1. OVERVIEW

1.1 INTRODUCTION

ZigBee focuses on defining a general purpose in expensive self-organizing mesh network that can be used for industrial control, embedded sensing, medical data collection, building automation and home automation.

Our project mainly deals with home automation, which is linking of appliances, monitoring and controlling them through an intelligent network. This project demonstrates home automation applications by making use of the wireless ZigBee protocol. The XBee OEM RF Modules were engineered to operate within the ZigBee protocol and provide reliable delivery of data between remote devices.

1.2 BLOCK DIAGRAM
ZigBee device can play a role as a master or a slave. Master tries to connect itself to other devices and slave is waiting to be connected from other devices. A ZigBee connection can always be made from pair of master and slave devices.

- XBee and XBee-Pro are the RF Modules which are used as ZigBee devices in the transmitter and receiver sections.
- The microcontroller used in this project is Philips 89C51 which belongs to the 8051 family. It takes input from the external sources and roots them to the appropriate devices as programmed in it.
- The LCD used here is PCD8544 which is a low power CMOS controller/device. It is used to display the name of the appliance which is being currently controlled.
- Each and every key in the keypad is designed for a particular function. A device can be controlled by pressing the corresponding key.
- Power Supply Unit is used to provide a constant 5Volts, 1.5A supply to different ICs. This is a standard circuit using external 12V DC adopter and fixed 3-pin voltage regulator.
2. ZIGBEE COMMUNICATIONS

2.1 INTRODUCTION

For the last few years, we've witnessed a great expansion of remote control devices. Five years ago, infrared remotes for the television were the only such devices in our homes. Now we quickly run out of fingers as we count the devices and appliances we can control remotely in our house. This number will only increase as more devices are controlled or monitored from a distance. To interact with all these remotely controlled devices, a new and emerging wireless modality, ZigBee was developed. ZigBee style networks began to be designed in 1998 when many engineers realized that both Wi-Fi and Bluetooth were going to be unsuitable for many applications. It is a published specification designed to use small and low power digital radios based on the IEEE 802.15.4 standard for wireless personal area networks (WPANs).
With its unique emphasis on reliability, low cost, long battery life and easy deployment, ZigBee is paving the way for intelligent sensors to provide greater control of lighting, heating, cooling, water and filtration, appliance-use and security systems from anywhere in and around the home. These additional controls play an instrumental role in dramatically improving energy efficiency as the world grapples with increasing power demands that exceed supply, as well as concerns around greenhouse gas emissions.

2.2 WHAT DOES ZIGBEE DO?

ZigBee is designed for wireless controls and sensors. It could be built into just about anything you have around your home or office, including lights, switches, doors and appliances. These devices can then interact without wires, and you can control them all from a remote control or even your mobile phone. It allows wireless two-way communications between lights and switches, thermostats and furnaces, hotel-room air-conditioners and the front desk, and central command posts. It travels across greater distances and handles many sensors that can be linked to perform different tasks. Figure below gives a great example of how ZigBee can be applied.

![SIX WAYS ZIGBEE WILL CHANGE YOUR HOME](image-url)
ZigBee works well because it aims low. Controls and sensors don't need to send and receive much data. ZigBee has been designed to transmit slowly. It has a data rate of 250kbps (kilobits per second), pitiful compared with Wi-Fi, which is hitting throughput of 20Mbps or more. But because ZigBee transmits slowly, it doesn't need much power, so batteries will last up to 10 years. Because ZigBee consumes very little power, a sensor and transmitter that reports whether a door is open or closed, for example, can run for up to five years on a single double-A battery. Also, operators are much happier about adding ZigBee to their phones than faster technologies such as Wi-Fi; therefore, the phone will be able to act as a remote control for all the ZigBee devices it encounters.

2.3 ZIGBEE PROTOCOL

The ZigBee protocol was engineered by the ZigBee Alliance, a non-profit consortium of leading semiconductor manufacturers, technology providers, OEMs and end-users worldwide. The ZigBee protocol carries all the benefits of the 802.15.4 protocol with added networking functionality. The 802.15.4 specification was developed at the Institute of Electrical and Electronics Engineers (IEEE). The specification is a packet-based radio protocol that meets the needs of low-cost, battery-operated devices. The protocol allows devices to intercommunicate and be powered by batteries that last years instead of hours.
ZigBee can be implemented in mesh networks larger than is possible with Bluetooth. ZigBee compliant wireless devices are expected to transmit 10-75 meters, depending on the RF environment and the power output consumption required for a given application, and will operate in the unlicensed RF worldwide (2.4GHz global, 915MHz Americas or 868 MHz Europe). The data rate is 250kbps at 2.4GHz, 40kbps at 915MHz and 20kbps at 868MHz. IEEE and ZigBee Alliance have been working closely to specify the entire protocol stack. IEEE 802.15.4 focuses on the specification of the lower two layers of the protocol (physical and data link layer). On the other hand, ZigBee Alliance aims to provide the upper layers of the protocol stack (from network to the application layer) for interoperable data networking, security services and a range of wireless home and building control solutions, provide interoperability compliance testing, marketing of the standard, advanced engineering for the evolution of the standard.

2.3.1 ZigBee Protocol Features:

- Low duty cycle - provides long battery life
- Low latency
- Support for multiple network topologies: static, dynamic, star and mesh
- Direct Sequence Spread Spectrum (DSSS)
- Up to 65,000 nodes on a network
- 128-bit AES (Advanced Encryption Standard) - provides secure connections between devices
- Collision avoidance
- Link quality indication
- Clear channel assessment
- Retries and acknowledgements
- Support for guaranteed time slots and packet freshness.
2.4 HOW ZIGBEE WORKS?

ZigBee basically uses digital radios to allow devices to communicate with one another. A typical ZigBee network consists of several types of devices. A network coordinator is a device that sets up the network, is aware of all the nodes within its network, and manages both the information about each node as well as the information that is being transmitted/received within the network. Every ZigBee network must contain a network coordinator. Other Full Function Devices (FFD's) may be found in the network, and these devices support all of the 802.15.4 functions. They can serve as network coordinators, network routers, or as devices that interact with the physical world. The final device found in these networks is the Reduced Function Device (RFD), which usually only serve as devices that interact with the physical world. An example of a ZigBee network is shown below in Figure.

