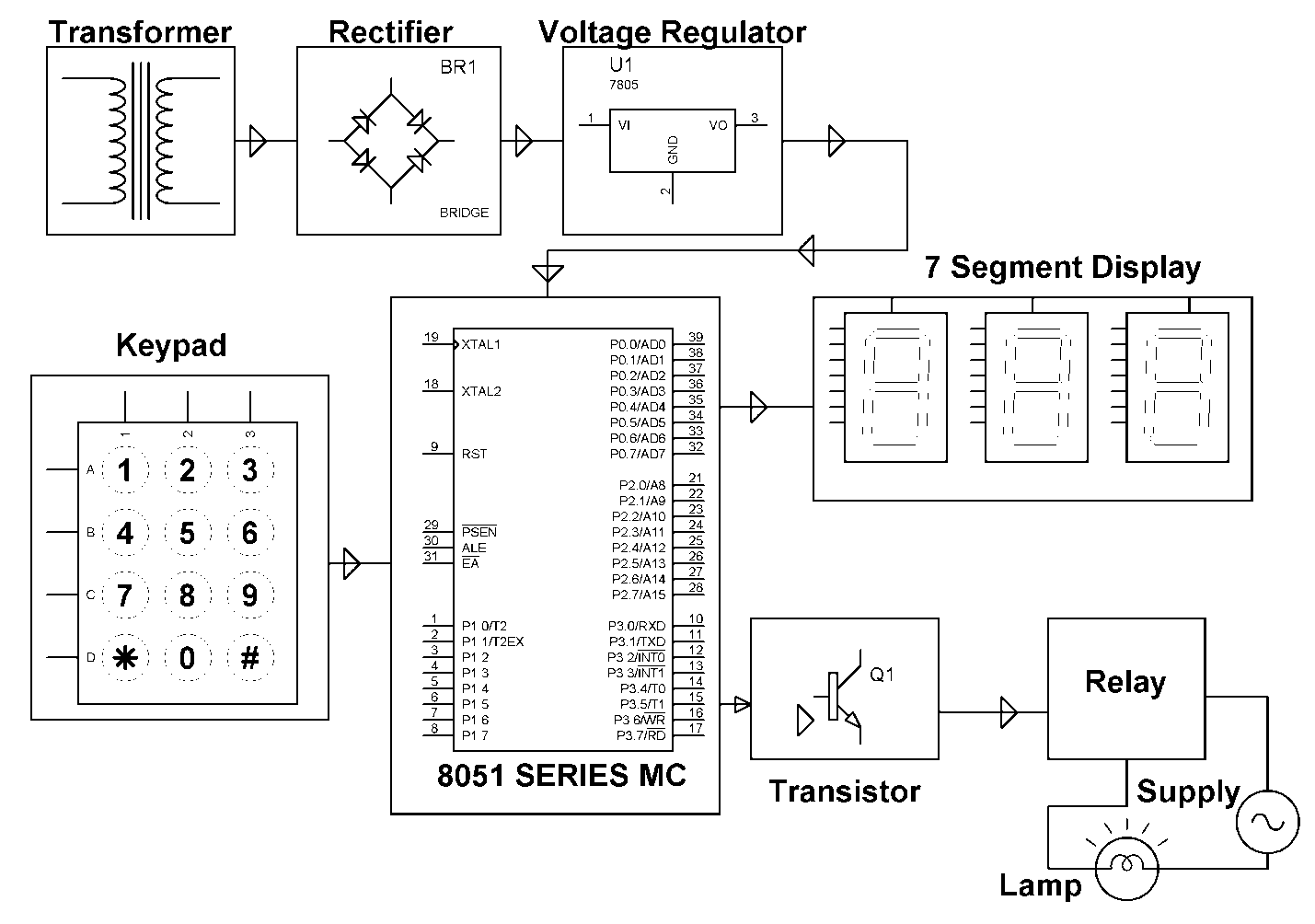
**HARD REAL TIME SYSTEM**

* "Hard" real-time systems have very narrow response time.
* Example: Nuclear power system, Cardiac pacemaker.

**SOFT REAL TIME SYSTEM**

* "Soft" real-time systems have reduced constrains on "lateness" but still must operate very quickly and repeatable.
* Example: Railway reservation system – takes a few extra seconds the data remains valid.

**3. PROJECT BLOCK DIAGRAM**



**4. HARDWARE REQUIREMENTS**

**HARDWARE COMPONENTS:**

1. TRANSFORMER (230 – 12 V AC)
2. VOLTAGE REGULATOR (LM 7805)
3. RECTIFIER
4. FILTER
5. MICROCONTROLLER (AT89S52/AT89C51)
6. SEVEN SEGMENT DISPLAY
7. KEYPAD
8. RELAY
9. BC547
10. 1N4007
11. LED
12. RESISTORS
13. CAPACITORS

**DESCRIPTION**

**4.1 TRANSFORMER**

Transformers convert AC electricity from one voltage to another with a little loss of power. Step-up transformers increase voltage, step-down transformers reduce voltage. Most power supplies use a step-down transformer to reduce the dangerously high voltage to a safer low voltage.



FIG 4.1: A TYPICAL TRANSFORMER

The input coil is called the primary and the output coil is called the secondary. There is no electrical connection between the two coils; instead they are linked by an alternating magnetic field created in the soft-iron core of the transformer. The two lines in the middle of the circuit symbol represent the core. Transformers waste very little power so the power out is (almost) equal to the power in. Note that as voltage is stepped down and current is stepped up.

The ratio of the number of turns on each coil, called the turn’s ratio, determines the ratio of the voltages. A step-down transformer has a large number of turns on its primary (input) coil which is connected to the high voltage mains supply, and a small number of turns on its secondary (output) coil to give a low output voltage.

TURNS RATIO = (Vp / Vs) = ( Np / Ns )

Where,

Vp = primary (input) voltage.

Vs = secondary (output) voltage

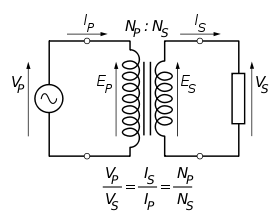
Np = number of turns on primary coil

Ns = number of turns on secondary coil

Ip = primary (input) current

Is = secondary (output) current.

**Ideal power equation**

[](http://en.wikipedia.org/wiki/File:Transformer_under_load.svg)

The ideal transformer as a circuit element

If the secondary coil is attached to a load that allows current to flow, electrical power is transmitted from the primary circuit to the secondary circuit. Ideally, the transformer is perfectly efficient; all the incoming energy is transformed from the primary circuit to the [magnetic field](http://en.wikipedia.org/wiki/Magnetic_field) and into the secondary circuit. If this condition is met, the incoming [electric power](http://en.wikipedia.org/wiki/Electric_power) must equal the outgoing power:


P_\text{incoming} = I_\text{p} V_\text{p} = P_\text{outgoing} = I_\text{s} V_\text{s},\!


Giving the ideal transformer equation


\frac{V_\text{s}}{V_\text{p}} = \frac{N_\text{s}}{N_\text{p}} = \frac{I_\text{p}}{I_\text{s}}.


Transformers normally have high efficiency, so this formula is a reasonable approximation.

If the voltage is increased, then the current is decreased by the same factor. The impedance in one circuit is transformed by the *square* of the turns ratio. For example, if an impedance *Z*s is attached across the terminals of the secondary coil, it appears to the primary circuit to have an impedance of (*N*p/*N*s)2*Z*s. This relationship is reciprocal, so that the impedance *Z*p of the primary circuit appears to the secondary to be (*N*s/*N*p)2*Z*p.

**4.2 VOLTAGE REGULATOR 7805**

**Features**

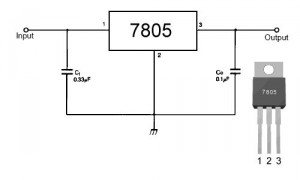
• Output Current up to 1A.

• Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V.

• Thermal Overload Protection.

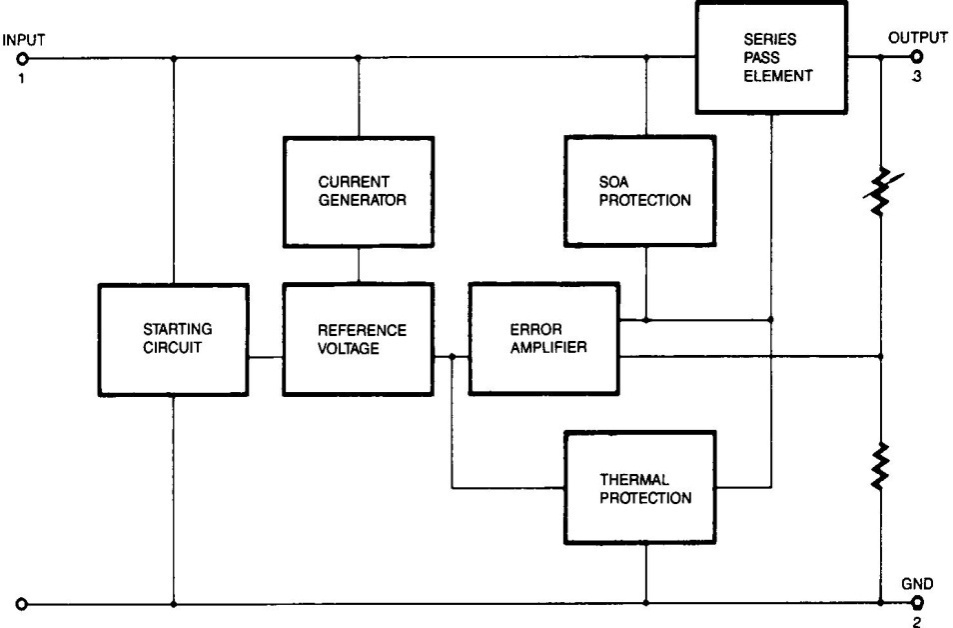
• Short Circuit Protection.

• Output Transistor Safe Operating Area Protection.



**Description**

The LM78XX/LM78XXA series of three-terminal positive regulators are available in the TO-220/D-PAK package and with several fixed output voltages, making them useful in a Wide range of applications. Each type employs internal current limiting, thermal shutdown and safe operating area protection, making it essentially indestructible. If adequate heat sinking is provided, they can deliver over 1A output Current. Although designed primarily as fixed voltage regulators, these devices can be used with external components to obtain adjustable voltages and currents.

****

**Absolute Maximum Ratings**

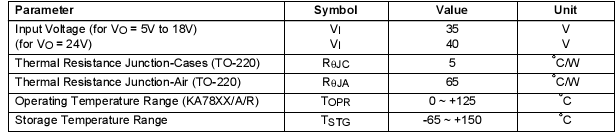
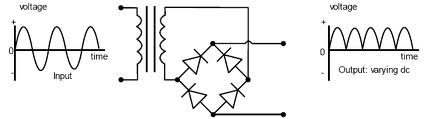
****

TABLE 4.2(b): RATINGS OF THE VOLTAGE REGULATOR

[](http://www.google.co.in/imgres?imgurl=http://www.evl.uic.edu/sjames/dvl/images/51262.jpg&imgrefurl=http://vinosysabores.com/voltage-regulator-7805&page=7&usg=__8M5wBB3ts7AF3gXgXNgYUtUKog4=&h=300&w=300&sz=7&hl=en&start=14&zoom=1&tbnid=gdGbqNw28x40UM:&tbnh=116&tbnw=116&ei=MRHeTdXKDo-8vgO735nABQ&prev=/search?q=voltage+regulator+7805&hl=en&sa=X&biw=1003&bih=539&tbm=isch&itbs=1)**4.3 RECTIFIER**

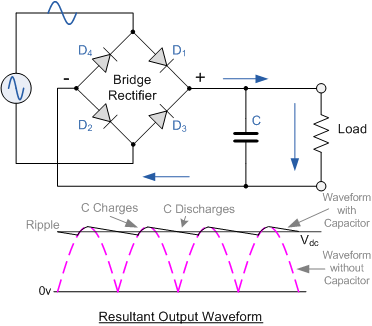
A rectifier is an electrical device that converts [alternating current](http://en.wikipedia.org/wiki/Alternating_current) (AC), which periodically reverses direction, to [direct current](http://en.wikipedia.org/wiki/Direct_current) (DC), current that flows in only one direction, a process known as rectification. Rectifiers have many uses including as components of [power supplies](http://en.wikipedia.org/wiki/Power_supply) and as [detectors](http://en.wikipedia.org/wiki/Detector_(radio)) of [radio](http://en.wikipedia.org/wiki/Radio) signals. Rectifiers may be made of [solid state](http://en.wikipedia.org/wiki/Solid_state_(electronics)) [diodes](http://en.wikipedia.org/wiki/Diode), [vacuum tube](http://en.wikipedia.org/wiki/Vacuum_tube) diodes, [mercury arc valves](http://en.wikipedia.org/wiki/Mercury_arc_valve), and other components. The output from the transformer is fed to the rectifier. It converts A.C. into pulsating D.C. The rectifier may be a half wave or a full wave rectifier. In this project, a bridge rectifier is used because of its merits like good stability and full wave rectification. In positive half cycleonly two diodes ( 1 set of parallel diodes) will conduct, in negative half cycle remaining two diodes will conduct and they will conduct only in forward bias only.



**4.4 FILTER**

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

The simple capacitor filter is the most basic type of power supply filter. The use of this filter is very limited. It is sometimes used on extremely high-voltage, low-current power supplies for cathode-ray and similar electron tubes that require very little load current from the supply. This filter is also used in circuits where the power-supply ripple frequency is not critical and can be relatively high. Below figure can show how the capacitor changes and discharges.



