WIRELESS FIRE SENSING FLYING ROBOT

A Project Report

Submitted by

M. Ganesh Kannan	-	71203105010
A.Gowri Shankar	-	71203105015
G.Naga Santhosh	-	71203105029
K. Vikram	-	71203105054

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

ELECTRICAL AND ELECTRONICS ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY
COIMBATORE - 641006

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2007

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project entitled **"Wireless Fire Sensing Flying Robot "** is the bonafide work of

Mr. M.Ganesh Kannan	- Reg.No. 71203105010
Mr. A.Gowri Shankar	- Reg.No. 71203105015
Mr. G.Naga Santhosh	- Reg.No. 71203105029
Mr. K.Vikram	- Reg.No. 71203105054

Who carried out the project work under my supervision.

Signature of the Head of the Department

Signature of the Supervisor

Prof..K.REGUPATHY SUBRAMANIAN

Ms.C.VIJAYANANDHI

Certified that the candidate with university Register No. ______________ was

examined in project viva voce Examination held on 26.04.2007

Internal Examiner 26/4/07

External Examiner 26/4/07

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY
COIMBATORE - 641006

ABSTRACT

In this project a wireless flying robot has been designed for fire detection and monitoring. A prototype for the same is developed. The robot helps to have bird view on the spot and assists in finding any one trapped inside the fire. The application can be extended for security purpose.

In case of fire, the destination to the flying machine is provided by the user through PC using appropriate control code .The control codes will be generated by a **(PIC16F877)** micro controller in response to the received (sensed) signal from the sensing unit of the flying kit. The generated code from the control unit will be transmitted to flying robot with the help of RS232 through the wireless RF module. The RF module aids in the wireless (remote) control of the robot for a limited range of 100 meters.

The robot will able to move in all direction depending upon the code received by them. The light dependent resistor (LDR) senses the fire and transmits the sensed signal to the control unit through RF transmitter for monitoring purpose. The camera connected with the flying robot also helps in monitoring. A PC is used for monitoring the received signals, also videos can be recorded for future references. In addition a Laser beam actuated by the LDR signal is used to exactly locate the fire point.

ACKNOWLEDGEMENT

We express our heartful thanks to our beloved correspondent **Dr.K.ARUMUGAM** for his kind patronage in pursuing this project work successfully.

We wish to express our sincere thanks to our Principal **Dr.JOSEPH V.THANIKAL**, for providing support and necessary facilities to carryout this work..

We are highly indebted and convey our most sincere thanks to our Head of the Department **Prof.K.REGUPATHY SUBRAMANIAN**, for his invaluable suggestions towards this project.

We are extremely grateful to our guide **Ms.C.VIJAYANANDHI M.E** for her guidance in completing this project most successfully.

We are very much grateful to our parents for their blessings and encouragement throughout the project.

Last, but not the least, we like to express our sincere thanks to all the faculty, and supporting staffs of the Department of Electrical and Electronics Engineering for their co-operation in completing this work successfully.

CONTENTS

.

List of figures

List of tables

List of Abbreviations

DCE - Data Communication Equipment

LCD – Liquid Crystal Display

LDR – Light Dependent Resistor

PIC – Peripheral Interface Control

RTS- Request to Send

RISC – Reduced Instruction Set Code

USART - Universal Synchronous Asynchronous Receiver Transmitter

WDT – Watch Dog Timer

WIRELESS FIRE SENSING FLYING ROBOT – AN INTRODUCTION

1.INTRODUCTION

Need for the project

In the present world the field of robotics is improving in leaps and bounds. There are enormous numbers of applications where the human beings are replaced by robots. There's also a need to free up human from hazardous and unsafe sites and utilize well-designed telerobot for those purpose. In practice, the surface vehicles are used to monitor the fire accidents. But in this project flying robot is used to sense and monitor the fire accidents.

Objective of the project

The main objective of the project is to transmit the image captured by the robot in the survey field to system and also to control the entire operation of the robot including fire detection.

Feature of the project

"Robot" comes from the Czech word "robota", meaning, "forced labor."

"A reprogram able, multifunctional manipulator designed to move material, parts, tools, or specialized devices through various programmed motions for the performance of a variety of tasks"

Scope of the project

Due to various problems encountered by human beings in dangerous areas, there is a need to employ robots in such sites. This robot can be used near oil wells and in nuclear & chemical plants as watchdog for security purposes. It is also used to investigate the unexplored areas contains toxic gases.

Project Overview

The over view of the Wireless fire sensing flying robot is given below.

Our project consists of two main units namely,

- Control unit
- Sensing & Flying unit.

In this project the designed robot can be controlled using PC with the help of control unit.

When a fire accident occurs, the user estimates the destination and gives the corresponding code, for the flying robot to reach the target. RS232 is used to interface between the PC and control unit, MAX232 is used for serial interfacing and the micro controller generates a corresponding pulse with the help of their ports. Each port have their corresponding code differs from each other.

When flying, the LDR senses for the fire and if the fire is detected, the circuit gets closed and the signal is transmitted through RF transmitter. The camera attached with robot helps in guiding the user to reach the destination. The RF receiver receives the RF signal transmitted and the signal is processed by micro controller and displayed in LCD as fire. The control codes can be continuously changed, so that the robot changes its position to view the exact location.

1.1 Control Unit

The components of control unit compises the signal transimitter and video reciver unit.The block diagram of the control unit is given below.

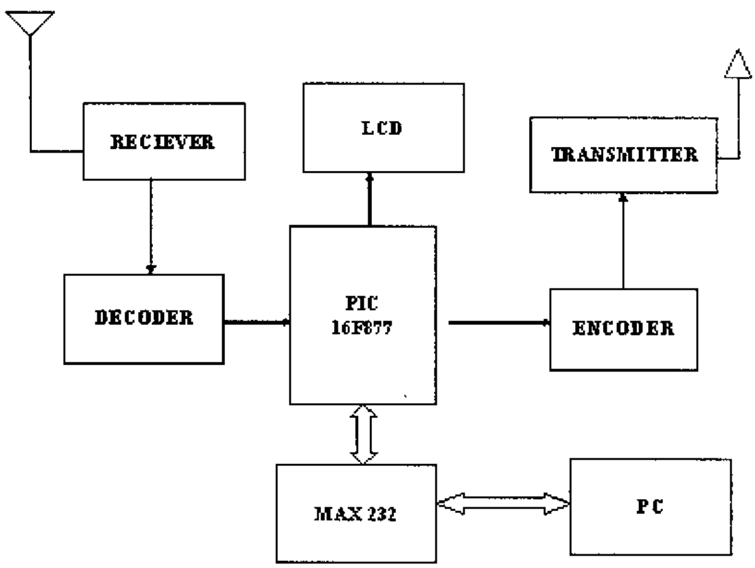


Fig.1.1 Block diagram of Control unit

Signal reception section

The Receiver section receives the signals transmitted by the transmitter section as RF wave. The Transmitter section has two main components namely, RF Receiver (RX 434), Decoder (HT 12D).

RF Receiver

The Receiver (RX 434), receives the serial data via RF transmitted by the control section and gives it to the decoder module.

Decoder

The Decoder (HT 12D) converts the serial input to parallel output and gives it to the microcontroller. Input to decoder is given from the RF receiver module.

Video receiver

This section receives the video of the survey field, sent from the Robot section via the video transmitter. Its output is given to the computer via TV Tuner Card,which acts as monitoring unit.

1.2 Sensing and flying unit

The components of sensing and flying unit compises the LDR and camera unit.The block diagram of the is sensing and flying unit given below

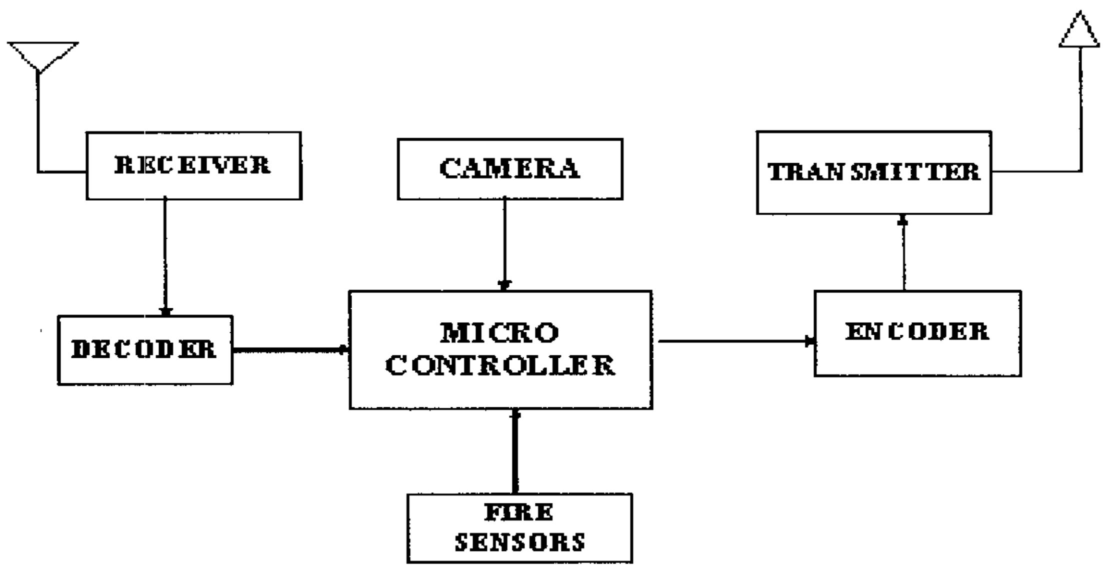


Fig 1.2. Block diagram of Flying unit

Signal transmitter section

The Transmitter section transmits the signals to the Robot unit. The input to this section is given through Parallel Port. The Transmitter section has two main components namely, Encoder (HT 12E),RF Transmitter (TX 434).

