# Chapter 1: Introduction to SMART HOME

Netuser.com/images/detailed/1/pgm-pickit3.jpg

Smart Home System is a project used to control any devices in home or in office or in other places can switch on or off. The goal of a smart home system on enabling those who live in them to control a large number of electronic devices easily and remotely. A simple smart home system might, for instance, turn on the kitchen lights and turn off the alarm when the garage door opener activates. Smart homes typically have many more complicated systems, but they all operate via the same principles.

Smart homes work via fairly simple systems: PIC and SMS. Normal home devices such as lights, entertainment systems, heaters, air conditioners, computers, security systems and radios are equipped with receivers. This PIC detects a certain signal initiated by the code SMS, which can be housed in a control device such a light switch or, most commonly, a remote control.

In order to design the smart home system in this project, this system can control by using Short Message Service (SMS). Nowadays, Short Message Service (SMS) is widely used as a form of data communication. It is about 2.4 billion active users which equals to 74% of mobile phone subscribers sending and receiving text messages on their phones. SMS is a communication application in Global System for Mobile communication (GSM) system. It allows interchange of short text messages between mobile telephone devices using standardized communication protocols.

The system in this project is design to receive the SMS from any mobile device to the GSM modem that connected to microcontroller. In order to prevent any occurrence of SMS likelihood control words, the sending SMS that contain control words should come between the specified codes that protocol between user of far mobile phone and the GSM modem that connected to the microcontroller. After the GSM modem which connected to the microcontroller receives the sent message, it sends this message to the microcontroller.

* 1. **Objective of Project**

As, the world around us is getting digitise, it becomes beneficial to automate the control of electronic appliances used in any environment such as home, industries, offices or any other. The main objective of this project is to design and develop an environment that is capable of controlling, home appliances using a wireless protocol-ZIGBEE along with GSM. We can monitor and control the status of appliances present at our home sitting from a far place. Living in such an environment makes it easy and convenient to control devices.

**Chapter 2: Study Phase**

**2.1 PIC32MX575F256L**

**2.1.1 Introduction**

PIC is a family of microcontrollers made by Microchip Technology. The original one was the PIC1650 developed by General Instruments. This device was called PIC for “Programmable Intelligent Computer” although it is now associated with “Programmable Interface Controller.” Microchip does not use PIC as an acronym. Instead they prefer the brand name PICmicro. Popular wisdom relates that PIC is a registered brand in Germany and Microchip is unable to use it internationally.

The PIC microcontrollers do not use the conventional von Neumann architecture but a different hardware design often referred to as Harvard architecture. Microcontrollers with Harvard architecture are also called "RISC Microcontrollers".

The PIC32MX Microcontroller Unit (MCU) is a complex system-on-a-chip that is based on a M4K™ core from MIPS® Technologies. M4K™ is a state-of-the-art 32-bit, low-power, RISC processor core with the enhanced MIPS32® Release 2 Instruction Set Architecture.

**2.1.2 Block Diagram**

 (i) Figure 2.1.2 Block Diagram of PIC32

The PIC32 employs the M4K® 32-bit core from MIPS Technologies which is based on Harvard architecture. It has separate Instruction and Data busses connected to the Bus Matrix. The core connects to the rest of the modules via Bus Matrix which is a high-speed switch. It establishes a point to point connection between modules such as the CPU core, USB and DMA connect to the SRAM, SPI, UART, etc., via the Bus Matrix and Peripheral Bus. The Bus Matrix runs at the same speed as the CPU. The PIC32 uses a 128-bit wide Flash memory that is specifically designed to increase the instruction throughput and improve overall CPU Performance. To further enhance the performance, the PIC32 employs a 128-bit Pre fetch Cache Module. This module can be programmed to look ahead and pre fetch the next 128-bits of instructions and store them in an on-chip cache memory. This module is the reason why the PIC32 can continue to provide high performance even when the CPU is running faster than Flash memory speed.

**2.1.3 Reason for Selecting PIC32M575F256L:**

* **Rich in peripherals**: The PIC microcontroller has many built in peripherals which can be utilised for various purposes. The 100 pins of PIC make it easier to use the peripherals as the functions are spread out over the pins. This makes it easier to decide what external devices to attach without worrying too much if there enough pins to do the job.
* **Reprogrammable controller:** The PIC32MX575F256L has 256kb flash memory which can be used to erase and rewrite the programs for the controller. Hence the devices can be re-programmed up to 10,000 times.
* **Low power consumption:** The controller works with a low power supply such as 3.3V DC. Power consumption will dictate the size and cost of the power supply circuit you will have to design. If battery operated, this parameter will dictate the

Size and cost of the battery, or vice versa, the life (hours of operation) of application.

* **Easy programming, cheap and reliable:** It is easy to program the PIC microcontroller in embedded C language or assembly level language.

**2.1.4 Key Features**

• Up to 1.5 DMIPS/MHz of performance

• Programmable pre fetch cache memory to enhance execution from Flash memory

• 16-bit Instruction mode (MIPS16e) for compact code

• Vectored interrupt controller with 63 priority levels

• Programmable User and Kernel modes of operation

• Atomic bit manipulations on peripheral registers (Single cycle)

• Multiply-Divide unit with a maximum issue rate of one 32 × 16 multiply per clock

• High speed Microchip ICD port with hardware-based non-intrusive data monitoring and

Application data streaming functions

• EJTAG debug port allows extensive third party debug, programming and test tools

Support.

• Instruction controlled power management modes

• Five stage pipelined instruction execution

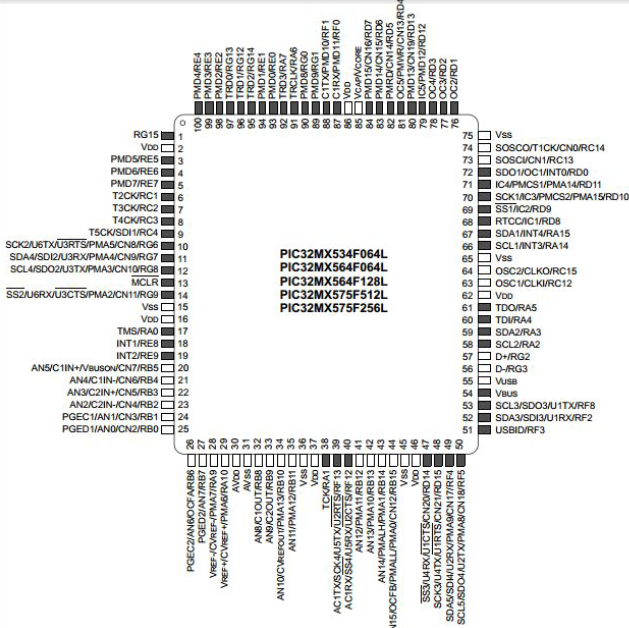
• Internal Code protection to help protect intellectual property

**2.1.5 PIC32 Peripheral Features**

Table 2.1.5 PIC32 Peripheral features

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Device  Name | pins | Prog.  Memory  (KB) | Data  Memory  (KB) | USB | Timers | UART | SPI | I2C | ADC | Comparators |
| PIC32MX575F256L | 100 | 256 | 64 | 1 | 5 | 6 | 4 | 5 | 16 | 2 |

**2.1.6 Pin Diagram of PIC32**



(ii) Figure 2.1.6 Pin diagram of PIC32

**2.2 SPI (Serial Peripheral Interface)**

**2.2.1 Introduction**

The SPI module is a synchronous serial interface that is useful for communicating with external peripherals and other microcontroller devices. These peripheral devices may be Serial EEPROMs, Shift registers, display drivers, Analog-to-Digital Converters, etc. The

PIC32 SPI module is compatible with Motorola® SPI and SIOP interfaces. The clock signal is provided by the master to provide synchronization. The clock signal controls when data can change and when it is valid for reading.