**4.5 MICROCONTROLLER AT89S52**

The AT89S52 is a low-power, high-performance CMOS 8-bit microcontroller with 8K bytes of in-system programmable Flash memory. The device is manufactured using Atmel’s high-density non volatile memory technology and is compatible with the industry standard 80C51 instruction set and pin out. The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional non volatile memory programmer. By combining a versatile 8-bit CPU with in-system programmable Flash on a monolithic chip, the Atmel AT89S52 is a powerful microcontroller which provides a highly-flexible and cost-effective solution to many embedded control applications. The AT89S52 provides the following standard features: 8K bytes of Flash, 256 bytes of RAM, 32 I/O lines, Watchdog timer, two data pointers, three 16-bit timer/counters, a six-vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator, and clock circuitry. In addition, the AT89S52 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port, and interrupt system to continue functioning. The Power-down mode saves the RAM contents but freezes the oscillator, disabling all other chip functions until the next interrupt or hardware reset.

**Features:**

• Compatible with MCS®-51 Products

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

– Endurance: 10,000 Write/Erase Cycles

• 4.0V to 5.5V Operating Range

• Fully Static Operation: 0 Hz to 33 MHz

• Three-level Program Memory Lock

• 256 x 8-bit Internal RAM

• 32 Programmable I/O Lines

• Three 16-bit Timer/Counters

• Eight Interrupt Sources

• Full Duplex UART Serial Channel

• Low-power Idle and Power-down Modes

• Interrupt Recovery from Power-down Mode

• Watchdog Timer

• Dual Data Pointer

• Power-off Flag

• Fast Programming Time

• Flexible ISP Programming (Byte and Page Mode)

• Green (Pb/Halide-free) Packaging Option

**Block Diagram of AT89S52:**

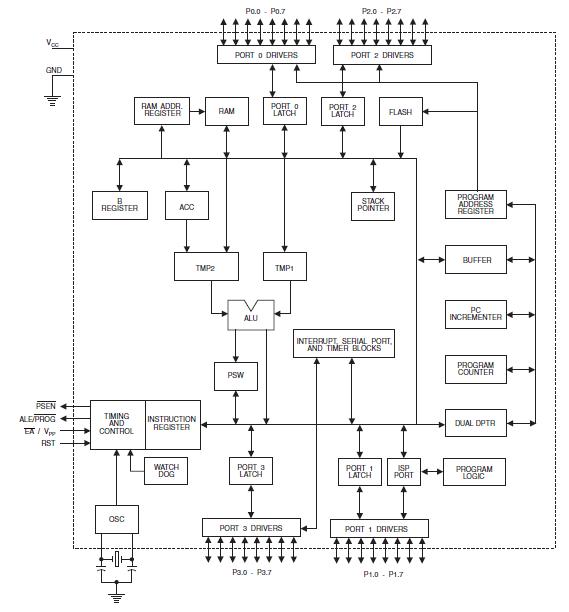
****

Fig 4.5(a): Block Diagram Of AT89S52

**Pin Configurations of AT89S52**

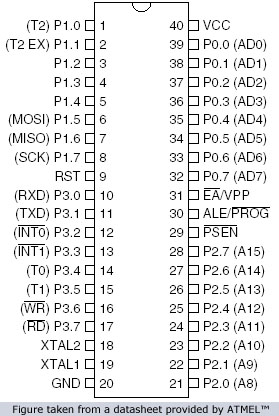


FIG 4.5(b): PIN DIAGRAM OF AT89S52

**Pin Description:**

**VCC:**

Supply voltage.

**GND:**

Ground

**Port 0:**

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

**Port 1:**

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

**Port 2:**

Port 2 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 2 output buffers can sink/source four TTL inputs. When 1s are written to Port 2 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current (IIL) because of the internal pull-ups. Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that uses 16-bit addresses (MOVX @ DPTR). In this application, Port 2 uses strong internal pull-ups when emitting 1s. During accesses to external data memory that uses 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register.

**Port 3:**

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

**RST:**

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

**ALE/PROG:**

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

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

**PSEN:**

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

**EA/VPP:**

External Access Enable. EA must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH. Note, however, that if lock bit 1 is programmed, EA will be internally latched on reset. EA should be strapped to VCC for internal program executions. This pin also receives the 12-volt programming enable voltage (VPP) during Flash programming.

**XTAL1:**

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

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

**Oscillator Characteristics:**

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 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 6.2. 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.

****

FIG 4.5(c): Oscillator Connections



FIG 4.5(d): External Clock Drive Configuration

**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.

**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 VCC is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize.

**4.6 SEVEN SEGMENT DISPLAY**

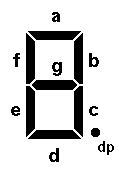
****

FIG 4.6(a): SEVEN SEGMENT DISPLAY LAY OUT

The illustration to the right shows the basic layout of the segments in a seven-segment display. The segments themselves are identified with lower-case letters "a" through "g," with segment "a" at the top and then counting clockwise. Segment "g" is the centre bar.

Most seven-segment digits also include a decimal point ("dp"), and some also include an extra triangle to turn the decimal point into a comma. This improves readability of large numbers on a calculator, for example. The decimal point is shown here on the right, but some display units put it on the left, or have a decimal point on each side.

In addition, most displays are actually slanted a bit, making them look as if they were in italics. This arrangement allows us to turn one digit upside down and place it next to another, so that the two decimal points look like a colon between the two digits. The technique is commonly used in LED clock displays.

****

FIG 4.10(b): PACKAGES OF SEVEN SEGMENT DISPLAY

Seven-segment displays can be packaged in a number of ways. Three typical packages are shown above. On the left we see three small digits in a single 12-pin DIP package. The individual digits are very small, so a clear plastic bubble is molded over each digit to act as a magnifying lens. The sides of the end bubbles are flattened so that additional packages of this type can be placed end-to-end to create a display of as many digits as may be needed.

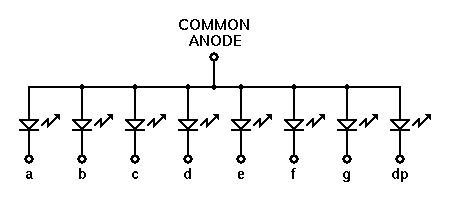
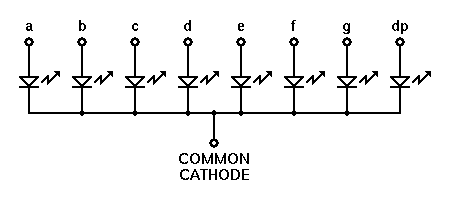
The second package is essentially a 14-pin DIP designed to be installed vertically. Note that for this particular device, the decimal point is on the left. This is not true of all seven-segment displays in this type of package.

One limitation of the DIP package is that it cannot support larger digits. To get larger displays for easy reading at a distance, it is necessary to change the package size and shape. The package on the right above is larger than the other two, and thus can display a digit that is significantly larger than will fit on a standard DIP footprint. Even larger displays are also available; some digital clocks sport digits that are two to five inches tall.

**CONSTRUCTION:**

Seven-segment displays can be constructed using any of a number of different technologies. The three most common methods are fluorescent displays (used in many line-powered devices such as microwave ovens and some clocks and clock radios), liquid crystal displays (used in many battery-powered devices such as watches and many digital instruments), and LED’S (used in either line-powered or battery-powered devices). However, fluorescent displays require a fairly high driving voltage to operate, and liquid crystal displays require special treatment that we are not yet ready to discuss. Therefore, we will work with a seven-segment LED display in this experiment.

Schematic Diagram

****

As shown in the two schematic diagrams above, the LED’S in a seven-segment display are not isolated from each other. Rather, either all of the cathodes, or all of the anodes, are connected together into a common lead, while the other end of each LED is individually available. This means fewer electrical connections to the package, and also allows us to easily enable or disable a particular digit by controlling the common lead. (In some cases, the common connections are made to groups of LED’S, and the external wiring must make the final connections between them. In other cases, the common connection is made available at more than one location for convenience in laying out printed circuit boards. When laying out circuits using such devices, you simply need to take the specific connection details into account.)There is no automatic advantage of the common-cathode seven-segment unit over the common-anode version, or vice-versa. Each type lends itself to certain applications, configurations, and logic families.

**4.7 KEYPAD**



A **keypad** is a set of buttons arranged in a block or "pad" which usually bear digits, symbols and usually a complete set of alphabetical letters. If it mostly contains numbers then it can also be called a **numeric keypad**. Keypads are found on many [alphanumeric keyboards](http://en.wikipedia.org/wiki/Alphanumeric_keyboard) and on other devices such as [calculators](http://en.wikipedia.org/wiki/Calculators), [push-button telephones](http://en.wikipedia.org/wiki/Push-button_telephone), [combination locks](http://en.wikipedia.org/wiki/Combination_locks), and [digital door locks](http://en.wikipedia.org/wiki/Digital_door_lock), which require mainly numeric input.

Keypads are a part of HMI or Human Machine Interface and play really important role in a small embedded system where human interaction or human input is needed. Matrix keypads are well known for their simple architecture and ease of interfacing with any microcontroller.

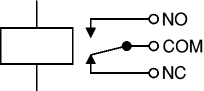
FIG 4.3: MATRIX KEYPAD

**SCANNING OF MATRIX KEYPAD:**

There are many methods depending on the connection keypad with micro controller, but the basic logic is same the columns are made as input and drive the rows making them as output; this whole procedure of reading the keyboard is called scanning. In order to detect which key is pressed from the matrix, the row lines are to be made low one by one and read the columns. Assume that if Row1 is made low, then read the columns. If any of the key in row1 is pressed then correspondingly the column 1will give low that is if second key is pressed in Row1, then column2 will give low. This is how Scanning is done. So to scan the keypad completely, we need to make rows low one by one and read the columns. If any of the buttons are pressed in a row, it will take the corresponding column to a low state which shows that a key is pressed in that row. If button 1 of a row is pressed then Column 1 will become low, if button 2 then column2 and so on...this is the way of working by a keypad.

**4.8 RELAY**

A relay is an electrically operated switch. Many relays use an electromagnet to operate a switching mechanism mechanically, but other operating principles are also used. Relays are used where it is necessary to control a circuit by a low-power signal (with complete electrical isolation between control and controlled circuits), or where several circuits must be controlled by one signal.

A relay is an electrically operated switch. Current flowing through the coil of the relay creates a magnetic field which attracts a lever and changes the switch contacts. The coil current can be on or off so relays have two switch positions and most have double throw (changeover) switch contacts as shown in the diagram.

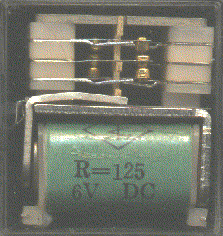


Fig 4.8 Relay showing coil and switch contacts

Relays allow one circuit to switch a second circuit which can be completely separate from the first. For example a low voltage battery circuit can use a relay to switch a 230V AC mains circuit. There is no electrical connection inside the relay between the two circuits; the link is magnetic and mechanical.

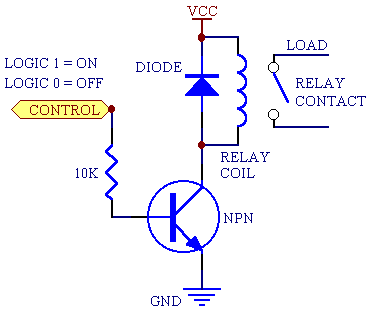
The coil of a relay passes a relatively large current, typically 30mA for a 12V relay, but it can be as much as 100mA for relays designed to operate from lower voltages. Most ICs (chips) cannot provide this current and a transistor is usually used to amplify the small IC current to the larger value required for the relay coil. The maximum output current for the popular 555 timer IC is 200mA so these devices can supply relay coils directly without amplification.

Relays are usually SPDT or DPDT but they can have many more sets of switch contacts, for example relays with 4 sets of changeover contacts are readily available. For further information about switch contacts and the terms used to describe them please see the page on switches.

Most relays are designed for PCB mounting but you can solder wires directly to the pins providing you take care to avoid melting the plastic case of the relay.

The supplier's catalogue should show you the relay's connections. The coil will be obvious and it may be connected either way round. Relay coils produce brief high voltage 'spikes' when they are switched off and this can destroy transistors and ICs in the circuit. To prevent damage you must connect a protection diode across the relay coil.

The figure shows a relay with its coil and switch contacts. You can see a lever on the left being attracted by magnetism when the coil is switched on. This lever moves the switch contacts.



There is one set of contacts (SPDT) in the foreground and another behind them, making the relay DPDT.

The relay's switch connections are usually labelled COM, NC and NO:

* COM = Common, always connect to this; it is the moving part of the switch.
* NC = Normally Closed, COM is connected to this when the relay coil is off.
* NO = Normally Open, COM is connected to this when the relay coil is on.

**Applications of relays**

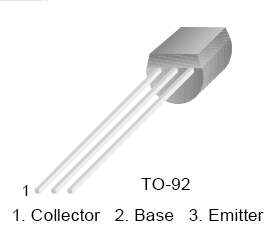
Relays are used to and for:

* Control a high-voltage circuit with a low-voltage signal, as in some types of modems or audio amplifiers.
* Control a high-current circuit with a low-current signal, as in the starter solenoid of an automobile.
* Detect and isolate faults on transmission and distribution lines by opening and closing circuit breakers.
* Time delay functions. Relays can be modified to delay opening or delay closing a set of contacts. A very short (a fraction of a second) delay would use a copper disk between the armature and moving blade assembly. Current flowing in the disk maintains magnetic field for a short time, lengthening release time. For a slightly longer (up to a minute) delay, a dashpot is used. A dashpot is a piston filled with fluid that is allowed to escape slowly. The time period can be varied by increasing or decreasing the flow rate. For longer time periods, a mechanical clockwork timer is installed.

**4.9 BC547**

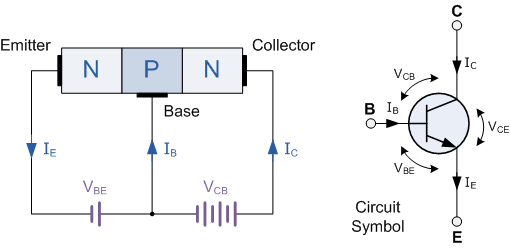
**TECHNICAL SPECIFICATIONS:**

The BC547 transistor is an NPN Epitaxial Silicon Transistor. The BC547 transistor is a general-purpose transistor in small plastic packages. It is used in general-purpose switching and amplification BC847/BC547 series 45 V, 100 mA NPN general-purpose transistors.



