



Jaypee University of Information Technology
Solan (H.P.)
LEARNING RESOURCE CENTER

Acc. Num. *SP03075* Call Num:

General Guidelines:

- ◆ Library books should be used with great care.
- ◆ Tearing, folding, cutting of library books or making any marks on them is not permitted and shall lead to disciplinary action.
- ◆ Any defect noticed at the time of borrowing books must be brought to the library staff immediately. Otherwise the borrower may be required to replace the book by a new copy.
- ◆ The loss of LRC book(s) must be immediately brought to the notice of the Librarian in writing.

Learning Resource Centre-JUIT



SP03075

SP3075

DESIGN OF AUTOMATED CAR PARKING SYSTEM

CERTIFICATE

BY

SOUVIK GHOSH – 031076
VIJAY KUMAR – 031064
KARAN BHIRANI--031051



Dr. Sahil Bhushan
Dept. of Electronics & Communication
(H.O.D)

Mr. Vivek Seghal
(Project Supervisor)

MAY-2007

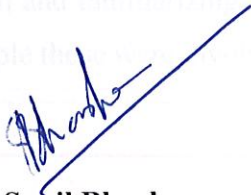
Submitted in partial fulfillment
of
the Degree of Bachelor of Technology

to

DEPARTMENT OF ELECTRONICS AND
COMMUNICATION ENGINEERING
JAYPEE UNIVERSITY OF INFORMATION
TECHNOLOGY
WAKNAGHAT

CERTIFICATE

This is to certify that the work entitled, "**DESIGN OF AUTOMATED CAR PARKING SYSTEM**" submitted by **SOUVIK GHOSH, VIJAY KUMAR** and **KARAN BHIRANI** in partial fulfillment for the award of degree of Bachelor of Technology in Electronics and communication Engineering of Jaypee University of Information Technology has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.



Dr. Sunil Bhushan
Dept. of Electronics & Communication
(H.O.D)



Mr. Vivek Seghal
(Project Supervisor)

1. Souvik Ghosh - 031076
2. Vijay Kumar - 031064
3. Karan Bhirani - 031051

ACKNOWLEDGEMENT

No task is a single man's effort .Various factors, situations, and people come together and sketch the background for the accomplishment of a task. We are grateful to **Mr. Vivek Seghal**, our project co-coordinator for his sincere guidance and are indebted to him for giving us some of his valuable time, for his initiative, encouragement, constructive criticism and familiarizing us with all technical aspects of the project .We also thank all the people those were involved directly or indirectly with our project.

- 1. Souvik Ghosh - 031076**
- 2. Vijay Kumar - 031064**
- 3. Karan Bhirani - 031051**

ABSTRACT

Automation is the use of scientific and technological principles in making machines that would take over the work normally done by humans. This definition has been disputed by professional scientists and engineers, but in any case, the term is derived from the longer term "Automatization" or from the phrase automatic operation. Delmar S. Harder, a plant manager for General Motors, is credited with first having used the term in 1935. This project deals with the Automation of a Car Parking Space. Normally in a garage, the guard is the person who tells a person about the availability of space on the floor, so we designed a system that would take over his job. Our parking space model tells any person, the number of cars parked on the floor. We have achieved this by the use of sensors, microcontroller, LCD display and LEDs. This was is just a rudimentary step towards the automation of a garage. The effort can be carried forward by incorporating lifts, doors (which are supposed to close when no more space is available), password protected VIP floors etc.

CONTENTS

CHAPTER 1. INTRODUCTION	1-5
1.1 Objective	1
1.2 Definition of Problem	1
1.3 Methodology	1
1.4 History of Automation	
1.4.1 Social Issues of Automation	2
1.4.2 Current Emphases in Automation	4
1.4.3 Safety issues of Automation	4
	1
CHAPTER 2. THE 8051 MICROCONTROLLER	6 -33
2.1 Introduction	6
2.1.1 A Note on ROM	6
2.2 Overview of the 8051 Family	9
2.3 Architecture of 8051	10
2.3.1 Features of 8051	12
2.4 Pin Functions of 8051	16
2.5 Special Function Registers	18
2.6 Instruction Set	19
CHAPTER 3. WORKING OF THE MODEL	34 - 49
3.1 Introduction	34
3.2 Infra Red Sensor	34
3.3 Infra Red Sensor Logic	40
3.4 Microcontroller Interfacing	43

CHAPTER 4. CONCLUSIONS AND FUTURE WORK	50
APPENDIX -1 Program of the Microcontroller	52
APPENDIX -2 8051 Instruction Set	56
APPENDIX -3 Data Sheet of IC 7805	65
APPENDIX -4 Data Sheet of BC547	67
BIBLIOGRAPHY	69

LIST OF FIGURES

- 2.1 Architecture of 8051
- 2.2 Internal RAM structure
- 3.1 Photo-Transistors
- 3.2 LEDs
- 3.3 IC of a 555 Timer
- 3.4 Structure of a 555 Timer
- 3.5 Circuit Diagram
- 3.6 Conversion of 200V AC to 9-0-9 AC

CHAPTER 1

INTRODUCTION

1.1 Objective

The objective of this project is to design and build an automated car parking system by using a microcontroller. All the functionalities of the model are to be governed by the microcontroller.

1.2 Definition of Problem

The design of the model can utilize any electronic component starting from resistors to ICs. The demo model should support a parking space of atleast five cars and it should be completely automated i.e. none of its featured functions should require any human assistance.

1.3 Methodology

The complete design process of the circuit was done on paper. The process required consultation of a numerous books and the internet. The program for controlling the operations was written on the “AceBus 8051 Integrated Development Environment”.

1.4 History of Automation

Automation (ancient Greek: = *self dictated*), **roboticization** or **industrial automation** or numerical control is the use of control systems such as computers to control industrial machinery and processes, replacing human operators. In the scope of industrialization, it is a step beyond mechanization. Whereas *mechanization* provided human operators with machinery to assist them with the *physical* requirements of work, *automation* greatly reduces the need for human *sensory* and *mental* requirements as well.

Automation plays an increasingly important role in the global economy and in daily experience. Engineers strive to combine automated devices with mathematical and organizational tools to create complex systems for a rapidly expanding range of applications and human activities.

There are still many jobs which are in no immediate danger of automation. No device has been invented which can match the human eye for accuracy and precision in many tasks; nor the human ear. Even the admittedly handicapped human is able to identify and distinguish among far more scents than any automated device. Human pattern recognition, language recognition, and language production ability is well beyond anything currently envisioned by automation engineers.

Automation is a field that is relatively young, yet it has met with more resistance than almost any other. It deals with the science and engineering of systems to automate tasks of all kinds, from the piloting of a starship through jumpspace, to the repetitive welding of particular parts on an assembly line. Throughout its short history, automation has been alternately hailed and despised, promoted and banned.

1.4.1 Social Issues of Automation

Automation raises several important social issues. Among them is automation's impact on employment. Indeed, the Luddites were a social movement of English textile workers in the early 1800s who protested against Jacquard's automated weaving looms— often by destroying such textile machines— that they felt threatened their jobs. Since then, the term *luddite* has come to be applied freely to anyone who is against any advance of technology. Some argue automation leads to *higher* employment. One author made the following case. When automation was first introduced, it caused widespread fear. It was thought that the displacement of human workers by computerized systems would lead to severe unemployment. In fact, the opposite has often been true, e.g., the freeing up of the labor force allowed more people to enter higher skilled jobs, which are typically higher paying. One odd

side effect of this shift is that "unskilled labor" now benefits in many "first-world" nations, because fewer people are available to fill such jobs.

Some argue the reverse, at least in the long term. They argue that automation has only just begun and short-term conditions might partially obscure its long-term impact. Many manufacturing jobs left the United States during the early 1990s, but a one-time massive increase in IT jobs (which are only now being outsourced), at the same time, offset this. It appears that automation does devalue labor through its replacement with less-expensive machines; however, the overall effect of this on the workforce as a whole remains unclear. Today automation of the workforce is quite advanced, and continues to advance increasingly more rapidly throughout the world and is encroaching on ever more skilled jobs, yet during the same period the general well-being of most people in the world (where political factors have not muddied the picture) has increased dramatically. What role automation has played in these changes has not been well studied.

One irony is that in recent years, outsourcing has been blamed for the loss of jobs in which automation is the more likely culprit. This argument is supported by the fact that in the U.S., the number of insourced jobs is increasing at a greater rate than those outsourced. Further, the rate of decline in U.S. manufacturing employment is no greater than the worldwide average: 11 percent between 1995 and 2002. In the same period, China, which has been frequently criticized for "stealing" American manufacturing jobs, lost 15 million manufacturing jobs of its own (about 15% of its total), compared with 2 million lost in the U.S..

Millions of human telephone operators and answerers, throughout the world, have been replaced wholly (or almost wholly) by automated telephone switchboards and answering machines (*not* by Indian or Chinese workers). Thousands of medical researchers have been replaced in many medical tasks from 'primary' screeners in electrocardiography or radiography, to laboratory analyses of human genes, sera, cells, and tissues by *automated systems*. Even physicians have been partly replaced by remote, automated robots and by highly sophisticated surgical robots that allow them to perform remotely and at levels of accuracy and precision otherwise not normally possible for the average physician.

1.4.2 Current Emphases in Automation

Currently, for manufacturing companies, the purpose of automation has shifted from increasing productivity and reducing costs, to broader issues, such as increasing quality and flexibility in the manufacturing process.

The old focus on using automation simply to increase productivity and reduce costs was seen to be short-sighted, because it is also necessary to provide a skilled workforce who can make repairs and manage the machinery. Moreover, the initial costs of automation were high and often could not be recovered by the time entirely new manufacturing processes replaced the old. (Japan's "robot junkyards" were once world famous in the manufacturing industry.)

Automation is now often applied primarily to increase quality in the manufacturing process, where automation can increase quality substantially. For example, automobile and truck pistons used to be installed into engines manually. This is rapidly being transitioned to automated machine installation, because the error rate for manual installment was around 1-1.5%, but has been reduced to 0.00001% with automation. Hazardous operations, such as oil refining, the manufacturing of industrial chemicals, and all forms of metal working, were always early contenders for automation.

Another major shift in automation is the increased emphasis on flexibility and convertibility in the manufacturing process. Manufacturers are increasingly demanding the ability to easily switch from manufacturing Product A to manufacturing Product B without having to completely rebuild the production lines.