The figure above introduces the concept of the ZigBee network topology. Several topologies are supported by ZigBee, including star, mesh, and cluster tree. Star and mesh networking are both shown in the figure above. As can be seen, star topology is most useful when several end devices are located close together so that they can communicate with a single router node. That node can then be a part of a larger mesh network that ultimately communicates with the network coordinator. Mesh networking allows for
redundancy in node links, so that if one node goes down, devices can find an alternative path to communicate with one another.

2.5 ZIGBEE Vs BLUETOOTH

ZigBee is broadly categorized as a low rate WPAN, and its closest technology is Bluetooth. A good bit of energy has been spent in analyzing whether ZigBee and Bluetooth are complementary or competing technologies, but after a quick look at the two, it can be seen that they fall a lot farther down the complementary side of the spectrum. They are two different technologies with very different areas of application and different means of designing for those applications. While ZigBee is focused on control and automation, Bluetooth is focused on connectivity between laptops, PDA’s, and the like, as well as more general cable replacement. ZigBee uses low data rate, low power consumption, and works with small packet devices; Bluetooth uses a higher data rate, higher power consumption, and works with large packet devices.

ZigBee networks can support a larger number of devices and a longer range between devices than Bluetooth. Because of these differences, the technologies are not only geared toward different applications, they don't have the capability to extend out to other applications. As an example, for its applications, Bluetooth must rely on fairly frequent battery recharging, while the whole goal of ZigBee is for a user to be able to put a couple of batteries in the devices and forget about them for months to years. In timing critical applications, ZigBee is designed to respond quickly, while Bluetooth takes much longer and could be detrimental to the application. Thus, a user could easily use both technologies as a wireless solution in a PAN to suit all types of applications within that network.

2.6 ZIGBEE APPLICATIONS

- **ZigBee Home Automation** is the global standard for the control of appliances, lighting, environment, energy management, safety, and security. It supports a diverse ecosystem of service providers, original device manufacturers (ODM),
and original equipment manufacturers (OEM) with a standards-based wireless solution for home and small office automation.

- **ZigBee Smart Energy** offers utilities and energy service providers secure, easy-to-use wireless home area networks (HAN) for managing energy. Smart Energy gives these groups and their customers the power to directly communicate with thermostats and other smart appliances.

- **ZigBee Health Care** provides a global standard for interoperable wireless devices enabling secure and reliable monitoring and management of non-critical, low-acuity healthcare services targeted at chronic disease. It promotes aging independence, overall health, wellness and fitness by providing more information about one's state of health. Designed for use in homes, fitness centers, retirement communities, nursing homes and a variety of medical care facilities.

- **ZigBee Telecommunication Services**: Innovative mobile devices for innovative lifestyles are in development and will be available soon.

- **ZigBee Industrial Automation** extends existing manufacturing and process control systems reliability.

### 2.7 XBee™/XBee-PRO™ RF MODULES

The XBee and XBee-PRO OEM RF Modules were engineered to meet IEEE 802.15.4 standards and support the unique needs of low-cost, low-power wireless sensor networks. The modules require minimal power and provide reliable delivery of data between devices.

The modules operate within the ISM 2.4 GHz frequency band and are pin-for-pin compatible with each other.
2.7.1 Key Features

Long Range Data Integrity:

- **XBee**
  - Indoor/Urban: up to 100’ (30 m)
  - Outdoor line-of-sight: up to 300’ (100 m)
  - Transmit Power: 1 mW (0 dBm)
  - Receiver Sensitivity: -92 dBm

- **XBee-PRO**
  - Indoor/Urban: up to 300’ (100 m)
  - Outdoor line-of-sight: up to 1 mile (1500 m)
  - Transmit Power: 100 mW (20 dBm) EIRP
  - Receiver Sensitivity: -100 dBm

- **RF Data Rate**
  - 250,000 bps

Advanced Networking & Security:

- Retries and Acknowledgements
- DSSS (Direct Sequence Spread Spectrum)
- Each direct sequence channels has over 65,000 unique network addresses available
- Source/Destination Addressing
- Unicast & Broadcast Communications
- Point-to-point, point-to-multipoint and peer-to-peer topologies supported
- Coordinator/End Device operations

**Low Power:**

- **XBee**
  - TX Current: 45 mA (@3.3 V)
  - RX Current: 50 mA (@3.3 V)
  - Power-down Current: < 10 μA

- **XBee-Pro**
  - TX Current: 215 mA (@3.3 V)
  - RX Current: 55 mA (@3.3 V)
  - Power-down Current: < 10 μA

**ADC and I/O line support:**

- Analog-to-digital conversion
- Digital I/O Line Passing

**Easy-to-Use:**

- No configuration necessary for out-of-box RF communications
- Free X-CTU Software (Testing and configuration software)
- AT and API Command Modes for configuring module parameters
- Extensive command set
2.7.2 Mechanical Drawings

The figure below shows mechanical drawings of the XBee/XBee-PRO OEM RF Modules. The XBee and XBee-PRO RF Modules are pin-for-pin compatible.