BC 547 TRANSISTOR PINOUTS

We know that the transistor is a "CURRENT" operated device and that a large current (Ic) flows freely through the device between the collector and the emitter terminals. However, this only happens when a small biasing current (Ib) is flowing into the base terminal of the transistor thus allowing the base to act as a sort of current control input. The ratio of these two currents (Ic/Ib) is called the DC Current Gain of the device and is given the symbol of hfe or nowadays Beta, (β). Beta has no units as it is a ratio. Also, the current gain from the emitter to the collector terminal, Ic/Ie, is called Alpha, (α), and is a function of the transistor itself. As the emitter current Ie is the product of a very small base current to a very large collector current the value of this parameter α is very close to unity, and for a typical low-power signal transistor this value ranges from about 0.950 to 0.999.

**An NPN Transistor Configuration**

**4.10 1N4007**

Diodes are used to convert AC into DC these are used as half wave rectifier or full wave rectifier. Three points must he kept in mind while using any type of diode.

1. Maximum forward current capacity
2. Maximum reverse voltage capacity
3. Maximum forward voltage capacity

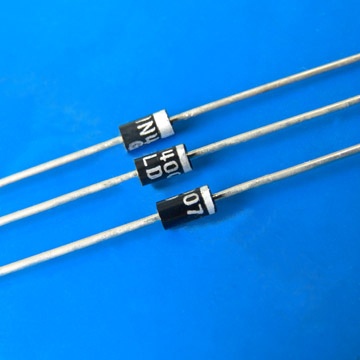


Fig: 1N4007 diodes

The number and voltage capacity of some of the important diodes available in the market are as follows:

* Diodes of number IN4001, IN4002, IN4003, IN4004, IN4005, IN4006 and IN4007 have maximum reverse bias voltage capacity of 50V and maximum forward current capacity of 1 Amp.
* Diode of same capacities can be used in place of one another. Besides this diode of more capacity can be used in place of diode of low capacity but diode of low capacity cannot be used in place of diode of high capacity. For example, in place of IN4002; IN4001 or IN4007 can be used but IN4001 or IN4002 cannot be used in place of IN4007.The diode BY125made by company BEL is equivalent of diode from IN4001 to IN4003. BY 126 is equivalent to diodes IN4004 to 4006 and BY 127 is equivalent to diode IN4007.

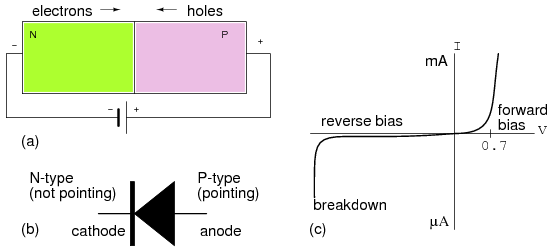


Fig:PN Junction diode

PN JUNCTION OPERATION

Now that you are familiar with P- and N-type materials, how these materials are joined together to form a diode, and the function of the diode, let us continue our discussion with the operation of the PN junction. But before we can understand how the PN junction works, we must first consider current flow in the materials that make up the junction and what happens initially within the junction when these two materials are joined together.

Current Flow in the N-Type Material

Conduction in the N-type semiconductor, or crystal, is similar to conduction in a copper wire. That is, with voltage applied across the material, electrons will move through the crystal just as current would flow in a copper wire. This is shown in figure 1-15. The positive potential of the battery will attract the free electrons in the crystal. These electrons will leave the crystal and flow into the positive terminal of the battery. As an electron leaves the crystal, an electron from the negative terminal of the battery will enter the crystal, thus completing the current path. Therefore, the majority current carriers in the N-type material (electrons) are repelled by the negative side of the battery and move through the crystal toward the positive side of the battery.

Current Flow in the P-Type Material

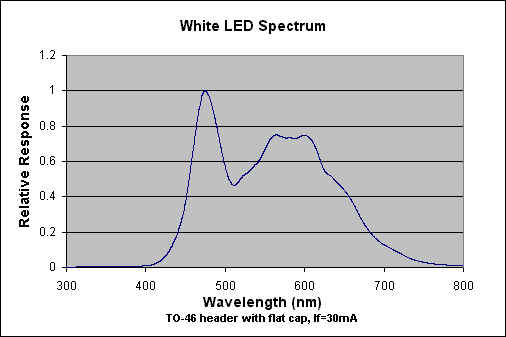
Current flow through the P-type material is illustrated. Conduction in the P material is by positive holes, instead of negative electrons. A hole moves from the positive terminal of the P material to the negative terminal. Electrons from the external circuit enter the negative terminal of the material and fill holes in the vicinity of this terminal. At the positive terminal, electrons are removed from the covalent bonds, thus creating new holes. This process continues as the steady stream of holes (hole current) moves toward the negative terminal

**4.11 LED**

Light Emitting Diodes (LED) have recently become available that are white and bright, so bright that they seriously compete with incandescent lamps in lighting applications. They are still pretty expensive as compared to a GOW lamp but draw much less current and project a fairly well focused beam.

The diode in the photo came with a neat little reflector that tends to sharpen the beam a little but doesn't seem to add much to the overall intensity.

When run within their ratings, they are more reliable than lamps as well. Red LEDs are now being used in automotive and truck tail lights and in red traffic signal lights. You will be able to detect them because they look like an array of point sources and they go on and off instantly as compared to conventional incandescent lamps.



LEDs are monochromatic (one color) devices. The color is determined by the band gap of the semiconductor used to make them. Red, green, yellow and blue LEDs are fairly common. White light contains all colors and cannot be directly created by a single LED. The most common form of "white" LED really isn't white. It is a Gallium Nitride blue LED coated with a phosphor that, when excited by the blue LED light, emits a broad range spectrum that in addition to the blue emission, makes a fairly white light.

There is [a claim](http://www.bivar.com/bullethtml/bull-nov2.htm) that these white LED's have a limited life. After 1000 hours or so of operation, they tend to yellow and dim to some extent. Running the LEDs at more than their rated current will certainly accelerate this process.

There are two primary ways of producing high intensity white-light using LED’S. One is to use individual LED’S that emit three [primary colours](http://en.wikipedia.org/wiki/Primary_color)—red, green, and blue—and then mix all the colours to form white light. The other is to use a phosphor material to convert monochromatic light from a blue or UV LED to broad-spectrum white light, much in the same way a fluorescent light bulb works. Due to [metamerism](http://en.wikipedia.org/wiki/Metamerism_(color)), it is possible to have quite different spectra that appear white.

LEDs are semiconductor devices. Like transistors, and other diodes, LEDs are made out of silicon. What makes an LED give off light are the small amounts of chemical impurities that are added to the silicon, such as gallium, arsenide, indium, and nitride.

When current passes through the LED, it emits photons as a byproduct. Normal light bulbs produce light by heating a metal filament until it is white hot. LEDs produce photons directly and not via heat, they are far more efficient than incandescent bulbs.

LED circuit symbol

Fig 3.1(a): circuit symbol

Not long ago LEDs were only bright enough to be used as indicators on dashboards or electronic equipment. But recent advances have made LEDs bright enough to rival traditional lighting technologies. Modern LEDs can replace incandescent bulbs in almost any application.

**Types of LED’S**

LEDs are produced in an array of shapes and sizes. The 5 mm cylindrical package is the most common, estimated at 80% of world production. The color of the plastic lens is often the same as the actual color of light emitted, but not always. For instance, purple plastic is often used for infrared LEDs, and most blue devices have clear housings. There are also LEDs in extremely tiny packages, such as those found on blinkers and on cell phone keypads. The main types of LEDs are miniature, high power devices and custom designs such as alphanumeric or multi-color.

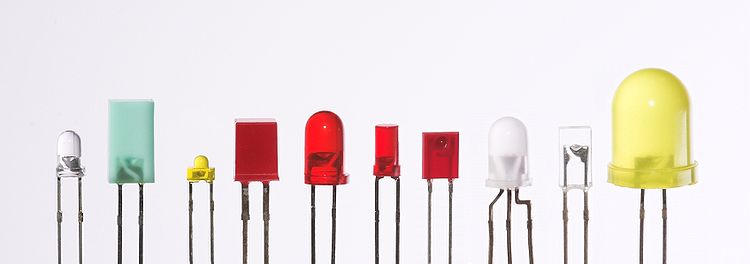
**[](http://en.wikipedia.org/wiki/File:Verschiedene_LEDs.jpg)**

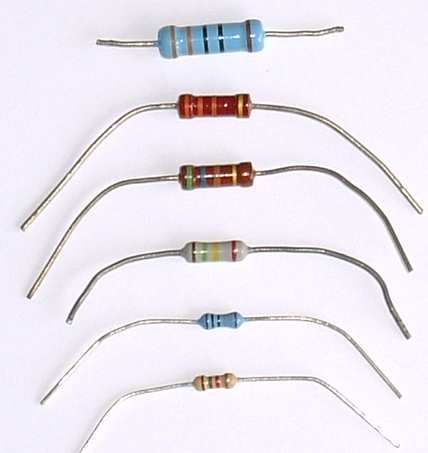
Fig 3.1(b) Different types of LED’S

**4.12 RESISTORS**

A resistor is a two-terminal electronic component designed to oppose an electric current by producing a voltage drop between its terminals in proportion to the current, that is, in accordance with Ohm's law:

V = IR

Resistors are used as part of electrical networks and electronic circuits. They are extremely commonplace in most electronic equipment. Practical resistors can be made of various compounds and films, as well as resistance wire (wire made of a high-resistivity alloy, such as nickel/chrome).

****

The primary characteristics of resistors are their resistance and the power they can dissipate. Other characteristics include temperature coefficient, noise, and inductance. Less well-known is critical resistance, the value below which power dissipation limits the maximum permitted current flow, and above which the limit is applied voltage. Critical resistance depends upon the materials constituting the resistor as well as its physical dimensions; it's determined by design.

Resistors can be integrated into hybrid and printed circuits, as well as integrated circuits. Size, and position of leads (or terminals) are relevant to equipment designers; resistors must be physically large enough not to overheat when dissipating their power.

A resistor is a two-[terminal](http://en.wikipedia.org/wiki/Terminal_%28electronics%29) [passive](http://en.wikipedia.org/wiki/Passivity_%28engineering%29) [electronic component](http://en.wikipedia.org/wiki/Electronic_component) which implements [electrical resistance](http://en.wikipedia.org/wiki/Electrical_resistance) as a circuit element. When a voltage V is applied across the terminals of a resistor, a current I will flow through the resistor in [direct proportion](http://en.wikipedia.org/wiki/Direct_proportion) to that voltage. The reciprocal of the constant of proportionality is known as the [resistance](http://en.wikipedia.org/wiki/Resistance) R, since, with a given voltage V, a larger value of R further "resists" the flow of current I as given by [Ohm's law](http://en.wikipedia.org/wiki/Ohm%27s_law):

I = {V \over R}

Resistors are common elements of [electrical networks](http://en.wikipedia.org/wiki/Electrical_networks) and electronic circuits and are ubiquitous in most electronic equipment. Practical resistors can be made of various compounds and films, as well as [resistance wire](http://en.wikipedia.org/wiki/Resistance_wire) (wire made of a high-resistivity alloy, such as nickel-chrome). Resistors are also implemented within [integrated circuits](http://en.wikipedia.org/wiki/Integrated_circuits), particularly analog devices, and can also be integrated into [hybrid](http://en.wikipedia.org/wiki/Hybrid_circuit) and [printed circuits](http://en.wikipedia.org/wiki/Printed_circuit_board).

The electrical functionality of a resistor is specified by its resistance: common commercial resistors are manufactured over a range of more than 9 [orders of magnitude](http://en.wikipedia.org/wiki/Orders_of_magnitude). When specifying that resistance in an electronic design, the required precision of the resistance may require attention to the [manufacturing tolerance](http://en.wikipedia.org/wiki/Engineering_tolerance#Electrical_component_tolerance) of the chosen resistor, according to its specific application. The [temperature coefficient](http://en.wikipedia.org/wiki/Temperature_coefficient) of the resistance may also be of concern in some precision applications. Practical resistors are also specified as having a maximum [power](http://en.wikipedia.org/wiki/Power_%28physics%29) rating which must exceed the anticipated power dissipation of that resistor in a particular circuit: this is mainly of concern in power electronics applications. Resistors with higher power ratings are physically larger and may require [heat sinking](http://en.wikipedia.org/wiki/Heat_sink). In a high voltage circuit, attention must sometimes be paid to the rated maximum working voltage of the resistor.

The series [inductance](http://en.wikipedia.org/wiki/Inductance) of a practical resistor causes its behavior to depart from ohms law; this specification can be important in some high-frequency applications for smaller values of resistance. In a [low-noise amplifier](http://en.wikipedia.org/wiki/Low-noise_amplifier) or [pre-amp](http://en.wikipedia.org/wiki/Pre-amp) the noise characteristics of a resistor may be an issue. The unwanted inductance, excess noise, and temperature coefficient are mainly dependent on the technology used in manufacturing the resistor. They are not normally specified individually for a particular family of resistors manufactured using a particular technology.[[1]](http://en.wikipedia.org/wiki/Resistor#cite_note-0) A family of discrete resistors is also characterized according to its form factor, that is, the size of the device and position of its leads (or terminals) which is relevant in the practical manufacturing of circuits using them.