1.4.3 Safety Issues of Automation

One safety issue with automation is that while it is often viewed as a way to *minimize* human error in a system, *increasing the degree and levels of automation also increases the consequences of error*. For example, The Three Mile Island nuclear event was largely due to over-reliance on "automated safety" systems. Unfortunately, in the event, the designers had never anticipated the actual failure mode which occurred, so both the "automated safety"

systems and their human overseers were inundated with vast amounts of largely irrelevant information. With automation we have machines designed by (fallible) people with high levels of expertise, which operate at speeds well beyond human ability to react, being operated by people with relatively more limited education (or other failings, as in the Bhopal disaster or Chernobyl disaster). Ultimately, with increasing levels of automation over ever larger domains of activities, when something goes wrong the consequences rapidly approach the catastrophic. This is true for all complex systems however, and one of the major goals of safety engineering for nuclear reactors, for example, is to make safety mechanisms as simple and as foolproof as possible.

CHAPTER 2

THE 8051 MICROCONTROLLER

2.1 The 8051

The 8051 developed and launched in the early 80's, is one of the most popular micro controller in use today. It has a reasonably large amount of built in ROM and RAM. In addition it has the ability to access external memory.

The generic term '8x51' is used to define the device. The value of x defining the kind of ROM, i.e. x=0, indicates none, x=3, indicates mask ROM, x=7, indicates EPROM and x=9 indicates EEPROM or Flash.

2.1.1 A Note on ROM

The early 8051, namely the 8031 was designed without any ROM. This device could run only with external memory connected to it. Subsequent developments lead to the development of the PROM or the programmable ROM. This type had the disadvantage of being highly unreliable.

The next in line, was the EPROM or Erasable Programmable ROM. These devices used ultraviolet light erasable memory cells. Thus a program could be loaded, tested and erased using ultra violet rays. A new program could then be loaded again.

An improved EPROM was the EEPROM or the electrically erasable PROM. This does not require ultra violet rays, and memory can be cleared using circuits within the chip

itself.

Finally there is the FLASH, which is an improvement over the EEPROM. While the terms EEPROM and flash are sometimes used interchangeably, the difference lies in the fact that flash erases the complete memory at one stroke, and not act on the individual cells. This results in reducing the time for erasure.

Different microcontrollers in the market:

- PIC: One of the famous microcontrollers used in the industries. It is based on RISC Architecture which makes the microcontroller process faster than other microcontroller.
- INTEL: These are the first to manufacture microcontrollers. These are not as sophisticated other microcontrollers but still the easiest one to learn.
- ATMEL : Atmel's AVR microcontrollers are one of the most powerful in the embedded industry. This is the only microcontroller having 1kb of ram even the entry stage. But it is unfortunate that in India we are unable to find this kind of microcontroller.

Intel 8051

Intel 8051 is CISC architecture which is easy to program in assembly language and also has a good support for High level languages.

The memory of the microcontroller can be extended up to 64k.

This microcontroller is one of the easiest microcontrollers to learn.

The 8051 microcontroller is in the field for more than 20 years. There are lots of books and study materials are readily available for 8051.

Derivatives

The best thing done by Intel is to give the designs of the 8051 microcontroller to everyone. So it is not the fact that Intel is the only manufacture for the 8051 there more than 20 manufactures, with each of minimum 20 models. Literally there are hundreds of models of 8051 microcontroller available in market to choose. Some of the major manufactures of 8051 are

- Atmel
- Philips

Philips

The Philips's 8051 derivatives has more number of features than in any microcontroller. The costs of the Philips microcontrollers are higher than the Atmel's which makes us to choose Atmel more often than Philips

Dallas

Dallas has made many revolutions in the semiconductor market. Dallas's 8051 derivative is the fastest one in the market. It works 3 times as fast as a 8051 can process. But we are unable to get more in India.

Atmel

These people were the one to master the flash devices. They are the cheapest microcontroller available in the market. Atmel's even introduced a 20pin variant of 8051 named 2051. The Atmel's 8051 derivatives can be got in India less than 70 rupees. There are lots of cheap programmers available in India for Atmel. So it is always good for students to stick with 8051 when you learn a new microcontroller.

2.2 OVERVIEW OF THE 8051 FAMILY

8051 MICROCONTROLLER:

The 8051 is the original member of the 8051 family. Intel refers to it as MCS-51.

Its features are mentioned

- ROM 4K bytes
- RAM 128 bytes
- Timer 2
- I/O pins 32
- Serial port 1
- Interrupt sources 6

8052 MICROCONTROLLER:

8052 is another member of 8051 family. It has all the standard features of the 8051 as well as an extra 128 bytes of RAM and an extra timer.

Features of 8052 are mentioned below

- ROM 8K bytes
- RAM 256 bytes
- Timer 3
- I/O pins 32
- Serial port 1
- Interrupt sources 8

8031 MICROCONTROLLER:

This chip is also referred to as a ROM-less 8051 since it has 0K of on-chip ROM. To use this chip you must add external ROM to it. This external ROM must contain the program that the 8031 will fetch and execute. The ROM containing the program attached to 8031 can be as large as 64K bytes. In the process of adding external ROM to the 8031, you lose two ports. That leaves only 2 ports (of the 4 ports) for I/O operations. To solve this problem you can add external I/O to the 8031.

Features of 8031 microcontroller are mentioned below

- ROM 0K bytes
- RAM 128 bytes
- Timer 2
- I/O pins 32
- Serial port 1
- Interrupt sources 6

8751 MICROCONTROLLER:

The 8751 chip has only 4K bytes of on-chip UV-EPROM.

2.3 ARCHITECTURE OF 8051

The 8051 doesn't have any special feature than other microcontroller. The only feature is that it is easy to learn. Architecture makes us to know about the hardware features of the microcontroller.

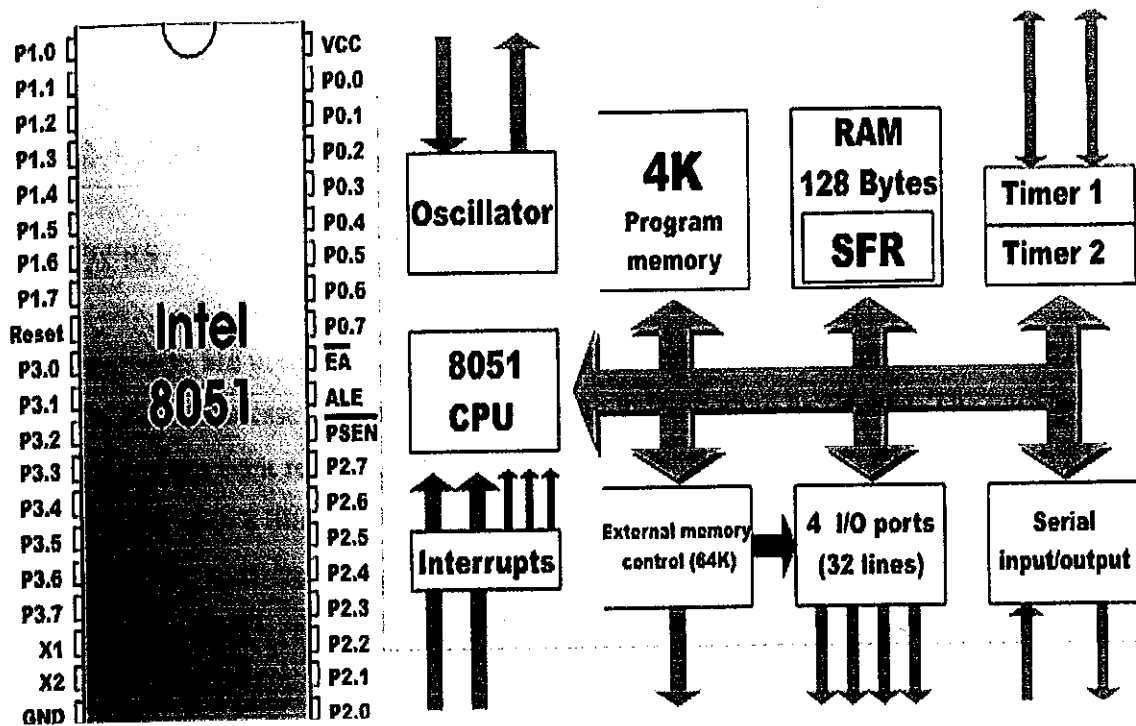


Fig.2.1: Internal Architecture of 8051.

The features of the 8051 are

- 4K Bytes of Flash Memory
- 128 x 8-Bit Internal RAM
- Fully Static Operation: 1 MHz to 24 MHz
- 32 Programmable I/O Lines
- Two 16-Bit Timer/Counters
- Six Interrupt Sources (5 Vectored)
- Programmable Serial Channel
- Low Power Idle and Power Down Modes

The 8051 has a 8-Bit CPU that means it is able to process 8 bit of data at a time. 8051 has 235 instructions in total.

2.3.1 FEATURES OF 8051 CORE IN DETAIL

REGISTERS:

In CPU registers are used to store information temporarily. That information could be a byte of data to be a processed or an address pointing to the data to be fetched. The vast majority of 8051 registers are 8-bit registers. IN 8051 there is only one data type : 8 bits. With an 8 bit data, any data larger than 8 bits must be broken into 8-bit chunks before it is processed. The most widely used registers of 8051 are A(accumulator),B,R0,R1,R2,R3,R4,R5,R6,R7,DPTR(data pointer), and PC(program counter). All above mentioned registers are 8 bit except except DPTR and PC which are 16 bit registers.

4K bytes of on-chip ROM

Once you have written out the instructions for the microcontroller, where do you put these instructions? Obviously you would like these instructions to be safe, and not get deleted or changed during execution. Hence you would load it into the 'ROM'

The size of the program you write is bound to vary depending on the application, and the number of lines. The 8051 microcontroller gives you space to load up to 4K of program size into the internal ROM. You could always extend the space by connecting to 64K of external ROM if required.

128 bytes on-chip RAM

This is the space provided for executing the program in terms of moving data, storing data etc.

RAM memory space allocation in 8051: The 128 bytes of RAM inside the 8051 are assigned addresses 00 to 7FH. These 128 bytes are divided into three different groups as follows

1. Total of 32 bytes from locations 00 to 1F hex are set aside for register banks and the stack. These 32 bytes are divided into 4 banks of registers in which each bank has 8 registers, R0 to R7
2. A total of 16 bytes from locations 20H to 2FH are set aside for bit addressable read/write memory.
3. A total of 80 bytes from locations 30H to 7FH are used for read and write storage, or what is normally called a scratch pad.

