LAPPEENRANTA UNIVERSITY OF TECHNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGY

Embedded Monitoring Server

The topic of this Thesis was approved by the council of the Department of Information Technology

-June 6th 2007-

Examiners: Adjunct Professor Jouni Ikonen and Professor Jari Porras

Lappeenranta October 16th 2007

Justin Kerosi Omwoyo
Ruskonlahdenkantu 13-15 C11,
53850 Lappeenranta
Justin.omwoyo@lut.fi

Abstract

Lappeenranta University of Technology
Department of Information Technology
Justin K Omwoyo

Embedded Monitoring Server

Master's thesis
2007
Xx pages, 28 figures, 3 tables.

Examiners: Adjunct Professor Jouni Ikonen and Professor Jari Porras
Supervisor: Adjunct Professor Jouni Ikonen

Keywords:
Home monitoring system, WRT54GL, OpenWrt

This study investigates, designs, and implements an inexpensive application that allows local and remote monitoring of a home. The application consists of an array of sensors for monitoring different conditions in a home environment and also for accessing the devices that might be connected to the system. Only a few sensors are initially involved in this study and information about the temperature level, forced entry detection, smoke and water leakage detection can be obtained at any time from any location with an Internet connection. The application software (coded in C language) runs on an embedded system which is basically a wireless Linksys router running on a GNU/Linux based firmware for embedded systems. Interaction between the sensors and the application software is achieved through an implemented sensor interfacing circuit. The communication with the sensor interfacing unit is done through the serial port, and accessibility of the connected sensors is achieved through a telnet client. The sensors can be accessed from local and remote locations with the sensors giving reliable information. The resulting application shows that it is possible to use the router for other applications other than what it is intended for.

Acknowledgements

This study was done in the Laboratory of Communication Engineering in the Department of Information Technology at Lappeenranta University of Technology. I am sincerely thankful to the Department for having funded this work and my studies.

I would also like to thank my supervisor Adjunct Professor Jouni Ikonen whose excellent guidance and knowledgeable comments helped me to get a grasp of this interesting topic. Special thanks to Professor Jari Porras for ensuring that I had a practical topic for my thesis work. I would like to express my gratitude to Riitta Salminen in the Faculty of Technology Management for also wish to express my gratitude to everybody in the Department who in one way or another supported me during this time.

My sincere appreciation to my family members brothers, sisters, and parents more especially my father Mr Omwoyo Mokono for their continued support both morally and financially. Much thanks to all my friends more especially D. Oganga, Dr J. Atina, C. Anam, E. Nyangari, E. Nyon’ga and H. Blanarova both for their moral and financial support.

Contents

Symbols and abbreviations

UART	Universal Asynchronous Receiver Transmitter
GSM	Global System for Mobile communications
LAN	Local Area Network
WAN	Wide Area Network
GUI	Graphical User Interface
ECHONET	Energy Conservation and Homecare Network
KNX	Konnex
EHS	European Home System
EIB	European Installation Bus
PSTN	Public Switched Telephone Network
ISDN	Integrated Service Digital Network
ADSL	Asymmetric Digital Subscriber Line
OSGi	Open Service Gateway initiative
PDA	Personal digital assistant
RF	Radio Frequency
IrDA	Infrared Data Association
WLAN	Wireless Local Area Network
ISM	Industrial Scientific and Medical
DTMF	Dual Tone Multi Frequency
GPRS	General Packet Radio Service
pSOS	plug-in Silicon Operating System
eCos	embedded Configurable operating system
API	Application programming interface
RPC	Remote Procedure Call
Sms	Short message service
PC	Personal Computer
ADC	Analog to Digital Converter
TV	Television
DOW	Dallas One-Wire

ROM	Read Only Memory
WR	Write
CS	Chip Select
RD	Read
INTR	Interrupt
A.GND	Analog ground
D.GND	Digital ground
RC	Resistor Capacitor
RS232	Recommended Standard 232
TTL	Transistor Transistor Logic
DSL	Digital Subscriber Line
JTAG	Joint Test Action Group
IDC	Insulation Displacement Connector
PCB	Printed Circuit Board
MSC	Message Sequence Charts
Rx	Receive
Tx	Transmit
JFFS2	Journalling Flash File System, version 2
USB	Universal Serial Bus
CPU	Central Processing Unit
RAM	Random Access Memory

1 Introduction

This is a study based on an application that allows local and remote monitoring of a home environment. It explores ways of employing an embedded operating system to implement an inexpensive home monitoring system consisting of an array of sensors. As the sensors work together, they should at the same time, be able to give individual signals for each state that is being monitored in the area of application. The objective is to provide a monitoring system that would be capable of measuring temperature, detecting water in case of a water leakage and smoke in case of fire. Where necessary the system should be able to activate alarm circuits when conditions such as the presence of smoke or water have been detected from the sensed signals, it should also send a notification message to the person responsible.

The advancement in sensor, regulator, activating element (actuator) and microprocessor technology both on the hardware and software levels have enabled distributed utilization of sensors. The distributed application has made control actions over sensor-actuator networks possible. By using a set of suitable sensors that are appropriately placed, it is possible to monitor different conditions such as temperature, humidity, voltages, currents, and switch closures in areas of interest. When these sensors are systematically connected together they form a network of sensors with simple data collection units that communicate the sensed value to a central point either for utilization or for an onward transmission. With this arrangement, some sensors can be connected to various devices in a home to facilitate the monitoring and control of these devices. If this network of sensors is connected to some global network e.g. the Internet then additional features such as remote monitoring and control from anywhere in the world are evident. While using the system, the user should be able to get the same response irrespective of whether the system has being accessed locally or from a remote or mobile location.

When the application of the sensor network is to monitor different conditions in a home and if it incorporates some level of device control then the network can alternatively be referred to as a home monitoring sensor network or generally as a home network (digital home, home automation [8]). In this case the term home network is a general term that is used to describe different but sometimes related technologies.

Remote and local access controls would contribute to keeping homes comfortable and providing support to the elderly, disabled, and the sick. With the World-Wide-Web evolving, it is clear that its underlying technologies can be suitable for a variety of applications other than just browsing the Web. As internet connection becomes an integral part of every household infrastructure it is also possible according to [14] that every device will have a network interface that allows it to be accessed and controlled regardless of its location.

Pervasive computing has contributed to embedding microcomputers in home appliances and factory devices which make the inclusion of embedded web servers in different devices possible [28]. The inclusion of the web server extends the capabilities of these devices to provide functions such as automatic generation of maintenance requests, remote diagnostics, and remote firmware upgrading. An embedded system refers to a machine that is small and works well routinely in a concealed place [20]. It is a system that is programmed to perform one or a few predefined tasks usually with specific requirements. These systems cover a wide range of applications from small computer-based systems to large systems monitoring and controlling complex jobs. As their applications become more complex, small control programs are not able to provide some required functionalities such as network connection, and multimedia functions.

More functionality can be achieved by using an embedded operating system. Different types of embedded operating systems exist, and decisions on the suitable system depend on the characteristics and requirements of the application. One example is the embedded Linux which is a Linux-based embedded operating system that was developed for systems with small memory and low performance processors [8]. According to [19], [22], Linux has several advantages that make it suitable for embedded system applications. It is scalable and can run on a broad range of devices including the small embedded devices that manage with small flash memories; it is customizable, and has neither loyalties nor a license fee. Linux displays a high degree of reliability with up-time measures in terms of weeks or years. Linux is a network operating system and it provides the possibility of accessing services from other machines thus providing full control of systems over the network.

The GSM and email messaging features that are suggested in this study for sending alarm messages to the user will not be implemented. Only a smoke alarm is included in the implementation. The sensor interface unit will be built to incorporate sensing elements for intrusion detection, temperature measurement, smoke and water leakage detection. The Telnet client will be employed in the application to facilitate communication between the user and the home monitoring server. The monitoring server instructions will be coded in C programming language.

In the remainder of this thesis, chapter two presents the related work on home monitoring networks. Chapter three gives the general description of the embedded monitoring server including some possible use cases, and chapter four contains the adopted design principles together with the general overview of the intended system. Chapter five presents the circuit implementation, operation description, system analysis and discussion. The conclusion about the overall design, implementation and analysis is given in chapter six.

2 Home monitoring networks

Various technologies are considered to be working together to realize the implementation and efficient operation of the home monitoring system. Different protocols are also employed at different levels of communication in the setup. Figure 2.1 shows the conceptual framework for the involved technologies.

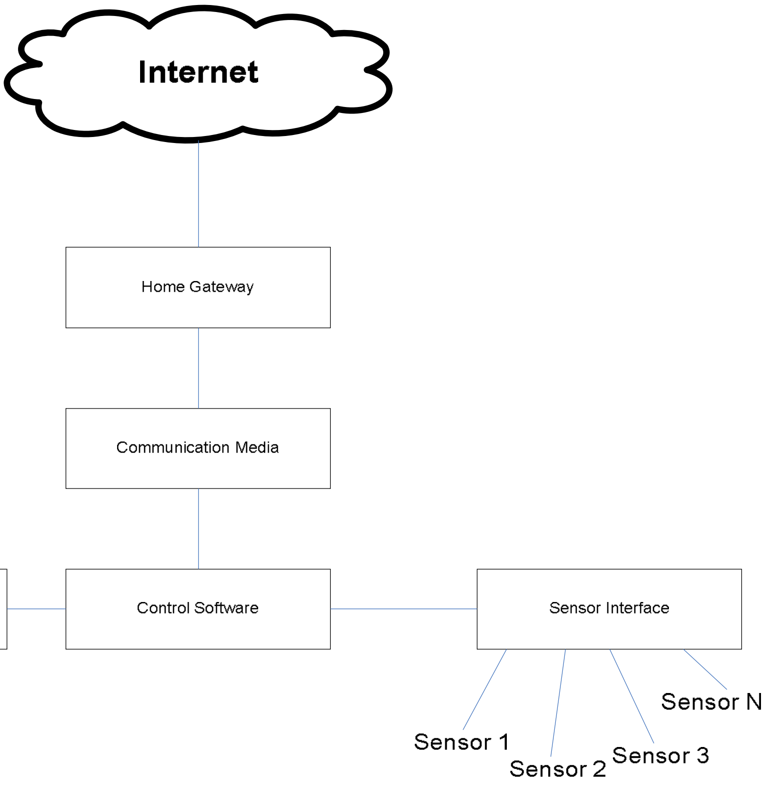


Figure 2.1 Home monitoring sensor network technologies

The home gateway is the hardware together with the software that connects the home network (LAN) with the Wide Area Network (WAN). The communication media provides the means of communication between the devices connected to the system and the user. The communication media integrates the devices into one comprehensive system with the control software providing means of reading, decoding and executing user requests.

User interface provides the means by which the user is able to interact with the system in terms of input and output. The sensor interface is an electronic circuit that provides an interface between the devices to be monitored and the controlling software. The interfacing circuitry detects the changes in the conditions under consideration through the sensors, processes the sensed signal, provides signal routing through multiplexing, and converts the sensed analog signal into digital form before transmission through serial or parallel connectivity.

The idea behind home networking for remote monitoring and control of home appliances sound exciting more especially when considering the convenience and the flexibilities it avails. Unfortunately only a few of the home networking solutions can be applied to real life. Various reasons are cited to be the cause although not all of them are well identified or properly understood. The cost of the application said to be one of the reasons because of the small ratio between the features offered and the cost [5], [29]. The cost factor is attributed to lack of a powerful standard technology with a wide market acceptance.

Since the applications software for each application is tailor made to suit individual user requirements, the software development costs very much dominates the final cost of the finished product. Software reusability approach can be employed to reduce the software development effort and according to [6], specification, configuration, and deployment processes can be transformed to a single process that is followed by a repetitive configuration process. A model capable of storing all the parameters relevant to a specific process is used to support the configuration process. For semi-automatic configuration of selected services based on the model, a specification is required before hand where the user gives the architectural information, the existing appliances and their location.

Complexity, lack of security in terms of device exposure, and the fact that no standard protocol has been developed for home networks are also said to be reasons why home automation has not received wider acceptance and attention [7], [8]. Several major Japanese consumer electronics manufacturers and other organizations have come up with a standard specification “ECHONET” for controlling and monitoring home appliances

[9]. The Konnex (KNX) Association [36] promotes the KNX standard, a system for Home and Building controls that is open and platform independent, guarantees multi-vendor and cross-discipline interoperability, and supports many configuration methods. KNX is said to be the successor to, and convergence of three previous standards: the European Home System Protocol (EHS), BatiBus, and the European Installation Bus (EIB) [36]. Accepted standards would allow the appearance of a multitude of different products, compatible with each other even when produced by different companies. Standards would also make it possible for the user to configure the system according to the prevailing needs to facilitate usability.