Encoder

The encoder (HT 12E) converts the parallel input to serial output and gives it to transmitter. Input to encoder is given from the parallel port.

Transmitter

The Transmitter (TX 434) transmits the serial data given to it from encoder through air via RF wave at 433 MHz.

Microcontroller

The heart of the Robot section is the microcontroller - ATMEL 89C51.Some of the advantages in using the ATMEL 89C51 microcontoller are inbuilt storage, PCB size reduced, low design cost. This module receives the input from the receiver section and processes the data. It controls the driver circuit and fire sensor. It has many features like On - Chip Flash Program Memory, On - Chip Data RAM, Watch dog timer etc. This is dealt in the later sections.

Driver circuit

The driver circuit is used to control the inputs to stepper motors in the Robot thereby controlling the movement of the Robot. The driver circuit is controlled by the micro controller. The driver circuit performs five operations based on the input given to it. The operations are given below

- Left
- Right
- Low speed
- Medium speed
- High speed
- Stop

Fire sensor circuit

There is a fire sensor (LDR) in the Robot section which senses any fire occurance in its path. If there is any fire, the sensor senses it & the sensor circuit triggers the relay. The relay actuates the laser beam light which points exact location of fire & also sends the sensed signal to the monitoring unit (CPU or LCD) which aids in continuous monitoring . For future enhancement, fire quenching devices can be installed in the flying unit to control the fire.

Video transmitter section

The video transmission circuit is used to transmit the video captured by the camera, in the survey field to the control section to be seen in the front end. The transmitted video is received in control section by the video receiver circuit.

These are the modules in the robot section.

2. SYSTEM REQUIREMENTS

2.1 Software requirements

2.2 Hardware requirements

2.SYSTEM REQUIREMENTS

2.1 Software requirements

The software used in the project are

1. Embedded C
2. C programming.

2.2 Hardware requirements

The hardware used in the project are

Control unit

1. PIC micro controller (16F877)
2. MAX 232
3. Video Reciever
4. LCD
5. RS 232

Sensing & Flying unit

1. Atmel (89C2051)
2. LDR
3. Camera
4. Driver Circuit
5. Video signal Transmitter

3. CONTROL MODULE

3.1 Micro controller(16F877A)

3.2.RS 232 Connector

3.3 MAX 232 Chip

3.4 Decoder (2^{12})

3.5 RF Receiver

3.6 LCD Display

3.CONTROL MODULE

3.1 Microcontroller(16F877A)

3.1.1 Concepts of microcontroller

Microcontroller is a general purpose device, which integrates a number of the components of a microprocessor system on to single chip. It has inbuilt CPU, memory and peripherals to make it as a mini computer. A microcontroller combines the CPU core, Memory(both ROM and RAM) and some parallel digital i/o on to the same microchip: Microcontrollers will combine other devices such as:

- A timer module to allow the microcontroller to perform tasks for certain time periods.
- A serial I/O port to allow data to flow between the controller and other devices such as a PIC or another microcontroller.
- An ADC to allow the microcontroller to accept analog input data for processing.

Microcontrollers are: Smaller in size, Consumes less power and Inexpensive. It is a stand alone unit, which can perform functions on its own without any requirement for additional hardware like i/o ports and external memory. The heart of the microcontroller is the CPU core. In the past, this has traditionally been based on an 8-bit microprocessor unit. For example Motorola uses a basic 6800 microprocessor core in their 6805/6808 microcontroller devices.

In the recent years, microcontrollers have been developed with specifically designed CPU cores. For example the microchip PIC range of microcontrollers have such facilities.

3.1.2 Role of PIC

The project uses five output ports of PIC for different speed and direction of the robot unit. Three port are used for speed and other two are used for direction. The PIC used here can be reprogrammed any number of times.

3.1.3 Introduction to PIC

The micro controller that has been used for this project is PIC series. PIC microcontroller is the first RISC based microcontroller fabricated in CMOS (Complimentary Metal Oxide Semiconductor) that uses separate bus for instruction and data allowing simultaneous access of program and data memory.

The main advantage of CMOS and RISC combination is low power consumption resulting in a very small chip size with a small pin count. The main advantage of CMOS is that it has immunity to noise than other fabrication techniques.

3.1.4 PIC 16F877:

Microcontrollers offers different kinds of memories of EEPROM, EPROM, and FLASH etc. are some of the memories of which FLASH is the most recently developed. Technology that is used in PIC16F877 is flash technology. The features of PIC 16F877 are data is retained even when the power is switched off, Easy Programming and Erasing.

3.1.5 PIC start plus programmer

The PIC start plus development system from microchip technology provides the product development engineer with a highly flexible low cost microcontroller design tool set for all microchip PIC micro devices. The PIC start plus development system includes PIC start plus development programmer and MPLAB.

The PIC start plus programmer gives the product developer ability to program user software in to any of the supported microcontrollers. The PIC start plus software running under MPLAB provides full interactive control over the programmer.

3.1.6 Special features of PIC microcontroller

3.1.6.1 Core features

- High-performance RISC CPU
- Only 35 single word instructions to learn
- All single cycle instructions except for program branches, which are two cycle
- Operating speed: DC - 20 MHz clock input
  DC - 200 ns instruction cycle
- Up to 8K x 14 words of Flash Program Memory,
  Up to 368 x 8 bytes of Data Memory (RAM)
  Up to 256 x 8 bytes of EEPROM data memory
- Pin out compatible to the PIC16C73/74/76/77
- Interrupt capability (up to 14 internal/external
- Eight level deep hardware stack
- Direct, indirect, and relative addressing modes

• Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)

• Watchdog Timer (WDT) with its own on-chip RC Oscillator for reliable operation

3.1.6.2 Peripheral features

• Timer0: 8-bit timer/counter with 8-bit prescaler

• Timer1: 16-bit timer/counter with prescaler, can be incremented during sleep via external crystal/clock

• Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler

• Two Capture, Compare, PWM modules

Capture is 16-bit, max resolution is 12.5 ns,

Compare is 16-bit, max resolution is 200 ns,

PWM max. resolution is 10-bit

• 10-bit multi-channel Analog-to-Digital converter

• Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9- Bit addresses detection.

• Brown-out detection circuitry for Brown-out Reset (BOR)

3.1.7 PIC 16F877 – pin configuration

The pin out diagram for 16F877 is shown

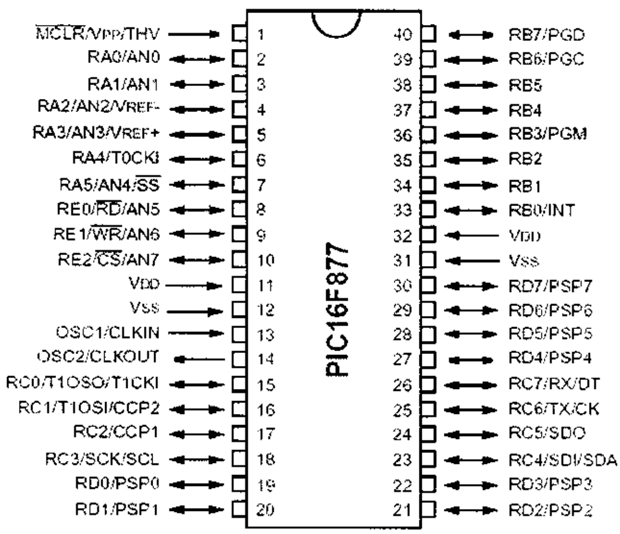


Fig 3.1 PIC 16F877 – Pin out diagram

3.1.8 I/O ports

3.1.8.1 port A and tris A register

PORTA is a 6-bit wide bi-directional port. The corresponding data direction register is TRISA. Setting a TRISA bit (=1) will make the corresponding PORTA pin an input, i.e., put the corresponding output driver in a Hi-impedance mode. Clearing a TRISA bit (=0) will make the corresponding PORTA pin an output, i.e., put the contents of the output latch on the selected pin. Reading the PORTA register reads the status of the pins whereas writing to it will write to the port latch. All write operations are read-modify-write operations. Therefore a write to a port implies that the port pins are read; this value is modified, and then written to the port data latch. PORTA pins are multiplexed with analog inputs and analog VREF input. The operation of each pin is selected by clearing/setting the control bits in the ADCON1 register (A/D Control Register1).