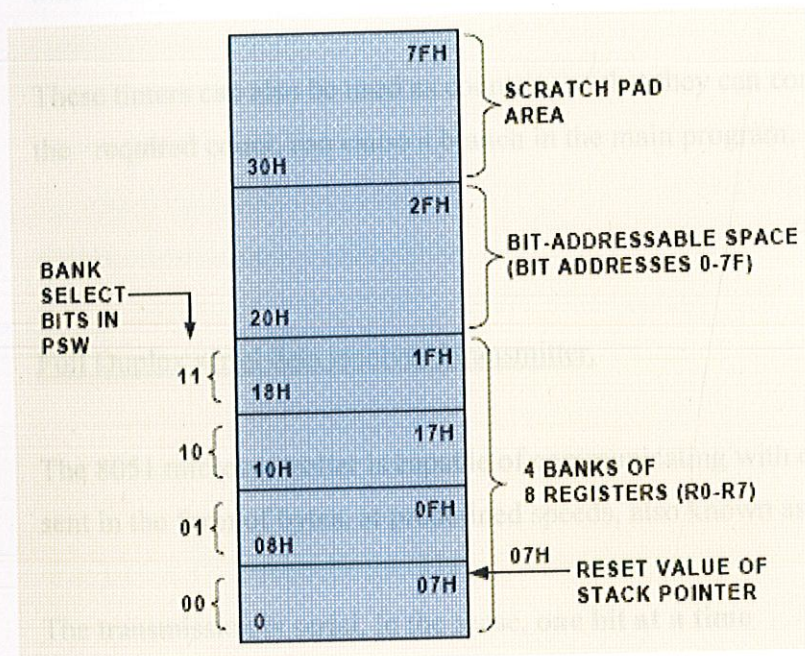


Fig.2.2: RAM structure of 8051.

32 I/O lines. (Four- 8 bit ports, labeled P0, P1, P2, P3)

in command line notation p1.0 means bit zero of port one. One bit controls one bulb. Thus port one would have 8 bits. There are a total of four ports named p0, p1, p2, p3, giving a total of 32 lines. These lines can be used both as input or output.

Two 16 bit timers / counters.

A microcontroller normally executes one instruction at a time. However certain applications would require that some event has to be tracked independent of the main program.

The manufacturers have provided a solution, by providing two timers. These timers execute in the background independent of the main program. Once the required time has been reached, (remember the time calculations described above?), they can trigger a branch in the main program.

These timers can also be used as counters, so that they can count the number of events, and on reaching the required count, can cause a branch in the main program.

Full Duplex serial data receiver / transmitter.

The 8051 microcontroller is capable of communicating with external devices like the PC etc. Here data is sent in the form of bytes, at predefined speeds, also known as baud rates.

The transmission is serial, in the sense, **one bit at a time**

5- interrupt sources with two priority levels (Two external and three internal)

During the discussion on the timers, we had indicated that the timers can trigger a branch in the main program. However, what would we do in case we would like the microcontroller to take the branch, and then return back to the main program, without having to constantly check whether the required time / count has been reached?

This is where the interrupts come into play. These can be set to either the timers, or to some external

events. Whenever the background program has reached the required criteria in terms of time or count of an external event, the branch is taken, and on completion of the branch, the control returns to the main program.

Priority levels indicate which interrupt is more important, and needs to be executed first in case two interrupts occur at the same time.

On-chip clock oscillator.

This represents the oscillator circuits within the microcontroller. Thus the hardware is reduced to just simply connecting an external crystal, to achieve the required pulsing rate. The speed with which a microcontroller executes instructions is determined by what is known as the crystal speed. A crystal is a component connected externally to the microcontroller. The crystal has different values, and some of the used values are 6MHZ, 10MHZ, and 11.059 MHz etc.

The time is calculated using the formula

No of cycles per second = Crystal frequency in HZ / 12.

For a 10MHZ crystal the number of cycles would be,

$10,000,000/12=833333.33333$ cycles.

This means that in one second, the microcontroller would execute 833333.33333 cycles.

2.4 PIN FUNCTION OF IC 89C51

1. **Supply** pin of this ic is pin no 40. Normally we apply a 5 volt regulated dc power supply to this pin. For this purpose either we use step down transformer power supply or we use 9 volt battery with 7805 regulator.
2. **Ground** pin of this ic is pin no 20. Pin no 20 is normally connected to the ground pin (normally negative point of the power supply).
3. **XTAL** is connected to the pin no 18 and pin no 19 of this ic. The quartz crystal oscillator connected to XTAL1 and XTAL2 PIN. These pins also needs two capacitors of 30 pf value. One side of each capacitor is connected to crystal and other pin is connected to the ground point. Normally we connect a 12 MHz or 11.0592 MHz crystal with this ic.. But we use crystal upto 20 MHz to this pins
4. **RESET PIN..** Pin no 9 is the reset pin of this ic.. It is an active high pin. On applying a high pulse to this pin, the micro controller will reset and terminate all activities. This is often referred to as a power on reset. The high pulse must be high for a minimum of 2 machine cycles before it is allowed to go low.
5. **PORT0** Port 0 occupies a total of 8 pins. Pin no 32 to pin no 39. It can be used for input or output. We connect all the pins of the port 0 with the pullup resistor (10 k ohm) externally. This is due to fact that port 0 is an open drain mode. It is just like a open collector transistor.
6. **PORT1.** ALL the ports in microcontroller is 8 bit wide pin no 1 to pin no 8 because it is a 8 bit controller. All the main register and sfr all is mainly 8 bit wide. Port 1 is also occupies a 8 pins. But there is no need of pull up resistor in this port. Upon reset port 1 act as a input port. Upon reset all the ports act as a input port
7. **PORT2.** port 2 also have a 8 pins. It can be used as a input or output. There is no need of any pull up resistor to this pin.

8. **PORT 3.** Port3 occupies a total 8 pins from pin no 10 to pin no 17. It can be used as input or output. Port 3 does not require any pull up resistor. The same as port 1 and port2. Port 3 is configured as an output port on reset. Port 3 has the additional function of providing some important signals such as interrupts. Port 3 also use for serial communication.

9. **ALE** ALE is an output pin and is active high. When connecting an 8031 to external memory, port 0 provides both address and data. In other words, the 8031 multiplexes address and data through port 0 to save pins. The ALE pin is used for demultiplexing the address and data by connecting to the ic 74ls373 chip.

10. **PSEN.** PSEN stands for program store enable. In an 8031 based system in which an external rom holds the program code, this pin is connected to the OE pin of the rom.

11. **EA.** EA. In 89c51 8751 or any other family member of the ateml 89c51 series all come with on-chip rom to store programs, in such cases the EA pin is connected to the Vcc. For family member 8031 and 8032 is which there is no on chip rom, code is stored in external memory and this is fetched by 8031. In that case EA pin must be connected to GND pin to indicate that the code is stored externally.

2.5 SPECIAL FUNCTION REGISTER (SFR) ADDRESSES.

ACC	ACCUMULATOR	0E0H
B	B REGISTER	0F0H
PSW	PROGRAM STATUS WORD	0D0H
SP	STACK POINTER	81H
DPTR	DATA POINTER 2 BYTES	
DPL	LOW BYTE OF DPTR	82H
DPH	HIGH BYTE OF DPTR	83H
P0	PORT0	80H
P1	PORT1	90H
P2	PORT2	0A0H
P3	PORT3	0B0H
TMOD	TIMER/COUNTER MODE CONTROL	89H
TCON	TIMER COUNTER CONTROL	88H
TH0	TIMER 0 HIGH BYTE	8CH

TLO	TIMER 0 LOW BYTE	8AH
TH1	TIMER 1 HIGH BYTE	8DH
TL1	TIMER 1 LOW BYTE	8BH
SCON	SERIAL CONTROL	98H
SBUF	SERIAL DATA BUFFER	99H
PCON	POWER CONTROL	87H

These registers are utilized by the programmer to store data. They are also used to check branching control conditions. Some of these registers are bit-addressable while some are non-bit addressable.

2.6 INSTRUCTIONS

SINGLE BIT INSTRUCTIONS::

SETB BIT	SET THE BIT =1
CLR BIT	CLEAR THE BIT =0
CPL BIT	COMPLIMENT THE BIT 0=1, 1=0
JB BIT, TARGET	JUMP TO TARGET IF BIT =1
JNB BIT, TARGET	JUMP TO TARGET IF BIT =0

JBC BIT,TARGET
BIT

JUMP TO TARGET IF BIT =1 & THEN CLEAR THE

MOV Instructions :

MOV instruction simply copy the data from one location to another location

MOV D,S

Copy the data from(S) source to D(destination)

MOV R0,A ; Copy contents of A into Register R0

MOV R1,A ; Copy contents of A into register R1

MOV A,R3 ; copy contents of Register R3 into Accnmulator.

Direct loading through MOV

MOV A,#23H ; Direct load the value of 23h in A

MOV R0,#12h ; direct load the value of 12h in R0

MOV R5,#0F9H ; Load the F9 value in the Register R5

ADD Instructions:

ADD instructions adds the source byte to the accumulator (A) and place the result in the Accumulator.

MOV A, #25H

ADD A, #42H ; BY this instructions we add the value 42h in Accumulator (42H+ 25H)

ADDA, R3 ;By this instructions we move the data from register r3 to accumulator and then add the contents of the register into accumulator .

SUBROUTINE CALL FUNCTION:

ACALL, TARGET ADDRESS

By this instructions we call subroutines with a target address within 2k bytes from the current program counter.

LCALL TARGET, ADDRESS.

ACALL is a limit for the 2 k byte program counter, but for upto 64k byte we use **LCALL** instructions.. Note that **LCALL** is a 3 byte instructions. **ACALL** is a two byte instructions.

AJMP Target, Address.

This is for absolute jump



AJMP stand for absolute jump. It transfers program execution to the target address unconditionally. The target address for this instruction must be within 2 k byte of program memory.

LJMP is also for absolute jump. It transfer program execution to the target address unconditionally. This is a 3 byte instructions LJMP jump to any address within 64 k byte location.

INSTRUCTIONS RELATED TO THE CARRY:

JC Target

Jump to target if CY Flag is =1

JNC Target

Jump to target if CY Flag is = 0

INSTRUCTIONS RELATED TO JUMP WITH ACCUMULATOR:

JZ Target

Jump to target if A=0.

JNZ Target

Jump if Accumulator is not zero.

This instructions jumps if registe A has a value other than zero

INSTRUCTIONS RELATED TO THE ROTATING THE BITS OF A REGISTER:

RL A

Rotate the accumulator left by one bit.

By this instructions we rotate the bits of A left. The bits rotated out of A are rotated back into A at the opposite end.

RR A

By this instruction we rotate the contents of the accumulator from right to left from LSB to MSB.

RRC A

This is same as RR A but difference is that the bit rotated out of register first enter in to carry and then enter into MSB.