2.1 Home Gateway

The home gateway interconnects the home monitoring network and the public (access) network. Various access networks such as public switched telephone network (PSTN), integrated service digital networks (ISDN), asymmetric digital subscriber line (ADSL), broadcasting networks, wireless access, and the (public) internet are available for utilization. Since the Gateway resides between the two networks i.e. LAN and WAN it provides the various communication function requirements. According to [38], some of these requirements are provision of the user interface to the user access from the public network, firewall function and policy control, home directory service function, remote control function, address and protocol translation functions, home router and communication server function, and provision of the medial translation function.

It is observed that home computers and communication devices connected to the internet are exposed to various kinds of attacks and therefore some form of protection should be instilled [12]. Denial of Service attack, a situation where an attacker attempts to make home network control resources unavailable to the end user is very common. Other possible security threats can be listed as hacking, malicious codes, worms, viruses, and eavesdropping, [12] proposes a security service framework in a home gateway that implements a secure channel between the home and the internet. The framework also implements traffic state-based firewall to protect home gateway from Denial of Service attacks.

The security requirements are addressed in terms of ensuring the confidentiality in transferring commands for controlling home appliances by using mobile devices, monitoring and controlling network traffic for the home gateway, access control and intruder detection for the wireless LAN within a home, and finally about managing the users[10]. The security framework in [10] is based on Open Service Gateway initiative (OSGi) home networking middleware. The protocol designed to authenticate the user and to use sequence numbering to prevent packets from possible attackers. The protocol also ensures that all packets between home gateway and the Personal Data Assistant (PDA) are encrypted, and each encrypted packet is given its own integrity value (data integrity). For traffic state-based firewall, [10] proposed traffic inspection using the security state graph and implemented a traffic inspection engine. The engine consists of traffic state information collector, and security policy manager. With the traffic state information manager collecting and analyzing the information using a statistical method and the later establishing security policy by using the security states graph that represents all the possible traffic situations that can appear on the network. In the same setup, each home network user is assigned a role such as administrator and guest with the administrator giving access rights to the guests. For external accessibility of the network each user must have an id and password.

No single security approach can be pointed out to be the best in all types of home monitoring network architectures or models [12], [13]. In the case of Gateway model architecture, as every packet passes through the home gateway, it authenticates home users and access permissions based on authentication information. The gateway therefore processes the security of the whole set of devices that follow in lump sum while with the Hierarchical model the cryptographic key is different at every level. In the individual model each devices carries out the security process individually and authenticates each other mutually.

The choice of using an email account for remote control gives room for a huge number of spam emails from hackers [5]. A program to accept and process only a few emails from senders with proper signatures, encryptions, and passwords serves to protect the system. Accessibility through the X10 power line control has the possibility of processing X10 commands from neighbors sharing the same power line, and it is also a possibility for an attacker to gain access. Installing a filter at the main electricity panel to block stray commands and placing the most critical devices on private lines provides some level of security. Security aspects would include password protection, proper signatures, and encryptions. Authentication and authorization can be used to block unauthorized users from accessing and controlling appliances at home [7], [28].

2.2 Communication Media

The main function of the communication media is to provide communication between the sensors, the user, and the controlling software. Devices in a home monitoring network system can be interconnected by using different communication media whose main function is provide communication between the sensors and the data acquisition system. Some of the possible means of communication are; power lines (X10), Wireless, Ethernet, Twisted pair wire, FireWire, RF, Phone lines, and Cellular network. X10 is an international and open industry standard for communication among electronic devices used for home automation [52]. The X10 standard allows compatible products to talk to each other using the existing electrical wiring in a home. No costly rewiring is necessary for this system that is also simple to install. The used media is selected appropriately to reduce the overall cost of the application

Bluetooth technology based home network is one suitable option on the basis that the technology has competing advantages over the other technologies like Infrared Data Association (IrDA), Home Radio Frequency (HomeRF) and Wireless Local Area Network (WLAN). Some of the advantages of Bluetooth technology are the low cost of installation and the easy implementation [3]. It also utilizes the unlicensed Industrial, Scientific, and Medical(ISM) frequency band which makes it suitable for electronic devices and data communication that uses short range radio links for operation.

Other communications options such as wireless LAN technology, dial-up modems, private radio networks, satellite communication, and cellular network can be used for monitoring and controlling machines in remote locations. With the cellular network attracting more attention for transporting data between machine to machine, man to machine or mobile to machine, more especially the widely adopted GSM digital standard [11]. Wider application of GSM is attributed to the wide coverage of GSM which makes the machine online for almost all the time, its low cost as compared to building a network or using satellite communication, and the high GSM network security. GSM also provides wide options such as the Dual Tone Multi Frequency (DTMF), Short Message Service (SMS), and the General Packet Radio Service (GPRS) for the designer to choose from. GPRS introduces packet data transmission which is one of the best options for some of the applications that may require online connection.

2.3 Controlling Software

The function of the controlling software is to coordinate the execution of different functions with respect to the requests made by the user. After gaining access to the system, the user can be able to address and monitor or control the connected devices at any time. While the user is still connected the software reads and interprets the different requests to generate corresponding control signals and address for a harmonious operation of the entire system. Additional requirements such as multitasking and the networking functions are best handled by an operating system. The Linux operating system is selected for this application.

Linux is an open-source Unix-like kernel that can be freely distributed under the terms of the General Public License, and which provides an alternative operating environment such as Windows and Unix. Linux was developed as an operating system for the desktop/server environment and the strengths it has displayed in this area makes it attractive in the embedded domain. An embedded operating system is under normal operating conditions expected to display a degree of robustness well beyond the requirements of the desktop domain [19]. Some of the operating systems that have been developed from scratch to support embedded systems are QNX, pSOS, VxWorks, ThreadX, Windows XP embedded, and Windows CE.NET [19], [20], [21]. NetBSD, FreeBSD, and eCos are listed under open source category with Embedded Linux especially said to have been developed to support the open-source policy, stability, powerful functionality, and high availability.

Software in embedded systems is in most cases composed of components that operate concurrently and in real time, often interacting remotely. It is implied in [23] that a framework that allows concurrency is particularly useful in designing embedded systems. As an example current cellular phones contain a web browser, a java virtual machine, e-mail software, and camera software [26]. Handling such kind of scenario requires an operating system that supports memory protection so that a software bug in one application does not crash the entire system, and malicious applications do not monopolize the entire system capacity.

However, real-time application requirement with Linux for the applications that require response times in microseconds and nanoseconds has not been addressed fully. With applications in the microsecond range well covered, several solutions currently available both commercially and free have been proposed to obtain real time capabilities. Two different approaches [24], [25] are employed, one introducing new software layer a real-time kernel called micro-kernel between the hardware and Linux, and the other employs a set of patches to standard Linux to make it a real-time kernel. The so-called preemption and low-latency patches, put together in a single patch and available as a standard kernel configuration creates an opportunity for the kernel scheduler to minimize the response time.

Embedded Linux can be obtained in three different ways, one is to select from a suitable distribution and the rest from vendors who support this type of Linux in various ways

[19]. There exist both commercial and non-commercial distributions with the commercial option adding value through enhanced development tools or support for additional software components such as embedded GUIs, real-time operations, and embedded Java virtual machines. The non-commercial distributions are targeted at specific application area such as in an embedded internet router with the OPENWRT project [27] serving as an example. Normally this is not an easy option as it might not be complete enough for other applications and would therefore require some additional input from the user. The second option is that of involving an Embedded Linux vendor to port their distribution to a chosen hardware platform. The third approach is to select hardware from a vendor who supports an embedded Linux 'out of the box' an option that allows the user to concentrate on adding value through application development rather than devoting time on the system software.

For convenient management of systems with embedded Linux, it is necessary to use a lightweight middleware that uses only basic operations of Linux. Where possible the middleware should be implemented by using the socket API and the RPC mechanism so that it can operate on embedded Linux which has limited resources and does not need specific platforms and development tools [20].

2.4 User Interface

The home user requires well defined user interface to provide the new means to control the objects and devices at home. The selected means should be convenient, easy to use, flexible, and where possible should not introduce any additional cost on the application. It therefore implies that, the means of user interface should be chosen from the devices that already exist in a home [5]. It is also observed in the same study [5] that, three types of remote user interface; instant message client, email client, and mobile phone through short message service (sms) can be considered to provide convenient interface.

It is suggested in [1] that there are two main types of activity patterns in a home setup when it comes to device and object control options; one is pattern control and the other is instant control. Pattern activities can be planned and determined in advance whereas with instant control activities it depends on what is suitable at a given point in time. From the existing home objects information appliances have standardized interaction capabilities embedded in them and so can be used to provide user interface. Three familiar information appliances commonly found in every household are a Personal Computer (PC), a media terminal, and a mobile phone. These three can be used to provide interface for selecting home functions. A media terminal would provide interaction via a TV through a remote control and therefore suitable for instant control. A PC is well suited for both instant and pattern control activities. A mobile phone has the advantage of providing remote control while the person is not at home. Since they all provide a graphical user interface (GUI), they build some degree of trust on the user's side in confirming that the function or command has been carried out as compared to when using the command line interface. Sending a control command to a device may not guarantee the successful operation of the device if it is defective and it therefore requires a feedback circuit to indicate the device's actual status after receiving a software command e.g. ON/OFF [17].

It was predicted that the Internet-enabled home was a likely in the 21st century with the most likely user interface forms proposed to be based on two requirements [4]. First the portability of the user interfaces, allowing access to the home network from a wide range of hardware and operating systems. The other requirement based on the thin size to facilitate dynamic uploading and operations on hand-held computers. Thin size also to facilitate utilization of resource constrained devices for interface. Embedded Java was considered as an application development platform for creating embedded devices and resource constrained environments, and the possibility to implement sophisticated user-interfaces using Java. The user interfacing format can take many different forms with some of the commonly used listed as: Graphical User Interface (GUI), Web-based user Interface, Voice user Interface, Text user Interface, Touch Interface, and the Command line Interface. OpenWrt and telnet primarily use command-based interface.

2.5 General User Attitudes

Because of the rapid rate of technology development large part of the population finds it impossible to utilize full functionality of the resulting products. New technology products are desirable and should also be accessible to the whole population in terms of functionality and usability. When designed to be user friendly, the home monitoring technology can potentially improve standards of living by promoting home comfort, convenience, security and entertainment [29]. If this potential is fulfilled the benefits will be apparent to many users more especially the elderly and the disabled people.

Before making digital home technology a reality, user requirements in terms of needs, attitudes, and expectations should be taken into consideration. In a survey [2], cost, reliability, security/privacy/safety, ease of use, flexibility, convenience, maintaining independence/keeping active, future proof of technology and issues regarding control of the digital home were identified as key factors concerning every potential user. A big part of the surveyed population was skeptical about smart home lifestyle that it would lead to a lazy and obese society. The older and the disabled participants felt the smart home would aid their home life provided it did not take away any activities they enjoyed and could still carry out. Users in this survey associated the reliability of the home networks with the problems they had experienced with computers and were suggesting need for back up systems in case of failure, while others cited the demand for simple customizable control systems. Initial set-up costs and security of the personal data privacy were considered to be of a greater importance to the users of the systems.

New developments and decreasing costs of electronic appliances have enabled the realization of pervasive computing systems in the daily environment thus leading to their wider application. It is suggested that future management of home networking systems would be expected to be more complex than managing today’s desktop machines because of their pervasiveness [5]. With security, reliability, and robustness pointed out to be the critical issues related more especially to changes in system configuration.

The issue of reliability with the home networking contributes to making the implementation expensive, a factor that makes the system unaffordable. Reliability in [5] is considered to incorporate a well-defined procedure for fault localization and provision of a “panic button” that will allow the user to disable the entire system when it malfunctions. In the event of power failure, the system should alternatively be supplied from an Uninterruptible Power Supply (UPS) source for some limited amount of time while waiting for the normal power supply restoration.

A home network system regularly experiences changes as users add, move, remove, or disable devices. The system should be designed in such a way that it supports self-reconfiguration in the presence of changes so as to provide continuous correct operation.
There still exist many challenges when creating home monitoring networks more especially when the network is growing, some of these the challenges are in terms of adding and deleting equipment, equipments joining and leaving the network [9].

3 General Description of the Monitoring System Framework

The proposed system is centered on an embedded system that is connected to a local area network (LAN) and eventually to the internet. The embedded system forms the Home Monitoring Server that connects to the sensor interface unit and runs a process that monitors the individual sensing elements. One or more sensor interface units can be connected to the server with each being responsible for a specific functional area. At the physical level the sensor interface unit is connected to the home monitoring server through a serial port. Figure 3.1 illustrates the general structure of the intended system.