The TRISA register controls the direction of the RA pins, even when they are being used as analog inputs. The user must ensure the bits in the TRISA register are maintained set when using them as analog inputs.

3.1.8.2 port B AND tris B register:

PORTB is an 8-bit wide bi-directional port. The corresponding data direction register is TRISB. Setting a TRISB bit (=1) will make the corresponding PORTB pin an input, i.e., put the corresponding output driver in a hi-impedance mode. Clearing a TRISB bit (=0) will make the corresponding PORTB pin an output, i.e., put the contents of the output latch on the selected pin. Three pins of PORTB RB3/PGM, RB6/PGC and RB7/PGD are multiplexed with the Low Voltage Programming function. The alternate functions of these pins are described in the Special Features Section. Each of the PORTB pins has a weak internal pull-up. A single control bit can turn on all the pull-ups.

Four of PORT B's pins, RB7:RB4, have an interrupt on change feature. Only pins configured as inputs can cause this interrupt to occur (i.e. any RB7:RB4 pin configured as an output is excluded from the interrupt on change comparison). The input pins (of RB7:RB4) are compared with the old value latched on the last read of PORTB. The "mismatch" outputs of RB7:RB4 are OR'ed together to generate the RB Port Change

Interrupt with flag bit RBIF (INTCON<0>). This interrupt can wake the device from SLEEP.

3.1.8.3 port C and tris c register

PORTC is an 8-bit wide bi-directional port. The corresponding data direction register is TRISC. Setting a TRISC bit (=1) will make the corresponding PORTC pin an input, i.e., put the corresponding output driver in a hi-impedance mode. Clearing a TRISC bit (=0) will make the corresponding PORTC pin an output, i.e., put the contents of the output latch on the selected pin. PORTC is multiplexed with several peripheral functions. PORTC pins have Schmitt Trigger input buffers.

3.1.8.4 port D and tris D register

This section is not applicable to the 28-pin devices. PORTD is an 8-bit port with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output. PORTD can be configured as an 8-bit wide microprocessor Port (parallel slave port) by setting control bit PSPMODE (TRISE<4>). In this mode, the input buffers are TTL.

3.1.8.5 Port E and tris E register

PORTE has three pins RE0/RD/AN5, RE1/WR/AN6 and RE2/CS/AN7, which are individually configurable as inputs or outputs. These pins have Schmitt Trigger input buffers. The PORTE pins become control inputs for the microprocessor port when bit PSPMODE (TRISE<4>) is set. In this mode, the user must make sure that the TRISE<2:0> bits are set (pins are configured as digital inputs). Ensure ADCON1 is configured for digital I/O. In this mode the input buffers are TTL.

3.2.RS 232 Connector

3.2.1 Overview

Modems and other devices used to send serial data are often referred to as data communication equipment or DCE. The terminals or computers that are sending or receiving the data are referred to as data terminal equipment or DTE. In response to the need for signal and handshake standards between DTE and DCE, the Electronic Industries Association (EIA) developed EIA standard RS-232C. This standard describes the functions of 25 signal and handshake pins for serial data transfer. Before we work our way through the 25 pin functions, we will take a brief look at some of the other hardware aspects of RS-232C.

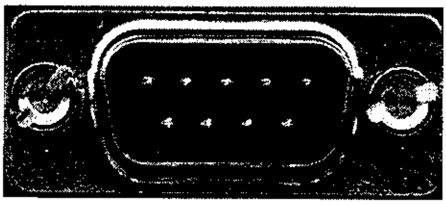
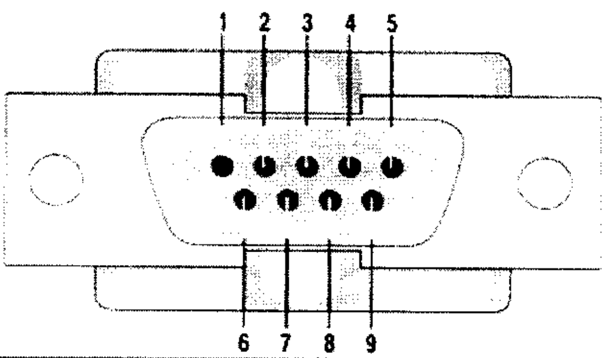


Fig 3.2 RS 232 pin Configuration

RS-232C specifies 25 signal pins and it specifies that the DTE connector should be a male, and the DCE connector should be a female. A specific connector is not given, but the most commonly used connectors are the DB-25P male and the DB-25S female.



Pin	Signal	Pin	Signal
1	Data Carrier Detect	6	Data Set Ready
2	Received Data	7	Request to Send
3	Transmitted Data	8	Clear to Send
4	Data Terminal Ready	9	Ring Indicator
5	Signal Ground		

Fig 3.3 Pin diagram of RS 232

When wiring these connectors, it is important to note the order in which the pins are numbered.

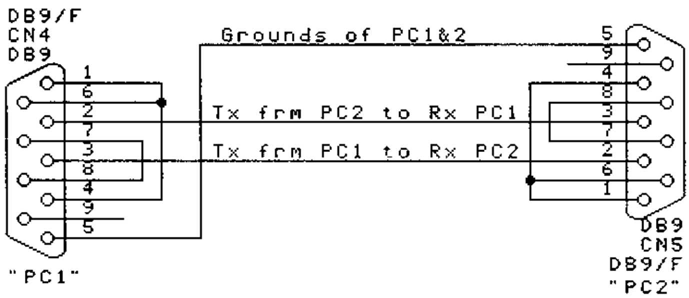


Fig3.4Connecting from PC to micro controller using RS232, without handshaking

Pin functions

Full Name	Function	
TD	Transmit Data	Serial Data Output (TXD)
RD	Receive Data	Serial Data Input (RXD)
CTS	Clear to Send	This line indicates that the Modem is ready to exchange data.
DCD	Data Carrier Detect	When the modem detects a "Carrier" from the modem at the other end of the phone line, this Line becomes active.
DSR	Data Set Ready	This tells the UART that the modem is ready to establish a link.
DTR	Data Terminal Ready	This is the opposite to DSR. This tells the Modem that the UART is ready to link.
RTS	Request To Send	This line informs the Modem that the UART is ready to exchange data.
RI	Ring Indicator	Goes active when modem detects a ringing signal from the PSTN.

Table 3.1 Pin functions of RS 232

3.3 MAX 232 Chip:

3.3.1 Description

The MAX232 is a dual driver/receiver that includes a capacitive voltage generator to supply EIA-232 voltage levels from a single 5-V supply. Each receiver converts EIA-232 inputs to 5-V TTL/CMOS levels. These receivers have a typical threshold of 1.3 V and a typical hysterics of 0.5 V, and can accept 30-V inputs. Each driver converts TTL/CMOS input levels into EIA-232 levels. The driver, receiver, and voltage-generator functions are available.

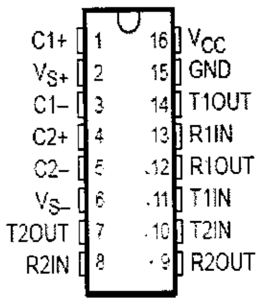


Fig3.5 Pin configuration

3.4 Decoder(2^{12})

The decoder IC that is made use of is the HT12D IC from HOLTEK. The main purpose of its installation in the reception circuitry is for the conversion of serial input to parallel form and to ensure error free reception by comparing the redundancy of data generated by the encoder.

3.4.1 Features

- Operating voltage: 2.4V~12V
- Low power and high noise immunity CMOS technology
- Low standby current
- Capable of decoding 12 bits of information
- 8 address bits and 4 data bits
- Built-in oscillator needs only 5% resistor
- Valid transmission indicator

3.4.2 Applications

- Burglar alarm system
- Smoke and fire alarm system
- Garage door controllers
- Car door controllers
- Car alarm system

3.4.3 General description

The 2^{12} decoders are a series of CMOS LSIs for remote control system applications. They are paired with Holtek's 2^{12} series of encoders.The decoders receive serial addresses and data from a programmed 2^{12} series of encoders that are transmitted by a carrier using an RF or an IR transmission medium.

They compare the serial input data three times continuously with their local addresses. If no error or unmatched codes are found, the input data codes are decoded and then transferred to the output pins. The VT pin also goes high to indicate a valid transmission.

The 2^{12} series of decoders are capable of decoding informations that consist of N bits of address and 12_N bits of data. Of this series, the HT12D is arranged to provide 8 address bits and 4 data bits. The fig below shows the basic pin out of the HT12D decoder

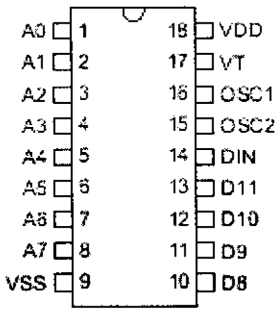


Fig 3.6.HT-12D pinout

3.4.4 Functional description

Operation

The 2^{12} series of decoders provides various combinations of addresses and data pins in different packages so as to pair with the 2^{12} series of encoders. The decoders receive data that are transmitted by an encoder and interpret the first N bits of code period as addresses and the last 12-N bits as data, where N is the address code number. A signal on the DIN pin activates the oscillator which in turn decodes the incoming address and data. The decoders will then check the received address three times continuously.