Since SPI is synchronous, it has a clock pulse along with the data. RS-232 and other

asynchronous protocols do not use a clock pulse, but the data must be timed very

accurately. Since SPI has a clock signal, the clock can vary without disrupting the data. The data rate will simply change along with the changes in the clock rate. This makes

SPI ideal when the microcontroller is being clocked imprecisely, such as by a RC oscillator.

Following are some of the key features of this module:

• Master and Slave modes support

• Four different clock formats

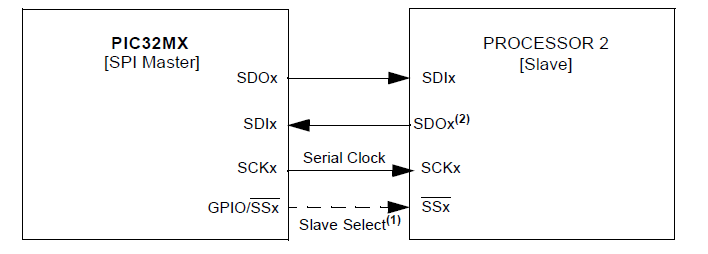
• Framed SPI protocol support

• User configurable 8-bit, 16-bit, and 32-bit data width

• Separate SPI shift registers for receive and transmit

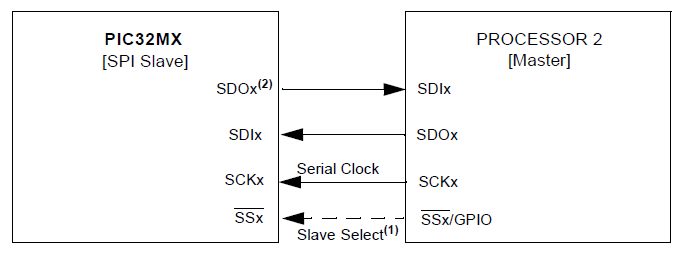
• Programmable interrupt event on every 8-bit, 16-bit, and 32-bit data transfer

**2.2.2 SPI Master-to-Slave Device Connection Diagram**



(iii) Figure 2.2.2 SPI Master-to-Slave Device Connection Diagram

**2.2.3 SPI Slave-to-Master Device Connection Diagram**



(iv) Figure 2.2.3 SPI Slave-to-Master Device Connection Diagram

SPI is a Serial Interface and uses the following signals to serially exchange data

with another device:

* SS - This signal is known as Slave Select. When it goes low, the slave device will

listen for SPI clock and data signals.

* SCK - This is the serial clock signal. It is generated by the master device and

controls when data is sent and when it is read.

* SDO - This is the Serial Data Output signal. SDO carries data out of a device.
* SDI - SDI is the Serial Data Input line. It carries data into a device.

**2.2.4 SPI WORKING**

SPI is a Master-Slave protocol. Only the master device can control the clock line, SCK.

No data will be transferred unless the clock is manipulated. All slaves are controlled by the clock which is manipulated by the master device. The slaves may not manipulate the clock.

The SSP configuration registers will control how a device will respond to the clock input.

SPI is a Data Exchange protocol. As data is being clocked out, new data is also

being clocked in.

When one “transmits” data, the incoming data must be read before attempting to

transmit again. If the incoming data is not read, then the data will be lost and the

SPI module may become disabled as a result. Always *read* the data after a transfer

has taken place, even if the data has no use in your application.

Data is always “exchanged” between devices. No device can just be a “transmitter”

or just a “receiver” in SPI. However, each device has two data lines, one for input

and one for output.

These data exchanges are controlled by the clock line, SCK, which is controlled by

the master device.

Often a slave select signal will control when a device is accessed. This signal must

be used for when more than one slave exists in a system, but can be optional when

only one slave exists in the circuit. As a general rule, it should be used.

This signal is known as the SS signal and stands for “Slave Select.” It indicates to a

slave that the master wishes to start an SPI data exchange between that slave device

and itself. The signal is most often active low, so a low on this line will indicate the

SPI is active, while a high will signal inactivity.

It is often used to improve noise immunity of the system. Its function is to reset the

SPI slave so that it is ready to receive the next byte.

SPI creates a data loop between two devices. Data leaving the master exits on the

SDO (serial data output) line. Data entering the master enters on the serial data

input, SDI line.

A clock (SCK), is generated by the master device. It controls when and how

quickly data is exchanged between the two devices.

SS, allows a master device to control when a particular slave is being addressed.

This allows the possibility of having more than one slave. When the SS signal goes low at a slave device, only that slave is accessed by SPI.

**2.3 UART (Universal Asynchronous Receiver Transmitter)**

**2.3.1 Introduction**

It is Universal Asynchronous Receiver Transmitter, is one of the most used serial protocols. It's almost as old as I am, and very simple.

Most controllers have hardware UART on board. It uses a single data line for transmitting and one for receiving data. Most often 8-bit data is transferred, as follows: 1 start bit, low level, 8 data bits, 1 stop bit, and high level.

The low level start bit and high level stop bit mean that there's always a high to low transition to start the communication.

That's what describes UART. No voltage level, so you can have it at 3.3 V or 5 V, whichever your microcontroller uses. Note that the microcontrollers which want to communicate via UART have to agree on the transmission speed, the bit-rate, as they only have the start bit's falling edge to synchronize. That's called asynchronous communication.

In synchronous there's not only data, but also a clock transmitted. With each bit a clock pulse tells the receiver it should latch that bit. Synchronous protocols either need a higher bandwidth, like in the case of Manchester encoding, or an extra wire for the clock, like SPI and I2C.

**2.3.2 Objective of UART**

INPUT: Parallel ASCII data to be transmitted, serial data in.

OUTPUT: Serial data out, Parallel ASCII character received.

Extension Done: - Flexible Baud Rate.

**2.3.3 UART functional elements**

The UART module consists of these important hardware elements:

• Baud Rate Generator

• Asynchronous transmitter

• Asynchronous receiver and IrDA encoder/decoder

**2.3.4 Features of UART**

The primary features of the UART module are:

• Full-duplex, 8-bit or 9-bit data transmission

• Even, Odd or No Parity options (for 8-bit data)

• One or two Stop bits

• Hardware auto-baud feature

• Hardware flow control option

• Fully integrated Baud Rate Generator (BRG) with

16-bit prescaler

• Baud rates ranging from 76 bps to 20 Mbps at 80 MHz

• 8-level deep First-In-First-Out (FIFO) transmit data buffer

• 8-level deep FIFO receive data buffer

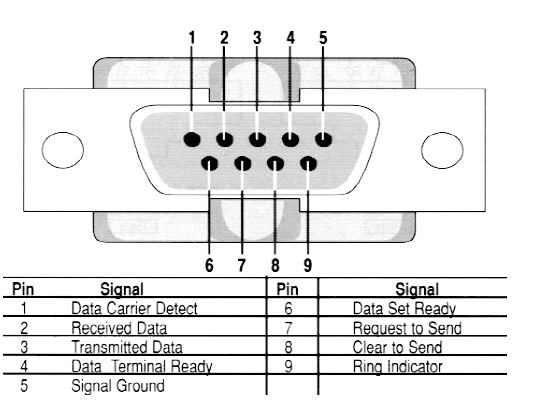
• Parity, framing and buffer overrun error detection

• Support for interrupt-only on address detect (9th bit = 1)

• Separate transmit and receive interrupts

**2.3.5 Serial connection**

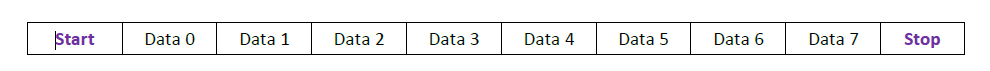
Serial Port used for UART is RS-232. It is a nine pin connector with each pin assigned with different functionality.