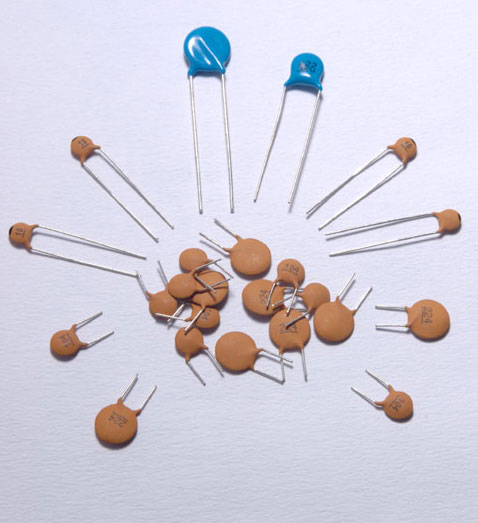
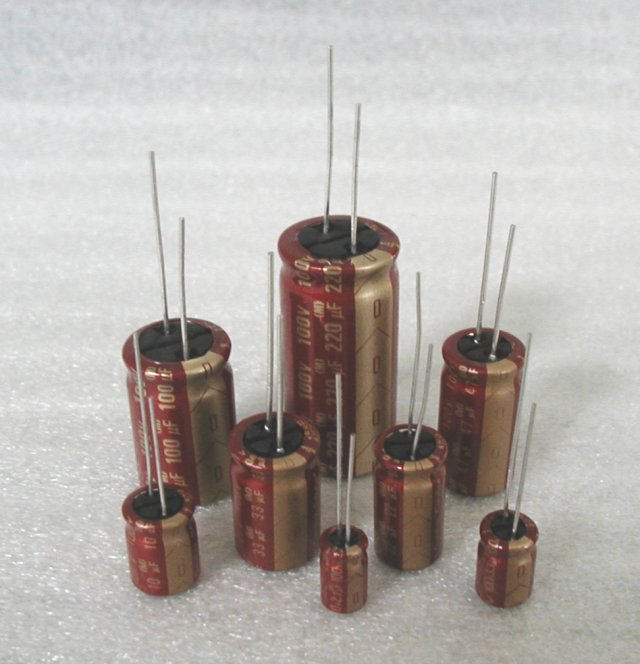
## Units

The [ohm](http://en.wikipedia.org/wiki/Ohm_%28unit%29) (symbol: [Ω](http://en.wikipedia.org/wiki/%CE%A9)) is the [SI](http://en.wikipedia.org/wiki/SI) unit of [electrical resistance](http://en.wikipedia.org/wiki/Electrical_resistance), named after [Georg Simon Ohm](http://en.wikipedia.org/wiki/Georg_Simon_Ohm). An ohm is equivalent to a [volt](http://en.wikipedia.org/wiki/Volt) per [ampere](http://en.wikipedia.org/wiki/Ampere). Since resistors are specified and manufactured over a very large range of values, the derived units of milliohm (1 mΩ = 10−3 Ω), kilohm (1 kΩ = 103 Ω), and megohm (1 MΩ = 106 Ω) are also in common usage.

The reciprocal of resistance R is called [conductance](http://en.wikipedia.org/wiki/Conductance) G = 1/R and is measured in [Siemens](http://en.wikipedia.org/wiki/Siemens_%28unit%29) ([SI](http://en.wikipedia.org/wiki/SI) unit), sometimes referred to as a [mho](http://en.wikipedia.org/wiki/Mho). Thus a Siemens is the reciprocal of an ohm: *S* = Ω − 1. Although the concept of conductance is often used in circuit analysis, practical resistors are always specified in terms of their resistance (ohms) rather than conductance.

**4.13 CAPACITORS**

capacitor or condenser is a passive electronic component consisting of a pair of conductors separated by a dielectric. When a voltage potential difference exists between the conductors, an electric field is present in the dielectric. This field stores energy and produces a mechanical force between the plates. The effect is greatest between wide, flat, parallel, narrowly separated cond



An ideal capacitor is characterized by a single constant value, capacitance, which is measured in farads. This is the ratio of the electric charge on each conductor to the potential difference between them. In practice, the dielectric between the plates passes a small amount of leakage current. The conductors and leads introduce an equivalent series resistance and the dielectric has an electric field strength limit resulting in a breakdown voltage.

The properties of capacitors in a circuit may determine the resonant frequency and quality factor of a resonant circuit, power dissipation and operating frequency in a digital logic circuit, energy capacity in a high-power system, and many other important aspects.

A capacitor (formerly known as condenser) is a device for storing electric charge. The forms of practical capacitors vary widely, but all contain at least two conductors separated by a non-conductor. Capacitors used as parts of electrical systems, for example, consist of metal foils separated by a layer of insulating film.

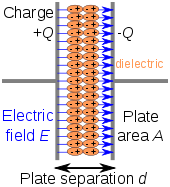
Capacitors are widely used in electronic circuits for blocking [direct current](http://en.wikipedia.org/wiki/Direct_current) while allowing [alternating current](http://en.wikipedia.org/wiki/Alternating_current) to pass, in filter networks, for smoothing the output of [power supplies](http://en.wikipedia.org/wiki/Power_supply), in the [resonant circuits](http://en.wikipedia.org/wiki/LC_circuit) that tune radios to particular [frequencies](http://en.wikipedia.org/wiki/Frequency) and for many other purposes.

A capacitor is a [passive](http://en.wikipedia.org/wiki/Passivity_%28engineering%29) [electronic component](http://en.wikipedia.org/wiki/Electronic_component) consisting of a pair of [conductors](http://en.wikipedia.org/wiki/Electrical_conductor) separated by a [dielectric](http://en.wikipedia.org/wiki/Dielectric) (insulator). When there is a [potential difference](http://en.wikipedia.org/wiki/Potential_difference) (voltage) across the conductors, a static [electric field](http://en.wikipedia.org/wiki/Electric_field) develops in the dielectric that stores [energy](http://en.wikipedia.org/wiki/Energy) and produces a mechanical force between the conductors. An ideal capacitor is characterized by a single constant value, [capacitance](http://en.wikipedia.org/wiki/Capacitance), measured in [farads](http://en.wikipedia.org/wiki/Farad). This is the ratio of the [electric charge](http://en.wikipedia.org/wiki/Electric_charge) on each conductor to the potential difference between them.

The capacitance is greatest when there is a narrow separation between large areas of conductor, hence capacitor conductors are often called "plates", referring to an early means of construction. In practice the dielectric between the plates passes a small amount of [leakage current](http://en.wikipedia.org/wiki/Leakage_%28electronics%29) and also has an electric field strength limit, resulting in a [breakdown voltage](http://en.wikipedia.org/wiki/Breakdown_voltage), while the conductors and [leads](http://en.wikipedia.org/wiki/Lead_%28electronics%29) introduce an undesired [inductance](http://en.wikipedia.org/wiki/Equivalent_series_inductance) and [resistance](http://en.wikipedia.org/wiki/Equivalent_series_resistance).

## Theory of operation

Main article: [Capacitance](http://en.wikipedia.org/wiki/Capacitance)

[](http://en.wikipedia.org/wiki/File:Capacitor_schematic_with_dielectric.svg)

Charge separation in a parallel-plate capacitor causes an internal electric field. A dielectric (orange) reduces the field and increases the capacitance.

[](http://en.wikipedia.org/wiki/File:Plattenkondensator_hg.jpg)

A simple demonstration of a parallel-plate capacitor

A capacitor consists of two [conductors](http://en.wikipedia.org/wiki/Electrical_conductor) separated by a non-conductive region[[8]](http://en.wikipedia.org/wiki/Capacitor#cite_note-Ulaby_p168-7). The non-conductive region is called the dielectric or sometimes the [dielectric medium](http://en.wikipedia.org/wiki/Dielectric_medium). In simpler terms, the dielectric is just an [electrical insulator](http://en.wikipedia.org/wiki/Insulator_%28electrical%29). Examples of dielectric mediums are glass, air, paper, [vacuum](http://en.wikipedia.org/wiki/Vacuum), and even a [semiconductor](http://en.wikipedia.org/wiki/Semiconductor) [depletion region](http://en.wikipedia.org/wiki/Depletion_region) chemically identical to the conductors. A capacitor is assumed to be self-contained and isolated, with no net [electric charge](http://en.wikipedia.org/wiki/Electric_charge) and no influence from any external electric field. The conductors thus hold equal and opposite charges on their facing surfaces,[[9]](http://en.wikipedia.org/wiki/Capacitor#cite_note-Ulaby_p157-8) and the dielectric develops an electric field. In [SI](http://en.wikipedia.org/wiki/SI) units, a capacitance of one [farad](http://en.wikipedia.org/wiki/Farad) means that one [coulomb](http://en.wikipedia.org/wiki/Coulomb) of charge on each conductor causes a voltage of one [volt](http://en.wikipedia.org/wiki/Volt) across the device.[[10]](http://en.wikipedia.org/wiki/Capacitor#cite_note-Ulaby_p169-9)

The capacitor is a reasonably general model for electric fields within electric circuits. An ideal capacitor is wholly characterized by a constant capacitance C, defined as the ratio of charge ±Q on each conductor to the voltage V between them:[[8]](http://en.wikipedia.org/wiki/Capacitor#cite_note-Ulaby_p168-7)

C= \frac{Q}{V}

Sometimes charge build-up affects the capacitor mechanically, causing its capacitance to vary. In this case, capacitance is defined in terms of incremental changes:

C= \frac{\mathrm{d}q}{\mathrm{d}v}

### Energy storage

[Work](http://en.wikipedia.org/wiki/Work_%28thermodynamics%29) must be done by an external influence to "move" charge between the conductors in a capacitor. When the external influence is removed the charge separation persists in the electric field and energy is stored to be released when the charge is allowed to return to its [equilibrium](http://en.wikipedia.org/wiki/Equilibrium) position. The work done in establishing the electric field, and hence the amount of energy stored, is given by:[[11]](http://en.wikipedia.org/wiki/Capacitor#cite_note-10)

W= \int_{q=0}^Q V \text{d}q = \int_{q=0}^Q \frac{q}{C} \text{d}q = {1 \over 2} {Q^2 \over C} = {1 \over 2}  C V^2 = {1 \over 2} VQ.

### Current-voltage relation

The current i(t) through any component in an electric circuit is defined as the rate of flow of a charge q(t) passing through it, but actual charges, electrons, cannot pass through the dielectric layer of a capacitor, rather an electron accumulates on the negative plate for each one that leaves the positive plate, resulting in an electron depletion and consequent positive charge on one electrode that is equal and opposite to the accumulated negative charge on the other. Thus the charge on the electrodes is equal to the [integral](http://en.wikipedia.org/wiki/Integral) of the current as well as proportional to the voltage as discussed above. As with any [antiderivative](http://en.wikipedia.org/wiki/Antiderivative), a [constant of integration](http://en.wikipedia.org/wiki/Constant_of_integration) is added to represent the initial voltage v (t0). This is the integral form of the capacitor equation,[[12]](http://en.wikipedia.org/wiki/Capacitor#cite_note-Dorf_p263-11)

v(t)= \frac{q(t)}{C} = \frac{1}{C}\int_{t_0}^t i(\tau) \mathrm{d}\tau+v(t_0).

Taking the derivative of this, and multiplying by C, yields the derivative form,[[13]](http://en.wikipedia.org/wiki/Capacitor#cite_note-Dorf_p260-12)

i(t)= \frac{\mathrm{d}q(t)}{\mathrm{d}t}=C\frac{\mathrm{d}v(t)}{\mathrm{d}t}.

The [dual](http://en.wikipedia.org/wiki/Duality_%28electrical_circuits%29) of the capacitor is the [inductor](http://en.wikipedia.org/wiki/Inductor), which stores energy in the [magnetic field](http://en.wikipedia.org/wiki/Magnetic_field) rather than the electric field. Its current-voltage relation is obtained by exchanging current and voltage in the capacitor equations and replacing C with the inductance L.

**5. SOFTWARE REQUIREMENTS**

**5.1 INTRODUCTION TO KEIL MICRO VISION (IDE)**

Keil an ARM Company makes C compilers, macro assemblers, real-time kernels, debuggers, simulators, integrated environments, evaluation boards, and emulators for ARM7/ARM9/Cortex-M3, XC16x/C16x/ST10, 251, and 8051 MCU families.

Keil development tools for the 8051 Microcontroller Architecture support every level of software developer from the professional applications engineer to the student just learning about embedded software development. When starting a new project, simply select the microcontroller you use from the Device Database and the µVision IDE sets all compiler, assembler, linker, and memory options for you.

  Keil is a cross compiler. So first we have to understand the concept of compilers and cross compilers. After then we shall learn how to work with keil.

**5.2 CONCEPT OF COMPILER**

Compilers are programs used to convert a High Level Language to object code. Desktop compilers produce an output object code for the underlying microprocessor, but not for other microprocessors. I.E the programs written in one of the HLL like ‘C’ will compile the code to run on the system for a particular processor like x86 (underlying microprocessor in the computer). For example compilers for Dos platform is different from the Compilers for Unix platform  So if one wants to define a compiler then compiler is a program that translates source code into object code.

The compiler derives its name from the way it works, looking at the entire piece of source code and collecting and reorganizing the instruction. See there is a bit little difference between compiler and an interpreter. Interpreter just interprets whole program at a time while compiler analyses and execute each line of source code in succession, without looking at the entire program.

The advantage of interpreters is that they can execute a program immediately. Secondly programs produced by compilers run much faster than the same programs executed by an interpreter. However compilers require some time before an executable program emerges. Now as compilers translate source code into object code, which is unique for each type of computer, many compilers are available for the same language.

**5.3 CONCEPT OF CROSS COMPILER**

  A cross compiler is similar to the compilers but we write a program for the target processor (like 8051 and its derivatives) on the host processors (like computer of x86). It means being in one environment you are writing a code for another environment is called cross development. And the compiler used for cross development is called cross compiler. So the definition of cross compiler is a compiler that runs on one computer but produces object code for a different type of computer.

**5.4 KEIL C CROSS COMPILER**

Keil is a German based Software development company. It provides several development tools like

•         IDE (Integrated Development environment)

•         Project Manager

•         Simulator

•         Debugger

•         C Cross Compiler, Cross Assembler, Locator/Linker

The Keil ARM tool kit includes three main tools, assembler, compiler and linker. An assembler is used to assemble the ARM assembly program. A compiler is used to compile the C source code into an object file. A linker is used to create an absolute object module suitable for our in-circuit emulator.

5.5 Building an Application in µVision2

To build (compile, assemble, and link) an application in µVision2, you must:

1. Select Project -(forexample,166\EXAMPLES\HELLO\HELLO.UV2).
2. Select Project - Rebuild all target files or Build target.µVision2 compiles, assembles, and links the files in your project.