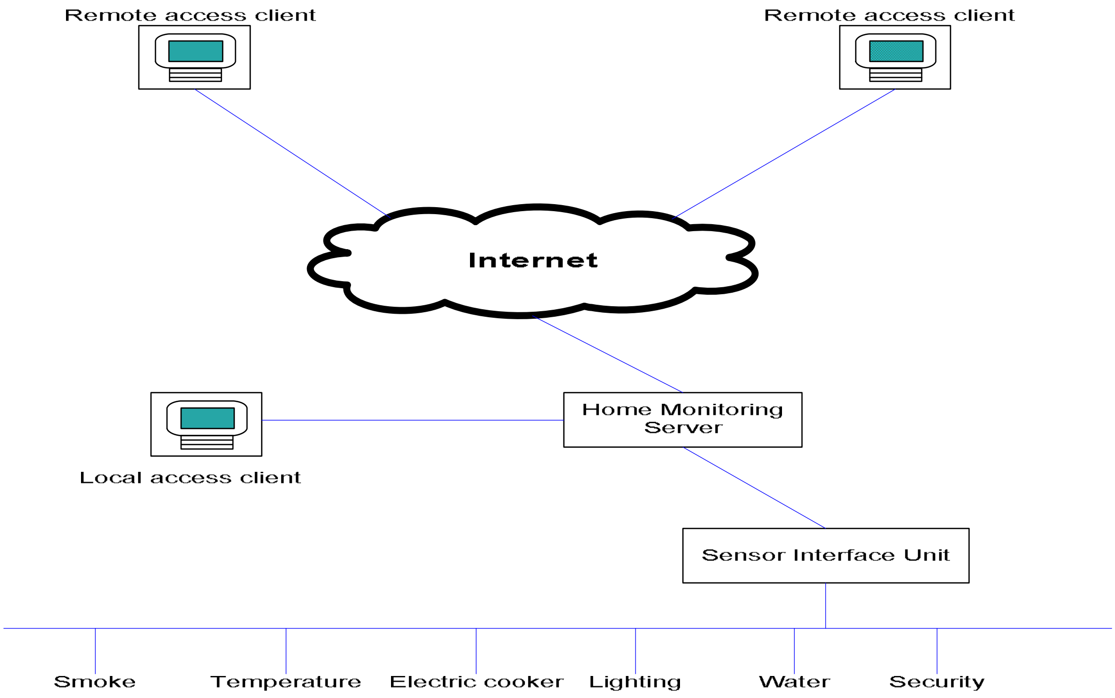


Figure 3.1 General structure of the system

3.1 Structure of the Home Monitoring Server

The diagram in Figure 3.2 describes the structure of the home monitoring server**.** The key element is the Home Monitor process that coordinates the interaction between the entire system and the users. The process maintains a textual database that keeps record of the general trend of events for each sensor connected to the interfacing unit. The textual database can be accessed at any time for detailed record information about each device or sensor.

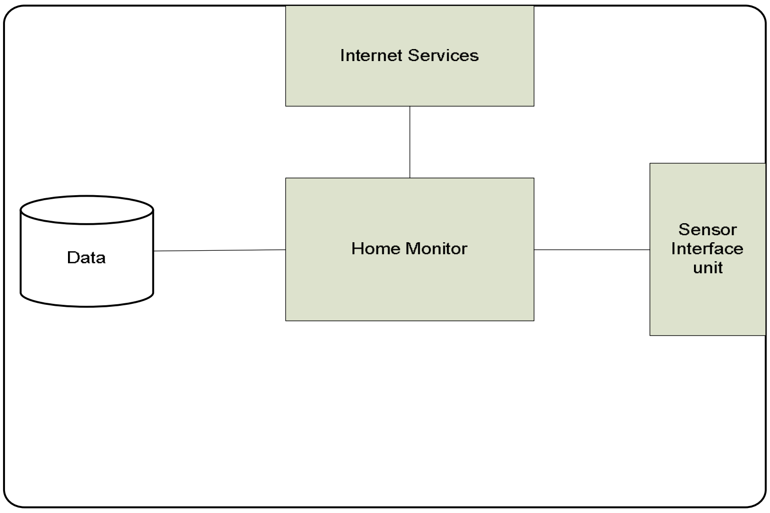


Figure 3.2 Internal Structure of the home monitoring server

The Home Monitor interacts with the world through the Internet services and the Sensor Interface Unit. Overall interaction is achieved in conjunction with the software function that dialogs with the specific hardware interface to access each home device current state.

3.2 Functional system overview

The different parts making a functional monitoring system can be seen to be consisting of blocks that are put together to give the layout in Figure 3.3. The layout consists of three functional units, an array of sensing elements for gathering a range of signals for the different conditions, means of communicating each sensed signal at a time (interfacing circuitry), and finally signal reception and processing at the receiving end. The processed signal information at the receiving end can be transmitted further or can be used to generate control signals for the connected devices.

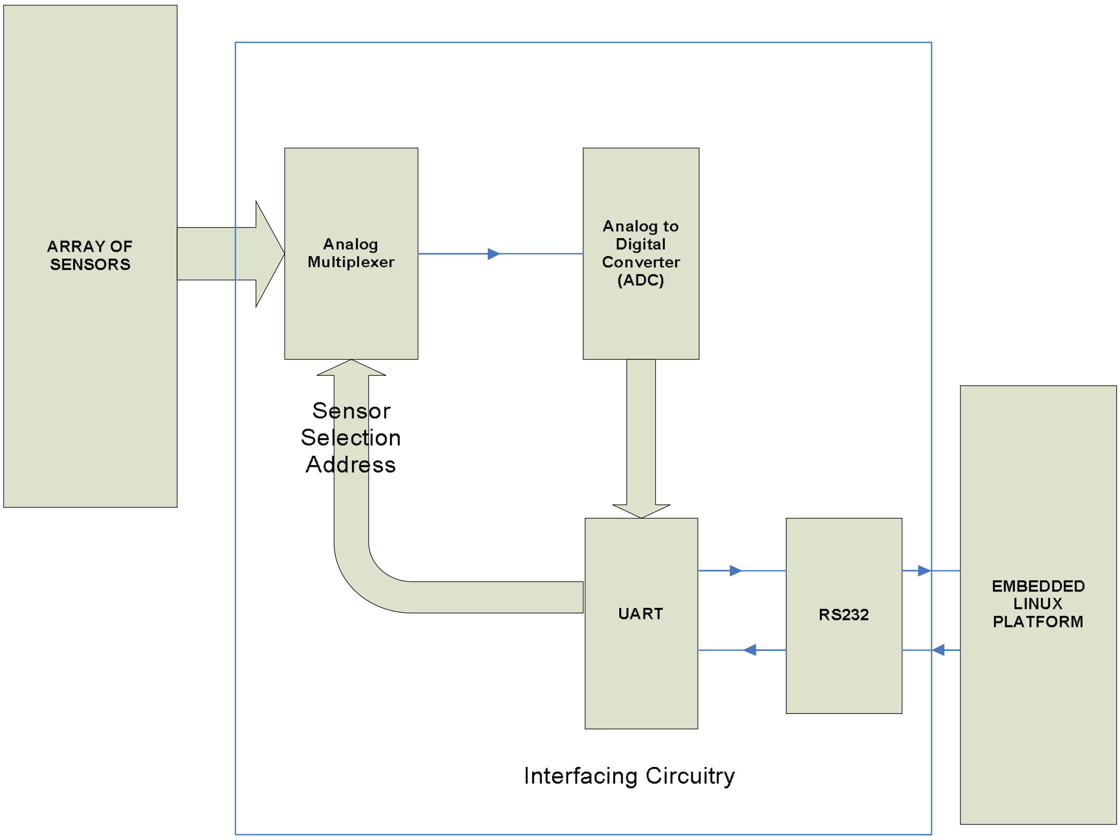


Figure 3.3 Block diagram for the Sensor Network

The array of sensors in Figure 3.3 is made up of a range of different sensing elements that are responsible for monitoring different applications, conditions or devices. In this application it consists of temperature, water, security and smoke sensors. The sensors are used for temperature measuring, water leakage detection, intrusion detection (security) and smoke detection in the event of fire. Each sensor output is selected for connection to the Analog to Digital Converter (ADC) at a time by a multiplexer. A unique address can be used to identify each sensing element to obtain the respective output at any time. The ADC generates an 8-bit digital equivalent of the sensed analog signal connected to its input. With the Universal Asynchronous Receiver Transmitter (UART) the ADC parallel output is converted into serial form and prepared for serial transmission by using the RS232 signal level converter to the Embedded Linux Platform for processing, monitoring, and possibly storage.

3.3 User interaction

Through a local or remote access client such as **telnet** or **ssh,** the user can be able to access the sensors to obtain information about the different physical conditions in a home. Some of the conditions can be temperature level, the ON/OFF state of electric appliances (ON/OFF status), general state of the house security for example whether the house main doors are locked, monitoring energy utilization, keeping track of the indoor luminosity to facilitate regulation, detecting when there is fire outbreak, detecting forced entry to the house, and detecting when there is a water leakage. Examples of these applications are illustrated with the aid of Figure 3.4.

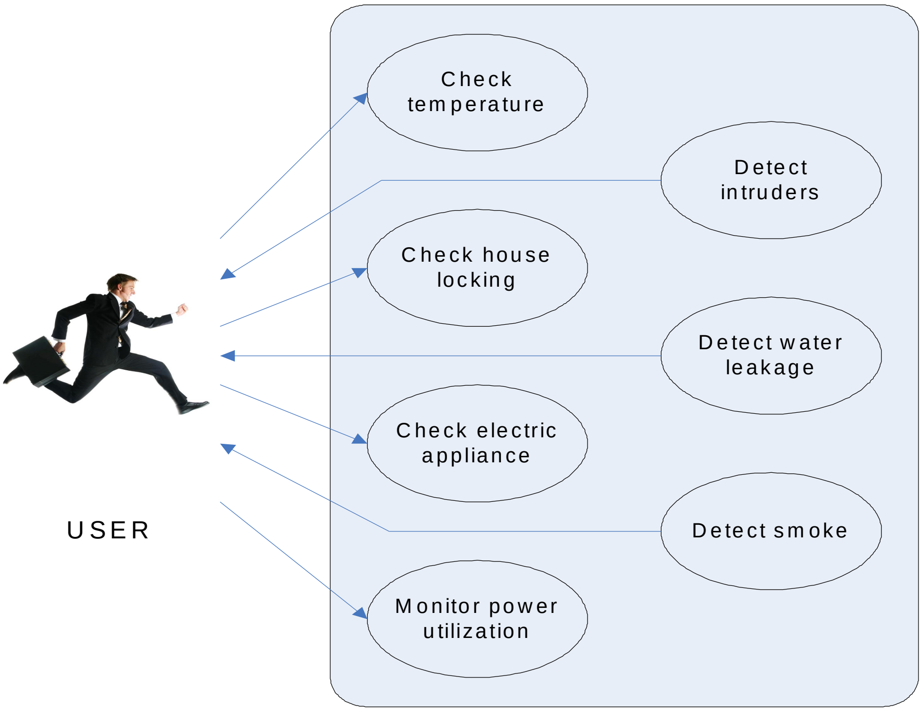


Figure 3.4 Possible use cases

Applications with the home monitoring sensor network in Figure 3.4 are listed here with explanations.

I. Check temperature

When the user selects to check the temperature level, the system should respond with some value that represents the level of temperature around the temperature sensing element. The application can incorporate more than one sensing element to enable temperature measuring in more than one place.

II. Monitoring Power Utilization

When the user selects to check the power utilization, the system responds with HIGH when the power consumption is higher than normal and NORMAL when within the limits. If the displayed message is HIGH then the user will be expected to check whether there is any electric appliance that has been left ON by mistake. Each electric appliance should be fitted with a sensing element around the power supply cable to generate a signal proportionate to the amount of current being drawn. When there are more appliances the total consumption would be the sum of the individual consumption, which is then compared with the normal average consumption to determine whether it is within the limit.

III. House Locking

The user should be able to obtain information on whether the house main door is locked. The display message should be either LOCKED or UNLOCKED and if there has been any attempted forced entry the system should inform the owner by a short text message where possible. Door opening and locking is done by operating electromagnetic contacts which when operated, the contacts would energize to open the door and de-energize when locking the door, and the different contact positions are detected by a micro switch that is fitted to indicate the position of the lock.