(v) Figure 2.3.5 Serial Connection

**2.3.6 Working of UART**

UART is a device that has the capability to both receive and transmit serial data. UART exchanges text data in an American Standard Code for Information Interchange (ASCII) format in which each alphabetical character is encoded by 7 bits and transmitted as 8 data bits. For transmission the UART protocol wraps this 8 bit sub word with a start bit in the least significant bit (LSB) and a stop bit in the most significant bit (MSB) resulting in a 10 bit word format.



UART transmitter controls transmission by fetching a data word in parallel format and directing the UART to transmit it in a serial format. Likewise, the Receiver must detect transmission, receive the data in serial format, strip of the start and stop bits, and store the data word in a parallel format.

Since the UART is asynchronous in working, the receiver does not know when the data will come, so receiver generate local clock in order to synchronize to transmitter whenever start bit is received. Asynchronous transmission allows data to be transmitted without the sender having to send a clock signal to the receiver. The transmitter and receiver agree on timing parameters in advance and special bits are added to each word which is used to synchronize the sending and receiving units.

When a word is given to the UART for Asynchronous transmission, a bit called the “Start Bit” is added to the beginning of each word that is to be transmitted. The Star Bit is used to alert the receiver that a word of data is about to be sent, and to force the clock in the receiver into synchronization with the clock in the transmitter. After the Start Bit, the individual bits of the word of data are sent, with the Least Significant Bit (LSB) being sent first. Each bit in the transmission is transmitted for exactly the same amount of time as all of the other bits, and the receiver “looks” at the wire at approximately halfway through the period assigned to each bit to determine if the bit is a 1 or a 0. For example, if it takes two seconds to send each bit, the receiver will examine the signal to determine if it is a 1 or a 0 after one second has passed, then it will wait two seconds and then examine the value of the next bit, and so on. Then at least oneStop Bit is sent by the transmitter. Because asynchronous data is “self-synchronous”, if there is no data to transmit, the transmission line can be idle.

**2.4 Comparison between UART & SPI**

|  |  |  |
| --- | --- | --- |
|  | **Synchronous** | **Asynchronous** |
| **Peripheral** | **SPI** | **UART** |
| Max bit rate | 20 Mbit/s | 500 kbit/s |
| Max bus size | Limited by number of  pins | Point to point  (RS232), 256 devices |
| Pros | Simple, low cost, high  speed | Longer distance  (use transceivers  for improved noise  immunity) |
| Cons | Single master, short  distance | Requires accurate  clock frequency |
| Typical application | Direct connection  to many common  peripherals on same  PCB | Interface with  terminals, personal  computers, and other  data acquisition  systems |
| Examples | Serial EEPROMs  (25CXXX series),  MCP320X A/D  converter, ENC28J60  Ethernet controller,  MCP251X CAN  controller . . . | RS232, RS422, RS485,  LIN bus, MCP2550  IrDA interface . . . |

Table 2.4 Comparison between UART & SPI

**2.5 ZIGBEE**

**2.5.1 What is Zigbee?**

Zigbee is a Technological Standard Created for Control and Sensor Networks based on the IEEE 802.15.4 specification for wireless personal area network .It is a New wireless technology that has application in various fields. ZigBee benefits are low cost and Range and obstruction issues avoidance. The main features of this standard are network flexibility, low cost, very low power consumption, and low data rate in an adhoc self-organizing network among inexpensive fixed, portable and moving devices.

**2.5.2 Origin of word Zigbee**

The erratic, zig-zagging patterns of bees between flowers while collecting pollens from the flowers Symbolizes communication between nodes in a mesh network of Zigbee network. The network components are analogous to queen bee, drones, worker bees.

Process on ZigBee-style networks began in 1998, when many engineers realized that both

WiFi and Bluetooth were going to be unsuitable for many applications. In particular, a need for self-organizing ad-hoc digital radio networks arose.

ZigBee is very low cost, low power consumption, two ways, wireless communication protocol. It adopts IEEE 802.15.4, as its lower protocol layers. The relative organization of the IEEE radio with respect to the ZigBee functionality. Wireless systems mostly use cell phone-style radio links, using point-to-point or point-to- multipoint transmission. These traditional wireless formats have drawbacks like rigid structure, signal dropping and meticulous planning requirements.

**2.5.3 Zigbee Alliance**

It is association of companies working together to enable reliable ,cost effective, low power wirelessly networked ,monitoring and control products based on an open global standard.

**2.5.4 Need of Zigbee**

ZigBee was created to satisfy the market’s need of a standards-based wireless network that is cost-effective, supports low data rates, low power consumption, secure and reliable.

We compare it to its closest competitor, Bluetooth. If we want to build a remote battery powered Bluetooth node, we need at least 250K of memory for the code and stack, and transmission speed of 720KB/s up to range of approximately 10 meters , if there are no cordless phones, VCRs etc. The battery life will be 7 days. Now, compare that to ZigBee. Though it is a lower-speed wireless protocol that’s targeted at transmission speeds of 20-250KB/s, it has a transmission range of over 50 meters. Battery life is 2 years and 32K of system resources is required. This is simple, effective, and very practical.

ZigBee is the only wireless standards-based technology that addresses the unique needs of remote monitoring control, and sensory network applications, enables broad-based deployment of wireless networks with low cost, low power solutions.

**2.5.5 Wireless sensor networking is one of the most exciting technology markets today**

They say that over the next five to ten years, wireless sensors will have a significant

Impact on almost all major industries as well as our home lives. Broadly, this technology

market includes application segments such as automated meter reading, home automation,

building automation, container security/tracking, and many others.

The main motivations for migrating these products to wireless communications are threefold:

**1.** Installation cost - The cost of running wires in a typical building automation project

in an existing facility can be as high as 80project cost.

**2.** Maintenance - It is easier to configurea hot-water heater controller with a hand-held remote than a keypad in the closet.

