A Novel Approach to Remote Sensing and Control

RAHUL PANDHI
Electronics and Communication
Delhi University
A-5/B, 305 Paschim Vihar
New Delhi
INDIA

MAYANK KAPUR
Manufacturing Process and Automation
Delhi University
408, Bhera Enclave
New Delhi
INDIA

SHWETA BANSAL
Electronics and Communication
Delhi University
392, Urban Estate
Kurukshetra
INDIA

ASOK BHATTACHRYA
Electronics and Communication
Delhi University
Delhi College of Engineering
New Campus
New Delhi
INDIA

Abstract: - This paper examines the application of embedded systems as an interfacing medium for mobile phone and a multitude of hardware devices. This would facilitate extension of the applications of a mobile phone to be used as a remote controlling and sensing device, so as to provide means to control different hardware devices and at the same time to provide vital feedback to the users in response to their queries through Short Message Service (SMS) from any place in the world. The response given by the mobile phone can be critical to the extent of being life saving as in the case of a fire out-break being reported and vital such as providing the position of a car in case of a theft. A micro-controller, programmed according to the required application, can be used for interfacing the mobile phone with the desired hardware. Working on the same lines, interface of mobile phone with a computer will facilitate many important applications such as data base management and computer peripheral control through a mobile phone from a remote place.

Key-Words: - FBUS, SMS, Remote Controlling, Sensing, TPDU, VP, SMSC

1. Introduction

Increased competition and busy life schedule has led to a comfort driven market. This necessitates long distance controlling of devices. Also it has become imperative for humans to get a feedback regarding vital information such as critical physical parameters and security status, from remote setups. In order to serve the above purpose, short messaging service (SMS) of the mobile phones can be used as a password protected controlling medium for hardware and other appliances

At the controlling end, the user sends an SMS (containing the code of the device to be controlled and the function to be performed on the device) to a mobile phone installed at the receiving end. At the receiver end, the mobile phone receives the SMS and passes the control signal in the form of data frame through serial port to the micro-controller. The micro-controller deciphers the code and performs the required action on the destination device. The device can also interrupt the micro-controller to provide certain feedback or alert. The micro-controller then sends a data frame to the mobile phone, which commands it to send an SMS to the user. The micro-controller thus acts as an interface between the mobile phone and a multitude of devices. The micro-controller and mobile phone grouped together make a controller which can take commands from the user in form of SMS and do the desired work, simultaneously keeping track of the sensing devices and providing vital feedback.

The controller once installed can be used to control any electronic device or any other hardware with an electronic transducer and provide feedback to a user regarding a specific query; the only condition being the availability of a GSM or CDMA networks. The method proposed in the paper uses Short Message Service (SMS) as the communicating tool for remote controlling and sensing. Another possible medium can be voice or a missed call.
The flowchart for the data flow at the receiver end is shown below:

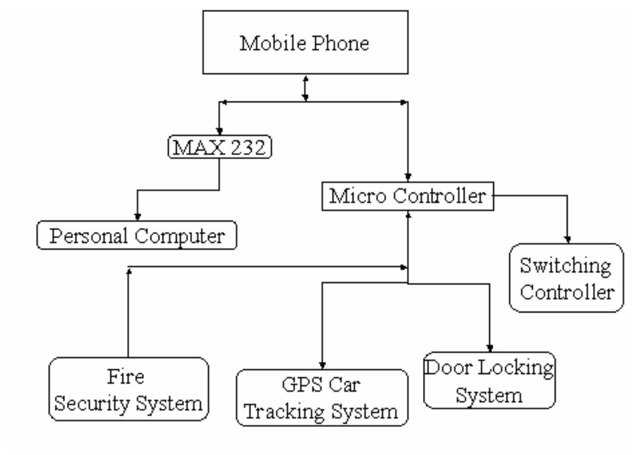

Fig. 1 Basic Framework Showing Various Applications

2 Serial port and communication protocols

Two popular protocols used for serial communication in mobile phones are F-Bus and M-Bus. Most mobile phones have F-Bus and M-Bus connections that can be used to connect a phone to a PC or a micro-controller. The connection can be used for controlling just about all functions of the phone, as well as uploading new firmware etc. This bus allows sending, reception and deletion of SMS messages. The data received by the mobile phone, through SMS, is encrypted using the protocols. This data can be transmitted to the micro-controller by an M-Bus or F-Bus cable. The basic advantage of F-Bus cable
over M-Bus cable is that it has two independent wires for receiving and transmitting data. Consequently the transmission through F-Bus cable takes place at a baud rate of 115200 bps as compared to a baud rate of 9600 bps in case of M-Bus cable. This advantage along with its easy availability makes F-Bus as the preferred protocol.