2.7.3 XBee/XBee-PRO Addressing

Every RF data packet sent over-the-air contains a Source Address and Destination Address field in its header. The RF module conforms to the 802.15.4 specification and supports both short 16-bit addresses and long 64-bit addresses. A unique 64-bit IEEE source address is assigned at the factory and can be read with the SL (Serial Number Low) and SH (Serial Number High) commands. Short addressing must be configured manually. A module will use its unique 64-bit address as its Source Address if its MY (16-bit Source Address) value is “0xFFFF” or “0xFFFE”. To send a packet to a specific module using 64-bit addressing: Set Destination Address (DL + DH) to match the Source Address (SL + SH) of the intended destination module. To send a packet to a specific module using 16-bit addressing: Set DL (Destination Address Low) parameter to equal the MY parameter and set the DH (Destination Address High) parameter to ‘0’.
3. MICROCONTROLLERS

3.1 INTRODUCTION

A microcontroller is an integrated chip that is often part of an embedded system. The microcontroller includes a CPU, RAM, ROM, I/O ports, and timers like a standard computer but they are designed to execute only a specific task to control a single system and are much smaller and simplified so that they can include all the functions required on a single chip.
3.2 DESCRIPTION OF PHILIPS 89C51 MICROCONTROLLER

Phillips 89C51 contains a non-volatile FLASH program memory that is parallel programmable. Phillips 89C51, 8-bit Micro controller from MHS-51 Intel family, with 4K bytes of flash and 128 bytes of internal RAM had been used. It has a 40-pin configuration and other components of interfaced to its ports. The Micro controller takes input from the external sources and routes them to the appropriate devices as programmed in it.

3.2.1 Features

- 89C51 Central Processing Unit
- On-chip FLASH Program Memory
- Speedup to 33 MHz
- RAM expandable externally up to 64 Kbytes
- Four interrupt priority levels and Six interrupt sources
- Fully static operation
- Four 8-bit input output ports
- Full-duplex enhanced UART
  - Framing error detection
  - Automatic address recognition
- Three 16-bit timers/counters T0, T1 and additional T2
- Power control modes
  - Clock can be stopped and resumed
  - Idle mode
• Power down mode

➤ Asynchronous port reset

3.2.2 Central Processing Unit

The CPU processes instructions for arithmetic and logical operations, bit manipulations, and data transfer operations, input and output operations, program flow control, program sequencing and supervising the system operations.

3.2.3 Arithmetic Logic Unit

The ALU performs arithmetic and logic functions on 8-bit variables. The ALU can perform addition, subtraction, multiplication and division and the logic unit can perform logical operations. An important and unique feature of the microcontroller architecture is that the ALU can also manipulate 1 bit as well as 8-bit data types. Individual bits may be set, cleared, complemented, moved, tested and used in logic computation.

3.2.4 Accumulator

It is returned as register A or Acc. It is an 8-bit Register. Accumulator holds a source of operand and stores the result of the arithmetic operations such as addition, subtraction, multiplication and division. The accumulator can be the source or destination register for logical operations. The accumulator has several exclusive functions such as rotate, parity computation; testing for 0, sign acceptor etc. and so on.

3.2.5 Program Counter

The program counter points to the address of the next instruction to be executed. As the CPU fetches the opcode from the program ROM, the program counter is implemented to point to the next instruction. The microcontroller can access program addresses 0000 to FFFFH, a total of 64K bytes of code.
3.3 PIN DIAGRAM

PIN DIAGRAM:

Pin Diagram of Philips 89C51 Microcontroller
3.3.1 PIN DESCRIPTIONS

- **1–8: Port 1**: Each of these pins can be used as either input or output according to your needs. Also, pins 1 and 2 (P1.0 and P1.1) have special functions associated with Timer.

- **9: Reset Signal**: High logical state on this input halts the MCU and clears all the registers. Bringing this pin back to logical state zero starts the program a new as if the power had just been turned on. In another words, positive voltage impulse on this pin resets the MCU. Depending on the device's purpose and environs, this pin is usually connected to the push-button, reset-upon-start circuit or a brown out reset circuit.

- **10-17: Port 3**: As with Port 1, each of these pins can be used as universal input or output. However, each pin of Port 3 has an alternative function:
  
  - **Pin 10: RXD**: serial input for asynchronous communication or serial output for synchronous communication.
  
  - **Pin 11: TXD**: serial output for asynchronous communication or clock output for synchronous communication

  - **Pin 12: INT0**: input for interrupt 0

  - **Pin 13: INT1**: input for interrupt 1

  - **Pin 14: T0**: clock input of counter 0
• **Pin 15: T1** - clock input of counter 1

• **Pin 16: WR** - signal for writing to external (add-on) RAM memory

• **Pin 17: RD** - signal for reading from external RAM memory.

  - **18-19: X2 and X1**: Input and output of internal oscillator. Quartz crystal controlling the frequency commonly connects to these pins. Capacitances within the oscillator mechanism are not critical and are normally about 30pF. Instead of a quartz crystal, miniature ceramic resonators can be used for dictating the pace. In that case, manufacturers recommend using somewhat higher capacitances.

  - **20: GND**: Ground

  - **21- 28: Port 2**: If external memory is not present, pins of Port 2 act as universal input/output. If external memory is present, this is the location of the higher address byte, i.e. addresses A8 – A15. In cases when not all the 8 bits are used for addressing the memory (i.e. memory is smaller than 64kB), the rest of the unused bits are not available as input/output.

  - **29: PSEN**: MCU activates this bit (brings to low state) upon each reading of byte (instruction) from program memory. If external ROM is used for storing the program, PSEN is directly connected to its control pins.

  - **31: EA**: Bringing this pin to the logical state zero (mass) designates the ports P2 and P3 for transferring addresses regardless of the presence of the internal memory. This means that even if there is a program loaded in the MCU it will not be executed, but the one from the external ROM will be used instead. Conversely, bringing the pin to the high logical state causes the controller to use both memories, first the internal, and then the external (if present).

  - **32-39: Port 0**: Similar to Port 2, pins of Port 0 can be used as universal input/output, if external memory is not used. If external memory is used, P0 behaves as address output (A0 – A7) when ALE pin is at high logical level, or as
data output (Data Bus) when ALE pin is at low logical level.

- 40: VCC: Power +5V.

3.4 MEMORY ARCHITECTURE

During the runtime, micro controller uses two different types of memory: one for holding the program being executed (ROM memory), and the other for temporary storage of data and auxiliary variables (RAM memory). Depending on the particular model from 8051 family, this is usually few kilobytes of ROM and 128/256 bytes of RAM. This amount is built-in and is sufficient for common tasks performed "independently" by the MCU. However, 8051 can address up to 64KB of external memory. These can be separate memory blocks, (separate RAM chip and ROM chip) totaling 128KB of memory on MCU, which is a real programming goody.