**3.** New markets - Eliminating the wire opens new markets that were previously unavailable to wired products.

Zigbee was introduced as an alternative to Bluetooth for devices with low power

consumption requirements and applications of lower bit rates. Although products based on the Bluetooth standard are often capable of operating at greater distances, the targeted

operational area is the one around an individual, (e.g. within a 10 meters diameter). Bluetooth utilizes a short range radio link that operates in the 2.4 GHz industrial scientific and medical (ISM) band similar to WLAN. However, the radio link in Bluetooth is based on frequency hop spread spectrum. Although at any point in time, the Bluetooth signal occupies only 1MHz, the signal changes the centre frequency (or hops) deterministically at a rate of 1600Hz. Bluetooth hops over 79 centre frequencies, so over time the Bluetooth signal actually occupies 79MHz. The new short range, low power, low rate wireless networking protocol, Zigbee, complements the high data rate technologies such as WLAN and open the door for many new applications. This standard operates at three bands, the 2.4 GHz band with a maximum rate of 250 kbps, the 915 MHz band with a data rate of 40 kbps, and the 868 MHz band with a data rate of 20 kbps. While Bluetooth devices are better suited for fairly high rate sensor and voice applications, Zigbee is better suited for low rate sensors and devices used for control applications that do not require high data rate but must have long battery life, low user interventions and mobile topology. Some of these applications are in the fields of medicine

ZigBee is a low-cost, low-power, wireless mesh networking proprietary standard. The lowcost allows the technology to be widely deployed in wireless control and monitoring applications, the low power-usage allows longer life with smaller batteries, and the mesh networking provides high reliability and larger range. The ZigBee Alliance, the standards body that defines ZigBee, also publishes application profiles that allow multiple OEM vendors to create interoperable products. The current list of application profiles either published or in the works are: Home Automation ZigBee Smart Energy 1.0/2.0 Commercial Building Automation Telecommunication Applications Personal, Home, and Hospital Care Toys.

ZigBee operates in the industrial, scientific and medical (ISM) radio bands; 868 MHz in

Europe, 915 MHz in the USA and Australia, and 2.4 GHz in most jurisdictions worldwide. The technology is intended to be simpler and less expensive than other WPANs such as Bluetooth. ZigBee chip vendors typically sell integrated radios and microcontrollers with between 60K and 128K flash memory, such as the Jennic JN5148, the Freescale MC13213, the Ember EM250, the Texas Instruments CC2430, the Samsung Electro-Mechanics ZBS240 and the Atmel ATmega128RFA1. Radios are also available stand-alone to be used with any processor or microcontroller. Generally, the chip vendors also offer the ZigBee software stack, although independent ones are also available. Because ZigBee can activate (go from sleep to active mode) in 15 msec or less, the latency can be very low and devices can be very responsive particularly compared to Bluetooth wake-up delays, which are typically around three seconds. BecauseZigBee can sleep most of the time, average power consumption can be very low, resulting in long battery life. The first stack release is now called ZigBee 2004. The second stack release is called ZigBee 2006, and mainly replaces the MSG/KVP structure used in 2004 with a “cluster library”. The 2004 stack is now more or less obsolete. Citation needed] ZigBee2007, now the current stack release, contains two stack profiles, stack profile 1 (simply called ZigBee), for home and light commercial use, and stack profile 2 (called ZigBee Pro). ZigBee Pro offers more features, such as multi-casting, many-to-onerouting and high security with Symmetric-Key Key Exchange (SKKE), while ZigBee (stackprofile 1) offers a smaller footprint in RAM and flash. Both offer full mesh networking and work with all ZigBee application profiles.[citation needed] ZigBee 2007 is fully backward compatible with ZigBee 2006 devices: A ZigBee 2007 device may join and operate on a Zig-Bee 2006 network and vice versa. Due to differences in routing options, ZigBee Pro devices must become non-routing ZigBee End-Devices (ZEDs) on a ZigBee 2006 or ZigBee 2007 network, the same as ZigBee 2006 or ZigBee 2007 devices must become ZEDs on a ZigBee Pro network. The applications running on those devices work the same, regardless of the stack profile beneath them.

**2.5.6 Zigbee MRF24J40**

**(**IEEE 802.15.4™ 2.4 GHz RF Transceiver)

2.5.6.1 Features of MRF24J40

IEEE 802.15.4™ Standard Compliant RF Transceiver

Simple, 4-Wire Serial Peripheral Interface (SPI)

Integrated 20 MHz and 32.768 kHz Crystal Oscillator Circuitry

Low-Current Consumption:

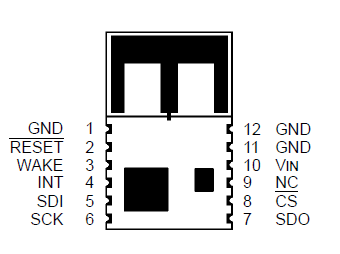
* RX mode: 19 mA (typical)
* TX mode: 23 mA (typical)
* Sleep: 2 μA (typical)

ISM Band 2.405-2.48 GHz Operation

Data Rate: 250 kbps (IEEE 802.15.4); 625 kbps (Turbo mode)

Operating Voltage: 2.4-3.6V (3.3V typical)

**2.5.6.2 Pin Diagram**



**(vi)** Figure 2.5.6.2 Pin Diagram

**2.5.6.3 Pin Description**

Table 2.5.6.3 Pin description of Zigbee MRF24J40

|  |  |  |  |
| --- | --- | --- | --- |
| **Pin** | **symbol** | **type** | **description** |
| **1** | GND | POWER | ground |
| **2** | RESET | DI | Global hardware Reset pin |
| **3** | WAKE | DI | External wake-up trigger |
| **4** | INT | DO | Interrupt pin to microcontroller |
| **5** | SDI | DI | Serial interface data input |
| **6** | SCK | DI | Serial interface clock |
| **7** | SDO | DO | Serial interface data output from MRF24J40 |
| **8** | CS | DI | Serial interface enable |
| **9** | NC | ---- | No connection |
| **10** | VIN | POWER | Power Supply |
| **11** | GND | GND | Ground |
| **12** | GND | GND | Ground |

**2.5.6.4 Uses**

ZigBee protocols are intended for use in embedded applications requiring low data rates andlow power consumption. ZigBee’s current focus is to define a general-purpose, inexpensive, self-organizing mesh network that can be used for industrial control, embedded sensing, medical data collection, smoke and intruder warning, building automation, home automation, etc. The resulting network will use very small amounts of power individual devices must have a battery life of at least two years to pass ZigBee certification.

Typical application areas includes-

Home Entertainment and Control Smart lighting, advanced temperature control, safety and security, movies and music, Home Awareness Water sensors, power sensors, energy monitoring, smoke and fire detectors, smart appliances and access sensors

Mobile Services m-payment, m-monitoring and control, m-security and access control, HealthCare and tele-assist Commercial Building Energy monitoring, HVAC, lighting,

access control Industrial Plant Process control, asset management, environmental management, energy management, industrial device control.

**2.6 GSM(Global System for Mobile Communication)**

**2.6.1 GSM INTRODUCTION**

A GSM modem is a wireless modem that works with a GSM wireless network. A wireless modem behaves like a dial-up modem. The main difference between them is that a dial-up modem sends and receives data through a fixed telephone line while a wireless modem sends and receives data through radio waves.

A GSM modem can be an external device or a PC Card / PCMCIA Card. Typically, an external GSM modem is connected to a computer through a serial cable or a USB cable. A GSM modem in the form of a PC Card / PCMCIA Card is designed for use with a laptop computer. It should be inserted into one of the PC Card / PCMCIA Card slots of a laptop computer. Like a GSM mobile phone, a GSM modem requires a SIM card from a wireless carrier in order to operate. Computers use AT commands to control modems. Both GSM modems and dial-up modems support a common set of standard AT commands. A GSM modem is just like a dial-up modem.