5.6 Creating Your Own Application in µVision2

To create a new project in µVision2, you must:

1. Select Project - New Project.
2. Select a directory and enter the name of the project file.
3. Select Project - Select Device and select an 8051, 251, or C16x/ST10 device from the Device Database™.
4. Create source files to add to the project.
5. Select Project - Targets, Groups, Files. Add/Files, select Source Group1, and add the source files to the project.
6. Select Project - Options and set the tool options. Note when you select the target device from the Device Database™ all special options are set automatically. You typically only need to configure the memory map of your target hardware. Default memory model settings are optimal for most applications.
7. Select Project - Rebuild all target files or Build target.

5.7 Debugging an Application in µVision2

To debug an application created using µVision2, you must:

1. Select Debug - Start/Stop Debug Session.
2. Use the Step toolbar buttons to single-step through your program. You may enter G, main in the Output Window to execute to the main C function.
3. Open the Serial Window using the Serial #1 button on the toolbar.

Debug your program using standard options like Step, Go, Break, and so on.

**5.8 Starting µVision2 and Creating a Project**

µVision2 is a standard Windows application and started by clicking on the program icon. To create a new project file select from the µVision2 menu Project – New Project…. This opens a standard Windows dialog that asks you for the new project file name. We suggest that you use a separate folder for each project. You can simply use the icon Create New Folder in this dialog to get a new empty folder. Then select this folder and enter the file name for the new project, i.e. Project1. µVision2 creates a new project file with the name PROJECT1.UV2 which contains a default target and file group name. You can see these names in the Project.

**5.9 Window – Files.**

Now use from the menu Project – Select Device for Target and select a CPU for your project. The Select Device dialog box shows the µVision2 device data base. Just select the microcontroller you use. We are using for our examples the Philips 80C51RD+ CPU. This selection sets necessary tool Options for the 80C51RD+ device and simplifies in this way the tool Configuration.

**5.10 Building Projects and Creating a HEX Files**

Typical, the tool settings under Options – Target are all you need to start a new application. You may translate all source files and line the application with a click on the Build Target toolbar icon. When you build an application with syntax errors, µVision2 will display errors and warning messages in the Output Window – Build page. A double click on a message line opens the source file on the correct location in a µVision2 editor window. Once you have successfully generated your application you can start debugging.

After you have tested your application, it is required to create an Intel HEX file to download the software into an EPROM programmer or simulator. µVision2 creates HEX files with each build process when Create HEX files under Options for Target – Output is enabled. You may start your PROM programming utility after the make process when you specify the program under the option Run User Program #1.

**5.11 CPU Simulation**

µVision2 simulates up to 16 Mbytes of memory from which areas can be mapped for read, write, or code execution access. The µVision2 simulator traps

and reports illegal memory accesses. In addition to memory mapping, the simulator also provides support for the integrated peripherals of the various 8051 derivatives. The on-chip peripherals of the CPU you have selected are configured from the Device.

**5.12 Database selection**

You have made when you create your project target. Refer to page 58 for more Information about selecting a device. You may select and display the on-chip peripheral components using the Debug menu. You can also change the aspects of each peripheral using the controls in the dialog boxes.

**5.13 Start Debugging**

You start the debug mode of µVision2 with the Debug – Start/Stop Debug

Session Command. Depending on the Options for Target – Debug Configuration, µVision2 will load the application program and run the startup code µVision2 saves the editor screen layout and restores the screen layout of the last debug session. If the program execution stops, µVision2 opens an editor window with the source text or shows CPU instructions in the disassembly window. The next executable statement is marked with a yellow arrow. During debugging, most editor features are still available.

For example, you can use the find command or correct program errors. Program source text of your application is shown in the same windows. The µVision2 debug mode differs from the edit mode in the following aspects:

\_ The “Debug Menu and Debug Commands” described on page 28 are available. The additional debug windows are discussed in the following.

\_ The project structure or tool parameters cannot be modified. All build commands are disabled.

**5.14 Disassembly Window**

The Disassembly window shows your target program as mixed source and assembly program or just assembly code. A trace history of previously executed instructions may be displayed with Debug – View Trace Records. To enable the trace history, set Debug – Enable/Disable Trace Recording.

If you select the Disassembly Window as the active window all program step commands work on CPU instruction level rather than program source lines. You can select a text line and set or modify code breakpoints using toolbar buttons or the context menu commands.

You may use the dialog Debug – Inline Assembly… to modify the CPU instructions. That allows you to correct mistakes or to make temporary changes to the target program you are debugging. Numerous example programs are included to help you get started with the most popular embedded 8051 devices.

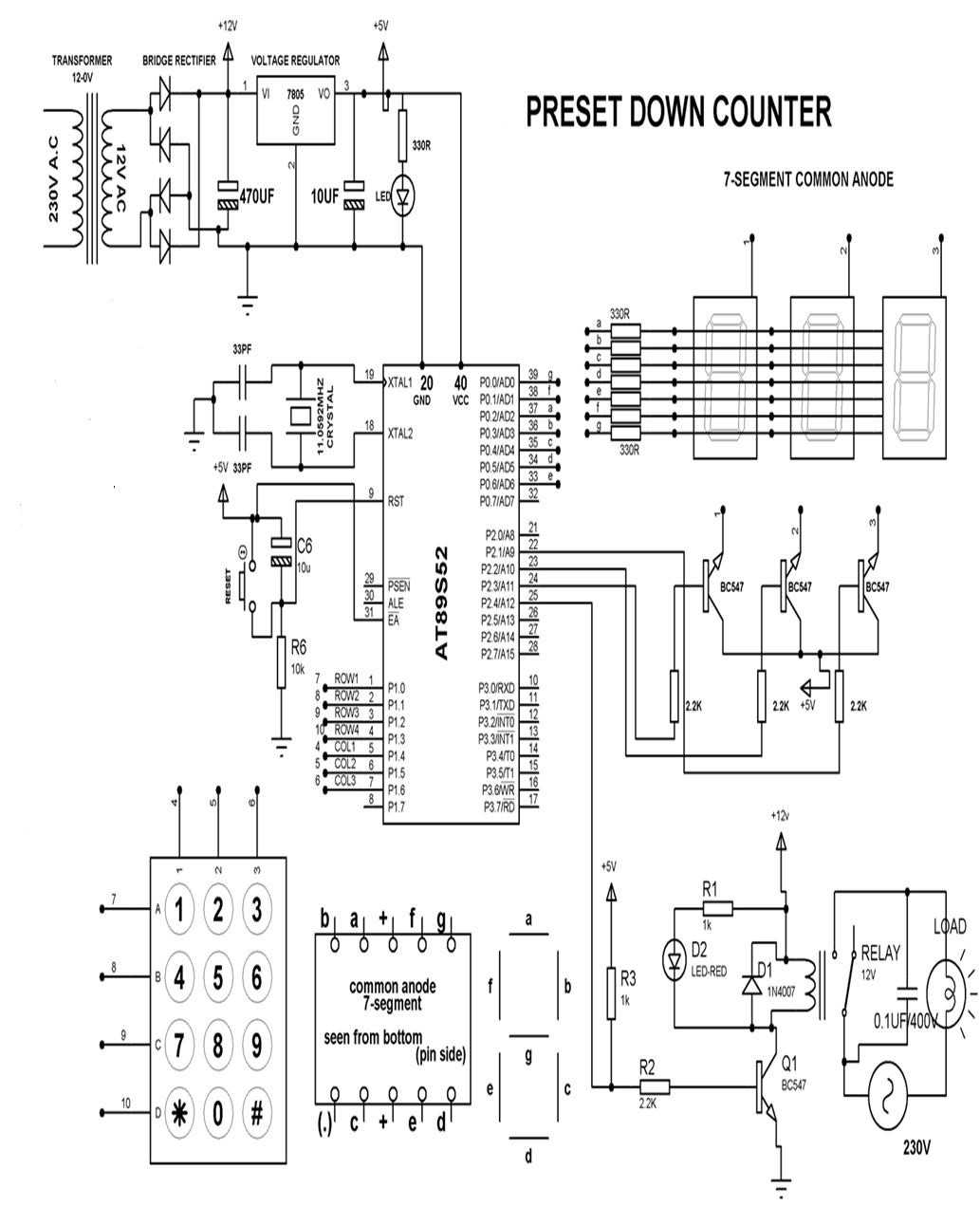
The Keil µVision Debugger accurately simulates on-chip peripherals (I²C, CAN, UART, SPI, Interrupts, I/O Ports, A/D Converter, D/A Converter, and PWM Modules) of your 8051 device. Simulation helps you understand hardware configurations and avoids time wasted on setup problems. Additionally, with simulation, you can write and test applications before target hardware is available.

**5.15 EMBEDDED C**

Use of embedded processors in passenger cars, mobile phones, medical equipment, aerospace systems and defense systems is widespread, and even everyday domestic appliances such as dish washers, televisions, washing machines and video recorders now include at least one such device.

Because most embedded projects have severe cost constraints, they tend to use low-cost processors like the 8051 family of devices considered in this book. These popular chips have very limited resources available most such devices have around 256 bytes (not megabytes!) of RAM, and the available processor power is around 1000 times less than that of a desktop processor. As a result, developing embedded software presents significant new challenges, even for experienced desktop programmers.

**6. SCHEMATIC DIAGRAM**



**6.1 DESCRIPTION**

**POWER SUPPLY**

The circuit uses standard power supply comprising of a step-down transformer from 230Vto 12V and 4 diodes forming a bridge rectifier that delivers pulsating dc which is then filtered by an electrolytic capacitor of about 470µF to 1000µF. The filtered dc being unregulated, IC LM7805 is used to get 5V DC constant at its pin no 3 irrespective of input DC varying from 7V to 15V. The input dc shall be varying in the event of input ac at 230volts section varies from 160V to 270V in the ratio of the transformer primary voltage V1 to secondary voltage V2 governed by the formula V1/V2=N1/N2. As N1/N2 i.e. no. of turns in the primary to the no. of turns in the secondary remains unchanged V2 is directly proportional to V1.Thus if the transformer delivers 12V at 220V input it will give 8.72V at 160V.Similarly at 270V it will give 14.72V.Thus the dc voltage at the input of the regulator changes from about 8V to 15V because of A.C voltage variation from 160V to 270V the regulator output will remain constant at 5V.

The regulated 5V DC is further filtered by a small electrolytic capacitor of 10µF for any noise so generated by the circuit. One LED is connected of this 5V point in series with a current limiting resistor of 330Ω to the ground i.e., negative voltage to indicate 5V power supply availability. The unregulated 12V point is used for other applications as and when required.

**STANDARD CONNECTIONS TO 8051 SERIES MICRO CONTROLLER**

ATMEL series of 8051 family of micro controllers need certain standard connections. The actual number of the Microcontroller could be “89C51” , “89C52”, “89S51”, “89S52”, and as regards to 20 pin configuration a number of “89C2051”. The 4 set of I/O ports are used based on the project requirement. Every microcontroller requires a timing reference for its internal program execution therefore an oscillator needs to be functional with a desired frequency to obtain the timing reference as t =1/f.

A crystal ranging from 2 to 20 MHz is required to be used at its pin number 18 and 19 for the internal oscillator. It may be noted here the crystal is not to be understood as crystal oscillator It is just a crystal, while connected to the appropriate pin of the microcontroller it results in oscillator function inside the microcontroller. Typically 11.0592 MHz crystal is used in general for most of the circuits using 8051 series microcontroller. Two small value ceramic capacitors of 33pF each is used as a standard connection for the crystal as shown in the circuit diagram.

**RESET**

Pin no 9 is provided with an re-set arrangement by a combination of an electrolytic capacitor and a register forming RC time constant. At the time of switch on, the capacitor gets charged, and it behaves as a full short circuit from the positive to the pin number 9. After the capacitor gets fully charged the current stops flowing and pin number 9 goes low which is pulled down by a 10k resistor to the ground. This arrangement of reset at pin 9 going high initially and then to logic 0 i.e., low helps the program execution to start from the beginning. In absence of this the program execution could have taken place arbitrarily anywhere from the program cycle. A pushbutton switch is connected across the capacitor so that at any given time as desired it can be pressed such that it discharges the capacitor and while released the capacitor starts charging again and then pin number 9 goes to high and then back to low, to enable the program execution from the beginning. This operation of high to low of the reset pin takes place in fraction of a second as decided by the time constant R and C.

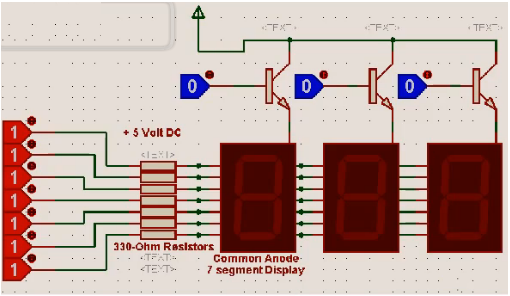
For example: A 10µF capacitor and a 10kΩ resistor would render a 100ms time to pin number 9 from logic high to low, there after the pin number 9 remains low.

**External Access(EA):**

Pin no 31 of 40 pin 8051 microcontroller termed as EA¯ is required to be connected to 5V for accessing the program form the on-chip program memory. If it is connected to ground then the controller accesses the program from external memory. However as we are using the internal memory it is always connected to +5V.