If the received address codes all match the contents of the decoder's local address, the 12_N bits of data are decoded to activate the output pins and the VT pin is set high to indicate a valid transmission. This will last unless the address code is incorrect or no signal is received. The output of the VT pin is high only when the transmission is valid. Otherwise it is always low.

3.5 RF Receiver

RLP 434A is used as a RF receiver. The RLP-434A are excellent for applications requiring short-range RF remote controls.

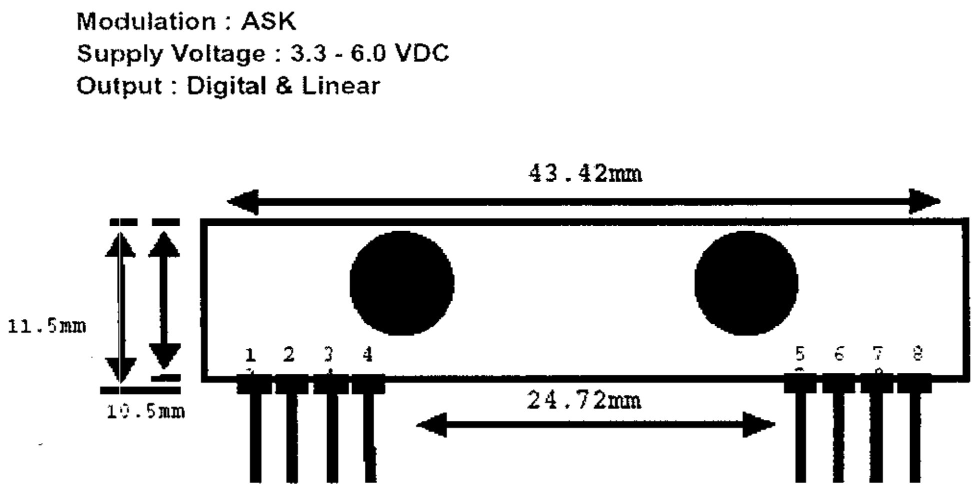


Fig 3.7 RF Receiver

3.5.1 General description

The following table depicts the technical specifications of the receiver module.

Symbol	Parameter	Conditions	Min	Typ	Max	
Vcc	Operating supply voltage		3.3	5.0V	6.0	V
Itot	Operating Current		-	4.5		mA
Vdata	Data Out	Idata = +200 uA (High)	Vcc-0.5	-	Vcc	V
		Idata = -10 uA (Low)	-	-	0.3	V

Table 3.2 General description of RF receiver

3.5.2 Electrical characteristics

Characteristics	SYM	Min	Typ	Max	Unit
Operation Radio Frequency	FC	315, 418 and 433.92			MHz
Sensitivity	Pref		-110		dBm
Channel Width			+-500		Khz
Noise Equivalent BW			4		Khz
Receiver Turn On Time			5		ms
Operation Temperature	Top	-20	-	80	C
Baseboard Data Rate			4.8		KHz

Table 3.3 RF Receiver characteristics

With reference to the above table, the receiver module could be effectively used by maintaining the allowable range of signal specifications and also by maintaining the optimum operating conditions specified for the efficient working. Also the sensitivity of the receiver in dB (decibels) is found to be quite appropriate for efficiency in reconstructing a signal from the transmitter at a vicinity of the receiver.

3.6 LCD Display

3.6.1 Interfacing with micro controller

The module, interfaced to the system, can be treated as RAM input/output, expanded or parallel I/O.

The 'E' signal must be a positive going digital strobe, which is active while data and control information are stable and true. The falling edge of the enable signal enables the data / instruction register of the controller. All module timings are referenced to specific edges of the 'E' signal. The 'E' signal is applied only when a specific module transaction is desired.

The read and write strobes of the host, which provides the 'E' signals, should not be linked to the module's R/W line. An address bit which sets up earlier in the host's machine cycle can be used as R/W.

When the host processor is so fast that the strobes are too narrow to serve as the 'E' pulse, the following action can be done Prolong these pulses by using the hosts 'Ready' input , Prolong the host by adding wait states , Decrease the Hosts Crystal frequency.

In spite of doing the above mentioned, if the problem continues, latch both the data and control information and then activate the 'E' signal

When the controller is performing an internal operation he busy flag (BF) will set and will not accept any instruction. The user should check the busy flag or should provide a delay of approximately 2ms after each instruction.The module presents no difficulties while interfacing slower MPUs.The liquid crystal display module can be interfaced, either to 4-bit or 8-bit MPUs. For 4-bit data interface, the bus lines DB4 to DB7 are used for data transfer, while DB0 to DB3 lines are disabled. The data transfer is complete when the 4-bit data has been transferred twice.The busy flag must be checked after the 4-bit data has been transferred twice. Two more 4-bit operations then transfer the busy flag and address counter data. For 8-bit data interface, all eight-bus lines (DB0 to DB7) are used.

4. ROBOT MODULE

4. ROBOT MODULE

4.1 AT89C2051

4.1.1 Features

- Compatible with MCS-51™ Products
- 2K Bytes of Reprogrammable Flash Memory
- 2.7V to 6V Operating Range
- Fully Static Operation: 0 Hz to 24 MHz
- Two-level Program Memory Lock
- 128 x 8-bit Internal RAM
- 15 Programmable I/O Lines
- Two 16-bit Timer/Counters
- Six Interrupt Sources

4.1.2 Description

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

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

The power-down mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next hardware reset.

4.1.3 Pin configuration

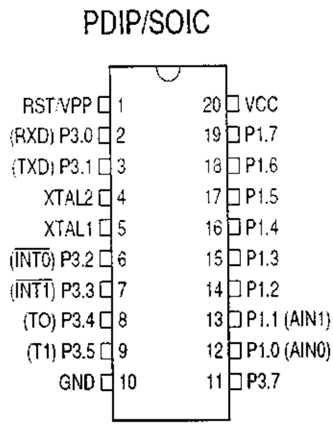


Fig 4.1. Pin configuration AT89C51

4.1.4 Programming the flash

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

4.1.5 PROGRAMMING THE AT89C2051

To program the AT89C2051, the following sequence is recommended.

1. Power-up sequence:

   Apply power between VCC and GND pins , Set RST and XTAL1 to GND

2. Set pin RST to "H" , Set pin P3.2 to "H"

3. Apply the appropriate combination of "H" or "L" logic levels to pins P3.3, P3.4, P3.5, P3.7 to select one of the programming operations shown in the PEROM Programming Modes table.

To Program and Verify the Array:

4. Apply data for Code byte at location 000H to P1.0 to P1.7.

5. Raise RST to 12V to enable programming.

6. Pulse P3.2 once to program a byte in the PEROM array or the lock bits. The byte-write cycle is self-timed and typically takes 1.2 ms.

7. To verify the programmed data, lower RST from 12V to logic "H" level and set pins P3.3 to P3.7 to the appropriate levels. Output data can be read at the port P1 pins.

8. To program a byte at the next address location, pulse XTAL1 pin once to advance the internal address counter. Apply new data to the port P1 pins.

9. Repeat steps 5 through 8, changing data and advancing the address counter for the entire 2K bytes array or until the end of the object file is reached.

10.Power-off sequence:

set XTAL1 to "L"

set RST to "L"

Turn VCC power off

4.1.6 FLASH PROGRAMMING MODES

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

Table 4.1 Flash programming modes of AT89C51

1. The internal PEROM address counter is reset to 000H on the rising edge of RST and is advanced by a positive pulse at XTAL 1 pin.

2. Chip Erase requires a 10 ms PROG pulse.

3. P3.1 is pulled Low during programming to indicate RDY/BSY.

4.2 Encoder (2^{12} Series)

The encoder IC that is made use of is the HT12E IC from HOLTEK. The main Purpose of its installation in the transmission circuitry is for the serialization of data from Pc and to ensure error free transmission by enabling redundancy of output data.

4.2.1 Features

- Operating voltage 2.4V~12V
- Low power and high noise immunity CMOS technology
- Low standby current: 0.1_A (typ.) at VDD=5V
- Minimum transmission word
- Four words for the HT12E
- Minimal external components

4.2.2 Applications

- Burglar alarm system
- Smoke and fire alarm system
- Garage door controllers
- Car door controllers
- Car alarm system

4.2.3 GENRAL DESCRIPTION

The 2^{12} encoders are a series of CMOS LSIs for remote control system applications. They are capable of encoding information which consists of N address bits and 12-N data bits. Each address/data input can be set to one of the two logic states

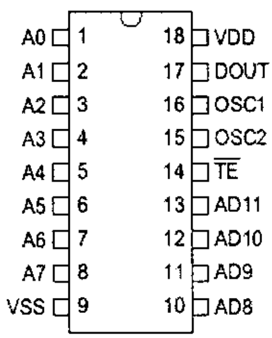


Fig 4.2. Pin configuration of HT12E

The programmed addresses/data are transmitted together with the header bits via an RF or an infrared transmission medium upon receipt of a trigger signal. The capability to select a TE trigger on the HT12E enhances the application flexibility of the 2^{12} series of encoders.