3.4.1 ROM memory

First models from 8051 families lacked the internal program memory, but it could be added externally in a form of a separate chip. New models have built-in ROM, although there are substantial variations. With some models internal memory cannot be programmed directly by the user. Instead, the user needs to precede the program to the manufacturer, so that the MCU can be programmed (masked) appropriately in the process of fabrication. Many manufacturers deliver controllers that can be programmed directly by the user. These come in a ceramic case with an opening (Erasable Programmable Read Only Memory, EPROM version) or in a plastic case without an opening (Electrically Erasable Programmable Read Only Memory, EEPROM version). Flash ROM is also another popular method of storing a program.

Flash memory features:

- FLASH EPROM internal program memory with chip erases.
• Up to 64K byte external program memory if the internal program memory is disabled.
• Programmable security bits.
• 10,000 minimum erase/program cycles for each byte & 10 year minimum data retention.

3.4.2 RAM memory

RAM is used for storing temporary data and auxiliary results generated during the runtime. Apart from that, RAM comprises a number of registers: hardware counters and timers, I/O ports, buffer for serial connection, etc. With older versions, RAM spanned 256 locations, while new models feature additional 128 registers. First 256 memory
locations form the basis of RAM (addresses 0 – Fifth) of every 8051 MCU. Locations that are available to the user span addresses from 0 to 7Fh, i.e. first 128 registers, and this part of RAM is split into several blocks.

- First block comprises 4 "banks" of 8 registers each, marked as R0 - R7. To address these, the parent bank has to be selected.

- Second memory block (range 20h – 2Fh) is bit-addressable, meaning that every belonging bit has its own address (0 to 7Fh). Since the block comprises 16 of these registers, there is a total of 128 addressable bits. (Bit 0 of byte 20h has bit address 0, while bit 7 of byte 2Fh has bit address 7Fh).

- Third is the group of available registers at addresses 2Fh – 7Fh (total of 80 locations) without special features or a preset purpose.

3.4.3 Memory Expanding

In case the built-in amount of memory (either RAM or ROM) is not sufficient for our needs, there is always an option of adding two external 64KB memory chips. When added, they are addressed and accessed via I/O ports P2 and P3.

8051 MCU has two separate read signals, RD# (P3.7) and PSEN#. The first one is active when reading byte from the external data memory (RAM), and the second one is active when reading byte from the external program memory (ROM). Both signals are active on low logical level. An expansion using separate chips for RAM and ROM is known as Harvard architecture.

Memory can be also mapped as a single block, functioning as both data memory and program memory simultaneously (only one memory chip is used). This approach is known as Von Neumann architecture. To be able to read the same block using RD# or PSEN#, these two signals were combined via logical AND.

Using the Hardware architecture effectively doubles MCU memory, but that's not the
only advantage offered by the method. Keeping the program code separated from the data makes the controller more reliable.

3.5 SFR (Special Function Registers)

SFR registers can be seen as a sort of control panel for managing and monitoring the micro controller. Every register and each of the belonging bits has its name, specified address in RAM and strictly defined role (e.g. controlling the timer, interrupt, serial connection, etc). Although there are 128 available memory slots for allocating SFR registers, the basic core shared by 8051 MCU has 22 registers. The rest has been left open intentionally to allow future upgrades while retaining the compatibility with earlier models.

<table>
<thead>
<tr>
<th>ADDR (HEX.)</th>
<th>MARK</th>
<th>FULL NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>P0</td>
<td>PORT 0</td>
</tr>
<tr>
<td>81</td>
<td>SP</td>
<td>STACK POINTER</td>
</tr>
<tr>
<td></td>
<td>DPTR</td>
<td>DATA POINTER</td>
</tr>
<tr>
<td>82</td>
<td>DPL</td>
<td>DATA LOW POINTER</td>
</tr>
<tr>
<td>83</td>
<td>DPH</td>
<td>DATA HIGH POINTER</td>
</tr>
<tr>
<td>87</td>
<td>PCON</td>
<td>POWER CONTROL</td>
</tr>
<tr>
<td>88</td>
<td>TCON</td>
<td>TIMER/COUNTER CONTROL</td>
</tr>
<tr>
<td>89</td>
<td>TMOD</td>
<td>TIMER/COUNTER MODE CONTROL</td>
</tr>
<tr>
<td>8A</td>
<td>TL0</td>
<td>TIMER/COUNTER0 LOW BYTE</td>
</tr>
<tr>
<td>8B</td>
<td>TL1</td>
<td>TIMER/COUNTER1 LOW BYTE</td>
</tr>
<tr>
<td>8C</td>
<td>TH0</td>
<td>TIMER/COUNTER0 HIGH BYTE</td>
</tr>
<tr>
<td>8D</td>
<td>TH1</td>
<td>TIMER/COUNTER1 HIGH BYTE</td>
</tr>
<tr>
<td>90</td>
<td>P1</td>
<td>PORT 1</td>
</tr>
<tr>
<td>98</td>
<td>SCON</td>
<td>SERIAL PORT CONTROL</td>
</tr>
<tr>
<td>99</td>
<td>SBUF</td>
<td>SERIAL DATA PORT</td>
</tr>
<tr>
<td>A0</td>
<td>P2</td>
<td>PORT 2</td>
</tr>
<tr>
<td>A8</td>
<td>IE</td>
<td>INTERRUPT ENABLE</td>
</tr>
<tr>
<td>B0</td>
<td>P3</td>
<td>PORT 3</td>
</tr>
<tr>
<td>B8</td>
<td>IP</td>
<td>INTERRUPT PRIORITY CONTROL</td>
</tr>
<tr>
<td>D0</td>
<td>PSW</td>
<td>PROGRAM STATUS WORD</td>
</tr>
<tr>
<td>E0</td>
<td>ACC(A)</td>
<td>ACCUMULATOR</td>
</tr>
<tr>
<td>F0</td>
<td>B</td>
<td>B REGISTER</td>
</tr>
</tbody>
</table>

3.5.1 PSW (program status word) Register:
The program status word (PSW) register is on 8-bit register. It is also referred to as the flag register.

\[
\begin{array}{cccccccc}
7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
CY & AC & ---- & RS1 & RS0 & OV & ---- & P \\
\end{array}
\]

- **CY** - Carry-flag.
- **AC** - Auxiliary carry-flag.
- **----** - Available to the user for general-purpose.
- **RS1** - register bank selector bit 1.
- **RS0** - register bank selector bit 0.
- **OV** - overflow flag.
- **----** - User definable bit.
- **P** - Parity flag. Set/cleared by hardware each instruction cycle to indicate an odd/even.