In addition to the standard AT commands, GSM modems support an extended set of AT commands. These extended AT commands are defined in the GSM standards. The

number of SMS messages that can be processed by a GSM modem per minute is very

low. They are only about six to ten SMS messages per minute.

**2.6.2 SIM900**

**2.6.2.1 Introduction**

SIM900 is a quad-band GSM/GPRS engine that works on frequency GSM 850MHz, EGSM 900MHz, DCS 1800MHz, PCS 1900 MHz. SIM900 features GPRS multi-slot class 10/class 8(optional) and supports the GPRS coding schemes CS-1, CS-2, CS-3,

CS-4.



(viii) Figure 2.6.2.1 Top view SIM900

With a tiny configuration of 24mmx24mmX3mm, SIM900 can meet almost all the space requirements of our applications, such as M2M, smart phone, PDA and other mobile devices.

SIM900 is designed with power saving technique so that the current consumption is as low as 1.5mA in SLEEP mode.

The SIM900 is a integrated with TCP/IP protocol; extended TCP/IP AT commands are develop for customers to use the TCP/IP protocol easily; which is very useful for those data transfer application.

**2.6.2.2 SIM900 Key Features**

Table 2.6.2.2 SIM900 key features

|  |  |
| --- | --- |
| **Features** | **Implementations** |
| Power supply | Single voltage 3.5-4.5 V |
| Power saving | Typical current consumption is 1.5mA in SLEEP mode. |
| Frequency Bands | SIM900 quad-band: GSM 850MHz, EGSM 900MHz, DCS 1800MHz, PCS 1900 MHz.  The SIM900 can search the 4 frequency bands automatically. |
| GSM Class | Small MS |
| Transmitting power | Class4(2W) at GSM 850 and EGSM 900  Class1(1W) at DCS 1800 and PCS 1900 |
| SMS | MT, MO, CT, text and PDU mode  SMS Storage: SIM card |
| SIM Interface | Support SIM card: 1.8V,3V |
| SIM Application Toolkit | Support SAT class3,GSM 11.14 release 99 |
| Real time clock | Implemented |
| Timer function | Programmable via AT command |
| Physical characteristic | Size : 24mmX24mmX3mm  Weight:3.4gm |

**2.6.2.3 SIM900 Functional Diagram**

The following figure shows a functional diagram of the SIM900.

* The GSM base band engine
* Flash and SRAM
* The GSM radio frequency part
* The antenna interface
* The other interfaces

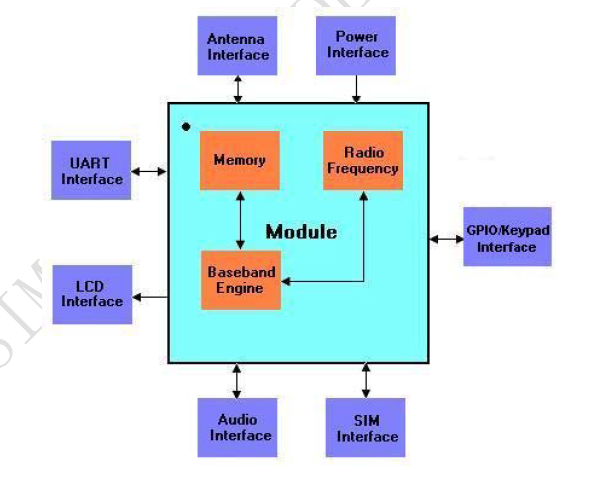


Figure 2.6.2.3 SIM900 Functional Diagram

**2.6.2.4 SIM900 Evaluation Board**

Here we use the SIMCOM Evaluation Board that interfaces the SIM900 directly with appropriate power supply, SIM card holder, RS232 Serial port, handset port, line in port, antenna and all GPIO of the SIM900.

****

(vx) Figure 2.6.2.4 SIM900 Evaluation Board

**2.6.2.5 Serial Interfaces**

The SIM900 is provide two unbalanced asynchronous serial ports. One is the serial port and other is a debug port.

Table 2.6.2.5 Pin definition of the serial interfaces

|  |  |  |
| --- | --- | --- |
| **Name** | **pin** | **Function** |
| **DTR** | **3** | **Data terminal ready** |
| **RI** | **4** | **Ring indicator** |
| **DCD** | **5** | **Data carrier detection** |
| **DCR** | **6** | **Data set ready** |
| **CTS** | **7** | **Clear to send** |
| **RTS** | **8** | **Request to send** |
| **TXD** | **9** | **Transmit data** |
| **RXD** | **10** | **Receive data** |
| **DBG\_RXD** | **11** | **Receive data** |
| **DBG\_TXD** | **12** | **Transmit data** |

(x) Figure 2.6.2.5 Serial Interface

****

TXD, RXD --- Data Line

CTS, RTS --- Hardware Flow Control Line

DTR --- Sleep Mode Control Line

DCD --- Data Mode

RI --- Incoming Call, SMS, Arouse host.

DSR --- Reserve

**2.6.2.6 AT commands**

AT command set implemented by SIM900 is a combination of GSM 07.05, GSM 07.07 and the AT commands developed by the SIMcom.

Table 2.6.2.6 Types of AT command

|  |  |
| --- | --- |
| Test command | **AT+<x>=?** |
| Read command | **AT+<x>?** |
| Write command | **AT+<x>=<…..>** |
| Execution command | **AT+<x>** |

**2.6.2.7 SMS commands**

Table 2.6.2.7 SMS commands

|  |  |  |
| --- | --- | --- |
| **Demonstration** | **Syntax** | **Response** |
| Set SMS system into text mode. | **AT+CMGF = 1** | OK |
| Send an SMS to myself. | **AT+CSCS = “GSM”** | OK |
| Unsolicited notification of the message arriving. | **AT+CMTI** | +CMTI:”SM”,1 |
| Read SMS that has just arrived. | **AT+CMGR** | OK |
| Send another SMS to myself. | **AT+CMGS=”8973xxxx”**  **> Test again Ctrl+Z** | +CMGS:35  OK |
| List all messages. | **AT+CMGL=”ALL”** | OK |
| Delete an SMS message | **AT+CMGD=1** | OK |

**2.7 RELAY**

**2.7.1 What is a relay?**

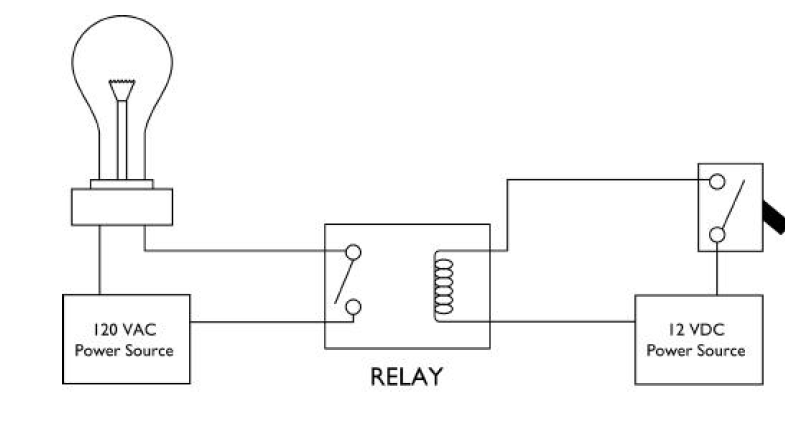
A relay is usually an electromechanical device that is actuated by an electrical current. The current flowing in one circuit causes the opening or closing of another circuit. Relays are like remote control switches and are used in many applications because of their relative simplicity, long life, and proven high reliability. Relays are used in a wide variety of applications throughout industry, such as in telephone exchanges, digital computers and automation systems. Highly sophisticated relays are utilized to protect electric power systems against trouble and power blackouts as well as to regulate and control the generation and distribution of power. In the home, relays are used in refrigerators,

washing machines and dishwashers, and heating and air-conditioning controls. Although

relays are generally associated with electrical circuitry, there are many other types, such as pneumatic and hydraulic. Input may be electrical and output directly mechanical, or vice versa.