**MULTIPLE SEVEN SEGMENT DISPLAY**

In a seven segment display all 7segments of each one are connected in parallel making each of their anodes as common point. Switching transistors are connected to each seven segment display anode common points in order to make that display ON when logic 1 is given to the switching transistor then that display appears and when logic 0 is given then it does not appear. So if there are three displays ,three transistors are used to make them on or off at a very fast rate. For example if we want a display as 456 we switch on the 1st transistor by logic 1 while data bus sends logical commands of number 4 at the seven segments input for displaying 4 . While data for 5 comes on the data bus transistor 1 is switched off and transistor 2 is switched on so that 5 is displayed .Similarly for the 6 is displayed .As the transistor switching is synchronized with the incoming data and the transistor switch at very high speed from the microcontroller, all the digits like 456 in this case appear to be stationary. This is because of the persistence of vision of human eye not able to know that one unit only is glowing at a time



**OPERATION**

**CONNECTIONS**

The output of the power supply which is 5v is connected to 40 pin of microcontroller and GND is connected to its 20 pin. Port 1.0 to 1.3 of microcontroller are connected to Rows A, B, C, D of keypad and port 1.4 to 1.6 of microcontroller are connected to column’s 1, 2, 3 of keypad. Port 0.0 - 0.6 of microcontroller are given to common anodes (a-g) of multiple seven segment display. Port 2.1 to 2.4 of microcontroller are given to BC547 transistors collector pin. Port 2.4 of microcontroller is given to base terminal of transistor Q1.

WORKING

The project uses 3 common anode 7 segment displays in parallel connected to port ‘0’ as explained above. It uses a 4x3 matrix keypad duly connected to port 1 for selecting the no of operations thus while executed by the program provides a logic high at pin no 25 of the microcontroller to drive a transistor for operating a relay the contacts of which are used to switch ON & switch OFF the load under test.

**OPERATING PROCEDURE**

After power on all the 3 seven segment are in off position. Enter the desired 3 digit number that gets displayed on the 7 segment display. By pressing ‘\*’ (star) button the program in the microcontroller advances to decrement from the set number every one sec ,till it reaches zero. Each decrement develops drive pulse for the Q1 such that the load is switched ON & OFF for every one second. Finally after reaching zero it stops.

**7. LAYOUT DIAGRAM**

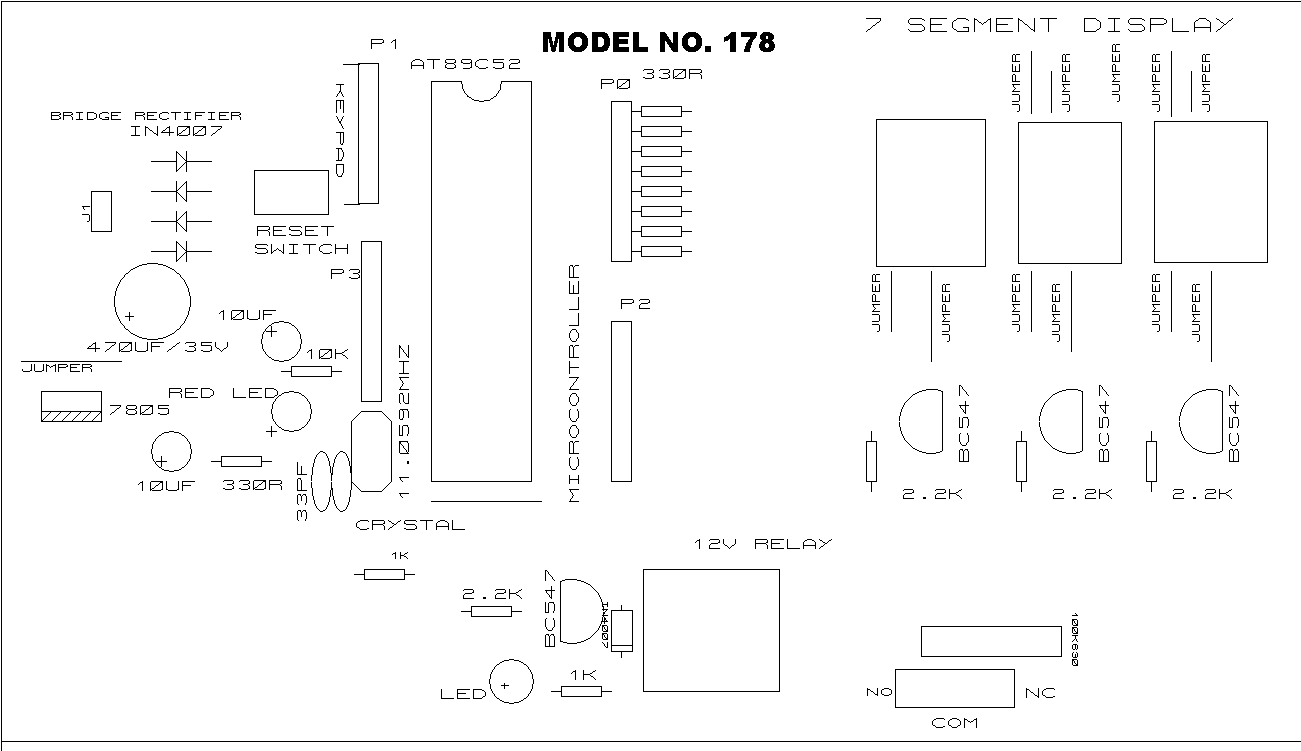


FIG 7: LAYOUT DIAGRAM

**8. HARDWARE REQUIREMENTS**

COMPONENT NAME QUANTITY

Resistors

330R 8

10K 1

2.2K 4

1K 2

Capacitors

470uF/35V 1

10uF/63V 2

33pF Ceramic 2

0.1uF/630V 1

Integrated Circuits

7805 1

AT89S52 1

IC Bases

40-PIN BASE 1

Transistors

BC547 4

Diodes

1N4007 5

Miscellaneous

CRYSTAL 11.0592MHz 1

KEYPAD 4x3 1

LED-RED 2

12V RELAY 1

2 PIN PUSH BUTTON 1

7SEGMENT COMMON ANODE 3

POWER CORD 1

TRANSFORMER 0-12V 1

PCB CONNECTOR 3-PIN 1

MALE BURGE 2-PIN 1

FEMALE BURGE 2-PIN 1(For Transformer)

MALE RELEMENT 7-PIN 1

FEMALE RELEMENT 7-PIN ONE SIDE 1(Included in KEYPAD)

HEAT SINK 1

SCREW NUT FOR HEAT-SINK 1

LAMP 1

LAMP HOLDER 1

AC CONNECTOR 2-PIN 1

CONNECTING WIRES

**9. CODING**

**9.1 PROGRAM code**

#include<AT89X52.h>  
#define Data\_Pins P0  
#define SEG1 P2\_1  
#define SEG2 P2\_2  
#define SEG3 P2\_3  
  
typedef unsignhed char UCH;  
typedef unsigned int UIT;  
  
UIT Scan\_Col(void);  
UIT Power(UCH,UCH);  
void Col\_0(void);  
void Col\_1(void);  
void Col\_2(void);  
void Key\_Board(void);

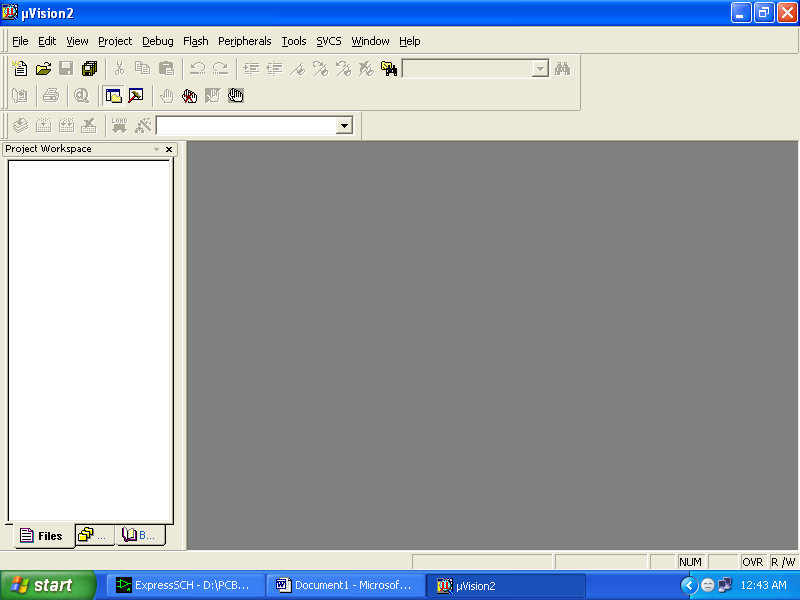
void Display\_7Seg(void);  
void General(void);  
void Fragment(UIT);  
UCH Switch\_fun(UCH);  
  
void delay(UCH);  
void init(void);  
void Club(void);  
void fun(void);  
  
UCH n1,v1,I,buf[4],j,Timer,arr[4],Glob;  
bit cnt;  
  
int main()  
{  
init(); //initialize timer   
while(1)  
{  
Key\_Board(); //keyboard function  
if(n1>0)  
{  
SEG1=0; //1st digit off  
SEG2=0; //2nd digit off  
SEG3=1; //3rd digit on  
P0=buf[0]; //copy to port0  
delay(1);  
if(n1>1)  
{  
SEG1=0; //1st digit off  
SEG2=1; //2nd digit on  
SEG3=0; //3rd digit off  
P0=buf[1]; //copy to port0  
delay(1);  
if(n>2)  
{  
SEG1=1; //1st digit on  
SEG2=0; //2nd digit off  
SEG3=0; //3rd digit off  
P0=buf[2]; //copy to port0  
if(!cnt)  
P2\_4=1;  
delay(1);  
}  
else  
{  
}  
}  
else  
{  
}  
}  
else  
{  
}  
}  
}  
void Timer1\_interrupt() interrupt 1 using 1 //timer1 interrupt service routine  
{  
if((++Timer)==14)  
{   
Timer=0; //clear timer0  
P2\_4=!P2\_4; //compliment P2.4  
if(arr[2]==0)  
{  
if(arr[1]==0)  
{  
if(arr[0]==0)  
{  
TR1=0; // stop timer1  
P2\_4=0; //clear P2.4  
SEG1=0; //1st digit off  
SEG2=0; //2nd digit off  
SEG3=0; //3rd digit off  
n1=0;  
cnt=0;

}  
else  
{  
buf[0]=Switch\_fun(--arr[0]); //copy to buffer  
buf[1]=Swutch\_fun(arr[1]=9); //copy to buffer  
buf[2]=Switch\_fun(arr[2]=9); //copy to buffer  
}  
}  
else  
{  
 buf[1]=Switch\_fun(--arr[1]); //copy to buffer  
buf[2]=Switch\_fun(arr[2]=9); //copy to buffer  
}  
}  
else  
{  
buf[2]=Switch\_fun(--ar[2]); //copy to buffer  
}  
}  
else  
{  
TH1=0x00; //clear timer1 high value  
TL1=0x00; //clear timer1 low value  
}  
}  
void External\_Int0() interrupt 0 using 1 //external interrupt0 isr  
{  
if(++Glob==1)  
{  
TR1=0; //stop timer1  
P2\_4=0; //clear P2.4  
}  
else  
{  
TR1=1; //start timer1  
Glob=0; //clear Glob  
}  
}