### 3.5.2 SCON (Serial Control) Register:

The Serial Control SFR is used to configure the behavior of the 8051's on-board serial port. This SFR controls the baud rate of the serial port, whether the serial port is activated to receive data, and also contains flags that are set when a byte is successfully sent or received.

\[
\begin{array}{ccccccccc}
7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
SM0/FE & SM1 & SM2 & REN & TB8 & RB8 & TI & RI \\
\end{array}
\]

- **REN** - set or cleared by software to enable or disable reception.
- **TB 8** - not widely used.
- **RB 8** - not widely used.
- **TI** - transmits interrupt flag. Set by hardware at the beginning of the stop bit in mode 1. It must be cleared by software.
RI-received interrupts flag. Set by hardware halfway through the stop bit time in mode 1. It must be cleared by software.

<table>
<thead>
<tr>
<th>SM0</th>
<th>SM1</th>
<th>Serial mode 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Synchronous mode</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>8-bit data, 1 start bit, 1 stop bit, variable baud rate</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>9- bit data, 1 start bit, 1 stop bit, fixed baud rate</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>9- bit data, 1 start bit, 1 stop bit, variable baud rate</td>
</tr>
</tbody>
</table>

3.5.3 PCON (Power Control) Register:

The Power Control SFR is used to control the 8051's power control modes. Certain operation modes of the 8051 allow the 8051 to go into a type of "sleep" mode which requires much less power. These modes of operation are controlled through PCON.

3.5.4 SBUF (Serial Control):

The Serial Buffer SFR is used to send and receive data via the on-board serial port. Any value written to SBUF will be sent out the serial port's TXD pin. Likewise, any value which the 8051 receives via the serial port's RXD pin will be delivered to the user program via SBUF. In other words, SBUF serves as the output port when written to and as an input port when read from.

3.6 TIMERS

Both Timer0 and Timer1 are 16-bits wide. 8051 has an 8-bit architecture, each
16-bit timer is accessed as two separate registers of Lower byte and Higher byte. The "timer" or "counter" function is selected by control bits C/T in the special function register TMOD. These two timer/counters have four operating modes, which are selected by bit-pairs (M1/M0) in TMOD. Modes 0, 1, and 2 are the same for both timers/counters. Mode 3 is different.

### 3.6.1 TMOD (Timer Mode) Registers

The Timer Mode SFR is used to configure the mode of operation of each of the two timers.

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>GATE</td>
<td>C/T</td>
<td>M1</td>
<td>M0</td>
<td>GATE</td>
<td>C/T</td>
<td>M1</td>
<td>M0</td>
</tr>
</tbody>
</table>

---------- TIMERR 1 ----------  ---------- TIMERR 0 ----------

**GATE** : When set, start and stop of timer by hardware
When reset, start and stop of timer by software

**C/T** : Cleared for timer operation
Set for counter operation

**Modes of TMOD Register**
<table>
<thead>
<tr>
<th>M1</th>
<th>M0</th>
<th>MODE</th>
<th>OPERATING MODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>13-bit timer mode</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>16-bit timer mode</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>2</td>
<td>8-bit auto reload mode</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td>Split timer mode</td>
</tr>
</tbody>
</table>

3.6.2 TCON (Timer Control) Register:

The Timer Control SFR is used to configure and modify the way in which the 8051's two timers operate. This SFR controls whether each of the two timers is running or stopped and contains a flag to indicate that each timer has overflowed.

7 6 5 4 3 2 1 0

| TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |

**TF**: timer overflows flag. Set by hardware when the timer/counter overflows. It is cleared by hardware, as the processor vectors to the interrupt service routine.

**TR**: timer run control bit. Set or cleared by software to turn timer or counter on/off.

**IE**: set by CPU when the external interrupt edge (H-to-L transition) is detected. It is cleared by CPU when the interrupt is processed.

**IT**: set/cleared by software to specify falling edge/low-level triggered external interrupt.

3.7 INTERRUPTS
A single microcontroller can serve several devices. In the interrupt method, whenever any device needs its service, the device notifies the microcontroller by sending it an interrupt signal. Upon receiving an interrupt signal, the microcontroller interrupts whatever it is doing and serves the device. The program associated with the interrupt is called the interrupt service routine (ISR). The advantageous of interrupts is that the microcontroller can serve many devices based on the priority assigned to it.

**Six Interrupts in 89C51**
1. Reset.
2. Two interrupts are set aside for the timers.
3. Two interrupts are set aside for hardware external hardware interrupts.
4. Serial Communications has a single interrupt (receive and transfer).

The Interrupt Enable SFR is used to enable and disable specific interrupts. The low 7 bits of the SFR are used to enable/disable the specific interrupts, where as the highest bit is used to enable or disable ALL interrupts.

**3.8 SERIAL COMMUNICATION**

Serial communication is the process of sending data one bit at one time, sequentially, over a communication channel or computer bus. One of the 8051s many powerful features is its integrated UART, otherwise known as a serial port. The fact that the 8051 has an integrated serial port means that you may very easily read and write values to the serial port.
There are two basic types of serial communications, synchronous and asynchronous. With Synchronous communications, the two devices initially synchronize themselves to each other, and then continually send characters to stay in sync.

Asynchronous means "no synchronization", and thus does not require sending and receiving idle characters. However, the beginning and end of each byte of data must be identified by start and stop bits. The start bit indicate when the data byte is about to begin and the stop bit signals when it ends.