RLC A

Rotate A left through carry.

Same as above but but shift the data from MSB to carry and carry to LSB.

RET

This is return from subroutine. This instructions is used to return from a subroutine previously entered by instructions LCALL and ACALL.

RET1

THIS is used at the end of an interrupt service routine. We use this instructions after intruupt routine.

PUSH

This copies the indicated byte onto the stack and increments SP by . This instructions supports only direct addressing mode.

POP

Pop from the stack.

This copies the byte pointed to be SP to the location whose direct address is indicated, and decrements SP by 1. Notice that this instructions supports only direct addressing mode.

DPTR INSTRUCTIONS.:

MOV DPTR,#16 BIT VALUE

Load data pointer with the 16 bit value.

This instructions load the 16 bit dptr register with a 16 bit immediate value

MOV C A,@A+DPTR

This instructions moves a byte of data located in program ROM into register A. This allows us to put strings of data, such as look up table elements.

MOVC A,@A+PC

This instructions moves a byte of data located in the program area to A. the address of the desired byte of data is formed by adding the program counter (PC) register to the original value of the accumulator.

INC BYTE

This instructions add 1 to the register or memory location specified by the operand.

INC A

INC R_n

INC DIRECT

DEC BYTE

This instructions subtracts 1 from the byte operand. Note that CY is unchanged

DEC A

DEC R_n

DEC DIRECT

ARITHMETIC INSTRUCTIONS:

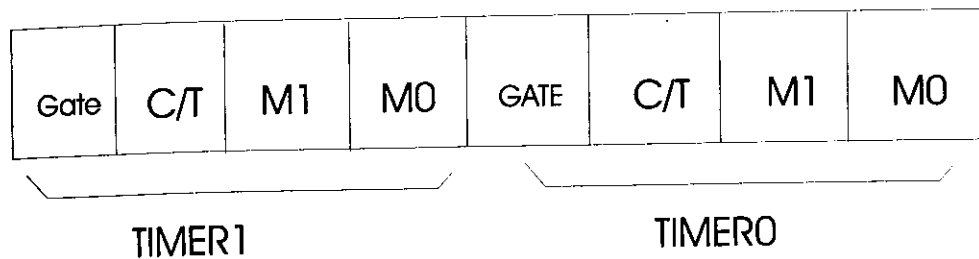
ANL dest-byte, source-byte

This performs a logical AND operation. This performs a logical AND on the operands, bit by bit, storing the result in the destination. Notice that both the source and destination values are byte -size only.

DIV AB

This instructions divides a byte accumulator by the byte in register B. It is assumed that both register A and B contain an unsigned byte. After the division the quotient will be in register A and the remainder in register B.

TMOD (TIMER MODE) REGISTER



Both timer is the 89c51 share the one register TMOD. 4 LSB bit for the timer 0 and 4 MSB for the timer 1.

In each case lower 2 bits set the mode of the timer

Upper two bits set the operations.

GATE: Gating control when set. Timer/counter is enabled only while the INTX pin is high and the TRx control pin is set. When cleared, the timer is enabled whenever the TRx control bit is set

C/T : Timer or counter selected cleared for timer operation (input from internal system clock)

M1 Mode bit 1

M0 Mode bit 0

M1	M0	MODE	OPERATING MODE
0	0	0	13 BIT TIMER/MODE
0	1	1	16 BIT TIMER MODE
1	0	2	8 BIT AUTO RELOAD
1	1	3	SPLIT TIMER MODE

PSW (PROGRAM STATUS WORD)

CY	AC	F0	RS1	RS0	OV	---	P
----	----	----	-----	-----	----	-----	---

CY	PSW.7	CARRY FLAG
AC	PSW.6	AUXILIARY CARRY
F0	PSW.5	AVAILABLE FOR THE USER FRO GENERAL PURPOSE
RS1	PSW.4	REGISTER BANK SELECTOR BIT 1
RS0	PSW.3	REGISTER BANK SELECTOR BIT 0

OV	PSW.2	OVERFLOW FLAG
--	PSW.1	USER DEFINABLE BIT
P	PSW.0	PARITY FLAG SET/CLEARED BY HARDWARE

PCON REGISATER (NON Bit-Addressable)

SMOD	---	---	---	GF1	GF0	PD	IDL
------	-----	-----	-----	-----	-----	----	-----

If the SMOD = 0 (DEFAULT ON RESET)

$$TH1 = \frac{\text{CRYSTAL FREQUENCY}}{256 \times \text{BAUD RATE}}$$

If the SMOD IS = 1

$$TH1 = \frac{\text{CRYSTAL FREQUENCY}}{256 \times \text{BAUD RATE}}$$

There are two ways to increase the baud rate of data transfer in the 8051

1. To use a higher frequency crystal
2. To change a bit in the PCON register

PCON register is an 8 bit register . Of the 8 bits, some are unused, and some are used for the power control capability of the 8051. the bit which is used for the serial communication is D7, the SMOD bit. When the 8051 is powered up, D7 (SMOD BIT) OF PCON register is zero. We can set it to high by software and thereby double the baud rate

BAUD RATE COMPARISION FOR SMOD = 0 AND SMOD =1

TH1 (DECIMAL)	HEX	SMOD =0	SMOD =1
-3	FD	9600	19200
-6	FA	4800	9600
-12	F4	2400	4800
-24	E8	1200	2400

XTAL = 11.0592 MHZ

IE (INTERRUPT ENABLE REGISTOR):

EA	---	ET2	ES	ET1	EX1	ETO	EXO
----	-----	-----	----	-----	-----	-----	-----

EA IE.7 Disable all interrupts if EA = 0, no interrupts is acknowledged
 If EA is 1, each interrupt source is individually enabled or disbaled
 By sending or clearing its enable bit.

IE.6 NOT implemented

ET2 IE.5 enables or disables timer 2 overflag in 89c52 only

ES IE.4 Enables or disables all serial interrupt

ET1 IE.3 Enables or Disables timer 1 overflow interrupt

EX1 IE.2 Enables or disables external interrupt

ET0 IE.1 Enables or Disables timer 0 interrupt.

EX0 IE.0 Enables or Disables external interrupt 0

INTERRUPT PRIORITY REGISTER

---	---	PT2	PS	PT1	PX1	PT0	PX0
-----	-----	-----	----	-----	-----	-----	-----

If the bit is 0, the corresponding interrupt has a lower priority and if the bit is 1 the corresponding interrupt has a higher priority

IP.7 NOT IMPLEMENTED, RESERVED FOR FUTURE USE.

IP.6 NOT IMPLEMENTED, RESERVED FOR FUTURE USE

PT2 IP.5 DEFINE THE TIMER 2 INTERRUPT PRIORITY LEVEL

PS IP.4 DEFINES THE SERIAL PORT INTERRUPT PRIORITY LEVEL

- PT1 IP.3 DEFINES THE TIMER 1 INTERRUPT PRIORITY LEVEL
- PX1 IP.2 DEFINES EXTERNAL INTERRUPT 1 PRIORITY LEVEL
- PT0 IP.1 DEFINES THE TIMER 0 INTERRUPT PRIORITY LEVEL
- PX0 IP.0 DEFINES THE EXTERNAL INTERRUPT 0 PRIORITY LEVEL

SCON: SERIAL PORT CONTROL REGISTER (Bit Addressable)

SCON

SM0	SM1	SM2	REN	TB8	RB8	T1	R1
-----	-----	-----	-----	-----	-----	----	----

- SM0 : SCON.7 Serial Port mode specifier
- SM1 : SCON.6 Serial Port mode specifier
- SM2 : SCON.5
- REN : SCON.4 Set/cleared by the software to Enable/disable reception
- TB8 : SCON.3 The 9th bit that will be transmitted in modes 2 and 3, Set/cleared By software
- RB8 : SCON.2 In modes 2 &3, is the 9th data bit that was received. In mode 1, If SM2 = 0, RB8 is the stop bit that was received. In mode 0 RB8 is not used
- T1 : SCON.1 Transmit interrupt flag. Set by hardware at the end of the 8th bit

Time in mode 0, or at the beginning of the stop bit in the other Modes. Must be cleared by software

CHAPTER 3

R1 SCON.0 Receive interrupt flag. Set by hardware at the end of the 8th bit
Time in mode 0, or halfway through the stop bit time in the other
Modes. Must be cleared by the software.

TCON TIMER COUNTER CONTROL REGISTER

This is a bit addressable register.

TF1 TCON.7 Timer 1 overflow flag. Set by hardware when the Timer/Counter 1
Overflows. Cleared by hardware as processor

TR1 TCON.6 Timer 1 run control bit. Set/cleared by software to turn Timer
Counter 1 On/off

TF0 TCON.5 Timer 0 overflow flag. Set by hardware when the timer/counter 0
Overflows. Cleared by hardware as processor

TR0 TCON.4 Timer 0 run control bit. Set/cleared by software to turn timer
Counter 0 on/off.

IE1 TCON.3 External interrupt 1 edge flag

ITI TCON.2 Interrupt 1 type control bit

IE0 TCON.1 External interrupt 0 edge

IT0 TCON.0 Interrupt 0 type control bit.

CHAPTER 3

WORKING OF THE MODEL

3.1 Introduction

In this project we show how we display the number of cars parked on a floor of a garage. This model has a single entry and single exit door system i.e. the same door is used to enter into the garage and the same is used to exit from it.

It's working is based on three logics basically viz:

1. Infra Red Sensor.
2. Infra Red Sensor Logic.
3. Microcontroller Interface.

3.2 Infra Red Sensor

The working of the infrared sensor logic is dependent on a photo-transistor and a IRED. A **phototransistor** is in essence nothing more than a normal bipolar transistor that is encased in a transparent case so that light can reach the Base-Collector diode. The phototransistor works like a photodiode, but with a much higher sensitivity for light, because the electrons that tunnel through the Base-Collector diode are amplified by the transistor function.

Phototransistors are specially designed transistors with the base region exposed. These transistors are light sensitive, especially when infrared source of light is used. They have only two leads (collector and emitter). When there is no light the phototransistor is closed and does not allow a collector-emitter current to go through. The phototransistor opens only with the presence of sufficient light.

An opto-electronic device that conducts current when exposed to light is the PHOTOTRANSISTOR. A phototransistor, however, is much more sensitive to light and produces more output current for a given light intensity than does a photodiode. Figure 3-32 shows one type of phototransistor, which is made by placing a photodiode in the base circuit of an NPN transistor. Light falling on the photodiode changes the base current of the transistor, causing the collector current to be amplified. Phototransistors may also be of the PNP type, with the photodiode placed in the base-collector circuit.