**2.7.2 Working of Relay**

All relays contain a sensing unit, the electric coil, which is powered by AC or DC current. When the applied current or voltage exceeds a threshold value, the coil activates the armature, which operates either to close the open contacts or to open the closed contacts. When a power is supplied to the coil, it generates a magnetic force that actuates the switch mechanism. The magnetic force is, in effect, relaying the action from one circuit to another. The first circuit is called the control circuit; the second is called the load circuit.

****

(xi) Figure 2.7.2 Circuit diagram of RELAY

There are three basic functions of a relay: On/Off Control, Limit Control and Logic Operation.

***On/Off Control*:** Example: Air conditioning control, used to limit and control a “high power load, such as a compressor

***Limit Control*:** Example: Motor Speed Control, used to disconnect a motor if it runs slower or faster than the desired speed.

***Logic Operation*:** Example: Test Equipment, used to connect the instrument to a number of testing points on the device under test.

**2.7.3 JQC-3FC/T73 model 12VDC PCB MOUNTED SUGAR CUBE RELAY**

****

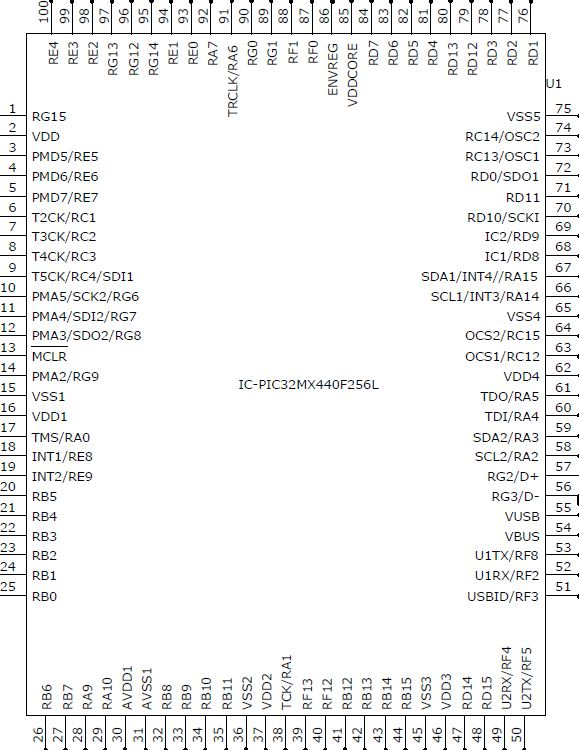
(xii) Figure 2.7.3 JQC-3FC/T73 12VDC

**2.7.3.1 Features of relay**

* Max. switching current:7a, 10a
* Max. switching voltage:28v dc/ 250v
* AC dielectric strength vr.m.s: between open contacts =750vac;
* Between coil and contacts =1000vac;
* Between contacts form =1000vac;
* Ambient temperature: -40-+85oc;
* Operation/release time:=10/8ms
* Contact capacity: 10a 125v, 7a 250v