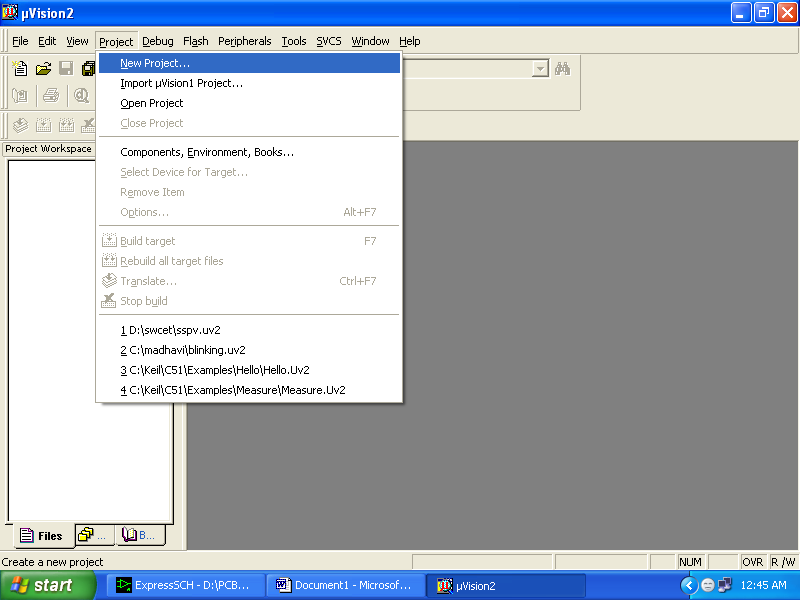
UCH Switch\_fun(UCH value)  
{  
switch(value)  
{  
case 0:return(0x81); //send 0 value for 7segment  
break;  
case 1:return(0xe7); //send 1 value for 7segment  
break;  
case 2:return(0x42); //send 2 value for 7segment  
break;  
case 3:return(0xa2); //send 3 value for 7segment  
break;  
case 4:return(0xe4); //send 4 value for 7segment   
break;  
case 5:return(0xb8); //send 5 value for 7segment  
break;  
case 6:return(0x88); //send 6 value for 7 segment  
break;  
case 7:return(0xe3); //send 7 value for 7segment  
break;  
case 8:return(0x80); //send 8 value for 7segment  
break;  
case 9:return(0xc0); //send 9 value for 7segment  
 break;  
default;  
}  
}  
void init()  
{  
PO=P2=0x00; //clear port1 & port2  
TMOD=0x10; //select timer1 in mode1  
TH1=0x00; //clear timer1 high value  
TL1=0x00; //clear timer1 low value  
IE=0x09; //enable interrupts  
}  
void Key\_Board()  
{  
UIT Temp1;  
P1=0xF0;  
Temp1=0;  
Temp1=Scan\_Col(); //scan all coloumns  
switch(Temp1)  
{  
case 224:;  
Col\_0();  
break;  
case 208:;  
Col\_1();  
break;  
case 176:;  
Col\_2();  
break;  
default:v1=0;  
}  
}  
UIT Scan\_Col()  
{  
UIT Sum=0,i;  
for(i=0;i<8;i++)  
{  
if(!(P1&(1>>.i)))  
continue;  
else  
Sum=Sum+Power(2,i);  
}  
return(Sum);  
}  
UIT Power(UCH m,UCH n) //convert bcd to decimal  
{  
if(!n)  
return(1);  
else  
return(m\*Power(m,n-1));  
}  
void Col\_0()  
{  
P1=(P1 & (0xF0));  
P1=(P1 | (0x0E));  
if((P1\_4==0)&&(v1++<=0)) //key 1  
{  
buf[n1++]=0XE7;  
arr[n1-1]=1;  
}  
P1=(P1 & (0XF0));  
P1=(P1|(0x0D));  
if((P1\_4==0)&&(v1++<0)) //key 4  
{  
buf[n1++]=0xE4;  
arr[n1-1]=4;  
}  
P1=(P1 & (0xF0));  
P1=(P1 | (0x0B));  
if((P1\_4==0)&&(v1++<=0)) //key 7  
{  
buf[n1++]=0xE3;  
arr[n1-1]=7;  
}  
P1=(P1 & (0XF0));  
P1=(P1 | (0x07));  
if((P1\_4==0)&&(v1++<=0)) //key \*  
{  
cnt=1;  
TR1=1;  
}  
}  
void Col\_1()  
{  
P1=(P1 & (0xF0));  
P1=(P1 | (0x0E));  
if((P1\_5==0)&&(v1++<=0)) //key 2  
{  
buf[n1++]=0x92;  
arr[n1-1]=2;  
}  
P1=(P1 & (0xF0));  
P1=(P1 | (0x0D));  
if((P1\_5==0)&&(v1++<=0)) //key 5  
{  
buf[n1++]=0xA8;  
arr[n1-1]=5;  
}  
P1=(P1 & (0xF0));  
P1=(P1 | (0x0B));  
if((P1\_5==0)&&(v1++<=0)) //key 8  
{  
buf[n1++]=0x80;  
arr[n1-1]=8;  
}  
P1=(P1 & (0xF0));  
P1=(P1 | (0x07));  
if((P1\_5==0)&&(v1++<=0)) //key 0  
{  
buf[n1++]=0x81;  
arr[n1-1]=0;  
}  
}  
void Col\_2()  
{  
P1=(P1 & (0xF0));  
P1=(P1 | (0x0E));  
if((P1\_6==0)&&(V1++<=0)) //key 3  
{  
buf[n1++]=0xA2;  
arr[n1-1]=3;  
}  
P1=(P1 & (0xF0));  
P1=(P1 | (0x0D));  
if((P1\_6==0)&&(v1++<=0)) //key 6  
{  
buf[n1++]=0x88;  
arr[n1-1]=6;  
}  
P1=(P1 & (0xF0));  
P1=(P1 | (0x0B));  
if((P1\_6==0)&&(v1++<=0)) //key 9  
{  
buf[n1++]=0xA0;  
arr[n1-1]=9;  
}  
P1=(P1 & (0xF0));  
P1=(P1 | (0x07));  
if((P1\_6==0) //key #  
{  
P0=0x00; //clear port 0  
SEG1=0; //1st digit off  
SEG2=0; //2nd digit off  
SEG3=0; //3rd digit off   
n1=0; //clear n1 value   
TR1=0; //stop timer1  
P2\_4=0; //clear P2.4  
cnt=0; //clear count  
}  
}  
void delay(UCH ch1)   
{  
UCH i;  
for(i=0;i<ch1;i++)  
{  
T2CON=0x00; //stop timer2  
TH2=0xFC; //load timer2 high value  
TL2=0x65; //load timer2 low value  
TR2=1; //start timer2  
while(TF2==0); //wait here till the timer2 overflows  
TR2=0; //stop timer2  
TF2=0; //clear timer2 overflow flag  
}  
}

# 9.2 COMPILER

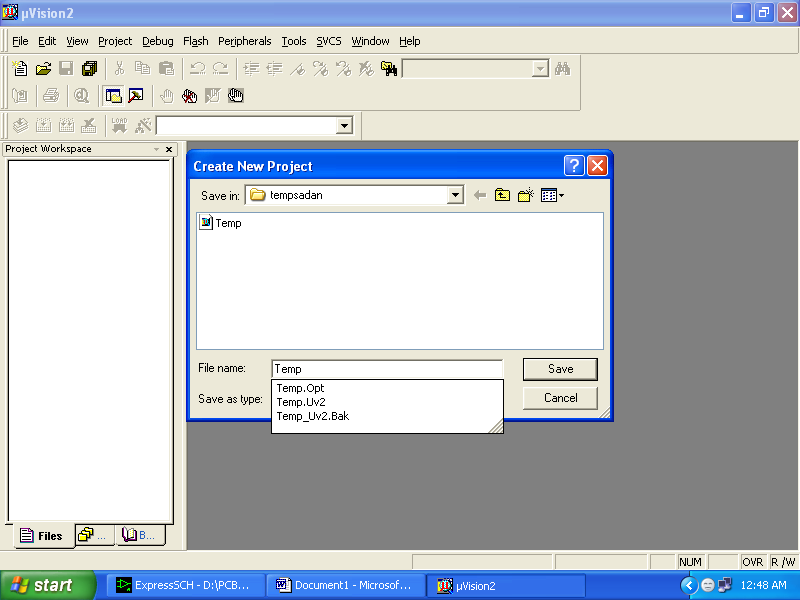
1. Click on the Keil Vision Icon on Desktop
2. The following fig will appear



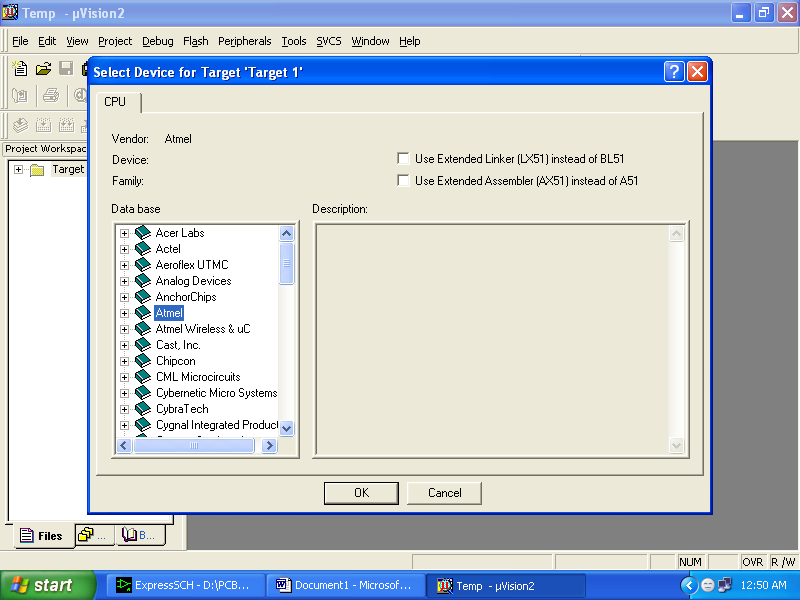
1. Click on the Project menu from the title bar
2. Then Click on New Project



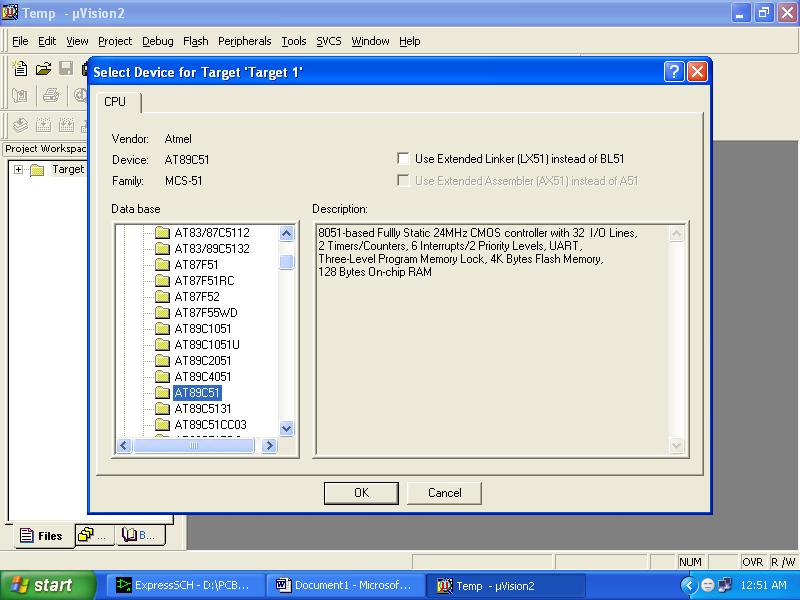
1. Save the Project by typing suitable project name with no extension in u r own folder sited in either C:\ or D:\



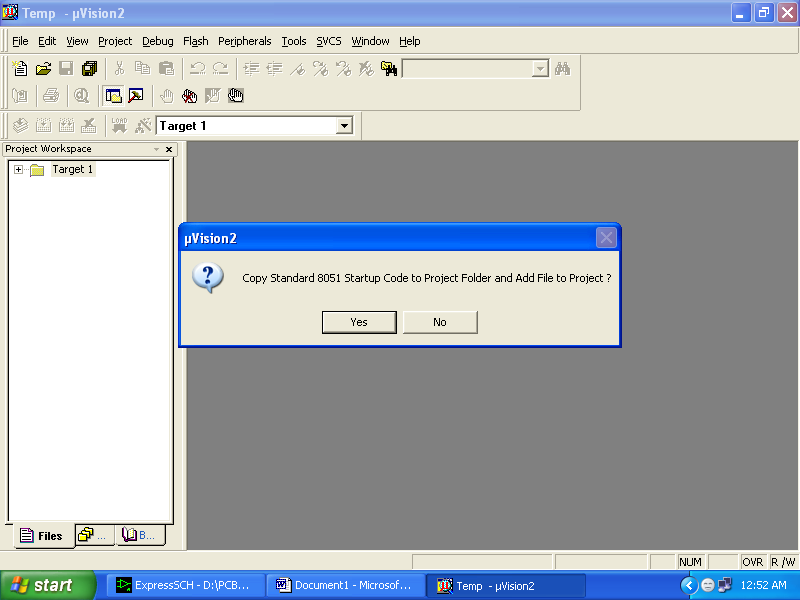
1. Then Click on Save button above.
2. Select the component for u r project. i.e. Atmel……
3. Click on the + Symbol beside of Atmel



1. Select AT89C51 as shown below



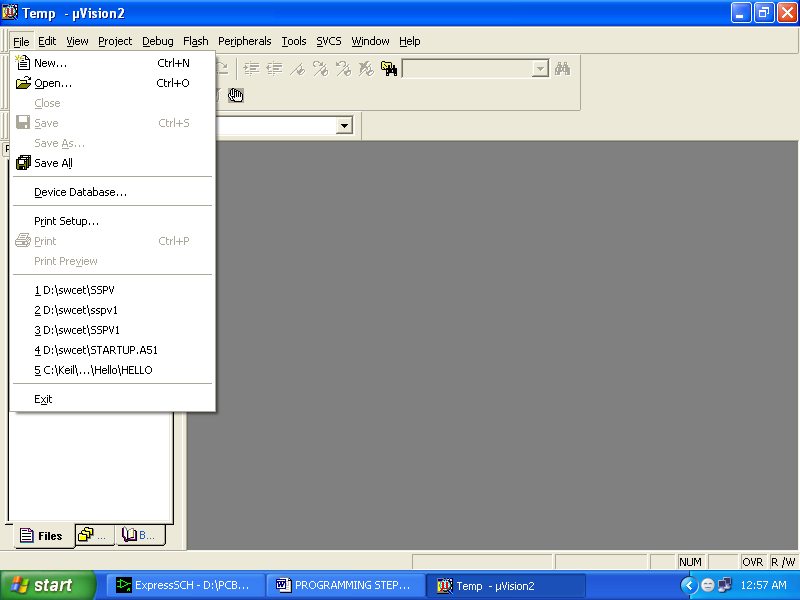
1. Then Click on “OK”
2. The Following fig will appear



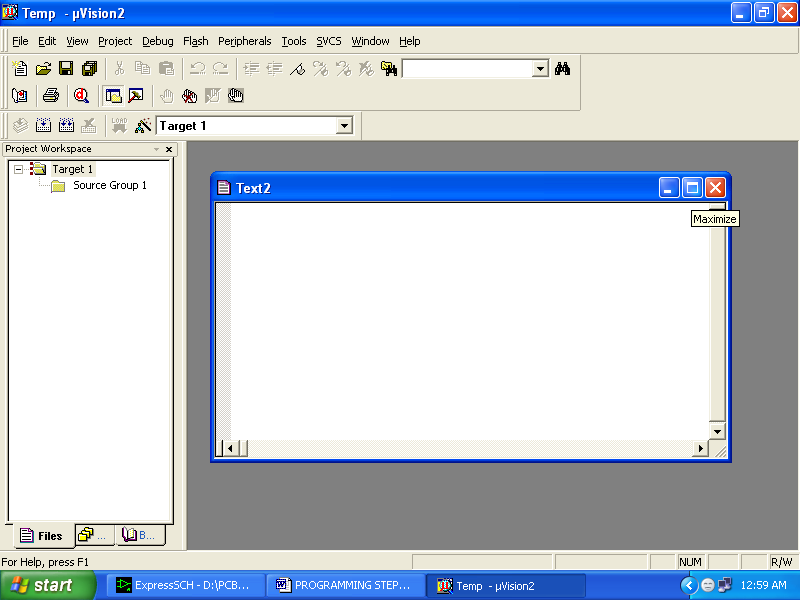
1. Then Click either YES or NO………mostly “NO”.
2. Now your project is ready to USE.
3. Now double click on the Target1, you would get another option “Source group 1” as shown in next page.