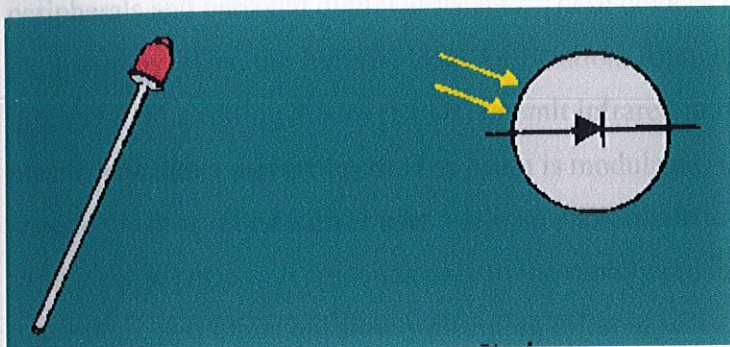


Fig.3.1: Photo-Transistor

Infrared (IR) radiation is electromagnetic radiation of a wavelength longer than visible light, but shorter than microwave radiation. The name means "below red" (from the Latin *infra*, "below"), red being the color of visible light of longest wavelength. Infrared radiation has wavelengths between 700 nm and 1 mm.

IR is often subdivided into near-IR (NIR, 0.7-5 μm in wavelength), mid-IR (MIR (also intermediate-IR (IIR)), 5 - 30 μm) and far-IR (FIR, 30 - 1000 μm). However, these terms are not precise, and are used differently in the various studies. Infrared radiation is often linked to heat, since objects at room temperature or above will emit radiation mostly concentrated in the mid-infrared band.

Infrared is used in night-vision equipment, when there is insufficient visible light to see an object. The radiation is detected and turned into an image on a screen, hotter objects showing up brighter, enabling the police and military to chase targets.

Smoke is more transparent to infrared than to visible light, so fire fighters apply infrared imaging equipment when working in smoke-filled areas.

A more common use of IR is in television remote controls. In this case it is used in preference to radio waves because it does not interfere with the television signal. IR data transmission is also employed in short-range communication among computer peripherals and personal digital assistants. These devices usually conform to standards published by IrDA, the Infrared Data Association. Remote controls and IrDA devices use infrared light-emitting diodes (LEDs) to emit infrared radiation which is focused by a plastic lens into a narrow beam. The beam is modulated, i.e. switched on and off, to encode the data. The receiver uses a silicon photodiode to convert the infrared radiation to an electric current. It responds only to the rapidly pulsing signal created by the transmitter, and filters out slowly changing infrared radiation from sunlight, people and other warm objects. The light used in fiber optic communication is typically infrared.

A **diode** functions as the electronic version of a one-way valve. By restricting the direction of movement of charge carriers, it allows an electric current to flow in one direction, but blocks it in the opposite direction.

A diode's current-voltage, or I-V, characteristic can be approximated by two regions of operation. Below a certain difference in potential between the two leads, the diode can be thought of as an open (non-conductive) circuit. As the potential difference is increased, at some stage the diode will become conductive and allow current to flow, at which point it can be thought of as a connection with zero (or at least very low) resistance.

A **light-emitting diode (LED)** is a semiconductor device that emits incoherent monochromatic light when electrically biased in the forward direction. This effect is a form of electroluminescence. The color depends on the semiconducting material used, and can be near-ultraviolet, visible or infrared. Nick Holonyak Jr. (1928 -) developed the first practical visible-spectrum LED in 1962.

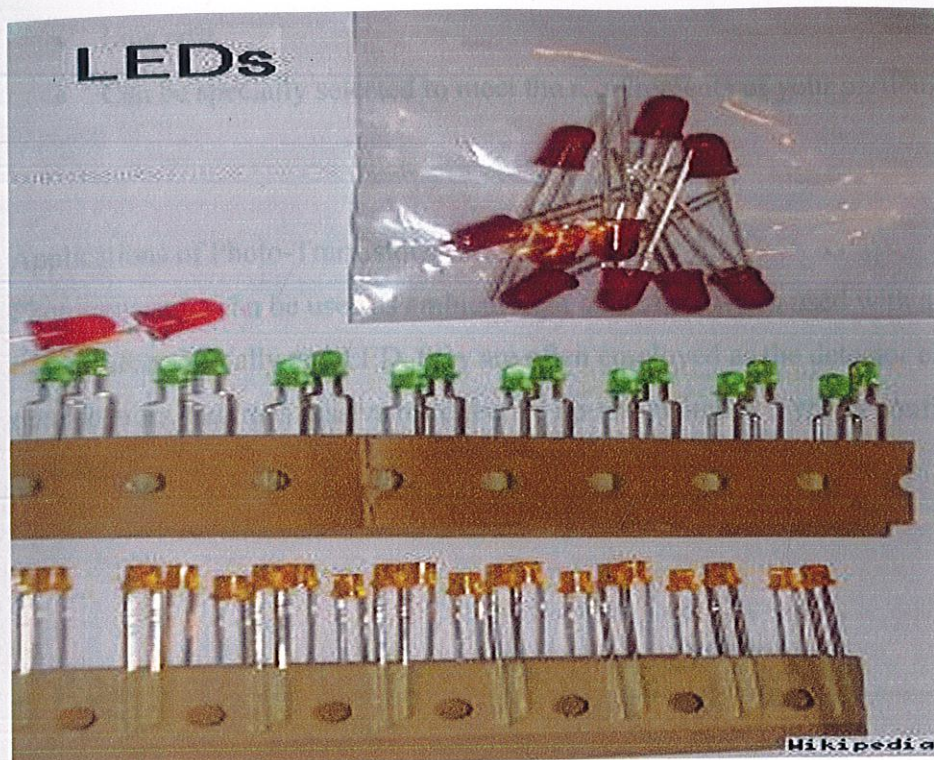


Fig 3.2: LEDs

Why Use IREDS?

IRED's are solid state light sources which emit light in the near-IR part of the spectrum. Because they emit at wavelengths which provide a close match to the peak spectral response of silicon photodetectors both GaAs and GaAlAs LEDs are often used with phototransistors and photodarlington. Key characteristics and features of these light sources include:

- Long operating lifetimes
- Low power consumption, compatible with solid state electronics
- Narrow band of emitted wavelengths
- Minimal generation of heat

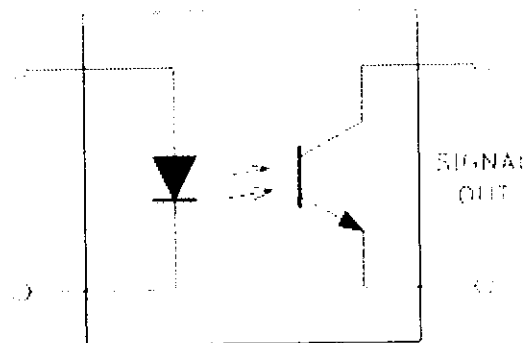
- Available in a wide range of packages including epoxy coated, transfer molded, cast and hermetic packages
- Low cost
- Can be specially selected to meet the requirements of your particular application

Applications of Photo-Transistors-

Phototransistors can be used as ambient light detectors. When used with a controllable light source, typically an LED, they are often employed as the detector element for optoisolators and transmissive or reflective optical switches. Typical configurations include:

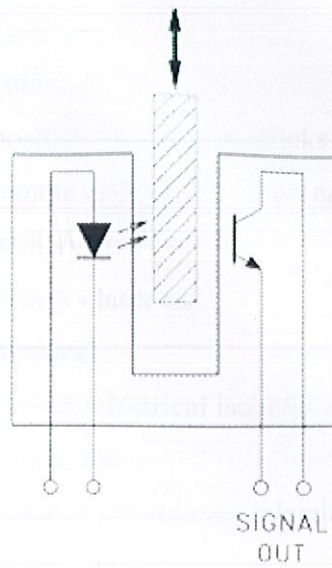
Optoisolator

The optoisolator is similar to a transformer in that the output is electrically isolated from the input.



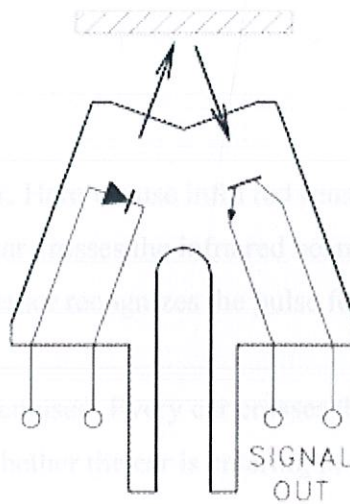
Optical Switch

An object is detected when it enters the gap of the optical switch and blocks the light path between the emitter and detector.



Retro Sensor

The retrosensor detects the presence of an object by generating light and then looking for its reflectance off of the object to be sensed.



Phototransistors and IREDS have been used in the following applications.

Computer/Business Equipment

track zero detector - floppy drive
margin controls - printers
read finger position - touch screen
detect holes - computer card
monitor paper position - copiers

Industrial

LED light source - light pens
security systems
safety shields
encoders - measure speed and direction
photoelectric controls

Consumer

coin counters
position sensors - joysticks
remote controllers - toys, appliances,
audio/visual equipment
games - laser tag

Medical

provide electrical isolation between patient and
equipment
monitor intravenous injection rates

3.3 Infra Red Sensor Logic

First part of this project is Infra red sensor. Here we use infra red sensor as a transmitter and photodiode as a receiver. When any car crosses the infra red beam then circuit provides a sharp pulse and the microprocessor recognizes the pulse for counting.

In this project two beams of light have been used. Every car crosses these two beams. By crossing these beams in steps we check whether the car is entering in the parking palace or exiting from there.

The photodiode in this sensor is connected to the IC 555. Here IC 555 work as a monostable timer. Here two 555 timers have been used.

First part of this project is to sense the beam and provide data to the MICROCONTROLLER circuit. For this purpose two IC circuits have been used. Here IC 555 is used as a sensor circuit where it is configured as a Monostable timer. In this project two infra red sensors have been used. Both have been connected to the same door.

Two photodiode's are also connected to the receiver circuit to detect the infra red signal. Both the infra red sensor are connected to the IC 555 which is working as a monostable timer. In this monostable IC 555. Pin no 8 of this IC is connected to the positive supply. Pin no 1 is connected to the negative supply. Pin no 2 is connected to the photodiode (which is the pin through which any input to the IC is given). In this project we use two IC 555.

Working of this project is as follows:

When anybody enter in the parking space, then one infra red sensor is interrupted and only one of the ICs is enabled. At this time second IC555 is disabled. When any vehicle comes out from the parking space, then the other IC555 gets enabled, while the first IC555 gets disabled. Note that only one sensor is ON at any particular time. With the help of this IC, we give an up and down pulse to the microcontroller.