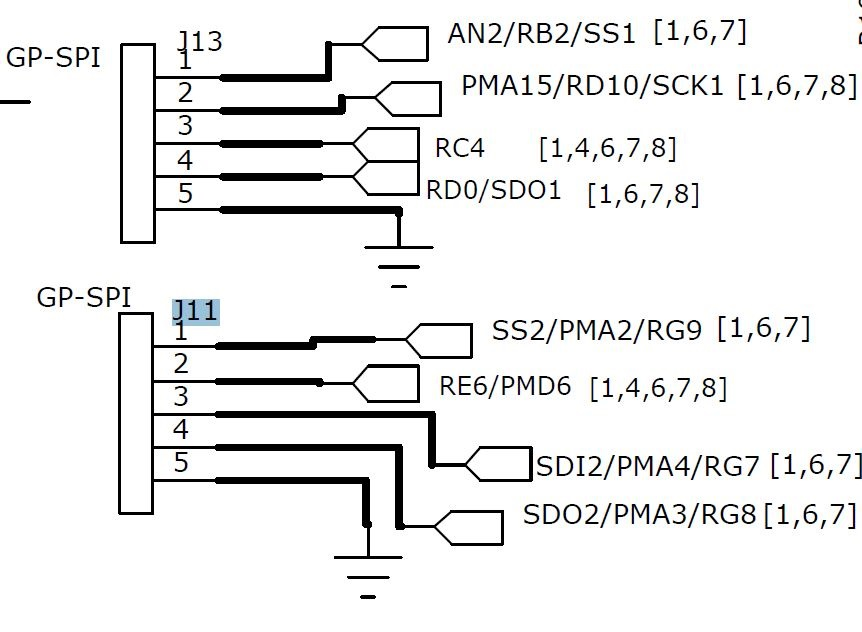
**Chapter 3: Hardware Design**

**3.1 PIC32 Hardware**

****

(xiii) Figure 3.1 Schematic PIC32

**3.2 SPI(Serial Pherifal Interface)**

****

**(** Figure 3.2 Schematic of SPI

**3.2.1 SPI Interfacing with PIC32 Master to Slave**

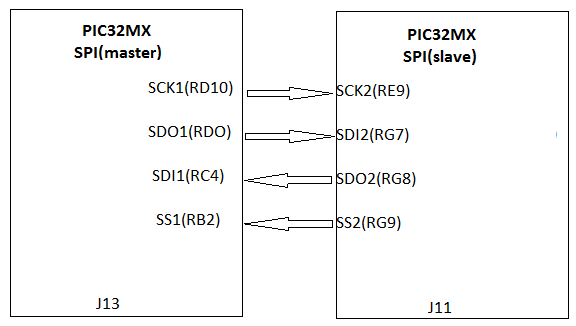
****

Figure 3.2.1 SPI Interfacing with PIC32 Master to Slave

**3.2.2 SPI Interfacing with PIC32 Slave to Master**

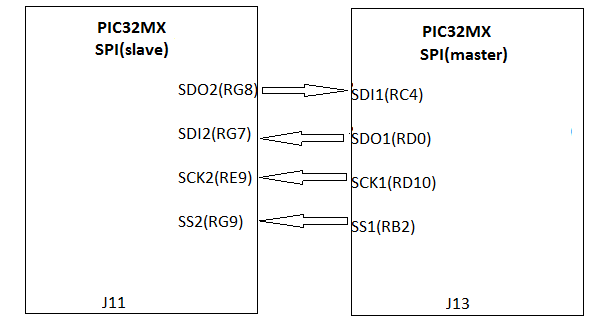
****

Figure 3.2.2 SPI Interfacing with PIC32 Slave to Master

**3.3 ZIGBEE**

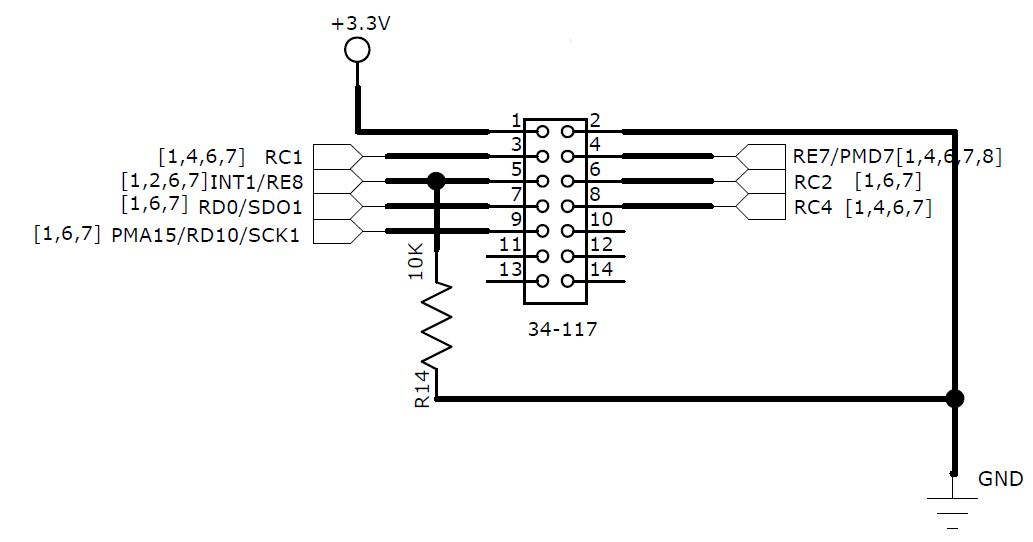
****

Figure 3.3 Schematic of Zigbee

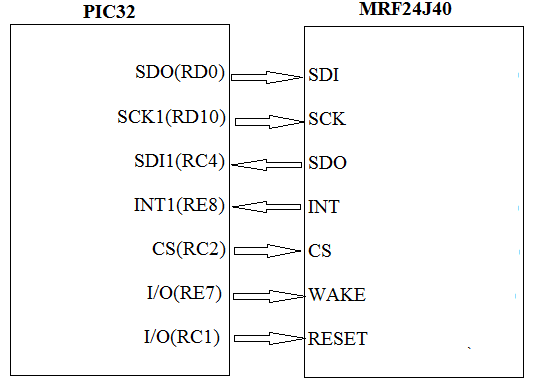
****

Figure 3.3 Interfacing PIC32 with Zigbee(MRF24J40)

**3.5 GSM**

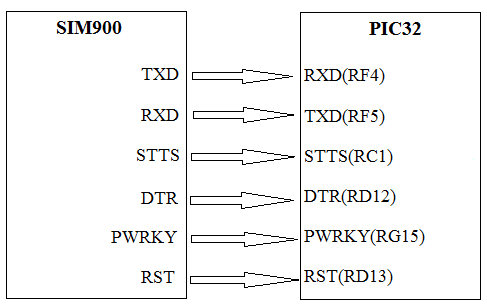
****

Figure 3.4 Interfacing PIC32 with GSM module

**3.6 LCD**

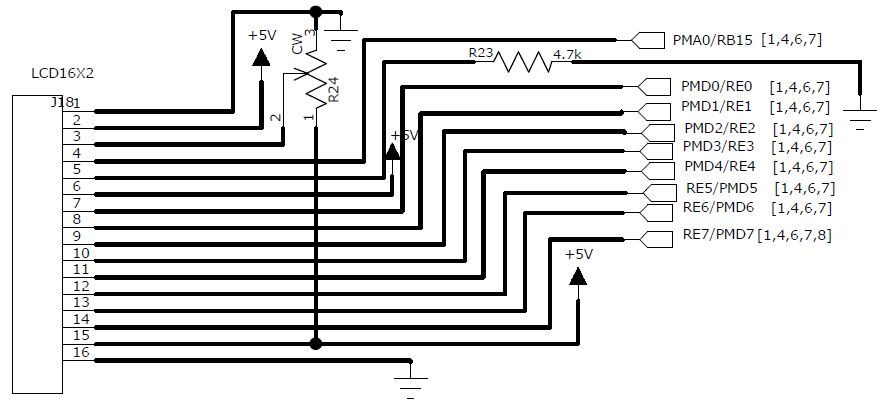
****

Figure 3.5 Interfacing PIC32 with LCD

**3.7 USB Power Supply**

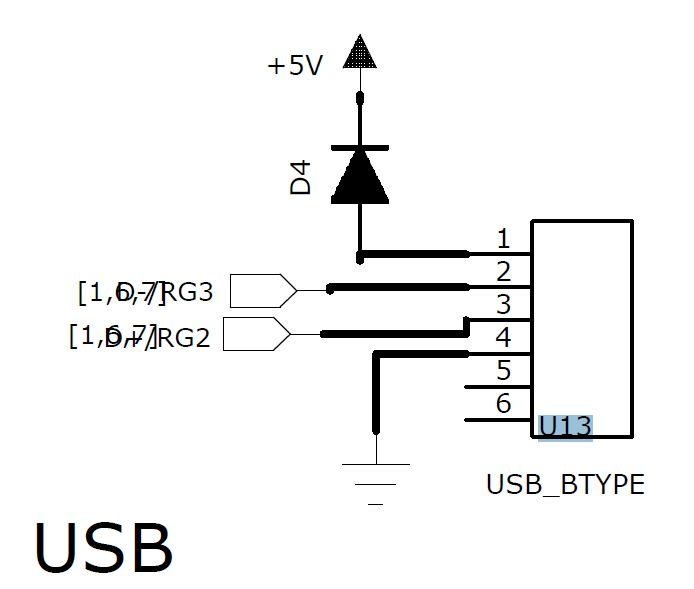
****

Figure 3.6 Interfacing PIC32 with USB

**Chapter 4: Software**

**4.1 MPLAB IDE**

**4.1.1 Introduction**

MPLAB Integrated Development Environment (IDE) is a comprehensive editor, project manager and design desktop for application development of embedded designs using Microchip PIC MCUs and dsPIC DSCs.The initial use of MPLAB IDE is how to make projects, edit code and test an application.

MPLAB® X IDE is a software program that runs on a PC develop application s for microchip microcontroller & digital signal controllers. Microchip offers programmers/debuggers under the MPLAB and PICKit series.

* + 1. **MPLAB® X IDE Features**
* Create and edit source code using the built-in editor.
* Assemble, compile and link source code.
* Debug the executable logic by watching program flow with the built-in simulator or in real time with in-circuit emulators or in-circuit ­debuggers.
* Make timing measurements with the simulator or emulator.
* View variables in Watch windows.
* Program firmware into devices with device programmers (for details, consult the user’s guide for the specific device programmer).