2.1 M-Bus

M-Bus is a one pin bi-directional bus for both transmitting and receiving data from the phone. It is slow (9600bps) and only half-duplex. Only two pins on the phone are used. One of the pins is ground and the other being data pin. M-Bus runs at 9600bps, 8 data bits, odd parity and one stop bit.

2.2 FBUS

F-Bus is high-speed full-duplex bus. It uses one pin for transmitting data, one pin for receiving data and a ground pin; very much like a standard serial port. It is faster than M-Bus with the baud rate of 115,200bps, 8 data bits, no parity, one stop bit.

3 FBUS Protocol and data frames

In order to allow a dialogue between micro-controller and mobile phone, the first step is to synchronize the UART in the phone with the PC or micro-controller. Sending a string of 0x55 or ‘U’ 128 times to the mobile phone, which is a series of zeroes and ones, does this. The next step is to pack the message to be sent in a form so that it could be represented as a set of hex codes. The packing of message to be sent as SMS follows GSM 03.08 convention.

3.1 Message packing and unpacking

To pack a string such as 'hello', first 'hello' is converted into hexadecimal using the character map provided by GSM 03.38. For A to Z and numbers it’s just the standard ASCII conversion.

```
h        e        l        l        o
 (ASCII characters)
68       65       6C       6C       6F
 (In hexadecimal)
1101000 1100101 1101100 1101100 1101111
(In Binary)
```

The first byte in the string is on the right. The least significant bit is then displayed on the left with the most significant bit on the left. Then the binary values are divided into bytes starting with the first character in the string. (Start from right and go to left.) The first decoded byte is simply the first 7 bits of the first character with the first bit of the second character added to the end as shown below. The next decoded byte in then the remaining 6 bits from the second character with two bits of the third byte added to the end. This process just keeps going until all characters are decoded. The last decoded byte is the remaining bits from the last character with the most significant bits packed with zeros.

```
 6F      6C      6C      65      68
1101111 1101100 1101100 1100101 1101000
```

(The ASCII characters shown in binary)

```
110  11111101  10011011  00110010  11101000
```

(The above binary just split into 8 bit segments)

```
 06      FD         9B          32          E8
```

(The 8 bit segments decoded into hex)

The message hello is therefore E8 32 9B FD 06 when packed.

The reverse process unpacks a packed SMS message.

3.2 Sending an SMS

It was then required to send a frame consisting of the packed message to the mobile phone so as to instruct it to send the message to a number specified by the frame.

Byte: 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15

Data: 1E 00 0C 02 00 30 00 01 00 01 02 00 07 91 19 89

Byte: 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Data: 01 50 91 41 FF 2B 2B 2B 15 00 00 00 04 0C 91 19

Byte: 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47

Data: 89 01 56 80 13 00 00 00 00 A7 00 00 00 00 00 00

Byte: 48 49 50 51 52 53 54 55

Data: C6 A4 B4 08 01 41 04 A0

F-Bus Frame Header

Byte 0: F-Bus Frame ID.

Byte 1: Destination address. (Mobile phone)

Byte 2: Source address. (Computer)

Byte 3:Message Type or 'command'.0x02 (SMS Handling).

Byte 4 & 5: Message length.

(SMS) Short Message Service Frame Header

Byte 6 to 8: Start of the SMS Frame Header

Byte 9 to 11: 0x01, 0x02, 0x00 = Send SMS Message

(SMSC) Short Message Service Center (12 Bytes)

Byte 12: SMS Center number length. 0x07 is 7 bytes long. (SMSC Number Type and SMS Center Phone Number)

Byte 13: SMSC number type
(0x81-unknown 0x91-international 0xA1-national)

Byte 14 to 23: (Octet format) SMS Center Phone Number

(TPDU) Transfer Protocol Data Unit

Byte 24: Message Type

XXXX XXX1 = SMS transmitted from the Mobile Station (MS) to the Service Center (SC).