Photodiode is connected to the pin no.2 via a 22 Kohm resistor. The 22 Kohm resistor is grounded from the pin no.2. In normal way when circuit is switched ON, both the infra red sensors are on, and light falls on the photodiode. Now when any vehicle enters into the parking space, then the circuit senses the interruption of the beam and at this time 555 gives its output. Output from the 555 is connected to the MICROCONTROLLER through a NPN transistor. Here we get this output from the collector of the transistor. This collector point is also connected to the pin no. 4 of second IC. Output available on the collector point is negative and due to this, pin no 4 of the next IC gets a negative voltage and this in turn switches OFF the other IC555. With the help of this logic we switch on one IC and switch off the second one i.e. by controlling the input to the pin no. 4 of the ICs.

Basic of the IC 555 as a monostable timer.

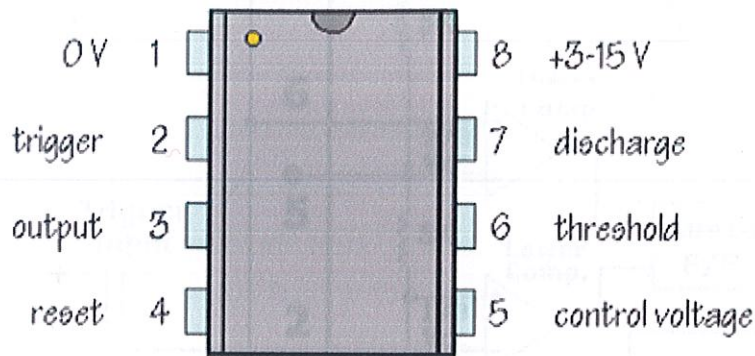


Fig.3.3: IC of a 555Timer.

The 555 timer IC was first introduced around 1971 by the Signetics Corporation as the SE555/NE555 and was called "**The IC Time Machine**" and was also the very first and only commercial timer IC available. It provided circuit designers and hobby tinkers with a relatively cheap, stable, and user-friendly integrated circuit for both monostable and astable applications. Since this device was first made commercially available, a myriad of novel and unique circuits have been developed and presented in several trade, professional, and hobby publications. The past ten years some manufacturers stopped making these timers because of competition or other reasons. Yet other companies, like NTE (a subdivision of Philips) picked up where some left off.

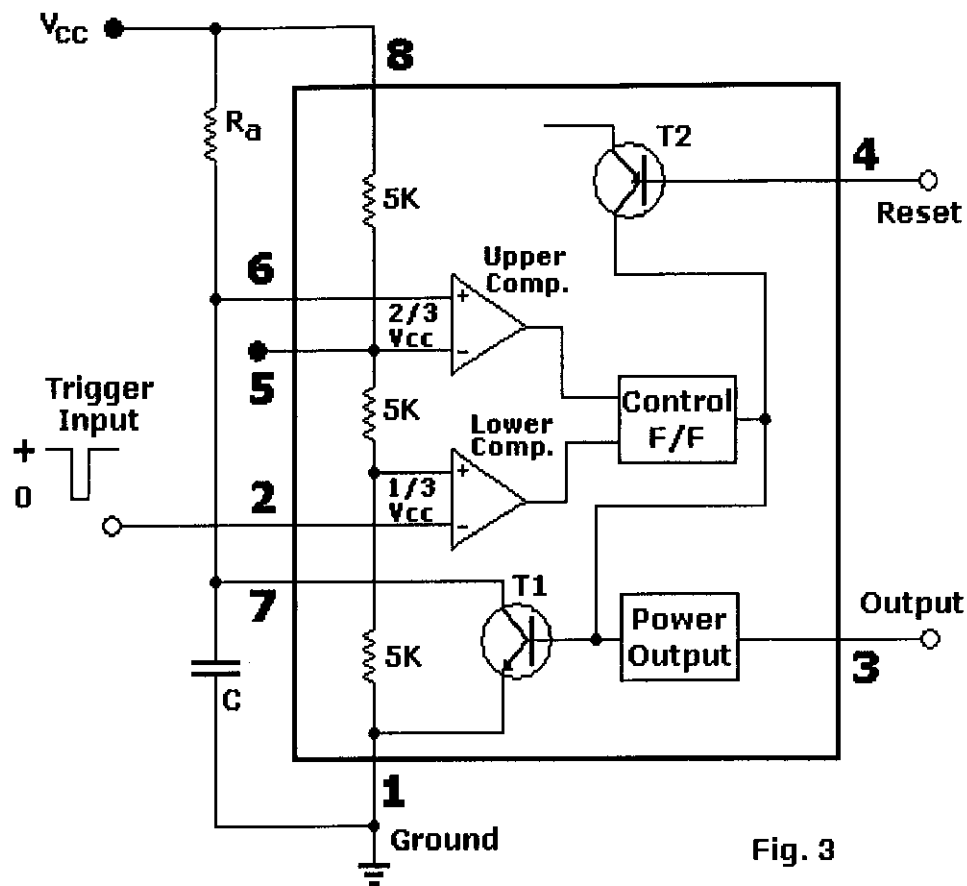


Fig. 3

Fig.3.4: Structure of IC 555

3.4 Microcontroller Interfacing

Here in this project IC 89c51 microcontroller has been used to interface with the IC555 circuit. In this project one 8051 family member IC named 89c51 has been used. This IC is a 40 pin microcontroller and has a built in flash memory. In this IC there are a total of four ports, of which two have been utilized for input and output of signals. We have used this IC for the control of all functions of the machine.

A program had to be written in the assembler software in the computer. The program is written in the assembly language. The assembly language program consists of assembly instructions or statements called directives. While instructions tell the CPU what to do,

the directives give instructions to the assembler After assembling the software in assembler, the file is stored as a “.asm “ file. Assembler automatically converts this ‘.asm’ file into a hex file (which is nothing but the same program written in hexadecimal language). Now this hex file had to be transferred from the computer to a blank IC with the help of a programmer kit which burns the hex file onto the ROM memory of the chip. The program for controlling the operations of the parking system is small enough to be accommodated into the built-in memory of IC 89c51. Pin no. 40 of the IC is connected to the 5 volt power supply (Vcc). For the 5 volt supply we use 5 volt regulated power supply, which is obtained from the 220V normal AC supply by the use of a step-down transformer and a rectifier. For this purpose a step down transformer along with a diode, a capacitor and regulator IC 7805 are used. This regulates the 220V AC supply voltage to 5 volt DC supply. This 5 volt dc is connected to the pin no 40 of the controller. Pin no 20 is connected to the ground.

The assembly language program for the parking system basically consists of four blocks viz:

1. Keycheck.
2. Increment.
3. Decrement.
4. Compare.

keycheck:

```
jnb p1.0, inc
jnb p1.1, dec
sjmp keycheck
```

In the microcontroller firstly we sense the input for entry and exit. Here we assign two pins for the input signal. For this purpose we use two pins as input. JNB p1.0 and JNB p1.1 are the commands used to check whether any of the sensors have been interrupted or

not. The program rotates in the keycheck function until there is a low signal on either of the pins.

inc:

```
jnb p1.0, inc
acall delay1
inc r1
mov a, r1
sjmp com
```

This block of code is invoked if the vehicle has crossed the beam and entered the parking space. When the pin P1.0 gets a low signal, the control jumps from the keycheck block to this Increment block. The second JNB P1.0 makes sure that the vehicle has crossed the beam and entered the parking space and that it is not standing at the door.

When the control enters this block, first a delay sub-routine is called after which the count of the vehicles in the garage is increased by one and the Compare block is called in order to display the appropriate count of vehicles in the garage.

dec:

```
jnb p1.1, dec
acall delay1
dec r1
mov a, r1
sjmp com
```

This block of code is invoked if the vehicle has crossed the beam and gone out of the parking space. When the pin P1.1 gets a low signal, the control jumps from the keycheck block to this Decrement block. The second JNB P1.1 makes sure that the vehicle has crossed the beam and entered the parking space and that it is not standing at the door.

When the control enters this block, first a delay sub-routine is called after which the count of the vehicles in the garage is decreased by one and the Compare block is called in order to display the appropriate count of vehicles in the garage.

By this way we assign two functions for increment and decrement of the count of vehicles.

Output of the microcontroller is connected to the seven segment display.

com:

```
mov a, r1
cjne a, #02, k1
mov p2, # 11111110b
mov p3, # 0ffh
mov p0, #11111001b ; display 1
sjmp keycheck
```

This block of code is invoked in order to display the appropriate number of vehicles that are parked in the parking space at a time. After the Increment / Decrement block has updated the count of vehicles appropriately, this block compares the count to the numbers one through eight and displays the proper number on the seven segment display.

k1:

```
mov a, r1
cjne a, # 03, K2
mov p2, #11111100b
mov p3, #0ffh
mov p0, #10100100b ; DISPLAY '2'
sjmp keycheck
```


k2:

```
mov a, r1
cjne a, #04, k3
mov p2, #11111000b
mov p0, #10110000b ; display '3"
mov p3, #0ffh
sjmp keycheck
```

After the Compare block is invoked, the control is sent to the appropriate display block K1 through K8.

The circuit diagram of the connections used to implement this Car Parking Model is given next.