**4.1.3 Steps for create a project**

All projects will have these basic steps:

|  |  |
| --- | --- |
| 1. | Select Device |

The capabilities of MPLAB IDE vary according to which device is selected. Device selection should be completed before starting a project.

|  |  |
| --- | --- |
| 2. | Create Project |

MPLAB IDE Project Wizard will be used to Create a Project.

|  |  |
| --- | --- |
| 3. | Select Language Tools |

In the Project Wizard the language tools will be selected. For this tutorial, the built-in assembler and linker will be used. For other projects, one of the Microchip compilers or other third party tools might be selected.

|  |  |
| --- | --- |
| 4. | Put Files in Project |

Two files will be put into the project, a template file and a linker script. Both of these files exist in sub-folders within the MPLAB IDE folder. It is easy to get started using these two files.

|  |  |
| --- | --- |
| 5. | Create Code |

Some code will be added to the template file to send an incrementing value out an I/O port.

|  |  |
| --- | --- |
| 6. | Build Project |

The project will be built – causing the source files to be assembled and linked into machine code that can run on the selected PIC MCU.

|  |  |
| --- | --- |
| 7. | Test Code with Simulator |

Finally, the code will be tested with the simulator.

**4.2 Boot loader**

**4.2.1 Introduction**

Many of the higher end flash based PICs can also self-program (write to their own program memory), a process known as boot loading. Demo boards are available with a small boot loader factory programmed that can be used to load user programs over an interface such as RS232 or USB, thus obviating the need for a programmer device.



Figure 4.2.1 Boot Loader

After programming the boot loader onto the PIC, the user can then reprogram the device using RS232 or USB, in conjunction with specialized computer software. The advantages of a boot loader over ICSP is faster programming speeds, immediate program execution following programming, and the ability to both debug and program using the same cable.

**4.3 Hyper Terminal**

This is a built in interface in Windows (therefore, could be found in DHD lab) which sends and receives data through the serial port. It has the option of sending through port COM1 and COM2 and also has flexible baud rate.

Hyper Terminal used in transmission and reception of UART.

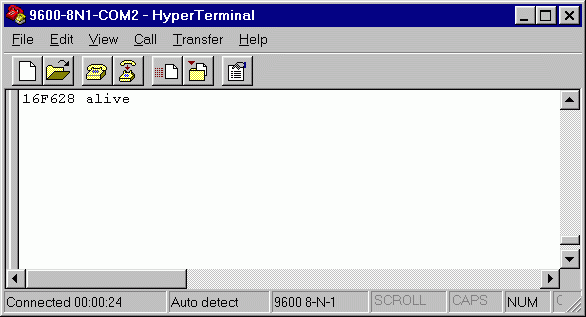
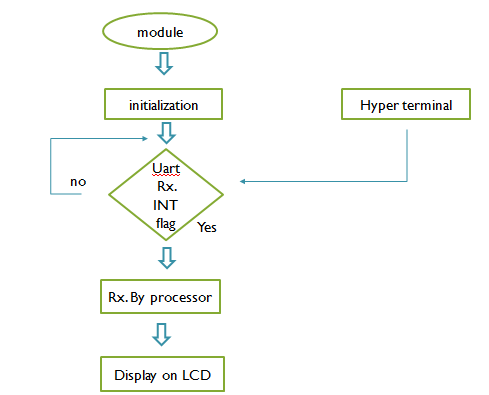


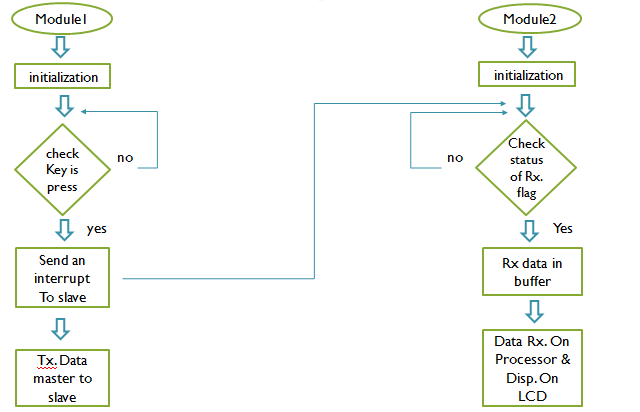
Figure 4.3 window of Hyper Terminal

**4.4 Software logic**

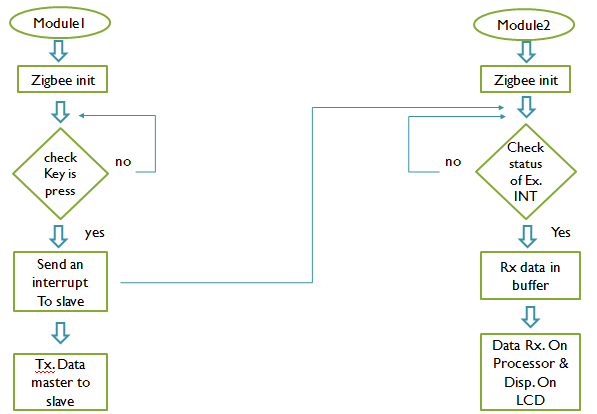
**4.4.1 flow chart of UART**

****

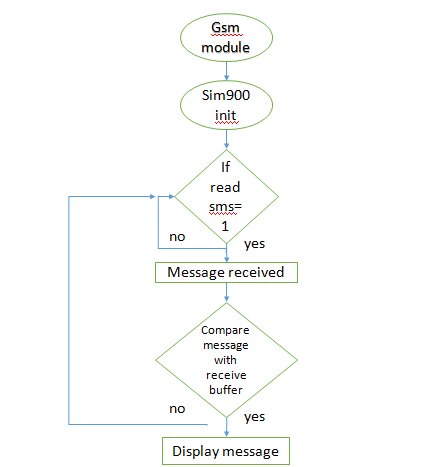
**4.4.2 SPI**

****

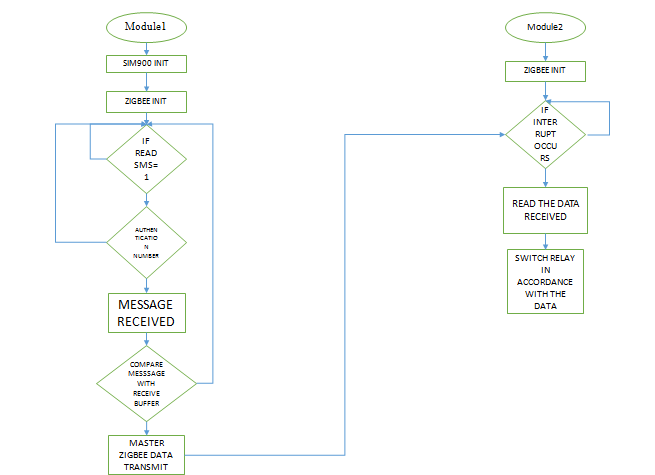
**4.4.3 ZIGBEE**

****

**4.4.4 GSM**

****

**4.4.5 Working of Smart Home**

****

**Chapter 5: Limitation and Future Enhancement**

**5.1 Limitations**

**. Costs:** It’s no secret that installing a home automation system can be quite costly. But, it all depends on the equipment you wish to have installed. Remember, the more advanced system you wish to have in your home the more expensive it will be.

**2. Human Error:** If the equipment is not handled & installed safely, this can lead to the equipment being damaged, and the risk of the system crashing is high.

**3.Reliability:** This occurs on a very rare occasions, depending on the age of the equipment it can have an effect on the system, but otherwise the technology in the home automation systems are all up to date.