XXXX XXX0 = SMS transmitted from the SC to the MS.

Byte 25: Message Reference

Byte 26: Protocol ID

Byte 27: Data Coding Scheme.

Byte 28: size of the unpacked message.

Destination's Phone Number (12 Bytes)

Byte 29: Destination's number length. Byte 30: Number type e.g. 0x81-unknown 0x91-international 0xa1-national

Byte 31 to 40: Destination's Phone Number

Validity Period (VP)

Byte 41: Validity-Period Code.

Byte 42 to 47: Service Center Time Stamp.

The SMS Message (SMS-SUBMIT)

Byte 48 to 52: This is the SMS message packed into 7 bit characters. SMS Point-to-Point Character Packing

The F-Bus frame ending

Byte 53: Packet Sequence Number

Byte 54: Padding Byte - String requires being even

Byte 55 & 56: Odd & even checksum bytes.

When the phone receives this frame it acknowledges it by sending the frame shown below:

Byte: 00 01 02 03 04 05 06 07 08 09

Data: 1E 0C 00 7F 00 02 02 03 1C 72

The destination and source addresses are swapped, as this is a frame from the phone to the PC. This message is two bytes long with the first byte representing the message type received (0x02) and the next byte, the sequence number (0x03). The last two bytes are the checksum.

After a short time the phone replies with a 'Message sent' frame shown below.

Byte: 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17

Data: 1E 0C 00 02 00 09 01 08 00 02 64 12 00 01 44 00 3F 1E

Byte 03: Message Type = 0x02 - SMS Handing

Byte 04 & 05: Message Length = 0x0009 - 9 Bytes long

Byte 09: 0x02 = Message Sent

Byte 10 to 14: Message center stamp

The PC then acknowledges the frame.
Byte: 00 01 02 03 04 05 06 07 08 09
Data: 1E 00 0C 7F 00 02 02 04 10 79
This completes the first phase of sending an SMS.

3.3 Receiving an SMS

The second phase involves obtaining the SMS received by the phone. The mobile phone can be instructed to send the received SMS, as soon as it is received, by sending it the following 'get SMS' query. However it is essential to synchronize the UART of the mobile phone.
Byte: 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
Data: 1E 00 0C 02 00 08 00 01 00 33 64 01 01 40 77 79
After receiving the SMS the mobile phone sends the following frame to the computer. However it is observed that immediately after the synchronization if the get SMS query is sent then an extra frame is received before the actual message-containing frame. This extra frame contains the message center details and is discarded. The actual message-containing frame is as follows:
Byte: 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
Data: 1E 0C 00 02 00 59 01 08 00 10 02 10 00 07 91 16
Byte: 16 17 18 19 20 21 22 23 24 25 26 27 28
Data: 14 91 09 10 F0 00 10 19 38 04 00 00 04
Byte: 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44
Data: 0B 91 16 04 73 08 70 F4 70 40 32 25 30 30 E8 32
Byte: 45 46 47 48 49 50 51
Data: 9B FD 06 45 00 4A 5C

F-Bus frame header
Byte 00 to 05: Hold same significance as explained in the sending of an SMS

(SMS) Short Message Service Frame Header
Byte 6 to 8:Start of SMS Frame Header0x00, 0x01, 0x00
Byte 09: SMS Message received
Byte 10: Memory Type = SIM
Byte 11: Location where SMS message stored

(SMSC) Short Message Service Center (12 Bytes)
Byte 12 to 23: Hold same significance as explained in the sending of an SMS

(TPDU) Transfer Protocol Data Unit
Byte 24: 0x38
Byte 25: 0x04
Byte 26: Protocol ID
Byte 27: Data Coding Scheme
Byte 28: Message Length.

Sender's Phone Number (12 Bytes)
Byte 29: Sender's number length. Byte 30: Number type e.g. 0x81-unknown 0x91-international 0xa1-national
Byte 31 to 40: (Octet format) Sender's Phone Number

The SMS Message (SMS-RECEIVED)
The frame received contains the packed message from the 43rd byte onwards and the size of the message received is specified by the 28th byte. The message can be unpacked to obtain the received message. Following the reverse of the packing procedure can do the unpacking.
This is subsequently followed by an acknowledgement, to this received frame, by the computer. The protocol for acknowledgement frame is same for any received frame, as mentioned in the sending SMS phase.