IV. Monitor electric appliances

The user should be able to check whether there are any appliances in the ON state and which should be OFF or those that are OFF and should be ON. The system displays all the electric appliances in the house with their respective status i.e. either ON or OFF. A switch operated by a magnetic flux caused by the current flowing in the appliance's supply cable can be used to detect when the electric appliance is ON or OFF.

V. Detect intruders

In case the system experiences events such as forceful entry, it should be able to immediately signal the owner through sms where possible or by activating an alarm circuit [8]. When the owner is not within reach the system should activate an alarm and at the same time send a text message or make a phone call to the security company assigned to the home. Phone calls and messaging alternatives are only possible when incorporated in the design. The system uses sensors that include magnetic contacts to detect the opening of doors and windows, and surveillance cameras for motion detection [15]. Sensing elements can also be strategically placed on the window panes to detect when the glass breaks in the event of forced entry.

VI. Smoke/Fire detection

The system detects when there is a fire potential through smoke detection. It activates an alarm circuit when smoke is detected, and at the same time signals the owner by sms or phone call depending on the implemented design. For smoke detection a set of smoke detectors placed at strategic points are used. It should be possible to efficiently locate the precise point or position of the activated detector. Once activated, the alarm switch stays ON until deactivated by the owner. Activated alarms should not stop other alarms from switching ON incase smoke is detected in a different location.

VII. Detect water leakage

When water spills on the floor due to a leaking pipe, valve, or an overflow in case of a valve failing to close, the condition should be detected immediately followed by an alarm signal. Where possible the system can also send a text message to the owner. By placing bare conducting materials supplied with a low voltage at vulnerable points any trace of conducting liquid such as water can be detected. There is water present whenever there is current flow however small and no water when there is no current.

4 System Design

It is suggested from the previous studies about home monitoring networks that cost is one of the main factors limiting their wider acceptance and application [2], [5], [29]. Software development costs, communication media, and the interfacing (controlling) hardware costs directly affect the overall cost. The design objective for this application is to realize a system that costs less than the available solutions, works independently with minimum interference (reliable), and convenient to work with in terms of accessibility (remote and local).

The interfacing hardware consists of the electronic circuits necessary for sensing, processing, and transmitting the information about the conditions that are being monitored. The system can be designed either to have all the available sensing elements working with a central Analog to Digital Converter (ADC) or for each sensor to have an independent converter at the sensing point. A central ADC design approach should have means of addressing an individual sensor to be able to select between the different sensor outputs.

The resulting signal from the converter can be transmitted in parallel or serially. Parallel transmission has the advantage of high speeds and no signal level conversion is necessary. However, it requires many lines depending on the number of bits used in representing each signal, and only works for very short distances. Serial transmission requires one line for sending and receiving respectively a factor that reduces the speed of transmission because of the conversions from parallel to serial and vice versa. It has the advantage of covering long distance compared with the parallel option. Cost, reliability, and availability are some of the main factors considered while selecting components for building the sensor interface unit.

The software part of the design consists of the controlling software (applications software) and the executing platform (operating system). One of the functions of the controlling software is to read and execute the user requests whereas the executing platform provides means of allocating and controlling the resources necessary for the execution of the controlling software.

4.1 Signal sensing

Appropriate sensing elements should be used to provide electrical output signals that vary with the conditions that are being monitored. In some cases depending on the sensed signal level and condition it might be necessary to carry out signal filtering and amplification before converting the signal into a digital form.

4.1.1 Temperature sensor

Temperature variations can be sensed by using different principles like a varying resistance with varying temperature. The varying resistance provides a voltage drop (voltage signal) that varies with temperature and the continuously varying voltage can be displayed on a scale calibrated in terms of temperature. A thermistor is one example of such a device whose resistance varies with temperature though in a non linear manner. The non linear variation makes it unsuitable for applications that require high accuracies. A thermocouple can be used as an alternative to a thermistor since it generates a voltage proportionate to the temperature.

There exists special integrated circuits specially intended for generating an output signal that varies directly with the varying temperatures and a popular example is the LM34 for the Fahrenheit range and LM35 for the Celsius range. With LM35, temperature can be measured more accurately than using a thermistor or thermocouple. The sensor circuitry for LM35 is sealed and not subject to oxidation, it generates a higher output voltage than the thermocouples that it may not require amplification.

The LM35 is fabricated in a three terminal transistor package and from the datasheet [32], the output signal is 10mV for every 1 ºC corresponding change in temperature above zero. Its rated full range is -40 to +110 ºC, and in Figure 4.1 is a circuit diagram that can be used for temperature measurement application.

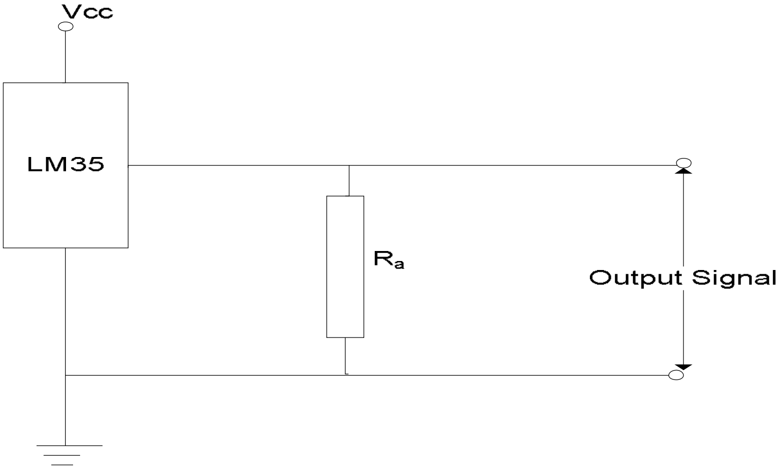


Figure 4.1 Temperature Sensor

For a linear relation between the temperature sensor output and the ADC output, the reference voltage for the ADC should be appropriately set. The reference voltage (V_{ref}) in this application should be set as 1.28V in order to correspond linearly with the 256 binary output steps provided by an 8 bit ADC. This is important especially when dealing with the temperature measurement. Table 1 illustrates the concept of V_{ref} in relation to linearity.

Table 1: Tabulation of temperature versus binary output for a linear temperature sensor and an ADC set up for 2560 mV full scale

Temperature(ºC)	$\mathbf{V_{in}}$(mV)	**Binary O/P**(b_7 to b_0)
0	0	0000 0000
1	10	0000 0001
2	20	0000 0010
15	150	0000 1111
25	250	0001 1001
45	450	0010 1101
65	650	0100 0001
100	1000	0110 0100
110	1100	0110 1110

4.1.2 Water Detector

For water detection, the sensing element generates an output signal (voltage signal) in case water is present in an area where there should be no water for example on the floor. The transistor circuit shown in Figure 4.2 can be used to generate an output signal when there is water or any conducting liquid capable of passing a minimum current between the conducting plates 1 and 2. The current I_B flows in the base-emitter junction as a result of the existence of a conducting liquid between the conducting plates. This switches ON transistor Q and current I_E that flows as a result produces a voltage drop across R_e which implies that water is present. The same signal can be used to activate an alarm circuit. Transistor Q is BC108 and its electrical characteristics are given in the datasheet [33].

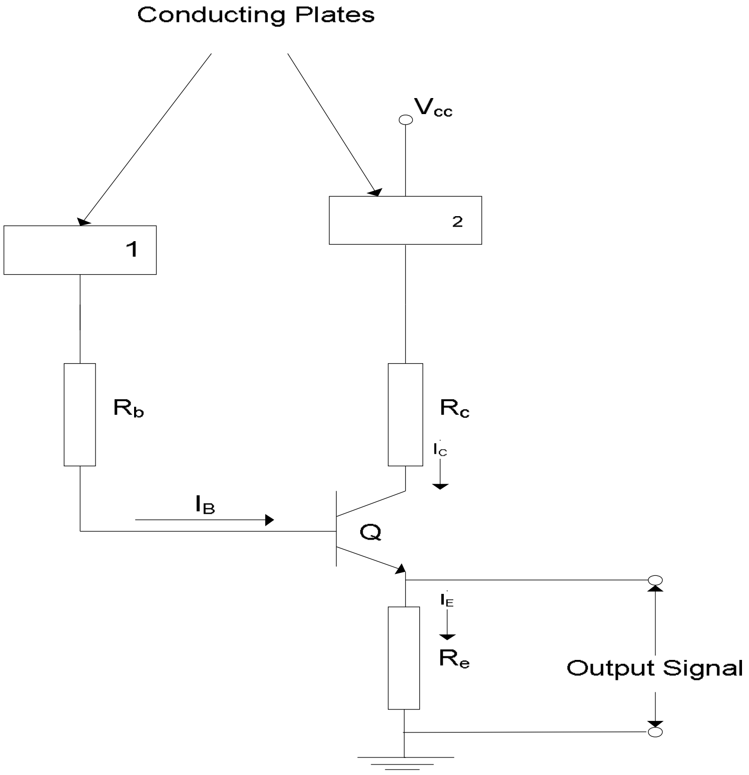


Figure 4.2 Water Sensor circuit

4.1.3 Smoke Detector

There are two basic types of smoke detectors, the photoelectric smoke detector that uses an optical beam to search for smoke, and the ionization chamber smoke detector (ICSD) where the presence of smoke affects the flow of ions between a pair of electrodes in the chamber. ICSD is efficient at sensing a fire that produces little smoke.

According to [16] optical smoke detectors can be of transmission or scattering type. Transmission type operation depends on the change of light absorption, and the other by the scattering of light by smoke particles in the air. The diagram in Figure 4.3 shows Light Dependent Resistor (LDR) circuit used for detecting smoke. An LED (D_L) creates a beam of infrared light (light source) in the smoke detection chamber and the LDR (light sensor) detects this light when there is no smoke. In case of fire for example, smoke particles would scatter the light beam which reduces the amount of light falling on the LDR. Less light increases the resistance of the LDR thus reducing the current flowing in the circuit which in turn reduces the output signal which is the voltage drop across the resistor R_S. Reduction of the output signal or no signal at all implies the presence of smoke, this state is used to activate an alarm and at the same time send a message to the owner.

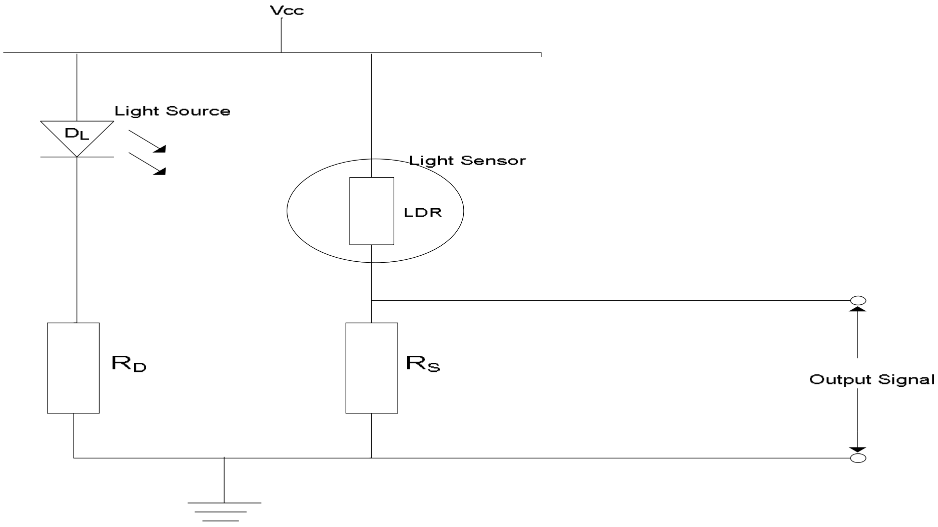


Figure 4.3 Smoke detecting circuits

Instead of placing the light source and light sensor at opposite ends, they can alternatively be placed at the same end with a suitable reflector placed at the opposite end to increase the effective transmission length [16]**.** The sensitivity of the light sensor can sometimes be affected by the presence of dust particles on its surface; it should therefore be kept clean.

The alternative method for detecting smoke is that of using an already built (commercial) smoke detector and measure the current drawn from the supply to determine when smoke is detected. When more current is drawn from the supply it means smoke is present because the detector draws more current for the alarm. A resistor with a small resistance value can be connected in series with the supply line so that the voltage drop across it can be used to indicate when the smoke detector starts drawing current. Under normal circumstances i.e. when there is no smoke there should be no voltage drop across the resistor. A commercial smoke detector with the said modifications is used in this application for smoke detection.