2.5 Circuit Diagram of Connections

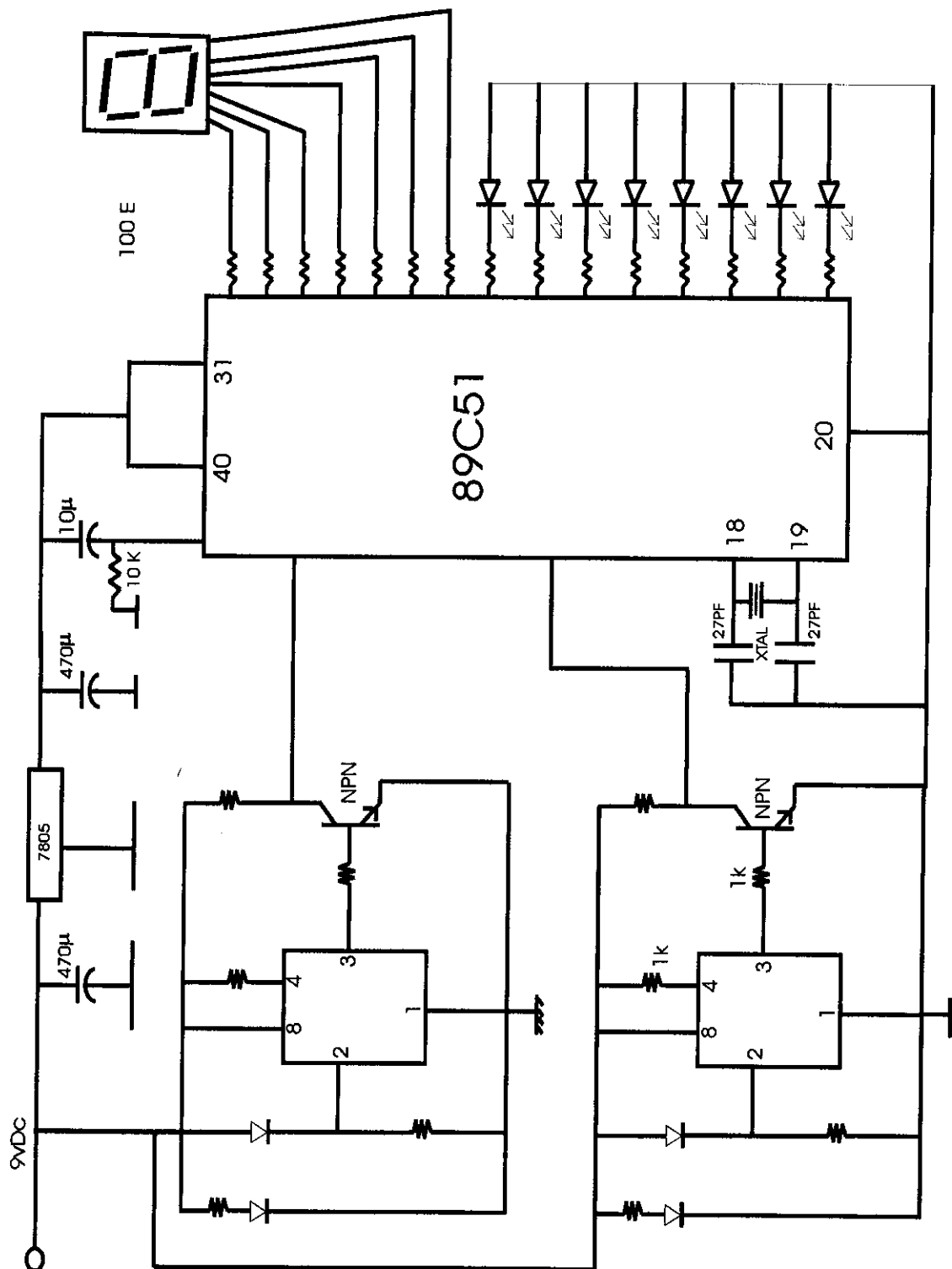


Fig.3.5: Circuit Diagram

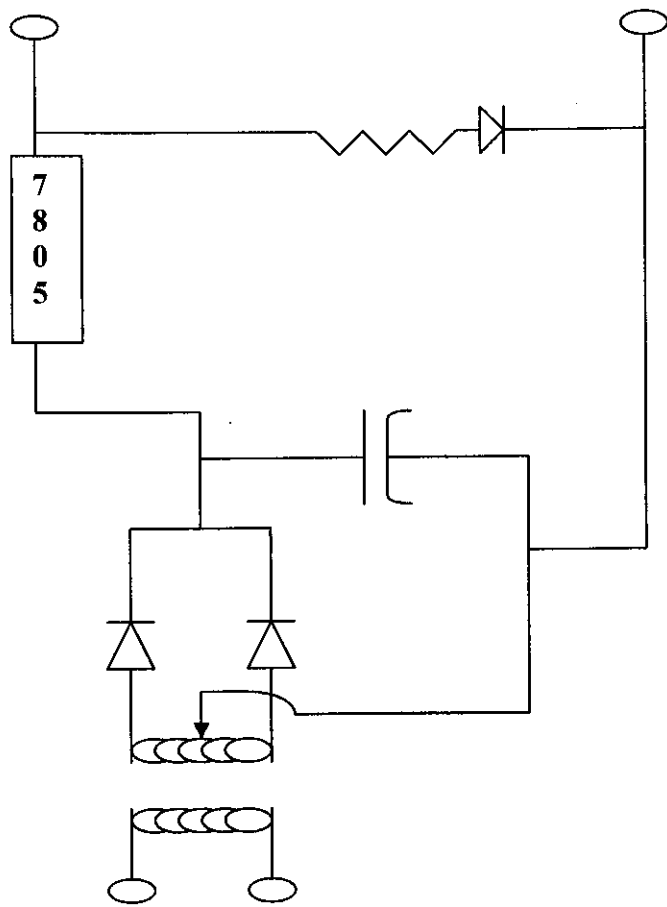


Fig.3.6: Conversion of 220V AC to 9-0-9 AC by the use of Centre Tap Transformer and Rectifier

CHAPTER 4

CONCLUSION AND FUTURE WORK

A combination of good strategy and proper choice of tools is necessary the faster implementation of the project.

As the present project presented the idea of an automated car parking system, the scope for the use of a microcontroller was fully exploited. But ultimately it is the job of the designer and programmer to tell the microcontroller as to what it should do. The microcontroller has a built in memory where the set of instructions can be stored The program can be written in either Assembly language or C language. Even if the size of the program exceeds the memory limit of the microcontroller, it can be stored on an external ROM and then interfaced with the microcontroller. So considering these flexibilities in the microcontroller and their versatility we can definitely think about automating many of our day to day chores. In fact microcontrollers are used in washing machines, remote controllers, card readers...etc the list goes on and on.

Programming the microcontroller is the easy part. The hard part to generate the input signals for the microcontroller. The biggest challenge was to make the model as a single door system. The microcontroller can recognize changes in the input within a fraction of a second. As both the sensor beams are located at the same place, any entering or exiting car would interrupt both of them, and that means the count gets incremented and decremented simultaneously. To avoid this problem, the second sensor had to be deactivated once the first sensor was interrupted. This was achieved by disabling one of the IC 555s by giving its pin number 4 a '-ve input.

Once this problem was solved, all the other pieces of this project just fell into their right places.

There can be many future enhancements to this project. For instance one could be the incorporation of a door placed outside the entry point. This door would close once the parking space is filled to its capacity. Whenever a car is about to leave the garage, the door would open upon the interruption of the exit sensor beam.

It can be further enhanced by making it multistoried by the incorporation of a lift. Furthermore, VIP floors can be added to this system. The door to these floors would open only if the correct password is entered.

APPENDIX - 1

Program to control the operations of the microcontroller:

```
org 0000h
```

```
keycheck:
```

```
  jnb p1.0,incr
```

```
  jnb p1.1,decr
```

```
  sjmp keycheck
```

```
incr:
```

```
  jnb p1.0,incr
```

```
  acall delay1
```

```
  inc r1
```

```
  mov a,r1
```

```
  sjmp com
```

```
decr:
```

```
  jnb p1.1,decr
```

```
  acall delay1
```

```
  dec r1
```

```
  mov a,r1
```

```
  sjmp com
```

```
com:
```

```
  mov a,r1
```

```
  cjne a,#02,k1
```

```
  mov p2,# 11111110b
```

```
  mov p3,# 0ffh
```

```
  mov p0,#11111001b ; display 1
```

```
  sjmp keycheck
```

```

k1:
  mov a,r1
  cjne a,# 03,k2
  mov p2,#11111100b
  mov p3,#0ffh
  mov p0,#10100100b ;DISPLAY '2'
  sjmp keycheck
k2:
  mov a,r1
  cjne a,#04,k3
  mov p2,#11111000b
  mov p0,#10110000b ; display '3"
  mov p3,#0ffh
  sjmp keycheck
k3:
  mov a,r1
  cjne a,#04,k4
  mov p2,# "code for 4"
  mov p0,# "code for 4" ; display '4"
  mov p3,#0ffh
  sjmp keycheck
k4:
  mov a,r1
  cjne a,#04,k5
  mov p2,# "code for 5"
  mov p0,# "code for 5" ; display '5"
  mov p3,#0ffh
  sjmp keycheck
k5:
  mov a,r1
  cjne a,#04,k6
  mov p2,# "code for 6"

```

```
mov p0,# "code for 6" ; display '6"
```

```
mov p3,#0ffh
```

```
sjmp keycheck
```

```
k6:
```

```
mov a,r1
```

```
cjne a,#04,k7
```

```
mov p2,# "code for 7"
```

```
mov p0,# "code for 7" ; display '7"
```

```
mov p3,#0ffh
```

```
ljmp keycheck
```

```
k7:
```

```
mov a,r1
```

```
cjne a,#04,k0
```

```
mov p2,# "code for 8"
```

```
mov p0,# "code for 8" ; display '8"
```

```
mov p3,#0ffh
```

```
ljmp keycheck
```

```
k0:
```

```
mov a,r1
```

```
mov p2,# "code for 0"
```

```
mov p0,# "code for 0" ; display '0"
```

```
mov p3,#0ffh
```

```
ljmp keycheck
```

```
DELAY1:
```

```
nop
```

```
nop
```

```
nop
```

```
nop
```

```
nop
```

nop
nop
nop
ret

APPENDIX - 2

8051 Instruction Set

Arithmetic Operations

Mnemonic	Description	Size	Cycles
ADD A,Rn	Add register to Accumulator (ACC).	1	1
ADD A,direct	Add direct byte to ACC.	2	1
ADD A,@Ri	Add indirect RAM to ACC	1	1
ADD A,#data	Add immediate data to ACC	2	1
ADDC A,Rn	Add register to ACC with carry	1	1
ADDC A,direct	Add direct byte to ACC with carry.	2	1
ADDC A,@Ri	Add indirect RAM to ACC with carry.	1	1
ADDC A,#data	Add immediate data to ACC with carry.	2	1
SUBB A,Rn	Subtract register from ACC with borrow.	1	1
SUBB A,direct	Subtract direct byte from ACC with borrow	2	1

SUBB A,@Ri	Subtract indirect RAM from ACC with borrow.	1	1
SUBB A,#data	Subtract immediate data from ACC with borrow.	2	1
INC A	Increment ACC.	1	1
INC Rn	Increment register.	1	1
INC direct	Increment direct byte.	2	1
INC @Ri	Increment indirect RAM.	1	1
DEC A	Decrement ACC.	1	1
DEC Rn	Decrement register.	1	1
DEC direct	Decrement direct byte.	2	1
DEC @Ri	Decrement indirect RAM.	1	1
INC DPTR	Increment data pointer.	1	2
MUL AB	Multiply A and B Result: A <- low byte, B <- high byte.	1	4
DIV AB	Divide A by B Result: A <- whole part, B <- remainder.	1	4
DA A	Decimal adjust ACC.	1	1