.4 Functional description

Operation

The 2^{12} series of encoders begin a 4-word transmission cycle upon receipt of a transmission enable (TE). This cycle will repeat itself as long as the transmission enable (TE) is held low. Once the transmission enable returns high the encoder output completes its final cycle and then stops as show below

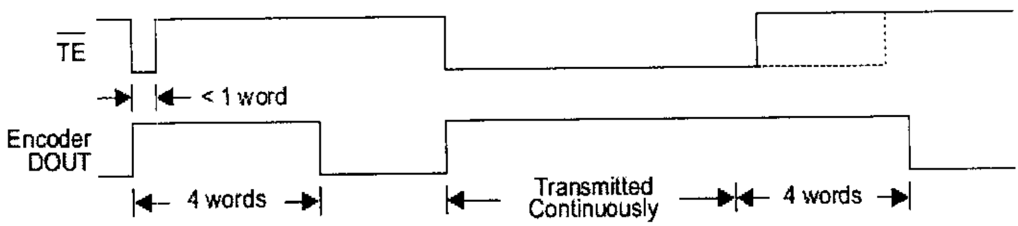


Fig. 4.3 Transmission timing for HT12E

4.3 RF Transmitter

The TWS-434 and RWS-434 are extremely small, and are excellent for applications requiring short-range RF remote controls. The transmitter module is only 1/3 the size of a standard postage stamp, and can easily be placed inside a small plastic enclosure.

4.3.1 Features

Fig 4.4 TWS-434A

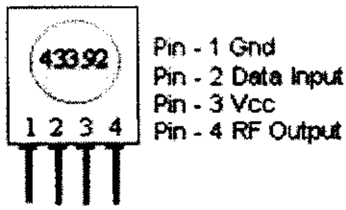


Fig 4.5 TWS-434 Pin Diagram

TWS-434: The transmitter output is up to 8mW at 433.92MHz with a range of approximately 400 foot (open area) outdoors.

4.3.2 Sample transmitter Application circuit

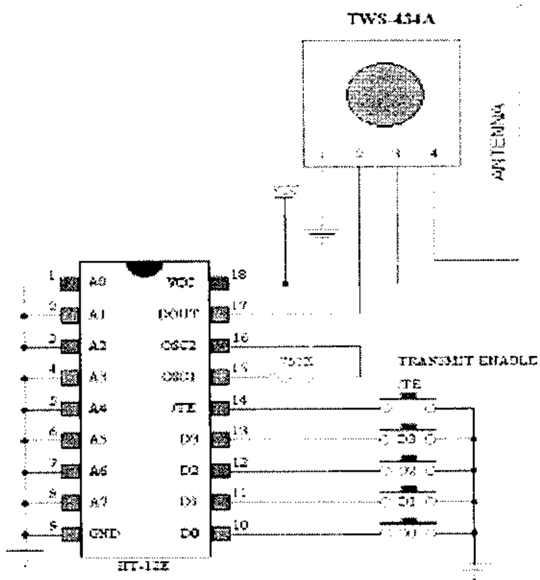


Fig 4.6 Sample transmitter Application circuit

4.3.3 General characteristics

The following table depicts the technical specification of the transmitter module.

Modulation : ASK

Operating voltage: 2 – 12 VDC

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
Vcc	Operating supply voltage		2.0	-	12.0	V
Icc 1	Peak Current (2V)		-	-	1.64	mA
Icc 2	Peak Current (12V)		-	-	19.4	mA
Vh	Input High Voltage	Idata= 100uA (High)	Vcc-0.5	Vcc	Vcc+0.5	V
Vl	Input Low Voltage	Idata= 0 uA (Low)	-	-	0.3	V
FO	Absolute Frequency	315Mhz module	314.8	315	315.2	MHz
PO	RF Output Power- 50ohm	Vcc = 9V-12V	-	16	-	dBm
		Vcc = 5V-6V	-	14	-	dBm
DR	Data Rate	External Encoding	512	4.8K	200K	bps

Table 4.2 RF Transmitter General Characteristics

With reference to the above table, the transmitter module could be effectively used by maintaining the allowable range of signal specifications and also by maintaining the optimum operating conditions specified for the efficient working. Also the output power level in dB (decibels) is found to be quite appropriate for efficiency in signal detection by the use of a tuned receiver in the vicinity of about 50m from the transmitter.

4.4 FIRE SENSOR CIRCUIT

When there is a flame in the survey field the flying robot senses it through the flame sensor and the output of differentiator (LM 741) becomes high. Now the transistor starts conducting and so the input to the NOT gate goes low. The output from NOT gate is then given to the microcontroller (ATMEL 89C51).Then the micro controller generates the corresponding control codes to the flying robot unit for any control action. The fire sensing circuit employed in the project is shown below.

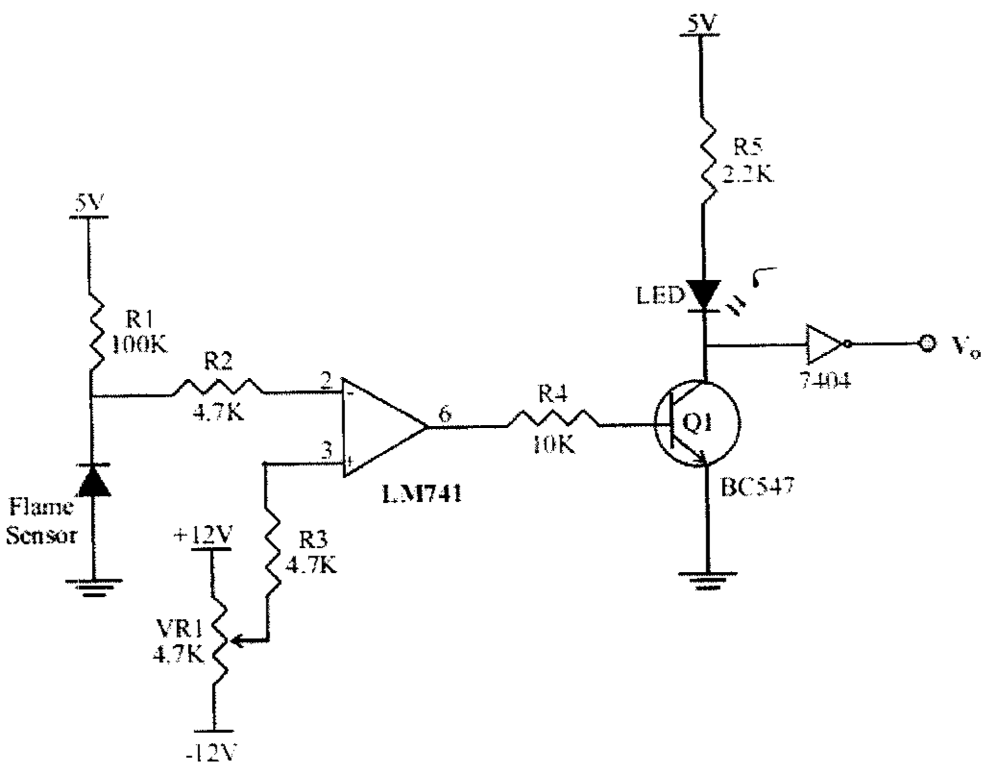


Fig 4.7 Fire sensing circuit

5. SOFTWARE MODULE

5. SOFTWARE MODULE

1.The following is a C program used to send a data from PC to Control unit.

```
#include<stdio.h>
#include<dos.h>
#include<conio.h>
#include<string.h>
#define PORT1 0x3F8
void control(unsigned char);
void main()
{       int k,c;
        char ch,cntrl,d[1];
        char *p;
        clrscr();
        outportb(PORT1 + 1 , 0);      //turn off interrput - port1
        /* PORT1 communication section*/
        outportb(PORT1 + 3, 0x80);  //set DLAB on*/
        outportb(PORT1 + 0 , 0x01); //set baud rate 115200
        outportb(PORT1 + 1 , 0x00); //set baud rate- divisor latch high byte
        outportb(PORT1 + 3 , 0x03); //8 bit, no parity, 1 stop bit
        outportb(PORT1 + 2 , 0xc7); //FIFO control register*/
        outportb(PORT1 + 4 , 0x0B);//Turn on DTR,TRS, and out2);
        while(1)
        {       clrscr();
                printf("\tpress 11 for speed 1\n");
                printf("\tpress 22 for speed 2\n");
                printf("\tpress 33 for speed 3\n");
                printf("\tpress 44 for right\n");
                printf("\tpress 55 for left\n");
                printf("\tpress 66 for stop\n");
                printf("\tpress 77 for exit\n");
```

```
scanf("%c",&cntrl);
switch (cntrl)
{                   case '11': printf("flying in speed 1");
                     d[0]='m';  //sending mm for speed1
                     d[1]='m';
                     control(d[0]);
                     control(d[1]);
                     delay(2000);
                     break;
          case '22': printf("flying in speed 2");
                     d[0]='n';  //sending nn for speed2
                     d[1]='n';
                     control(d[0]);
                     control(d[1]);
                     delay(2000);
                     break;
          case '33': printf("flying in speed 3");
                     d[0]='o'; //sending oo for speed3
                     d[1]='o';
                     control(d[0]);
                     control(d[1]);
                     delay(2000);
                     break;
          case '44': printf("flying right");
                     d[0]='p';//sending pp for right
                     d[1]='p';
                     control(d[0]);
                     control(d[1]);
                     delay(2000);
                     break;
```

```
case '5': printf("flying left");
        d[0]='q'; //sending qq for left
        d[1]='q';
        control(d[0]);
        control(d[1]);
        delay(2000);
        break;
case '6': printf("stop flying");
        d[0]='t';  //sending tt for stop flying
        d[1]='t';
        control(d[0]);
        control(d[1]);
        delay(2000);
        break;
case '7': exit(0);
default:  printf("invalid entry");
        continue;
}}}
void control(unsigned char d)//function for sending data
{       outportb(PORT1,d);
}
```