4.2 Signal Multiplexing

Signal multiplexing technique is a design requirement for systems that need to select between different inputs. In this application it enables the system to select a specific sensor output at a time for reading. It also facilitates the utilization of a central Analog to Digital Converter for all the sensors. Each sensor is assigned a unique address for identification and selection.

It requires a multiplexer with many input channels to serve applications with many sensors, the other option is to use a multiple set of multiplexers with small number of channels. An 8 channel multiplexer for example can appropriately serve the number of sensors that are used in this application and Figure 4.4 illustrates an 8 channel multiplexer with A_0, A_1, and A_2 as the sensor address lines.

Sensor signal multiplexing can also be used for scaling the number of sensing elements. For scalability, the sensors should be organized in groups with sensors carrying out similar functions grouped together, for example in this application there are four sensor groups, smoke detection, temperature sensing, water sensing and security sensing (door and window sensors). Sensors in the same group are also expected to use similar methods/functions for sensor signal processing. Spare sensor (extra sensor inputs) slots can be tied to the ground (earth) or can be connected to some special signal source. This is necessary in determining when the sensor is not connected or when there is a new sensor connected for the controlling software to start addressing and reading the new sensor output. The disadvantages here are the difficult with which to predict the number of extra sensors that would possibly be required in any application for future expansion, and the time taken by the software in checking whether there is any new sensor connected. User requirement specifications can be used to estimate the possible number of spare sensor slots to be set a side for expansion.

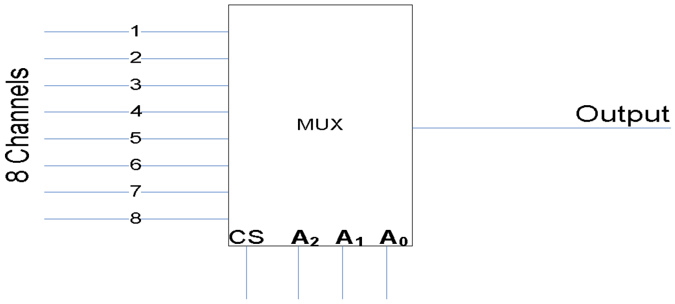


Figure 4.4 8-Channel Multiplexer

Two multiplexers each with 8 channels can be used to serve a sixteen sensor input system as illustrated with Figure 4.5. The multiplexers are connected to use three address lines for sensor addressing and an extra line for selecting between the two multiplexers (chip select CS). The remaining four lines in case of an 8-bit address bus can possibly be used for sending control signals to the connected devices. For example in case of a smoke detector sensing a fire potential, part of the address lines can be used to address a water sprinkler and the rest for switching an alarm circuit. For this application ADG428 analog multiplexer with 8 channels is employed [34].

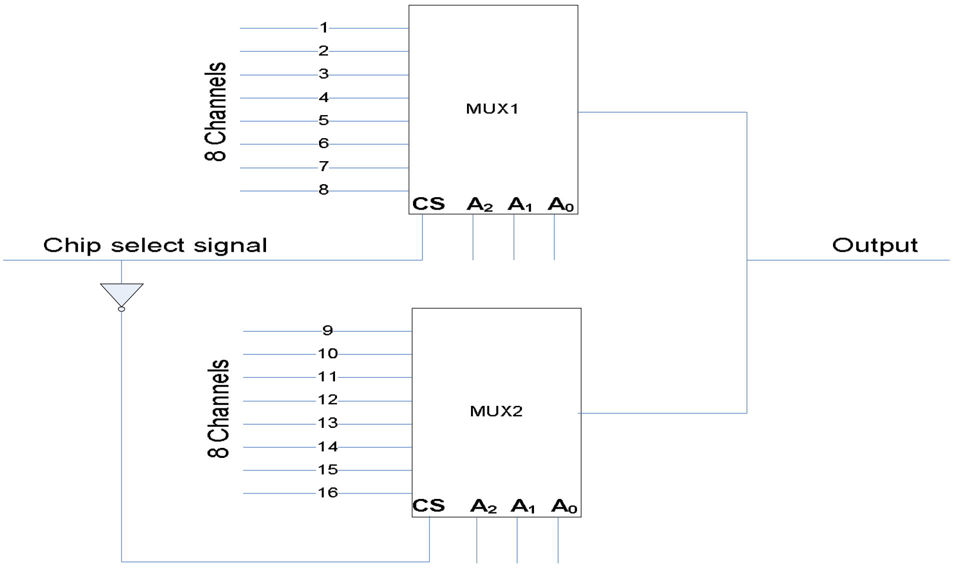


Figure 4.5 Two 8 channel multiplexer serving 16-inputs

An application involving a large number of sensors leads to a complex wiring/routing system and possibly makes the signal noisy. According to [30], noise can be reduced by placing the Analog to Digital Converter at the signal sensing source and communicate the signal serially. Serial data communication often reduces the wiring needed but the utilization of multi-drop serial bus architecture in case of several sensors is often difficult to manage. There are different variants of serial bus communication having different addressing and communication formats with the Dallas One-Wire (DOW) being the only type that uses a single wire for data communication [30], [31]. The second wire in the DOW approach is for the ground (earthing). It communicates data through zero and one timeslots, with messages bracketed with reset and presence scheme which allows the bus master to know whether there are devices (slave devices) attached to the bus or not (i.e. whether there are devices that need to communicate).

The DOW was originally designed for communication with memory devices using a minimal contact count but has now extended to include other applications such as digital temperature sensors, pressure sensors, humidity sensors, and specialized battery management devices. The DOW is capable of operating on cable lengths of up to 300m, this makes it a good candidate for extended sensing applications. Every device within the DOW standard has a 64-bit serial number encoded in its ROM for identification (address). For consistence these numbers are maintained in a central registry at the Dallas Semiconductor.

DOW has advantages that it derives its power from the signal bus (parasitic power), uses inexpensive and flexible devices, and supports multiple devices which are also self configuring on single line. The self configuring factor provides a scalable system and a minimization of interconnections. It however works with relatively low speed, needs an adapter and requires a careful layout to avoid signal reflections. Apart from speed it is an appropriate system for communication since it uses a bus system.

4.3 Analog to Digital Conversion

This study utilizes a centralized ADC and therefore requires an analog multiplexer. The basic function of the ADC is to convert an analog value which is typically represented by a voltage into bits that gives a good approximation of the input analog value. According to [17], [18] the digital conversion process can conceptually be viewed as forming a ratio between the input signal and a known reference voltage $V_{ref,}$ and then rounding the result to the nearest n-bit binary integer as mathematically depicted by the expression

$$D = \text{round}\,(\frac{V_{in}}{V_{ref}} 2^n)$$

Where V_{in} is the input analog, value,

V_{ref} is the reference voltage,

D is the data output word,

and n is the number of bits in D

The “round” function represents a rounding of the terms in the parenthesis to the nearest integer. The reference voltage can be generated internally as is the case with commercial converters or may be externally supplied. Figure 4.6 gives a pictorial view of the concept. The resolution of the converter’s approximation is better when n is larger so as to give a smaller rounding error [17].

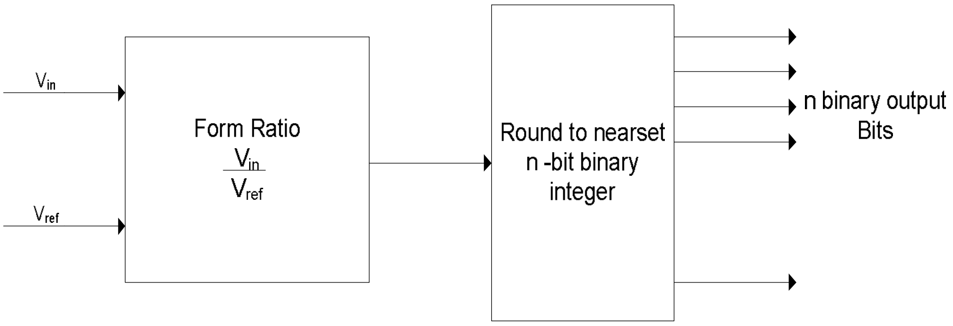


Figure 4.6 Conceptual view of the function provided by an ADC

In this application ADC0801 is used and Figure 4.7 shows its functional block diagram configured for continuous conversion.

It is possible to change the analog input voltage range to other values other than 0 to 5 V by using the $V_{ref}/2$ input (pin 9). Normally the $V_{ref}/2$ input is not connected, and it is kept at 2.500V (V_{cc}/2) [18]. The V_{ref} in this case is 2.56 volts and $V_{ref}/2$ is 1.28 volts.

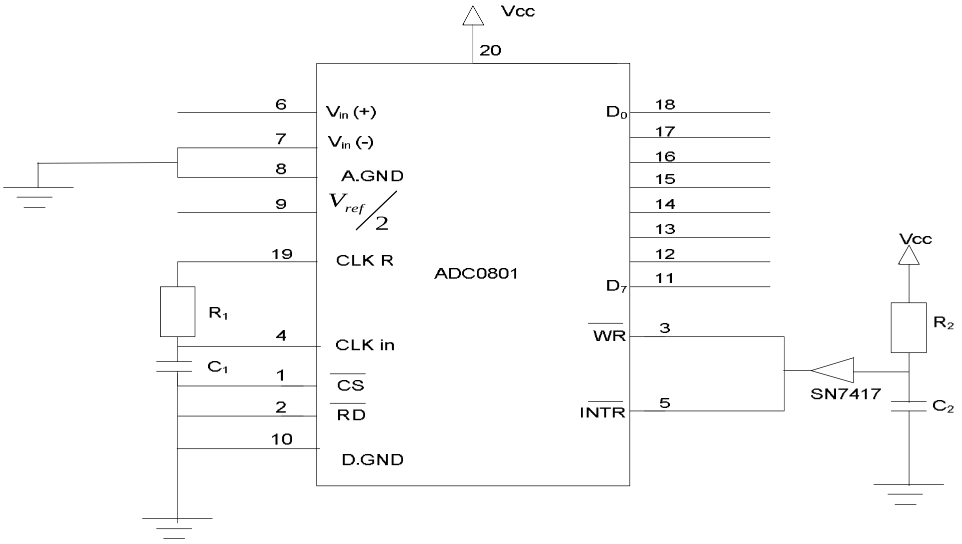


Figure 4.7 ADC0801 Block Diagram

Pin functions for the diagram in Figure 4.7 are listed here below.

$\overline{WR}$	Active LOW, Start Conversion
$\overline{CS}$	Active LOW, Chip Select
$\overline{RD}$	Active LOW, Output Enable
$\overline{INTR}$	Active LOW, End of Conversion
A.GND	Analog Ground
D.GND	Digital Ground
$V_{in(+)}$, $V_{in(-)}$	differential analog inputs (ground one pin for single ended measuring)
$V_{ref}/2$	Optional reference Voltage (Used to override the reference voltage V_{cc})
V_{cc}	5Volts Supply and assumed reference Voltage
D_0 to D_7	Digital Outputs

The R_1C_1 circuit determines the clock frequency, R_1 is 10KΩ and C_1 is 150pF. $\overline{CS}$ and $\overline{RD}$ control pins are connected to ground to enable the ADC chip all the time. The conversion starts by setting the $\overline{WR}$ pin to 0 and then to 1 immediately. The connection from $\overline{INTR}$ to $\overline{WR}$ causes the ADC to start a new conversion each time the $\overline{INTR}$ (end-of-conversion) line goes LOW. The RC circuit with the SN7417N open-collector buffer issues a LOW-to-float pulse at power-up to ensure initial startup [35]. The Analog signal from each sensor is connected to pin 6 at a time for conversion to a digital format. Datasheet for the ADC0801 provides detailed information about pin connection [37].

4.4 Signal Transmission

To transmit the sensed signal serially, it should first be converted for the parallel output format of the ADC to serial. For processing at the receiving end it should first be converted back to the parallel format. A Universal Asynchronous Receiver Transmitter (UART) circuit can be used to convert between parallel to serial and vice versa before serial transmission. The serial signal requires some signal level conversion so as to be conveniently transmitted over the serial link and an RS232 signal level converter circuit is used for the voltage level conversion.

4.4.1 Universal Asynchronous Receiver Transmitter

The purpose of the Universal Asynchronous Receiver Transmitter (UART) is to receive a serial transmission and convert it into parallel data and take parallel data and transmit it as a serial transmission. It prepares the received data for processing by converting the serial start, data, parity and stop bits into parallel form. The transmitting part converts parallel data into serial form and automatically adds start, parity and stop bits. The word length to be serially transmitted can be 5, 6, 7 or 8 bits with odd, even or no parity bit. The stop bit may be 1, 2 or 1.5 depending on the word length being transmitted.