### 3.8.1 RS232

To allow compatibility among the data communication equipment made by various manufacturers; an interfacing standard called RS232, was set by the electronics industries association (EIA) in 1960. RS 232 is the standard defined for the connection of "Data Terminal Equipment" (DTE) to "Data Communications Equipment" (DCE). DTE (Data Terminal Equipment) is a generic term for an item which forms part of the "information processing" portions of a system. Examples are: computer, printer, and terminal. DCE (Data Communications Equipment) is a device, which provides an interface between a DTE and a communications link.

### 3.9 ADDRESSING MODES

An "addressing mode" refers to how you are addressing a given memory location.

**Immediate Addressing**

Immediate addressing is so-named because the value to be stored in memory immediately follows the operation code in memory.

For example, the instruction:

```
MOV A,#20h
```

This instruction uses Immediate Addressing because the Accumulator will be loaded with the value that immediately follows; in this case 20 (hexadecimal).
**Direct Addressing**

Direct addressing is so-named because the value to be stored in memory is obtained by directly retrieving it from another memory location.

**MOV A,30h**

This instruction will read the data out of Internal RAM address 30 (hexadecimal) and store it in the Accumulator.

**Indirect Addressing**

Indirect addressing is a very powerful addressing mode which in many cases provides an exceptional level of flexibility. Indirect addressing appears as follows:

**MOV A,@R0**

This instruction causes the 8051 to analyze the value of the R0 register. The 8051 will then load the accumulator with the value from Internal RAM which is found at the address indicated by R0.

**Register Addressing**

It involves the use of Registers to hold the data to be manipulated. The source and destination Registers must match in size. We can move the data between Accumulator and register, but movement of data between registers is not allowed.

**MOV A, R0**

This instruction will copy the contents of R0 into A
4. LIQUID CRYSTAL DISPLAY

4.1 INTRODUCTION

The PCD8544 is a low power CMOS LCD controller/driver, designed to drive a graphic display of 48 rows and 84 columns. All necessary functions for the display are provided in a single chip, including on-chip generation of LCD supply and bias voltages, resulting in a minimum of external components and low power consumption. The PCD8544 interfaces to micro controllers through a serial bus interface. The PCD8544 is manufactured in n-well CMOS technology.

4.2 FEATURES

- Single chip LCD controller/driver
- 48 row, 84 column outputs
- Display data RAM 48 ’ 84 bits
- On-chip:
  - Generation of LCD supply voltage (external supply also possible)
  - Generation of intermediate LCD bias voltages
  - Oscillator requires no external components (external clock also possible).
- External RES (reset) input pin
- Serial interface maximum 4.0 M bits/s
- CMOS compatible inputs
- Logic supply voltage range VDD to VSS: 2.7 to 3.3 V
- Display supply voltage range VLCD to VSS
- 6.0 to 8.5 V with LCD voltage internally generated (voltage generator enabled)
- 6.0 to 9.0 V with LCD voltage externally supplied (voltage generator switched-off).
- Low power consumption, suitable for battery operated systems
4.3 BLOCK DIAGRAM

4.4 FUNCTIONAL DESCRIPTION OF GRAPHICAL LCD

- **Oscillator**
  
The on-chip oscillator provides the clock signal for the display system. No external components are required and the OSC input must be connected to VDD. An external clock signal, if used, is connected to this input.

- **Address Counter (AC)**
  
The address counter assigns addresses to the display data RAM for writing. The X-address X6 to X0 and the Y-address Y2 to Y0 are set separately. After a write operation, the address counter is automatically incremented by 1, according to the V flag.
➤ **Display Data RAM (DDRAM)**

The DDRAM is a 48 ´ 84 bit static RAM which stores the display data. The RAM is divided into six banks of 84 bytes (6 ´ 8 ´ 84 bits). During RAM access, data is transferred to the RAM through the serial interface. There is a direct correspondence between the X-address and the column output number.

➤ **Timing generator**

The timing generator produces the various signals required to drive the internal circuits. Internal chip operation is not affected by operations on the data buses.

➤ **Display address counter**

The display is generated by continuously shifting rows of RAM data to the dot matrix LCD through the column outputs. The display status (all dots on/off and normal/inverse video) is set by bits E and D by the ‘display control’ command.

➤ **LCD row and column drivers**

The PCD8544 contains 48 row and 84 column drivers, which connect the appropriate LCD bias voltages in sequence to the display in accordance with the data to be displayed. Unused outputs should be left unconnected.

### 4.5 LCD PIN DESCRIPTION

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>R0 to R47</td>
<td>LCD ROW DRIVER OUTPUTS</td>
</tr>
<tr>
<td>C0 to C83</td>
<td>LCD COLUMN DRIVER OUTPUTS</td>
</tr>
<tr>
<td>VSS1,VSS2</td>
<td>GROUND</td>
</tr>
<tr>
<td>VDD1,VDD2</td>
<td>SUPPLY VOLTAGE</td>
</tr>
<tr>
<td>VLCD1,VLCD2</td>
<td>LCD SUPPLY VOLTAGE</td>
</tr>
<tr>
<td>T1</td>
<td>TEST 1 INPUT</td>
</tr>
<tr>
<td>T2</td>
<td>TEST2 OUTPUT</td>
</tr>
<tr>
<td>T3</td>
<td>TEST3 INPUT/OUTPUT</td>
</tr>
</tbody>
</table>

32
4.6 DESCRIPTION OF INSTRUCTION SET

4.6.1 Initialization

Immediately following power-on, the contents of all internal registers and of the RAM are undefined. A RES pulse must be applied. Attention should be paid to the possibility that the device may be damaged if not properly reset. All internal registers are by applying an external RES pulse (external low) at pad 31, within the specified time. However, the RAM contents are still undefined. The state after reset is described in section below.

The RES input must be ≤0.3VDD when VDD reaches VDDmin (or higher) within a maximum time of 100ms after VDD goes HIGH.