3.4 Deleting an SMS

The third phase of the experiment involves the deletion of an SMS. This can be achieved by sending the following frame to the mobile phone.
Byte: 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
Data:1E 00 0C 14 00 08 00 01 00 0A 02 02 01 41 11 54
Byte 0: F-Bus Frame ID.
Byte 1: Destination address. (Mobile phone)
Byte 2: Source address. (Computer)
Byte 3: Message Type = 0x14 - SMS Functions
Byte 4 & 5: Message Length
Byte 6 to 8: Start of SMS Frame Header. 0x00, 0x01, 0x00
Byte 9: 0x0A Delete SMS Message
Byte 10: 0x02 = Memory Type = SIM (0x03 = phone)
Byte 11: 0x02 = Location where SMS message is stored.
Byte 12: 0x01
Byte 13: Packet Sequence Number
Byte 14 & 15: Odd & even checksum bytes.
With the deletion the device can have enough space to allow any number of messages.

4 Hardware Implementation

In order to extend the applications of the experiment to hardware devices, a micro controller is used as an interface between mobile phone and hardware devices. The process of receiving, transmitting and deletion of an SMS using a micro controller is essentially the same as that followed while using a computer. The only difference being, that the micro-controller and the mobile phone both operate at same logic levels thus eliminating the need for logic conversion. The setup used to carry out experiments included a mobile phone connected to the serial port of a computer through an RS232 cable The serial port of the computer follows the RS-232 logic i.e. logic high from –10 to –15 Volts and logic low at 10 to 15 Volts, whereas the serial port of the mobile phone follows CMOS logic i.e. logic high from 3 to 5 Volts and logic 0 at 0 Volts. This necessitates the use of a logic converter. The IC used for logic conversion is MAX-232, which is based on the principle of voltage doubling and voltage inversion to facilitate logic conversion.

Keeping in mind the constraints of a micro-controller such as its memory size and its processing speed, the micro-controller used for the experiments was ‘Phillips 89C51RD2BN’.This micro-controller has 64KB of flash memory and 1KB of RAM as compared to ‘Atmel 89C51’ which has 4KB of flash memory and 128 byte of RAM. Due to the large number of variables used in the software and the large size of SMS frames being received, ‘Phillips 89C51RD2BN’ became the preferred choice. It was required to further improve the software in order to ensure the repeatability and continuity of the aforesaid processes. This was accomplished by the combined use of polling method and external hardware interrupts.

The controlling of hardware devices was accomplished by decoding the SMS received, comparing it with the already stored strings in the micro-controller and accordingly providing an output on the ports of a micro-controller. This output is used to switch on off a given hardware device. In another case a sensing device such as a thermistor can be used to provide an interrupt signal. This interrupt signal then causes the micro-controller to process the interrupt service routine (ISR), which in turn provides a feedback to the user through an SMS.

5 Applications

Although the applications of the experiment are limitless but since the scope of this paper is limited we would like to concentrate on the following ones:

- A methodology to detect the occurrence of fire in a premise and notify the owner of the establishment and the fire security personnel of the event and automatically open the doors and windows of a building by incorporating an embodiment of Door Operating System (DOS). A further provision to automatically activate the Fire Extinguishing System after determining the cause of the fire can be made.