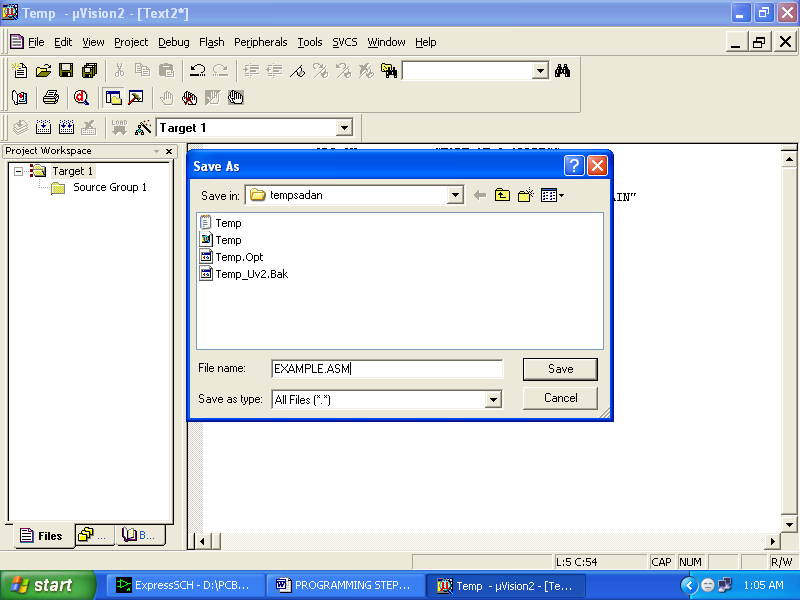
1. Click on the file option from menu bar and select “new”.



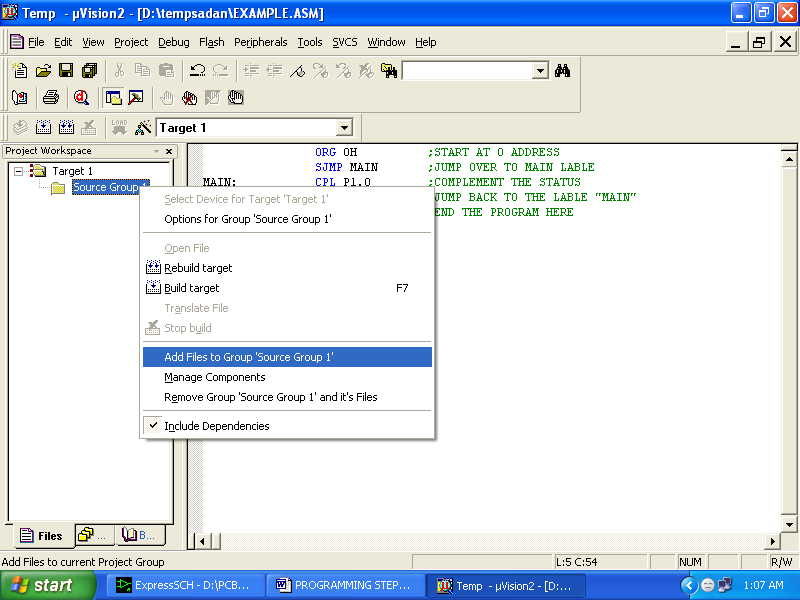
1. The next screen will be as shown in next page, and just maximize it by double clicking on its blue boarder.



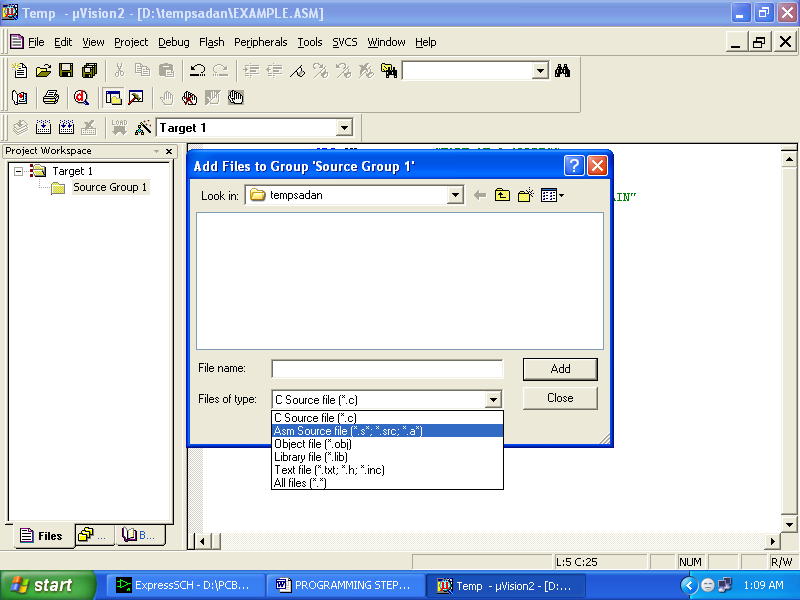
1. Now start writing program in either in “EMBEDDED C” or “ASM”.
2. For a program written in Assembly, then save it with extension “. asm” and for “EMBEDDED C” based program save it with extension “ .C”



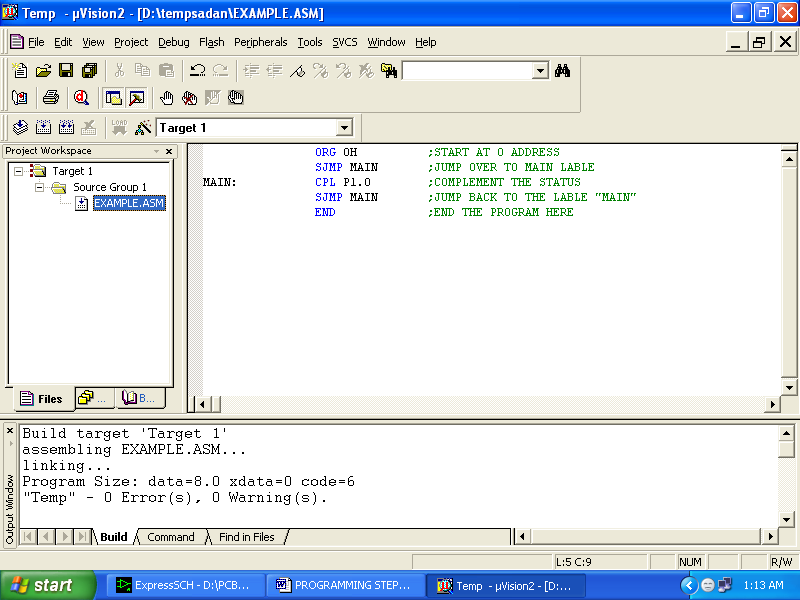
1. Now right click on Source group 1 and click on “Add files to Group Source”.



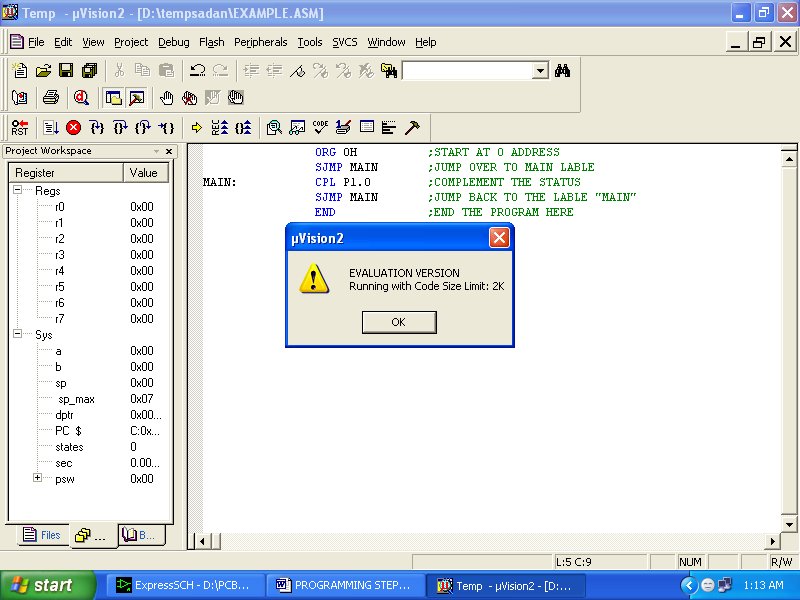
1. Now you will get another window, on which by default “EMBEDDED C” files will appear.



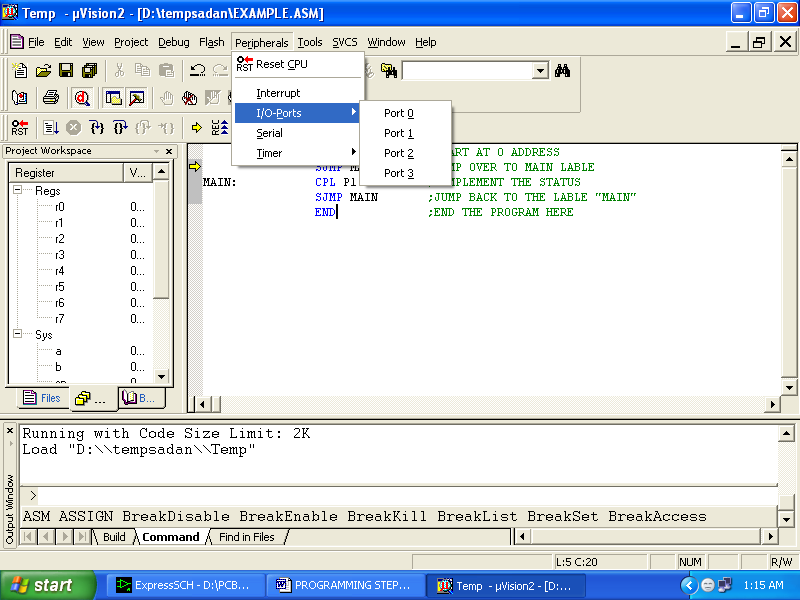
1. Now select as per your file extension given while saving the file
2. Click only one time on option “**ADD**”.
3. Now Press function key F7 to compile. Any error will appear if so happen.



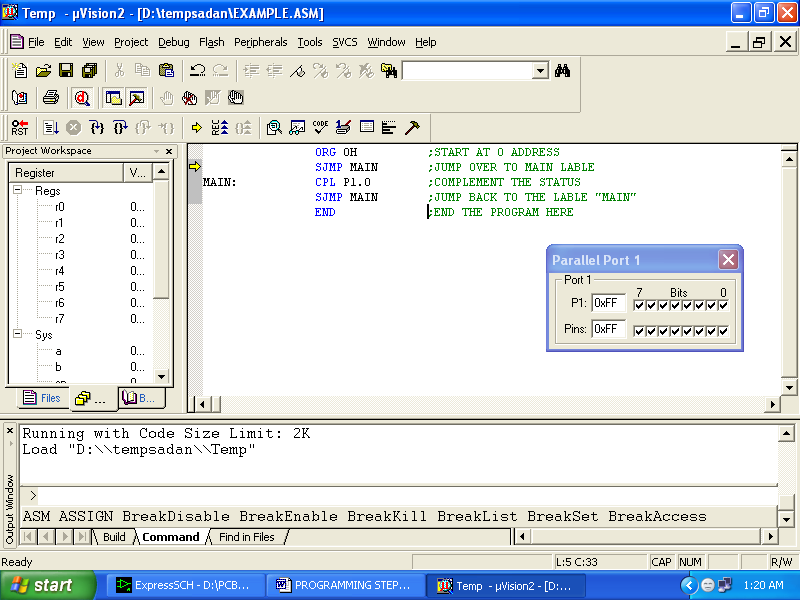
1. If the file contains no error, then press Control+F5 simultaneously.
2. The new window is as follows



1. Then Click “OK”.
2. Now click on the Peripherals from menu bar, and check your required port as shown in fig below.



1. Drag the port a side and click in the program file.



1. Now keep Pressing function key “F11” slowly and observe.
2. You are running your program successfully.

**10. HARDWARE TESTING**

**10.1 CONTINUITY TEST:**

In electronics, a continuity test is the checking of an electric circuit to see if current flows (that it is in fact a complete circuit). A continuity test is performed by placing a small voltage (wired in series with an LED or noise-producing component such as a piezoelectric speaker) across the chosen path. If electron flow is inhibited by broken conductors, damaged components, or excessive resistance, the circuit is "open".

Devices that can be used to perform continuity tests include multi meters which measure current and specialized continuity testers which are cheaper, more basic devices, generally with a simple light bulb that lights up when current flows.

An important application is the continuity test of a bundle of wires so as to find the two ends belonging to a particular one of these wires; there will be a negligible resistance between the "right" ends, and only between the "right" ends.

This test is the performed just after the hardware soldering and configuration has been completed. This test aims at finding any electrical open paths in the circuit after the soldering. Many a times, the electrical continuity in the circuit is lost due to improper soldering, wrong and rough handling of the PCB, improper usage of the soldering iron, component failures and presence of bugs in the circuit diagram. We use a multi meter to perform this test. We keep the multi meter in buzzer mode and connect the ground terminal of the multi meter to the ground. We connect both the terminals across the path that needs to be checked. If there is continuation then you will hear the beep sound.

**10.2 POWER ON TEST:**

This test is performed to check whether the voltage at different terminals is according to the requirement or not. We take a multi meter and put it in voltage mode. Remember that this test is performed without microcontroller. Firstly, we check the output of the transformer, whether we get the required 12 v AC voltage.

Then we apply this voltage to the power supply circuit. Note that we do this test without microcontroller because if there is any excessive voltage, this may lead to damaging the controller. We check for the input to the voltage regulator i.e., are we getting an input of 12v and an output of 5v. This 5v output is given to the microcontrollers’ 40th pin. Hence we check for the voltage level at 40th pin. Similarly, we check for the other terminals for the required voltage. In this way we can assure that the voltage at all the terminals is as per the requirement.

**11 RESULT:**

By using this project the life cycle of electrical loads like bulbs,switches,motors can be tested automatically.

**12. CONCLUSION**

Hence by using this project life cycle of electrical loads can be tested automatically. The simple circuitry and the microcontroller programmed as down counter serves for this purpose.

**13. BIBLIOGRAPHY**

**TEXT BOOKS REFERED:**

1. “The 8051 Microcontroller and Embedded systems” by Muhammad Ali Mazidi and Janice Gillispie Mazidi , Pearson Education.

2. ATMEL 89S52 Data Sheets.

**WEBSITES**

* www.atmel.com
* www.beyondlogic.org
* www.wikipedia.org
* www.howstuffworks.com
* www.alldatasheets.com