4.6.2 Reset Function

After reset, the LCD driver has the following state:

- Power-down mode (bit PD=1)
- Horizontal addressing (bit V=0) normal instruction set (bit H=0)
- Display blank (bit E=D=0)
- Address counter X5 to X0 = 0; Y2 to Y0 = 0
- Temperature control mode (TC1 TC0 = 0)
- Bias system (BS2 to BS1 = 0)
- VLCD is equal to 0, the HV generator is switched off (Vop6 to Vop0)
- After power-on, the RAM contents are undefined.
5. BRIEF DESCRIPTION OF SOFTWARE MODULES

5.1 INTRODUCTION

Programming Language : C Language
Software used : KEIL μVision2 IDE, Flash Magic

System requirements

There are minimum hardware and software requirements that must be satisfied to ensure that the compiler and utilities function properly.

For our Windows-based tools, we must have the following:

- PC with Pentium, Pentium-II or compatible processor,
- Windows 95, Windows-98, Windows NT 4.0, or higher
- 16 MB RAM minimum,
- 20 MB free disk space.

5.2 EVALUATION OF KEIL SOFTWARE

The KEIL μVision2 IDE combines project management, a rich-featured editor with Interactive error correction, option setup, makes facility, and on-line help. We use μVision2 to create our source files and organize them into a project that defines our target application. μVision2 automatically compiles, assembles, and links our embedded application and provides a single focal point for our development efforts.

1. Start the μVision Program
2. After the program has started:
3. Select File, New… from the program menu. Type your assembly file.
4. Select File, Save… from the program menu
5. The first time you save the program a dialog box will popup and allow you to name your file and file type.
6. Save program with filename: Toggle.a51
7. Change the file type: Assembler (*.a51)
8. Select Project, New Project… from the program menu
9. Give project name: Toggle.prj
10. Click on the Add button
11. A dialog-box appears, allowing you to add files to the project
12. Change the file type to Assembly.
14. Click on the Add button then close the Add dialog box.
15. Click on Save in your Project dialog box.
16. Select Project, Make: Build Project from the program menu
17. This creates the HEX file you need for the 8051

5.2.1 Using The Keil Dscope Debugger
1. Select Run, dScope debugger… from the program menu
   The debug program will start a new session
2. Select File, load CPU driver from the program menu
   Choose the 8051.dll from the drop down list box; you can also select this directly.
3. Select File, load object file from the program menu.
   Change the file type to HEX
Select your hex file, e.g. Toggle. Hex
Click OK
4. You should now see the source code of the file typed in earlier
5. Select Peripherals, I/O Ports from the program menu.
   Select Port 0, Port 1, Port 2 and Port 3
   This will bring up 4 boxes that display the status of the ports on the microcontroller.
6. Click on go to see the real time update of the I/O ports.
7. Click on stop when you are finished.

5.3 FLASH MAGIC:

The Flash Magic is feature rich in windows based tool for downloading of code into flash micro controller. It utilizes feature of microcontrollers called ISP, which allow the transfer of data serially between a PC and the device.

Flash Magic can erase device, program, read data and read and set various configuration information. rather than providing the basic feature of ISP, Flash Magic adds additional features and intelligence, allowing complex operation to be performed. For example, erasing can be any collection pages, blocks, the hex file to be programmed or the entire device.

Additional advanced feature of Flash Magic include the automatic programming, checksum, entering ISP mode via serial command, execution of just time module allowing endless flexibility in the data programmed, control over RS232 signal to place devices into ISP mode and control over the timing of such signal.

Flash magic supports all current 8-bit (8051), 16 bit and 32 bit flash microcontrollers.
6. CODE
6.1 ZIGBEE COMMUNICATION TRANSMITTER

```c
#include <reg51.h>
#include<intrins.h>

sbit lcd_rs = P1^5;
sbit lcd_rw = P1^6;
sbit lcd_en = P1^7;
#define lcd_data P0
sbit lcd_busy = P0^7;

void lcdline1();
void lcdline2();
void lcdline3();
void lcdline4();
void lcd_clr();
void busy_check();
void lcdinit();
void displaystring(char *ptr);
void lcd_cmmnd( unsigned char val);

sbit sw1=P1^1;
sbit sw2=P1^2;
sbit sw3=P1^3;
sbit sw4=P1^4;

sbit led1=P2^1;
sbit led2=P2^2;

void loop1();
void loop2();
void loop3();
void loop4();
void delay_ms(unsigned char);

void main()
{
    lcdinit();
    TMOD=0x20;
    SCON=0x50;

    // Main code here
}
```
TH1=0x0FD;
TR1=1;
P2=0x00;
delay_ms(500);
  lcdline1();
  displaystring("ZIGBEE COMUNICATION");
  delay_ms(500);
lcdline2();
displaystring("developed by");
  delay_ms(500);
  lcdline4();
displaystring("US");
  delay_ms(500);
lcd_clr();

while(1)
{
  if (sw1==0)
    loop1();
  if (sw2==0)
    loop2();
  if (sw3==0)
    loop3();
  if (sw4==0)
    loop4();
}

void loop1()
{
  lcdline1();
  displaystring("50W LIGHT ON ");
  SBUF='a';
  while(T1==0);
  T1=0;
  while(1)
  {
    led1=1;
    if (sw2==0)
      loop2();
    if (sw3==0)
      loop3();
    if (sw4==0)
      loop4();
  }
}
void loop2()
{
    lcdline1();
    displaystring("50W LIGHT OFF ");
    SBUF='b';
    while(TI==0);
    TI=0;
    while(1)
    {
        led1=0;
        if (sw1==0)
            loop1();
        if (sw3==0)
            loop3();
        if (sw4==0)
            loop4();
    }
}

void loop3()
{
    lcdline2();
    displaystring("fan ON ");
    SBUF='c';
    while(TI==0);
    TI=0;
    while(1)
    {
        led2=1;
        if (sw1==0)
            loop1();
        if (sw2==0)
            loop2();
        if (sw4==0)
            loop4();
    }
}

void loop4()
{
    lcdline2();
displaystring(" fan Off ");
SBUF='d';
while(TI==0);
TI=0;

while(1)
{
    led2=0;
    if (sw1==0)
        loop1();
    if (sw2==0)
        loop2();
    if (sw3==0)
        loop3();
}