Logical Operations

Mnemonic	Description	Size	Cycles
ANL A,Rn	AND Register to ACC.	1	1
ANL A,direct	AND direct byte to ACC.	2	1
ANL A,@Ri	AND indirect RAM to ACC.	1	1
ANL A,#data	AND immediate data to ACC.	2	1
ANL direct,A	AND ACC to direct byte.	2	1
ANL direct,#data	AND immediate data to direct byte.	3	2
ORL A,Rn	OR Register to ACC.	1	1
ORL A,direct	OR direct byte to ACC.	2	1
ORL A,@Ri	OR indirect RAM to ACC.	1	1
ORL A,#data	OR immediate data to ACC.	2	1
ORL direct,A	OR ACC to direct byte.	2	1
ORL direct,#data	OR immediate data to direct byte.	3	2
XRL A,Rn	Exclusive OR Register to ACC.	1	1
XRL A,direct	Exclusive OR direct byte to ACC.	2	1

XRL A,@Ri	Exclusive OR indirect RAM to ACC.	1	1
XRL A,#data	Exclusive OR immediate data to ACC.	2	1
XRL direct,A	Exclusive OR ACC to direct byte.	2	1
XRL direct,#data	XOR immediate data to direct byte.	3	2
CLR A	Clear ACC (set all bits to zero).	1	1
CPL A	Compliment ACC.	1	1
RL A	Rotate ACC left.	1	1
RLC A	Rotate ACC left through carry.	1	1
RR A	Rotate ACC right.	1	1
RRC A	Rotate ACC right through carry.	1	1
SWAP A	Swap nibbles within ACC.	1	1

Data Transfer

Mnemonic	Description	Size	Cycles
MOV A,Rn	Move register to ACC.	1	1

MOV A,direct	Move direct byte to ACC.	2	1
MOV A,@Ri	Move indirect RAM to ACC.	1	1
MOV A,#data	Move immediate data to ACC.	2	1
MOV Rn,A	Move ACC to register.	1	1
MOV Rn,direct	Move direct byte to register.	2	2
MOV Rn,#data	Move immediate data to register.	2	1
MOV direct,A	Move ACC to direct byte.	2	1
MOV direct,Rn	Move register to direct byte.	2	2
MOV direct,direct	Move direct byte to direct byte.	3	2
MOV direct,@Ri	Move indirect RAM to direct byte.	2	2
MOV direct,#data	Move immediate data to direct byte.	3	2
MOV @Ri,A	Move ACC to indirect RAM.	1	1
MOV @Ri,direct	Move direct byte to indirect RAM.	2	2
MOV @Ri,#data	Move immediate data to indirect RAM.	2	1
MOV DPTR,#data16	Move immediate 16 bit data to data pointer register.	3	2
MOVC A,@A+DPTR	Move code byte relative to DPTR to ACC (16 bit address).	2	1

MOVC A,@A+PC	Move code byte relative to PC to ACC (16 bit address).	1	2
MOVX A,@Ri	Move external RAM to ACC (8 bit address).	1	2
MOVX A,@DPTR	Move external RAM to ACC (16 bit address).	1	2
MOVX @Ri,A	Move ACC to external RAM (8 bit address).	1	2
MOVX @DPTR,A	Move ACC to external RAM (16 bit address).	1	2
PUSH direct	Push direct byte onto stack.	2	2
POP direct	Pop direct byte from stack.	2	2
XCH A,Rn	Exchange register with ACC.	1	1
XCH A,direct	Exchange direct byte with ACC.	2	1
XCH A,@Ri	Exchange indirect RAM with ACC.	1	1
XCHD A,@Ri	Exchange low order nibble of indirect RAM with low order nibble of ACC	1	1

Boolean Variable Manipulation

Mnemonic	Description	Size	Cycles
CLR C	Clear carry flag.	1	1
CLR bit	Clear direct bit.	2	1

SETB C	Set carry flag.	1	1
SETB	bitSet direct bit	2	1
CPL C	Compliment carry flag.	1	1
CPL bit	Compliment direct bit.	2	1
ANL C,bit	AND direct bit to carry flag.	2	2
ANL C,/bit	AND compliment of direct bit to carry.	2	2
ORL C,bit	OR direct bit to carry flag.	2	2
ORL C,/bit	OR compliment of direct bit to carry.	2	2
MOV C,bit	Move direct bit to carry flag.	2	1
MOV bit,C	Move carry to direct bit.	2	2
JC rel	Jump if carry is set.	2	2
JNC rel	Jump if carry is not set.	2	2
JB bit,rel	Jump if direct bit is set.	3	2
JNB bit,rel	Jump if direct bit is not set.	3	2
JBC bit,rel	Jump if direct bit is set & clear bit.	3	2

Program Branching

Mnemonic	Description	Size	Cycles
ACALL	addr11 Absolute subroutine call.	2	2
LCALL	addr16 Long subroutine call.	3	2
RET	Return from subroutine.	1	2
RETI	Return from interrupt.	1	2
AJMP	addr11 Absolute jump.	2	2
LJMP	addr16 Long jump.	3	2
SJMP	rel Short jump (relative address).	2	2
JMP	@A+DPTR Jump indirect relative to the DPTR.	1	2
JZ	rel Jump relative if ACC is zero.	2	2
JNZ	rel Jump relative if ACC is not zero.	2	2
CJNE	A,direct,rel Compare direct byte to ACC and jump if not equal.	3	2
CJNE	A,#data,rel Compare immediate byte to ACC and jump if not equal.	3	2
CJNE	Rn,#data,rel Compare immediate byte to register and jump if not equal.	3	2

CJNE @Ri,#data,rel Compare immediate byte to indirect and jump if not equal.32

DJNZ Rn,rel Decrement register and jump if not zero. 2 2

DJNZ direct,rel Decrement direct byte and jump if not zero. 3 2

Other Instructions

Mnemonic	Description	Size	Cycles
NOP	No operation.	1	1

APPENDIX – 3

Datasheet of IC 7805:

KA78XX/KA78XXA

3-Terminal 1A Positive Voltage Regulator

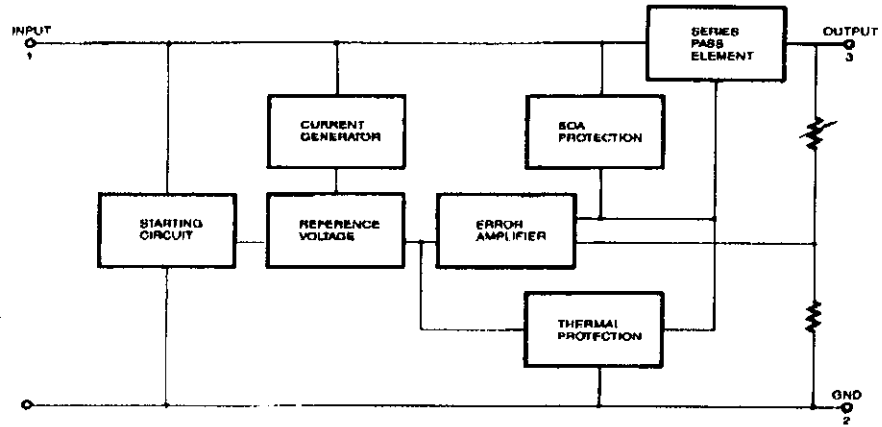
Features

- 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 KA78XX/KA78XXA series of three-terminal positive regulator 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 shut down 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.

Internal Block Diagram



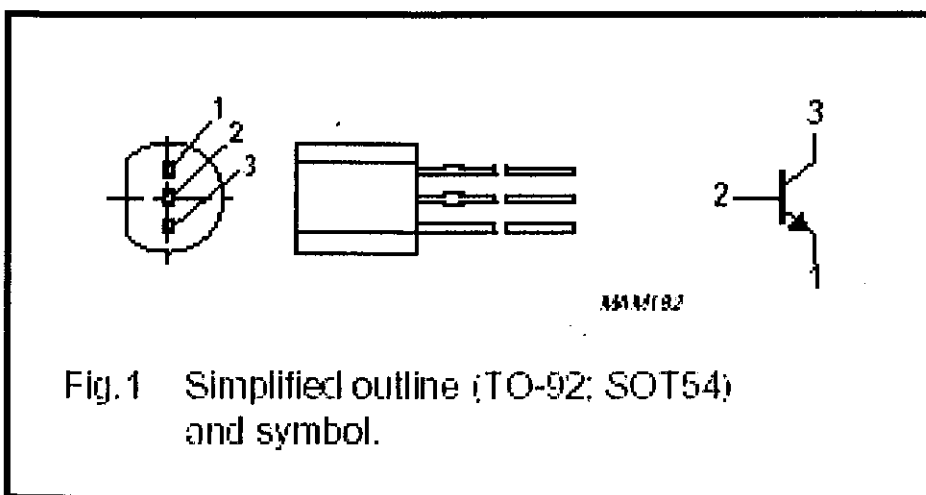
APPENDIX – 4

BC546; BC547

NPN general purpose transistors

PINNING

PIN	DESCRIPTION
1	emitter
2	base
3	collector



FEATURES

- Low current (max. 100 mA)
- Low voltage (max. 65 V).

APPLICATIONS

- General purpose switching and amplification.

DESCRIPTION

NPN transistor in a TO-92; SOT54 plastic package.

PNP complements: BC556 and BC557.

LIMITING VALUES

In accordance with the Absolute Maximum Rating System (IEC 134).

SYMBOL	PARAMETER	CONDITIONS	MIN.	MAX.	UNIT
V _{CB0}	collector-base voltage	open emitter			
	BC546		-	80	V
	BC547		-	50	V
V _{CE0}	collector-emitter voltage	open base			
	BC546		-	65	V
	BC547		-	45	V
V _{EB0}	emitter-base voltage	open collector			
	BC546		-	6	V
	BC547		-	6	V
I _C	collector current (DC)		-	100	mA
I _{CM}	peak collector current		-	200	mA
I _{BM}	peak base current		-	200	mA
P _{tot}	total power dissipation	T _{amb} ≤ 25 °C; note 1	-	500	mW
T _{stg}	storage temperature		-65	+150	°C
T _j	junction temperature		-	150	°C
T _{amb}	operating ambient temperature		-65	+150	°C

BIBLIOGRAPHY

- 1) M.A Mazidi , J.G Mazidi , R.D Mckinlay *The 8051 Microcontroller and Embedded Systems*. Pearson Prentice Hall Publications(P). New Delhi Second Edition ninth reprint 2006
- 2) IEEE Transactions on Automation Science and Engineering
- 3) www.google.com
- 4) www.altavista.com
- 5) www.wikipedia.com