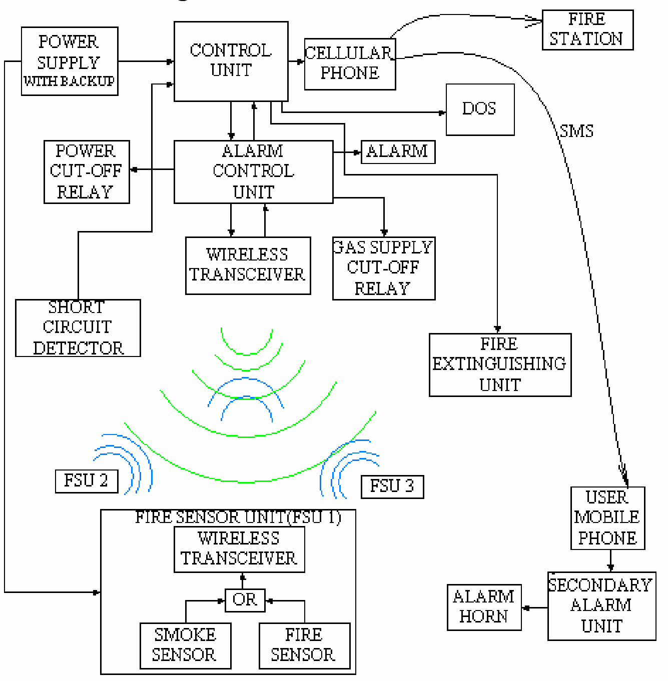

Fig. 2 Security Alert System

- A methodology to detect a medical emergency or a vehicle accident and to inform the relatives of the victim. It also includes a provision to continuously monitor the physiological parameters of a subject and provide a means through which the subject or a doctor can access the data. It may further provide a means to inform a patient of the required medication and its alternatives in case of a medical emergency.

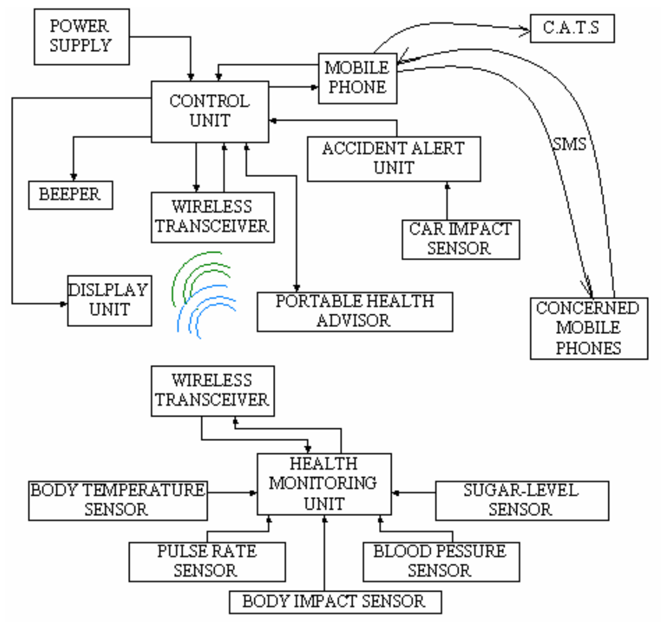

Fig. 3 Health Monitoring and Notification System

- Controlling the switching on or off of electrical devices.
- It can be used for remotely controlling door locking system and giving a warning signal, through SMS, in the event an intrusion.
- It can be used in the service sector such as car servicing in order inform the current status of their vehicles.
- Controlling the performance parameters of electrical devices such as the tripping temperature of an air-conditioner.
- Providing critical information such as position of a car in case of a car theft.
- It can be used as remote control to manipulate the movements of a robot with the added advantage of infinite distance control.

6 Conclusion

In this paper we discussed the use of embedded systems in extending the applications of mobile phones to control a multitude of devices and receiving vital feedback from them from a remote place using the Short Message Service (SMS). With the help of an appropriate electronic circuit and software, mobile phones can be interfaced to any electronic device or other devices using appropriate electronic transducers. It can be further said that the adaptation of this concept in our day-to-day lives can provide better security and time management.

7 Acknowledgements

We would like to express our sincere gratitude towards Mr. Krishnendu Chatterjee for invaluable support and guidance, and for his constant motivation, which lead to successful completion of this experiment. We would like to appreciate the suggestions given to us by Professor Suneet Tuli, Associate Dean, Indian Institute of Technology, New Delhi .We would like to thank the Electronics and Communication Department of Delhi College of Engineering for providing the necessary instruments for carrying out the experiment.

References:

[1] Yi-Bing Lin, Ai-Chun Pang, *Wireless and Mobile All-IP Networks*

[2] Charles E. Perkins, *Mobile IP Design Principles and Practices*

[3] Gunnar Heine, *GSM Networks: Protocols, Terminology and Implementation*

[4] Peter Stuckmann, *The GSM Evolution: Mobile Packet Data Services*

[5] Jorg Eberspacher, *GSM: Architecture, Protocols and Services*

[6] Rogier Noldus, *Intelligent Networks for the GSM, GPRS and UMTS Network*