PIC CODING

The following c programming is used for PIC coding

```
#include <16F877a.h>
#include <string.h>
#include <stdlib.h>
#fuses  HS,NOWDT,NOPROTECT,NOBROWNOUT,PUT,NOLVP
#use delay (clock=20000000)
#use rs232(baud=115200, xmit=PIN_C6, rcv=PIN_C7)
//lcd register select,Enable
#define RS  PIN_E0
```

```
#define E  PIN_E1
#define check  PIN_D1
#define RFS PIN_C7
#define serial_error()  OERR                          // USART error
#define serial_fix()     {CREN=0;CREN=1;}
#define VTXN PIN_A0
#define RIGHT PIN_D2      //pins for controlling device
#define LEFT PIN_D0
#define SPEED1 PIN_C5
#define SPEED2 PIN_D4
#define SPEED3 PIN_D3
#byte PORTC=0x07
#byte TMR2=0X11
#bit TMR2ON = 0x12.2 //' TIMER1 Enable/Disable
#byte RCREG=0x1a
#bit CREN=0x18.4
#bit OERR=0x18.1
#bit RCIF=0x0c.5
#bit RCIE=0x8c.5
#bit GIE=0x0b.7
extern byte s[75];
byte s3[3],i,l,s2,o;
unsigned char dat[8];
byte mem_loc=0,status1[18];
byte lng,count,reg;
void Bin (byte hex, byte bin[], byte bit) //changes bytes to bits
{  byte i;
 for (i=0; i < bit; i++)
 {   if (hex%2)
  bin[i] = 1;
  else bin[i] = 0;
```

```
hex = hex/2;
} }
byte Debin (byte bin[], byte bit)     //changes first 8 bits to byte
{ byte i, d = 1, hex = 0;
for (i=0; i < bit; i++)
{  hex = hex + d * bin[i];
d = d * 2;
} return (hex);
} byte EToSe (byte inp[])  //changes eight bit words to seven bit words
{ byte i, j = 0, sev[7], eight[8], shift = 0, x = 0, end = 0;
byte buffer[15] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
do
{   do
  {      if (inp [j + x] == 0xff)
    {         end = 1;
      inp [j + x] = 0x00;
    }      Bin (inp [j + x], eight, 8);
    for (i=0; i < 8; i++) buffer[i + shift] = eight[i];
    for (i=0; i < 7; i++) sev[i] = buffer[i];
    shift++;
    for (i=0; i < shift; i++) buffer[i] = buffer[i + 7];
    inp[j + x] = Debin(sev, 7);
    j++;
     if (end) break;
  }    while (j < 7);
  j = 0;
  x = x + 7;
} while (!end);
return (x);
}
void HEXtoGSM(unsigned char *dat)
```

```
{ int i,j;
unsigned char d[10],temp=0;
for(i=0;i<strlen(dat);i++)
{   unsigned char o=0;
 j=i+1;
 while(j)
 {      o=(dat[i+1] & 0x01);
        temp=(o<<(8-j));
        dat[i]|=temp;
        dat[i+1]>>=1;
        j--;
        temp=0;
 } } }
void  cmd()
 {   //command for lcd display
  output_low(RS);
  output_high(E);
  output_low(E);
  delay_ms(2);
 } void  dwr()
  {   //character to LCD display
  output_high(RS);
  output_high(E);
  output_low(E);
  delay_ms(2);
  } void  LCDInit()
 {   output_b(0x38);
   cmd();
   output_b(0x10);
   cmd();
   output_b(0x0f);
```

```
 cmd();
 output_b(0x06);
 cmd();
 output_b(0x01);
 cmd();
} void  clear()
{   output_b(0x01);
 cmd();
 output_b(0x80);
 cmd();
}  LCDPutchar( int c)
{     delay_ms(25);
   output_b((unsigned char)c);
  dwr();
  delay_ms(25);
} void Crc (int buffer[])  // calculates checksum
{  byte pad = 0, len = 0, i, chk = 0;
 len = buffer[5];
 if (len%2 != 0)
 pad = 1;
 chk = 0;
 for (i = 0; i < len + pad + 5; i+=2)
 chk ^= buffer[i];
 buffer[len + pad + 6] = chk;
 chk = 0;
 for (i = 1; i < len + pad + 6; i+=2)
 chk ^= buffer[i];
 buffer[len + pad + 7] = chk;
} /*Inserts data in 8 bytes*/
void ins_8(char *chr)
{  HEXtoGSM(chr);
```

```
for(i=0;i<7;i++)
{       s[lng]=chr[i];
        lng++;
} count++;
} void stop()  // function for stopping robot
{   printf(LCDPutchar,"STOP ROBOT");
  TMR2ON=0; //Disable TIMER2
output_high(SPEED1);
output_high(SPEED2);
output_high(SPEED3);
output_high(LEFT);
output_high(RIGHT);
  TMR2ON=0; //Disable TIMER2
  sprintf(dat," Stop  ");
  HEXtoGSM(dat);
  sms();
  delay_ms(10);
 }
 void main()
 {
  byte temp[3]; // local variable
  unsigned int rcvd=0;
  o=0;
  set_tris_a(0x0f);
  set_tris_b(0x00);
  set_tris_c(0x0f);
  set_tris_e(0x00);
  set_tris_d(0x00);
  LCDInit();
  output_c(0x00);
  output_low(VTXN);
```

```
output_high(check);
output_high(SPEED1);
output_high(SPEED2);
output_high(SPEED3);
output_high(LEFT);
output_high(RIGHT);
delay_ms(10);
//def();     //start flying forward at speed 1
while(1)
{     clear();
  delay_ms(10);
  printf(LCDPutchar,"  INTELLIGENT  ");
  delay_ms(10);
  clear();
  printf(LCDPutchar,"ROBOTIC VEHICLE");
  if(input(RFS))
    {  clear();
      printf(LCDPutchar,"in rfs");
      delay_ms(100);
      for (i=0; i < 34; i++)
        putchar(0x55);
      status1[0]=0x1e;
      status1[1]=0x00;
      status1[2]=0x0c;
      status1[3]=0xD1;
      status1[4]=0x00;
      status1[5]=0x07;
      status1[6]=0x00;
      status1[7]=0x01;
      status1[8]=0x00;
      status1[9]=0x03;
```

```
status1[10]=0x00;
status1[11]=0x01;
status1[12]=0x60;
status1[13]=0x00;
status1[14]=0x72;
status1[15]=0xd5;
for (i=0; i < 16; i++)
 putchar(status1[i]);
delay_ms (1000);
GIE=1;          // interrupt -- receiving sms
RCIE=1;
CREN=1;
RCIF=0;
while(1)
{          //checking any valid reception has happened
  if(input(VTXN))
  {              rcvd=0;
    clear();
    printf(LCDPutchar,"  DATA RECIEVED  ");
    delay_ms(100);
    reg=input_d();
    reg=reg&0x0f;
    clear();
    printf(LCDPutchar,"%x",reg);
    delay_ms(100);
    output_low(VTXN);
    if(reg==0x0A)
    {  clear();
      printf(LCDPutchar,"  FIRE  ");
      delay_ms(10000);
     /* clear();
```

```
    printf(LCDPutchar,"  LIGHT SWITCHED ON  ");
    delay_ms(1000);*/
    reg=0x00;
    output_low(VTXN);
  }   continue;          }
//collecting data from serial port
 while(!RCIF)
{       if(input(VTXN))
  {    rcvd=1;
    break;
  }       }
 if(serial_error())
     serial_fix();
 i=0;
 s[i]=RCREG;
 RCIF=0;
 if(rcvd==1)
   continue;
 i++ ;
 while(!RCIF)
 {              if(input(VTXN))
   {    rcvd=1;
     break;
   }      }
 s[i]=RCREG;
 RCIF=0;
 if(rcvd==1)
   continue;
 strcpy(s3,"mm");
 if((strncmp(s,s3,2)==0))              //checking whether data is from pc
  {
```

```
  clear();
  printf(LCDPutchar,"speed 1");
  delay_ms(5);
  output_low(SPEED1);
  output_high(SPEED2);
  output_high(SPEED3);
  output_high(LEFT);
      output_high(RIGHT);
  delay_ms(10);
  continue;
}    strcpy(s3,"nn");            //checking whether the data is from pc
if((strncmp(s,s3,2)==0))
{  clear();
  printf(LCDPutchar,"speed 2");
  delay_ms(5);
  output_high(SPEED1);
  output_low(SPEED2);
   output_high(SPEED3);
  output_high(LEFT);
  output_high(RIGHT);
  delay_ms(10);
  continue;
}   strcpy(s3,"oo");            //checking whether the data is from pc
if((strncmp(s,s3,2)==0))
{
  clear();
  printf(LCDPutchar,"speed 3");
  delay_ms(5);
  output_high(SPEED1);
  output_high(SPEED2);
  output_low(SPEED3);
```

```
  output_high(LEFT);
  output_high(RIGHT);
  delay_ms(10);
  continue;
}   strcpy(s3,"pp");
if((strncmp(s,s3,2)==0))     //checking whether the data is from pc
{  clear();
  printf(LCDPutchar,"right");
  delay_ms(5);
  output_high(SPEED1);
  output_high(SPEED2);
  output_low(SPEED3);
  output_high(LEFT);
  output_low(RIGHT);
  delay_ms(10);
  continue;
} strcpy(s3,"qq");
if((strncmp(s,s3,2)==0))     //checking whether the data is from pc
{   clear();
  printf(LCDPutchar,"left");
  delay_ms(5);
  output_high(SPEED1);
  output_high(SPEED2);
  output_low(SPEED3);
  output_low(LEFT);
  output_high(RIGHT);
  delay_ms(10);
  continue;
}    strcpy(s3,"tt");          //checking whether the data is from pc
if((strncmp(s,s3,2)==0))
{   clear();
```

```
  stop();
  delay_ms(10);
  continue;
} if(serial_error())
serial_fix();
if(s[0]==0x1e)
{ i++;  }
else
  continue;
do
{  s[i]=getc();
  if(serial_error())
    serial_fix();
  i++;
}while(i<4);
if(s[3]==0x02)
{   do
  {  s[i]=getc();
    if(serial_error())
      serial_fix();
    i++;
  }while(i<10);
}  if(s[9]==0x10)   //getting the sms from the serial port
 {  do    {
     s[i]=getc();
     if(serial_error())
     serial_fix();
     i++;
     //*** m=s[11];****
   }while(i<s[5]+8);
   break;
```

```
}     else
  continue;
} write_eeprom(0x40,s[32]);
write_eeprom(0x41,s[33]);
write_eeprom(0x42,s[34]);
write_eeprom(0x43,s[35]);
write_eeprom(0x44,s[36]);
l=0;
clear();
printf(LCDPutchar,"inner 3");
for (i=48; i<s[28]+48; i++)  //decoding  text portion
{   s[l] = s[i];
  l++;    }
s[l+1] =0xff;
EToSe(s);
strcpy(s3,"s1");
if((strncmp(s,s3,2)==0)&&(s[28]==0x02))            {
  clear();
  printf(LCDPutchar,"speed 1");
  output_low(SPEED1);
  output_high(SPEED2);
  output_high(SPEED3);
  output_high(LEFT);
  output_high(RIGHT);
  sprintf(dat,"speed1");
  delay_ms(10);
  HEXtoGSM(dat);
  sms();
  continue;
 }
 else
```

```
  strcpy(s3,"s2");
if((strncmp(s,s3,2)==0)&&(s[28]==0x02))
{   clear();
  printf(LCDPutchar,"speed 2");
  delay_ms(200);
  output_high(SPEED1);
  output_low(SPEED2);
  output_high(SPEED3);
  output_high(LEFT);
  output_high(RIGHT);
   sprintf(dat,"speed2");
  delay_ms(10);
  HEXtoGSM(dat);
  sms();
  S[0];
  continue;
 }  else
   strcpy(s3,"s3");
 if((strncmp(s,s3,2)==0)&&(s[28]==0x02))          {
   clear();
   printf(LCDPutchar,"speed 3");
   delay_ms(200);
   output_high(SPEED1);
   output_high(SPEED2);
   output_low(SPEED3);
   output_high(LEFT);
   output_high(RIGHT);
   sprintf(dat,"speed3");
   delay_ms(10);
         HEXtoGSM(dat);
       sms();
```

```
continue;
}
else
  strcpy(s3,"r");
if((strncmp(s,s3,1)==0)&&(s[28]==0x01))  //checking the data is for turning
right
{
  printf(LCDPutchar,"RIGHT ROBOT");
  output_high(SPEED1);
  output_high(SPEED2);
  output_low(SPEED3);
  output_high(LEFT);
  output_low(RIGHT);
  sprintf(dat," RIGHT ");
  delay_ms(10);
  HEXtoGSM(dat);
  sms();
  continue;
}   else
  strcpy(s3,"l");
if((strncmp(s,s3,1)==0)&&(s[28]==0x01))
{   printf(LCDPutchar,"LEFT ROBOT");
  output_high(SPEED1);
  output_high(SPEED2);
  output_low(SPEED3);
  output_low(LEFT);
  output_high(RIGHT);
  sprintf(dat," LEFT ");
  delay_ms(10);
  HEXtoGSM(dat);
  sms();
```

```
        continue;
      }   else
        strcpy(s3,"s");
      if((strncmp(s,s3,1)==0)&&(s[28]==0x01))
        {     stop();
        continue;
      }     }
    else
    {   clear();
      printf(LCDPutchar,"rfs is present");
      delay_ms(1000);
      if(!input(check))
      {  clear();
         delay_ms(10400);
      }  if(o==25)
         o=0;
      else
         o++;
    }    continue;
} }
```