The parallel data to be transmitted by the UART is the digital equivalent of the sensor output signal and the serial data that it receives in this case will be the address information for each sensor to facilitate sensor selection. The UART is a 40-pin chip, and there exist a variety of different types with UART 16550 and 6402 series serving as examples. UART6402 is used for this application and the main reason for selecting the 6402 UART is because it has a separate receive and transmit data buses, and the other reason is that connecting pins to various logic levels can easily configure the UART [39]. However, it requires an external baud rate generator, and its frequency should be 16 times faster than the baud rate of the serial transmission. It is therefore necessary to generate a clock frequency for the communication. For this application the communication is set for 9600 baud (bits/sec) implying that the clock frequency must be 153.6 KHz (16x9600). One alternative to realize a baud rate generator is to use a 555 timer as astable multivibrator but variation in frequency due to temperature changes makes it unsuitable since only a variation of 3% is allowed. A suitable option is to use a crystal oscillator since it is more stable.

There is no crystal oscillator for 153.6 KHz but by using a 2.4576 MHz oscillator and putting the generated signal through a binary counter it is possible to generate a 153.6 KHz oscillation. A range of frequencies can be obtained by using a number of binary counter stages and Table 2 can be used as a guideline for the frequency selection and Figure 4.8 shows the 2.4576 MHz crystal oscillator with a 14-stage binary ripple counter circuit. Details about the respective output frequency at each counter stage together with the pin configuration for the HCT4060 14-stage counter can be obtained from the datasheet [40] and [41] for the 2.4576 MHz crystal oscillator.

Table 2 Baud rate with corresponding frequency

Baud(bits/sec)	**Clock Frequency(KHz)**
300	4.8
1200	19.2
4800	76.8
9600	153.6

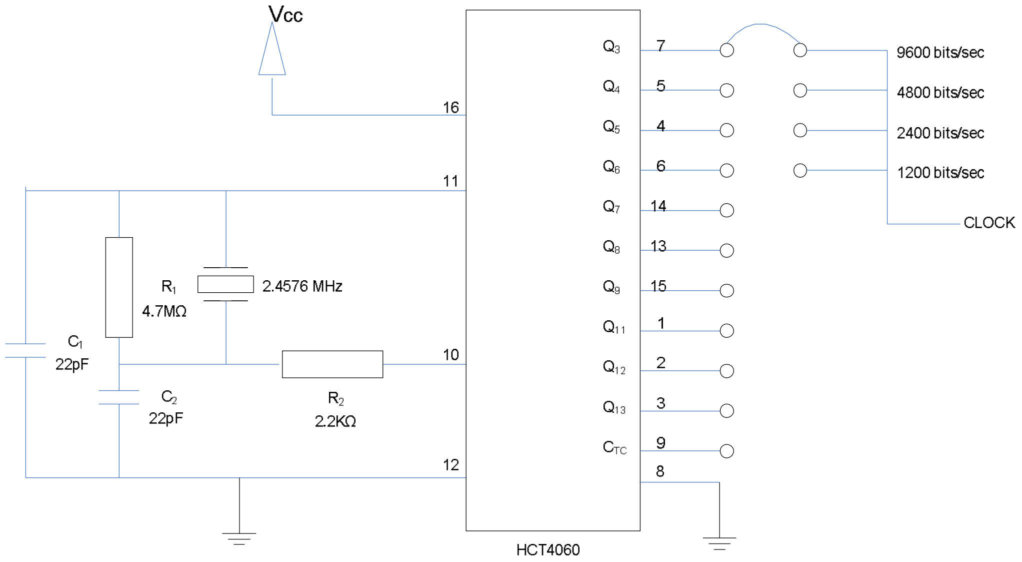


Figure 4.8 Baud rate generating circuit

Figure 4.9 shows the pin chart for the UART HD-6402 and the pin configuration details can be obtained from the datasheet [39]. In this application the UART is configured for 8 data bits, 1 stop bit, and no parity checking. The unused pin 2 is taken low.

4.4.2 RS-232

The RS232 is one of the most commonly used standards for serial communication between computers and peripherals or to other computers. It is one of the standard types of interface which is also supported by computer operating systems and a variety of software tools. With this form of interface the instruments are physically independent of the computer and can be optimized for measurement performance. The other advantage is that the instrument can be remote from the computer.

For serial communications each word of data that is sent or received is sent one bit at a time. Commonly used terms with serial communication are **mark** for the ON state and **space** for the OFF state. Serial RS-232 communication works with voltages (-15V...-3V) for HIGH and (+3…+15V) for LOW. These voltages are not compatible with normal computer logic voltages. On the other hand, classic TTL computer logic operates with voltages in the range of 0V…+5V (approximately 0V…+0.8V for LOW, +2V…+5V for HIGH). Modern low-power logic operates in the range of 0V…+3.3V or even lower. RS-232C is the commonly used standard where a mark bit is defined as a voltage between -12V and -3V, and a space bit as a voltage between +3V and +12V. It is evident that the maximum RS-232 signal levels are far too high for computer logic, and the negative RS-232 voltage cannot be understood at all by computer logic. Therefore after receiving serial data from an RS-232 interface the voltage has to be reduced, and the LOW and HIGH voltage level inverted. Also when sending computer logic signals (TTL) over the RS-232 the logic voltage has to be "pumped up", and a negative voltage has to be generated too. Sipex SP3232 chip is used to carry out these voltage level conversions in this application, and Figure 4.10 shows a typical operating circuit.

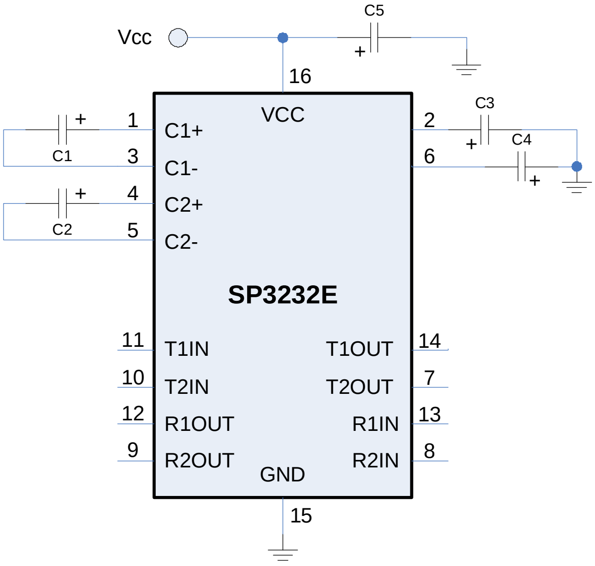


Figure 4.10 SP3232E Typical Operating Circuit

Pins 10 and 11 are for logic inputs whereas pins 9 and 12 are for logic outputs. Pins 7 and 14 gives RS-232 outputs for the respective logic signals at the input pins, and pins 8 and 13 takes RS-232 inputs for conversion to the logic level.

The logic input/output is represented as 0 volts for ‘0’ and 5 volts for ‘1’. From the datasheet specifications in [42], the SP3232E series is ideal for +3.3V systems mixed with +3.3V to 5.5V systems.

4.5 The controlling software

Open source software can be used in some applications to reduce the overall cost of the applications software. For this application OpenWrt [27] which is a GNU/Linux distribution for wireless routers will be used to provide a platform for executing the controlling software. Initially OpenWrt started as a replacement firmware for the Linksys WRT54G and compatible (Broadcom chipsets), but currently it is being ported to other platforms. Most popular routers with OpenWrt seem to be the Linksys WRT54GL /GS and the Asus WL500G.

To build and install additional packages in OpenWrt system it requires a desktop computer with installed recent GNU/Linux distribution, GNU make, and GNU gcc. OpenWrt Software Development Kit (SDK) should first be installed on the desktop to facilitate cross-compilation. The SDK contains a precompiled version of the complete toolchain, libraries, and header files to cross-compile application for OpenWrt. An appropriate download for the OpenWrt SDK can be obtained from [51].

4.5.1 WRT54GL/GS Linksys Router and the OpenWrt Platform

The Linksys router has various features that can be manipulated to further extend its functionalities for a wider application. It has a wireless access point that connects fast Wireless-G (802.11g at 54Mbps) and Wireless-B (802.11b at 11Mbps) devices to the network, a 4-port full-duplex 10/100 Switch to connect wired-Ethernet devices together thus providing room for expanding the networks. Figure 4.11 shows the Linksys WRT54GL wireless router.

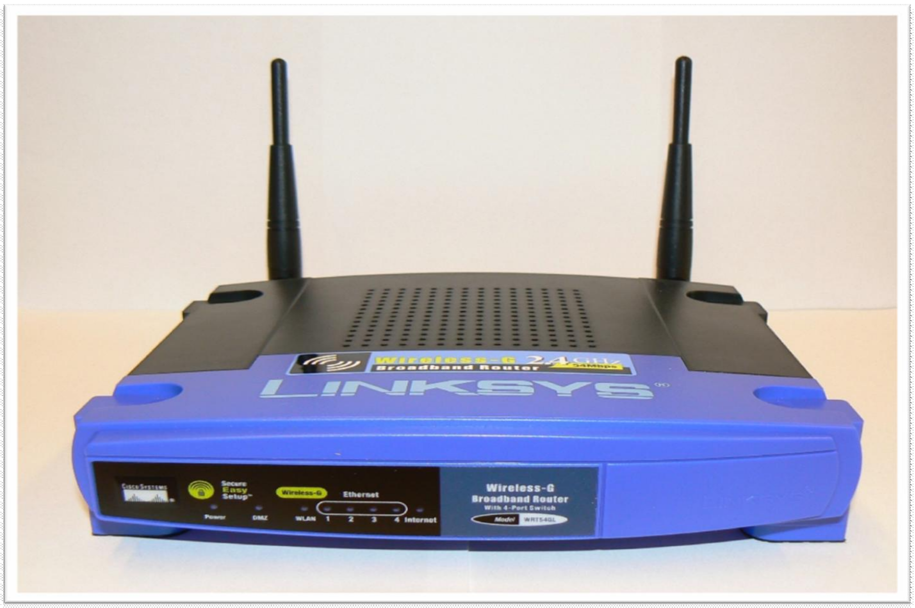


Figure 4.11 Linksys WRT54GL Router

Other details about the board information and the CPU model for the WRT54GL v1.1 Linksys router that is particularly used for this application are 200MHz CPU, 4MB flash memory, 16MB RAM, no USB, 2xRS-232 serial ports and the JTAG port [50]. The Router function ties all the features together to enable the whole network to share a high-speed cable or DSL Internet connection.

OpenWrt is said to exhibit various characteristics some of which are stated in [27]. It employs writable file system which means that firmware is no longer a static compilation of software but can instead be dynamically adjusted to fit specific needs. It gives freedom from the application selection and configuration provided by the vendor and allows the user to customize the device through the use of packages to suit any application requirement. OpenWrt turns the Linksys WRT54G device into a Linux box with OpenWrt acting as the distribution with almost all traditional Linux Commands and package management for loading extra software for more functionality. It thus provides a framework to build an application without having to build a complete firmware.

Instead of having only a web-controlled wireless access point, OpenWrt provides a fully interactive Linux system with some notable features such as the ability to telnet/SSH the router to facilitate easy accessibility. Either JFFS2 or SquashFS files systems can be used for the firmware but JFFS2 is more appropriate because it enables a fully writable file system and it is also simpler to work with than its counterpart.

The Linksys router WRT54GL/GS together with OpenWrt provides a platform for executing the controlling software. This eliminates the need for a PC specifically for the home monitoring application.

Utilization of the built in serial port on the router can be achieved by soldering a 10 pin IDC male PCB-mount header at JP2 on the board in case it is to be used as a peripheral serial port. For this application the signal carrying lines are directly soldered to the receiving and transmitting terminals of the router's serial port. External accessibility of the router for communication is achieved through Ethernet connections provided by the available four ports. Figure 4.12 shows the JP2 for the serial ports connection, and the JP1 for the JTAG port. The RS232 ports on board can be accessed for communication.