void lcdinit()
{
    lcd_cmmnd(0x30);
    lcd_cmmnd(0x38);
    lcd_cmmnd(0x06);
    lcd_cmmnd(0x0c);
    lcd_cmmnd(0x01);
}

void lcd_cmmnd( unsigned char val)
{
    busy_check();
    lcd_data =val;
    lcd_rs=0;
    lcd_rw=0;
    lcd_en=1;
    _nop_();
    lcd_en=0;
}

void lcd_dataout(unsigned char val)
{
    busy_check();
    lcd_data =val;
    lcd_rs=1;
    lcd_rw=0;
    lcd_en=0;
    _nop_();
}
void lcdline1()
{
    lcd_cmmnd(0x80);
}

void lcdline2()
{
    lcd_cmmnd(0xc0);
}

void lcdline3()
{
    lcd_cmmnd(0x94);
}

void lcdline4()
{
    lcd_cmmnd(0xd4);
}

void lcd_clr()
{
    lcd_cmmnd(0x01);
}

void displaystring(char *ptr)
{
    while(*ptr)
    {
        lcd_dataout(*ptr);
        delay_ms(2);
        ptr++;
    }
}

void busy_check(void)
{
    lcd_data = 0xff;
    lcd_rs = 0;
```c
    lcd_rw = 1;
    while(lcd_busy==1)
    {
        lcd_en = 0;
        _nop_();
        _nop_();
        lcd_en = 1;
    }

    void delay_ms(unsigned char p)
    {
        unsigned int i, j;
        for(i=0; i<1273; i++)
            for(j=0; j<p; j++);
    }

6.2 ZIGBEE COMMUNICATION RECEIVER

#include <reg51.h>
#include<intrins.h>
sbit lcd_rs = P1^5;
sbit lcd_rw = P1^6;
sbit lcd_en = P1^7;
#define lcd_data P0
sbit lcd_busy = P0^7;

void lcdline1();
void lcdline2();
void lcdline3();
void lcdline4();
void lcd_clr();
void delay_1msc();
void lcdinit();
void displaystring(char *ptr);
void lcd_cmmnd( unsigned char val);

    sbit led1 = P2^1;
sbit led2 = P2^2;
```
void loop1();
void loop2();
void loop3();
void loop4();

unsigned char x;

void main()
{
    lcdinit();
    TMOD=0x20;
    SCON=0x50;
    TH1=0x0FD;
    TR1=1;
    P2=0x00;
    while(1)
    {
        while (RI==0);
        x=SBUF;
        RI=0;
        if(x=='a')
            loop1();
        if(x=='b')
            loop2();
        if(x=='c')
            loop3();
        if(x=='d')
            loop4();
    }
}

void loop1()
{
    led1=1;
    lcdline1();
displaystring("50W LIGHT ON ");
    while(RI==0);
    x=SBUF;
    RI=0;
    while(1)
    {
        if (x=='b')
            loop2();

43
if (x=='c')
loop3();
if (x=='d')
loop4();
}
}

void loop2()
{

led1=0;

lcdline1();
displaystring("50W LIGHT OFF");
while(RI==0);

x=SBUF;

RI=0;
while(1)
{

if (x=='a')
loop1();
if (x=='c')
loop3();
if (x=='d')
loop4();
}
}

void loop3()
{

led2=1;

lcdline2();
displaystring("FAN ON ");
while(RI==0);

x=SBUF;

RI=0;
while(1)
{

if (x=='a')
loop1();
}
if (x=='b')
  loop2();
if (x=='d')
  loop4();
}
}

void loop4()
{
  led2=0;
  lcdline2();
displaystring("FAN OFF");
  while(RI==0);
  x=SBUF;
  RI=0;
  while(1)
  {
    if (x=='a')
      loop1();
    if (x=='b')
      loop2();
    if (x=='c')
      loop3();

  }
}

void lcdinit()
{
  lcd_cmmnd(0x30);
  lcd_cmmnd(0x38);
  lcd_cmmnd(0x06);
  lcd_cmmnd(0x0c);
  lcd_cmmnd(0x01);
}

void lcd_cmmnd( unsigned char val)
{
  delay_1msc();
  lcd_data =val;
  lcd_rs=0;
  lcd_rw=0;
void lcd_dataout(unsigned char val)
{
    delay_1msc();
    lcd_data = val;
    lcd_rs = 1;
    lcd_rw = 0;
    lcd_en = 1;
    _nop_();
    lcd_en = 0;
}

void lcdline1()
{
    lcd_cmmnd(0x80);
}

void lcdline2()
{
    lcd_cmmnd(0xc0);
}

void lcdline3()
{
    lcd_cmmnd(0x94);
}

void lcdline4()
{
    lcd_cmmnd(0xd4);
}

void lcd_clr()
{
    lcd_cmmnd(0x01);
}
void displaystring(char *ptr)
{

while(*ptr)
{
    lcd_dataout(*ptr);
    ptr++;
}

}

void delay_1msc(void)
{
    lcd_data = 0xff;
    lcd_rs = 0;
    lcd_rw =1;
    while(lcd_busy==1)
    {
        lcd_en =0;
        _nop_();
        _nop_();
        _nop_();
        lcd_en=1;
    }
}
7. CONCLUSION

This project has combined the characteristics of the IEEE 802.15.4 standard with the maturing ZigBee specification in defining the wireless profiles for low data rate monitoring and control applications. The capabilities of both will result in the availability of a technology tailored specifically for the low power, low cost, and low complexity applications in the industrial, residential, and home today and in the future.

If the application strictly needs to communicate in a point-to-point or point-to-multipoint fashion ZigBee will be able to handle all the communications between your devices and will be simpler to implement. Flexible management of home appliances from anywhere in the house is possible using the ZigBee remote.
8. REFERENCES

- Homepage of ZigBee™ Alliance, http://www.zigbee.org/
- Muhammad Ali Mazidi “The 8051 Microcontroller and Embedded Systems”
- Myke Predko “Programming and customizing the 8051”
- www.atmel.com
- www.analogicgroup.com
- www.philips.com