6. CONCLUSION

CONCLUSION

The designed fire sensing machine in the form of flying robot can be controlled effectively up to distance of 100m using a PC. It can also fly in predefined paths with out humans intervention for long period of time using wireless RF transmission

The images captured by the camera in the robot can also be used for remote sensing purposes. The main advantage of the project is the sensing and monitoring exact location of the occurrence of a fire accident also the time taken to reach the destination in less

The project is a prototype model used for monitoring the fire in the areas such as forest, residential buildings, corporate and so on…

This project can be used in various application with some modifications.

- This robot is just a prototype and for the real time application the laser beam in the flying kit can be replaced by any of fire quenching kits.
- The project can be extended to control the flying kit with the help of mobile services like GPRS, GSM …….
- The limitation of the camera used in this project is , it can capture the image only in open space not in a closed room. To enhance the application the camera can be replaced with thermal sensing camera which give better performance.

APPENDICES

Appendix A - Encoder(2^{12} Series)

Appendix B - Decoder(2^{12} Series)

Appendix C -Photos

7.APPENDIX A

NCODER(2[12] SERIES)

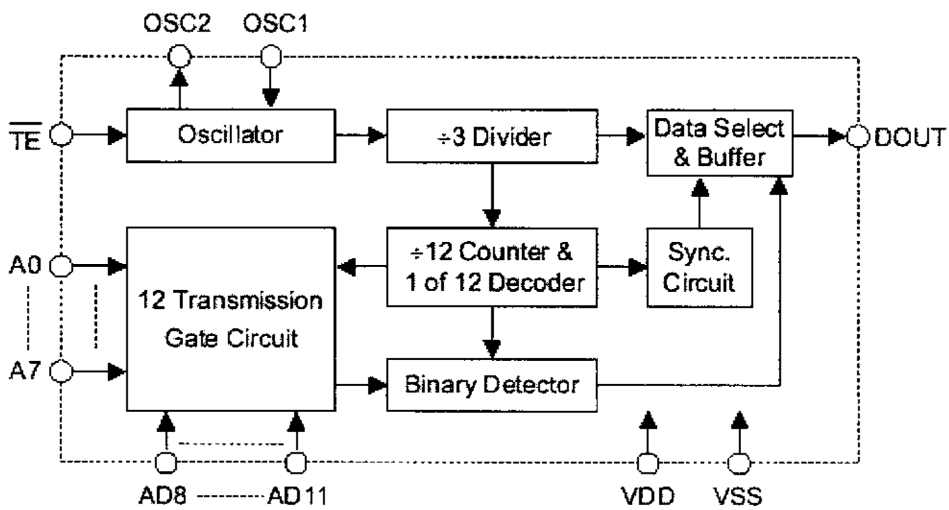
ELCTION TABLE

Function / Part No.	Address No.	Address/ Data No.	Data No.	Oscillator	Trigger	Package	Carrier Output	Negative Polarity
HT12E	8	4	0	RC oscillator	$\overline{TE}$	18 DIP 20 SOP	No	No