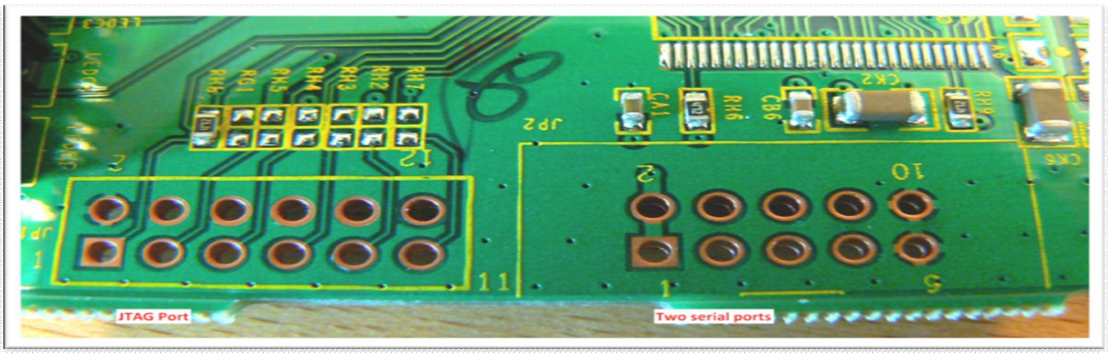


Figure 4.12 Serial ports JP2 and the JTAG port on board

The pin connections of JP2 (serial port) on the Linksys PCB together with the respective functions are as shown in Table 3 [49]. Devices connected to the serial port of the router can also have an access to a supply voltage of 3.3 volts from pins 1 and 2. Pins 7 and 8 are not connected.

Table 3 Serial port pin functions

Serial Port 1 (tts/0)			Serial Port 2(tts/1)		
Pin	**Function**	**Explanation**	**Pin**	**Function**	**Explanation**
2	Vcc	3.3volts Supply	1	Vcc	3.3volts Supply
4	Tx	Transmit data	3	Tx	Transmit data
6	Rx	Receive data	5	Rx	Receive data
8	-	Not connected	7	-	Not connected
10	GND	Ground	9	GND	Ground

4.5.2 The monitoring server software

The communication between the user and the sensing elements is based on the client-server architecture. With **telnet** or **ssh** clients, the user can be able to connect with the home monitoring server. Once a connection is established, the user is then able to communicate with the sensors by issuing properly defined requests. Message Sequence Charts (MSC) in Figures 4.13 and Figure 4.14 can be used to visualize the communication between the user and the server.

In Figure 4.13, the user gives the sensor name and the server addresses the selected sensor in order to read its output. Once the server reads the sensor's output, it returns either a value if the sensor is used for measuring temperature or a state of a condition for example smoke or no smoke in case of a smoke detector. The server returns an error message if it fails to access the named sensor.

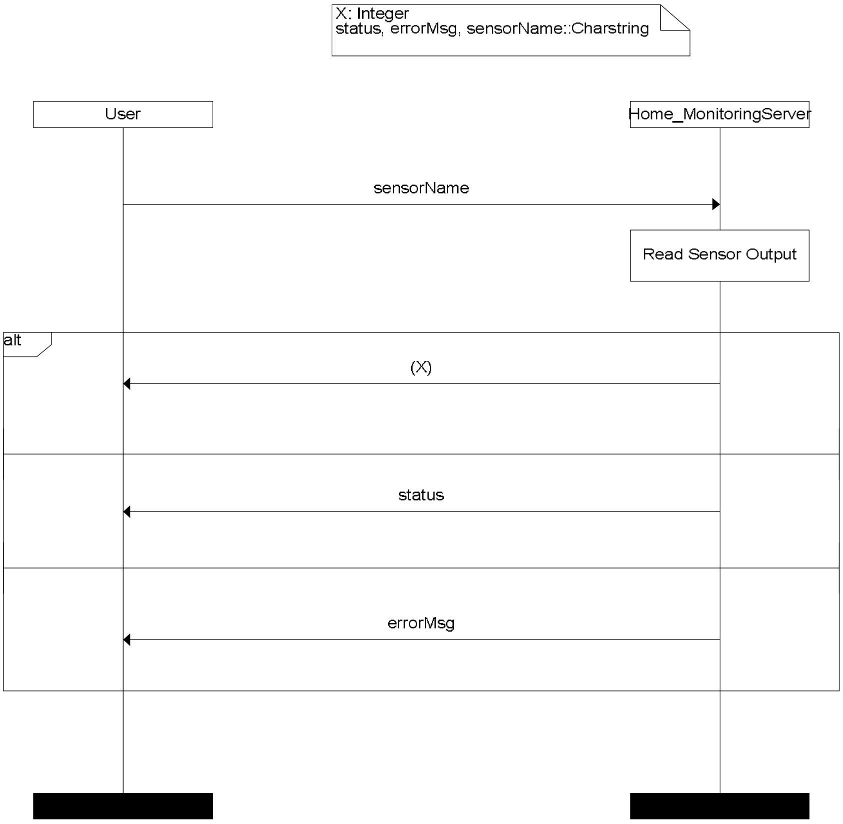


Figure 4.13 Home Monitoring Server response for each sensor

In Figure 4.14, the user can request for help, record or can decide to quit the communication. When requested for help, the server returns instructions about the usage. Since the server keeps record of the sensors' outputs, the user can always access the stored information about the sensors. When the user requests for the stored sensor information, the server returns the sensor name and address together with the respective outputs at specific times. The user has the option of closing the communication at any time by issuing a "quit" command. Once the server successfully closes the communication after receiving the quit command it returns a confirmation or an error message when it fails.

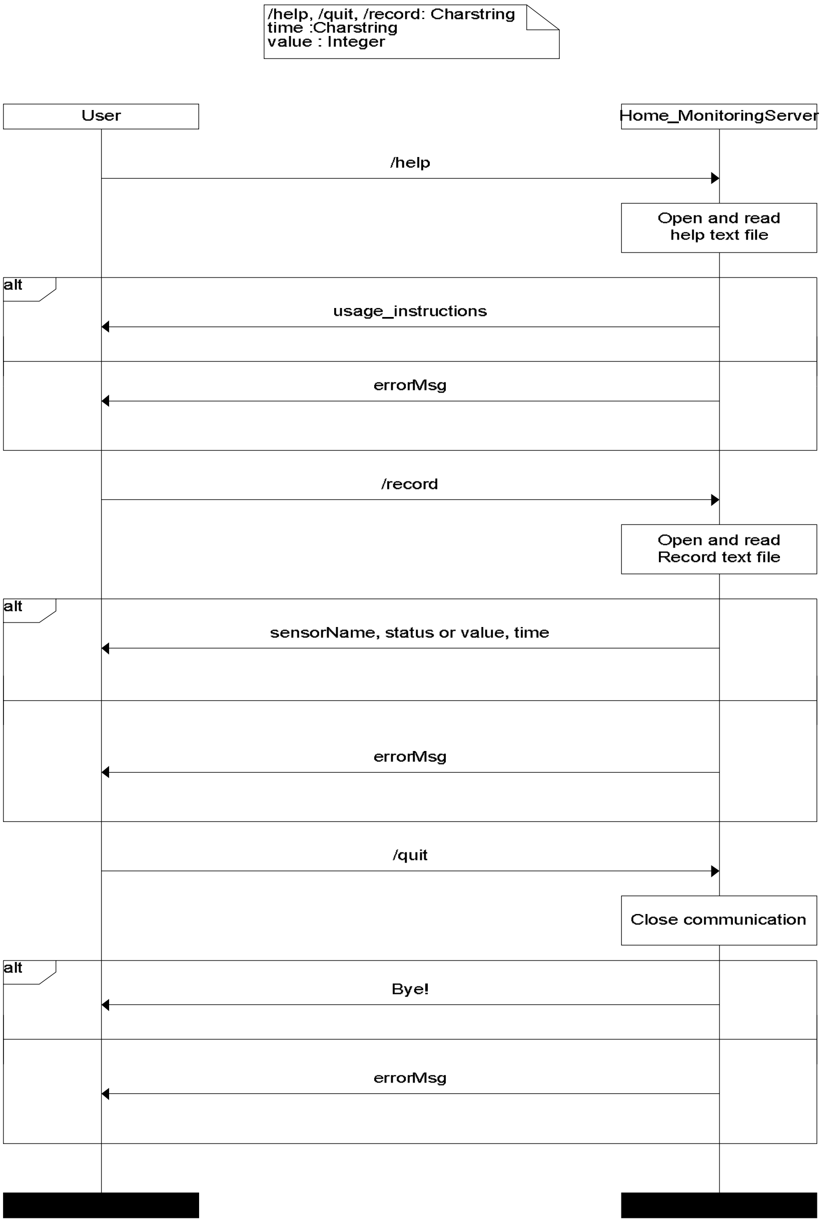


Figure 4.14 Home Monitoring Server extra functions response

The serial port requires appropriate configurations to facilitate effective and efficient communication between the home monitoring server and the sensor interfacing unit. Basically there are two modes in which the serial port can be set to operate, canonical and noncanonical [43], [44]. In the canonical mode, the input is processed as lines whereas in the noncanonical mode the input characters are not assembled into lines and some input character processing is turned off. For this application the serial port is set to operate in noncanonical mode with the following settings:

-The input and output speeds are set to 9600 bits/sec.

-For the serial port control flags, enable the receiver and set to local.

Set the serial port to operate in raw input/output mode with the character size set to 8 bits, with no parity and 1 stop bit.

-Set the input control flags as follows:

Clear ISTRIP to get all the eight bits, also clear input parity checking (IPCK)

Clear IEXTEN to turn off extended character processing

Clear IXON so that there is no flow control

Clear BRKINT

-Set the output control flags as follows:

Clear OPOST to turn off output processing.

-Set local flags as follows:

Clear ECHO to avoid echoing back characters.

Clear ISIG.

-Set control characters as follows:

Set TIME to 0, TIME specifies an inter-byte time.

Set MIN to 0, makes the read function to return without blocking

Information about the different flags governing the operation of the serial port and details about their configurations can be found in [43], [44]. The flags are, input flags, output flags, control flags and the local flags. For setting the control characters there is a special character array with enough space to hold all the special characters, and their number is said to be between 15 and 20 [43]. The code for the serial port configuration and the monitoring server is compiled and uploaded to the router for execution. The monitoring server uses a tcp socket to serve the connecting telnet clients.

4.5.3 The controlling software operation description

The server opens and configures the serial port for reading and writing. Thereafter it starts listening for the connecting clients. Once a connecting client is accepted the server keeps record of its file descriptor before starting to receive and process the requests. From figures 4.13 and 4.14 the requests can be polling a specific sensor, asking for help about the usage of the application, accessing the recorded sensors' output information or stopping the communication.

Each sensor is assigned a specific address for identification. When the user gives the sensor name, the server locates the corresponding address and uses it to select the named sensor. The server invokes a function for addressing and reading the named sensor with the serial port file descriptor and sensor address as its parameters. Shown in appendix is a C coded function for addressing and reading sensor outputs. Note that the same sensor is addressed twice and read twice with some delay in between. This gives time for the transients to settle down to keep the sensor output steady so as to obtain the correct signal value. Thereafter the server takes the average of the sensor output values obtained during the second reading before using it for processing.

To maintain a textual database for the sensors' signals, the server reads the output of all the sensors regularly at some given time intervals and records the information in a text file. The time between each record keeping event is set by the user and is always given in minutes with 1 minute being the minimum time. The server automatically deletes the textual database (Outputfile.txt) once the memory occupied by this file is 80KB before starting the process again. The amount of time it maintains the database will depend on the frequency with which the database is updated. To facilitate the reading of the sensors' outputs, the server obtains the sensor address from a text file maintained for keeping the sensor addresses. It also makes use of the sensor reading function to obtain the reading from each sensor. The reading is done systematically starting with the first sensor and the process repeats after a given time interval that has been set by the user. Updating of the database is maintained for as long as the server is kept running.

5 System Implementation and Analysis

The different circuit parts built according to the application requirement specifications are carefully assembled together to realize the overall application circuit. Sensing elements are not part of the sensor interface unit and relevant information about each sensing element has been given in the design chapter. The interfacing unit is designed to work with many different types of sensing elements and it incorporates a multiplexer to provide means for isolating each sensor output. The final and complete circuit diagram for the sensor interface unit is shown in Figure 5.2.

The resulting circuit works with a supply voltage of 5volts obtained from a voltage regulator with an input voltage of 12 volts from the Linksys router's power supply. Figure 5.1 shows the 5 volts voltage regulator circuit.