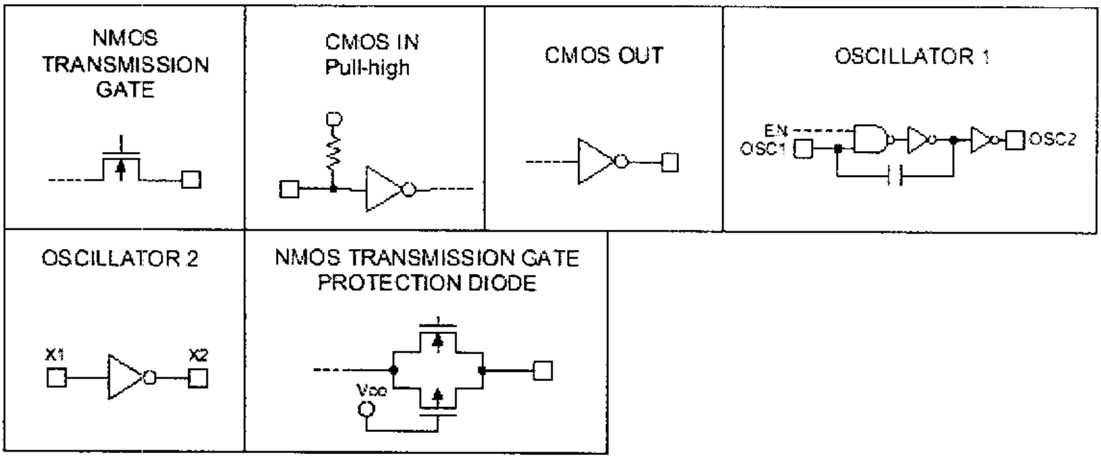
LOCK DIAGRAM

$\overline{TE}$ trigger

HT12E



PPROXIMATE INTERNAL CONNECTIONS



PIN DESCRIPTION

Pin Name	I/O	Internal Connection	Description
A0~A7	I	CMOS IN Pull-high (HT12A) NMOS TRANSMISSION GATE PROTECTION DIODE (HT12E)	Input pins for address A0~A7 setting These pins can be externally set to VSS or left open
AD8~AD11	I	NMOS TRANSMISSION GATE PROTECTION DIODE (HT12E)	Input pins for address/data AD8~AD11 setting These pins can be externally set to VSS or left open
D8~D11	I	CMOS IN Pull-high	Input pins for data D8~D11 setting and transmission enable, active low These pins should be externally set to VSS or left open (see Note)
DOUT	O	CMOS OUT	Encoder data serial transmission output
L/MB	I	CMOS IN Pull-high	Latch/Momentary transmission format selection pin: Latch: Floating or VDD Momentary: VSS

LECTRICAL CHARACTERISTICS

T12E

Operating temp at 25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.4	5	12	V
I_{STB}	Standby Current	3V	Oscillator stops	—	0.1	1	μA
		12V		—	2	4	μA
I_{DD}	Operating Current	3V	No load f_{OSC}=3kHz	—	40	80	μA
		12V		—	150	300	μA
I_{DOUT}	Output Drive Current	5V	V_{OH}=0.9V_{DD} (Source)	-1	-1.6	—	mA
			V_{OL}=0.1V_{DD} (Sink)	1	1.6	—	mA
V_{IH}	"H" Input Voltage	—	—	0.8V_{DD}	—	V_{DD}	V
V_{IL}	"L" Input Voltage	—	—	0	—	0.2V_{DD}	V
f_{OSC}	Oscillator Frequency	5V	R_{OSC}=1.1MΩ	—	3	—	kHz
$R_{\overline{TE}}$	$\overline{TE}$ Pull-high Resistance	5V	$V_{\overline{TE}}$=0V	—	1.5	3	MΩ

DDRESS/DATA PROGRAMMING(PRESET)

The status of each address/data pin can be individually pre-set to logic "high" or "low". If a transmission enable signal is applied, the encoder scans and transmits the status of the 12 bits of address/data serially in the order A0 to AD11 for the HT12E encoder. During information transmission these bits are transmitted with a preceding synchronization bit. If the trigger signal is not applied, the chip enters the standby mode and consumes a reduced current of less than 1A for a supply voltage of 5V.

The transmitted information is as shown

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

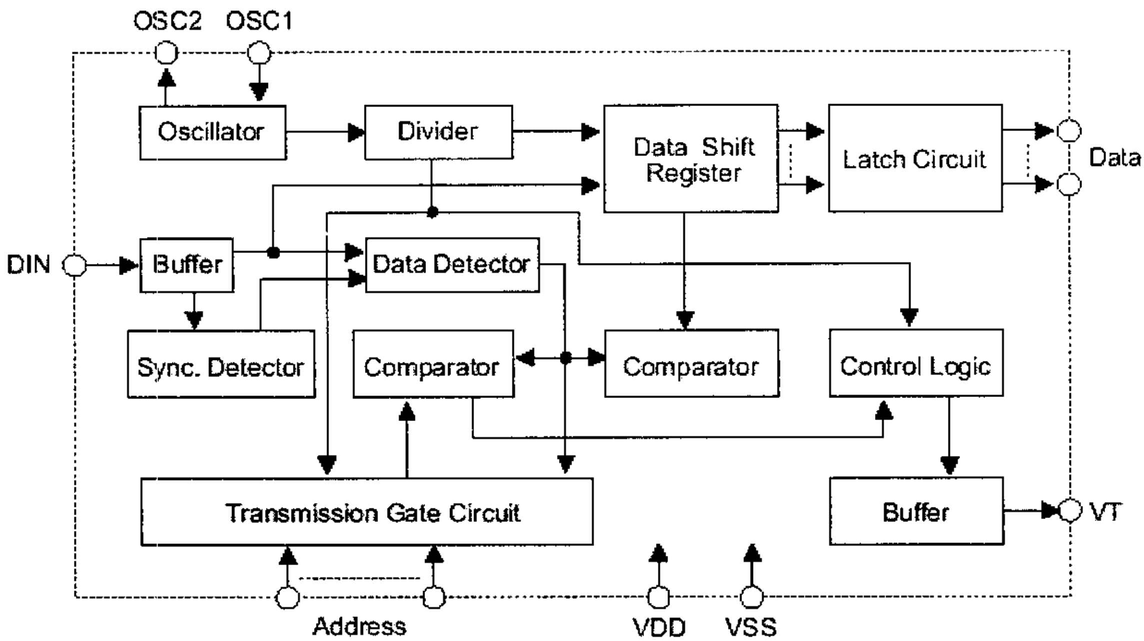
APPENDIX B

DECODER(2^{12} SERIES)

SELECTION TABLE

Function / Part No.	Address No.	Data		VT	Oscillator	Trigger	Package
		No.	Type				
HT12D	8	4	L	√	RC oscillator	DIN active "Hi"	18 DIP/20 SOP

BLOCK DIAGRAM



ELECTRICAL CHARACTERISTICS

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.4	5	12	V
I_{STB}	Standby Current	5V	Oscillator stops	—	0.1	1	µA
		12V		—	2	4	µA
I_{DD}	Operating Current	5V	No load f_{OSC}=150kHz	—	200	400	µA
I_O	Data Output Source Current (D8~D11)	5V	V_{OH}=4.5V	−1	−1.6	—	mA
	Data Output Sink Current (D8~D11)	5V	V_{OL}=0.5V	1	1.6	—	mA
I_{VT}	VT Output Source Current	5V	V_{OH}=4.5V	−1	−1.6	—	mA
	VT Output Sink Current		V_{OL}=0.5V	1	1.6	—	mA
V_{IH}	"H" Input Voltage	5V	—	3.5	—	5	V
V_{IL}	"L" Input Voltage	5V	—	0	—	1	V
f_{OSC}	Oscillator Frequency	5V	R_{OSC}=51kΩ	—	150	—	kHz

Table . . HT-12D Electrical characteristics

PIN DESCRIPTION

Pin Name	I/O	Internal Connection	Description
A0~A11	I	NMOS TRANSMISSION GATE	Input pins for address A0~A11 setting They can be externally set to VDD or VSS.
D8~D11	O	CMOS OUT	Output data pins
DIN	I	CMOS IN	Serial data input pin
VT	O	CMOS OUT	Valid transmission, active high
OSC1	I	OSCILLATOR	Oscillator input pin
OSC2	O	OSCILLATOR	Oscillator output pin
VSS	I	—	Negative power supply (GND)
VDD	I	—	Positive power supply

OUTPUT TYPE

The HT 12D provides 4 latch type data pins whose data remain unchanged until new data are received.

Part No.	Data Pins	Address Pins	Output Type	Operating Voltage
HT12D	4	8	Latch	2.4V~12V

ADDRESS/DATA SEQUENCE

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

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

APPENDIX -C

CONTROL UNIT

SENSING & FLYING UNIT

8. BIBILOGRAPHY

1.Singh.M.D and Kanchandani-'Power Electronics'-Tata McGraw-Hill & Hill publication Company Ltd New Delhi-2002

2.Mohammed Rafiquzzaman,' Micro-processor and Micro-computer Based system design',Universal Bookstall,New Delhi,1990.

3.A.Brece Carlson et al," Communication Systems", McGraw-Hill Int., 4th Edision,2002.

4.E.Balagurusamy,"Programming in ANSI C" Tata McGraw-Hill publishing Company Ltd New Delhi-1992

WEBSITES

1.www.wikipedia.com

2.www.microcontroller.com

3.www.embedded.com

4.www.howstuffswork.com