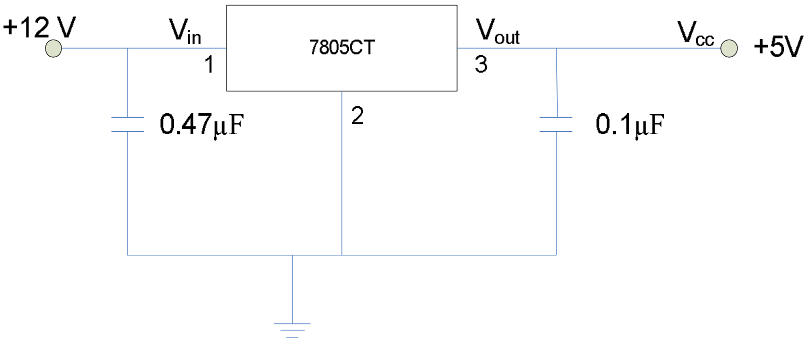


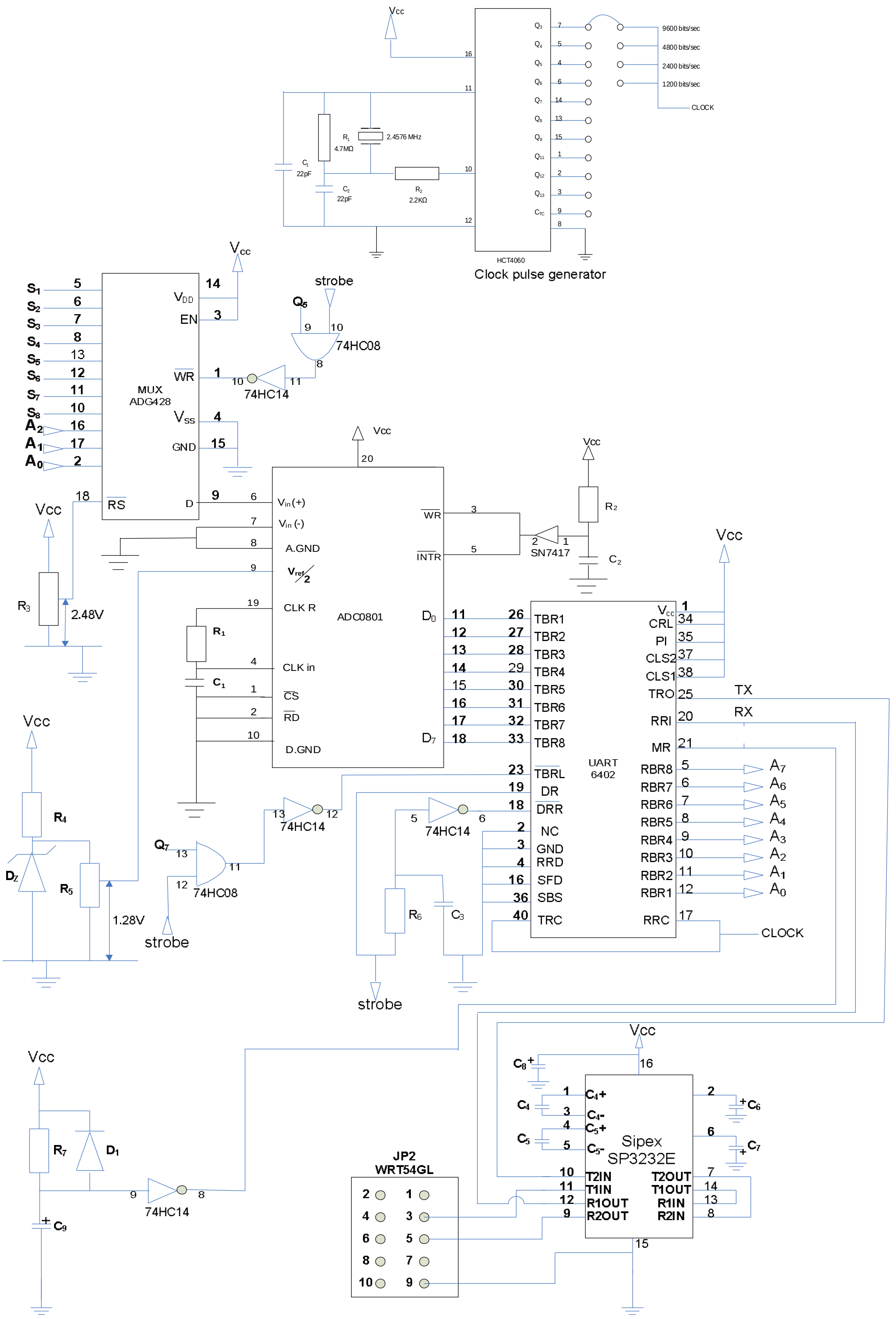
Figure 5.1 Circuit diagram for the 5 volts supply

5.1 Circuit operation

The circuit in Figure 5.2 for the sensor interface unit becomes active immediately it is supplied with power. Sensing elements' outputs are connected to the input terminals S_1 to S_8 of the analog multiplexer ADG428. Unused multiplexer channels should be connected to the ground potential. Each input to the multiplexer is connected to the output through pin 9 (D) depending on the address signal A_2, A_1, A_0. The address signal is part of an eight bit address that is used by this system. Only three bits are effective in this application since the application is using an 8-channel multiplexer. The address signal gates the appropriate channel once the clocking signal connected to pin 1 is made available. Activation is determined by the pulse Q_5 from the clock pulse generator circuit together with the strobe signal that temporarily exists after the UART receives a serial signal. The strobe signal is obtained from pin 19 of the UART and is normally used to indicate when a character is received. It exists for a period of time determined by R_6C_3. The baud rate generator circuit facilitates a transmission rate of 9600 bits/sec, it also provides the Q_5 and Q_7 clock pulses that are used for pulsing the multiplexer and the UART at appropriate intervals.

The multiplexer output signal is next connected to pin 6 of the analog to digital converter for conversion to digital form. The resulting digital equivalent of the sensed signal is obtained from the ADC pins 11 to 18 with pin 11 holding the LSB. The UART transmits the parallel data connected to pins 26 to 33 once it receives the activating signal at pin 23. Clocking pulse Q_7 and the strobe signal determines when to transmit the signal serially through pin 25 of the UART. Serial signal from pin 25 is fed through a signal level converting circuit SP3232E to provide a signal with a voltage level that is within the specifications of the WRT54GL serial port i.e. 0 V and 3.3 V. After the level conversion the signal is transmitted to the Rx terminal (pin 3) of the router's serial port. Information from the serial port is transmitted through the Tx terminal (pin5) through the signal level converter to pin 20 of the UART. The UART gives the received serial signal in parallel through pins 12 to 5 to form the address A_0 to A_7 as shown in the UART part of Figure 5.2.

Figure 5.3 shows a prototype of the intended home monitoring sensor interface unit. The server executes on the desktop computer that is running on Linux operating system and it communicates through the computer's RS232 serial port. The final and functioning circuit is shown in Figure 5.4 together with the Linksys router.



R_1, R_2, R_3 and R_5 are 10KΩ, R_4 = 2.5KΩ, R_6 = 5MΩ, R_7 = 47KΩ, D_Z =2.5 Volts Zener diode
C_1 = 150pF, C_2 = 0.001µF, C_3 = 22nF, C_4, C_5, C_6, C_7, C_8 are 0.1µF, C_9 = 10µF, D_1 = 1N4148 Signal diode

Figure 5.2 Circuit diagram for the sensor interface unit

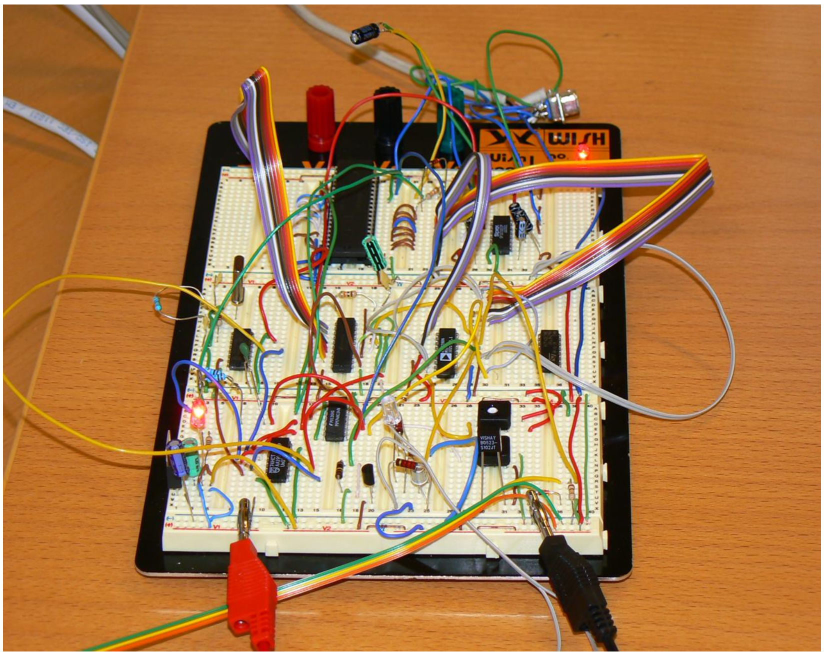


Figure 5.3 Prototype of the Sensor Interface Unit

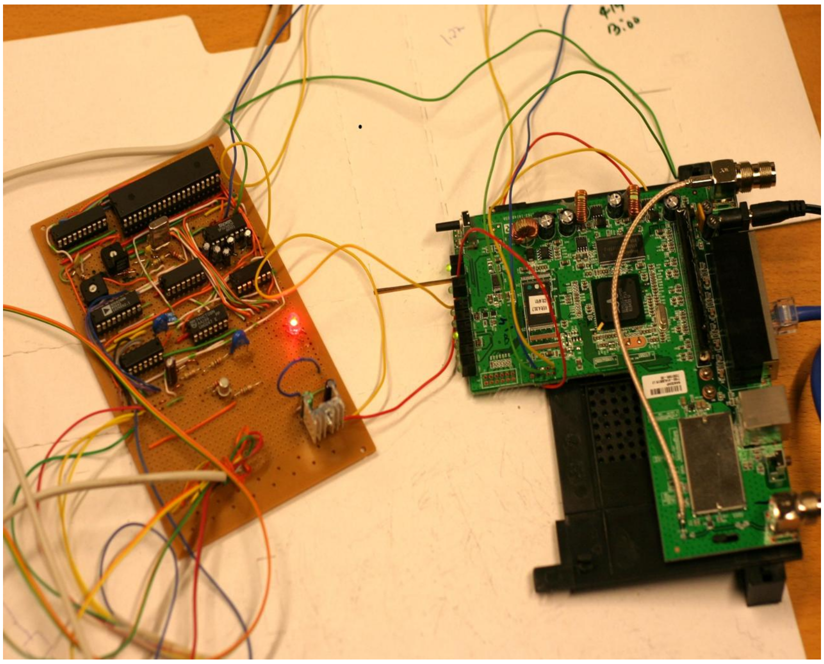
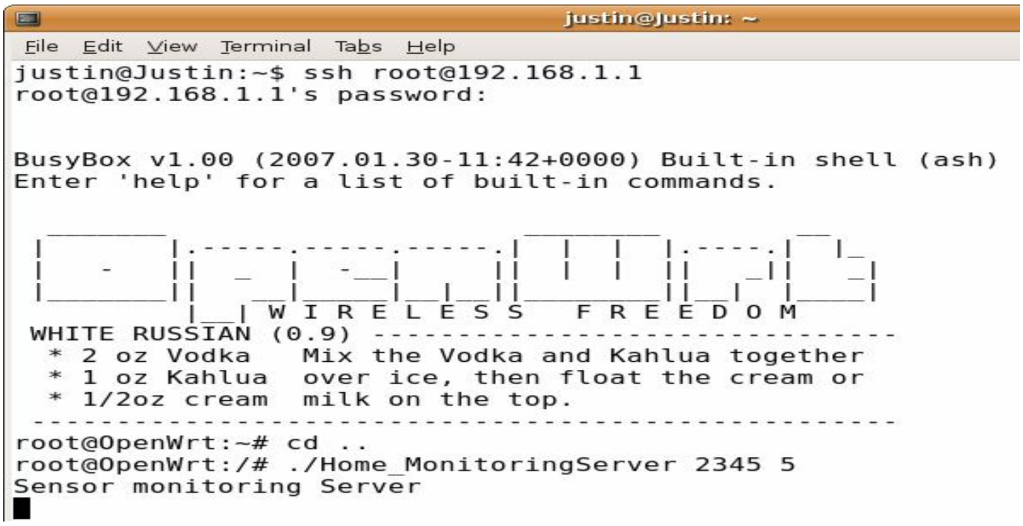


Figure 5.4 A complete and functioning Sensor Interface Unit

Screen shots covering different scenarios are used to give a general picture of the different operation stages of the application. Important events are covered right from the server initiation time to the point where the user receives a response from the sensors. Initially the router is accessed by using ssh client and 192.168.1.1 as its default address and once the login process is complete only then can the server be started. The steps are illustrated in Figure 5.5. The monitoring server in this illustration is started with 2345 as the port address for the connecting clients, and it is set to read all the sensors and record their respective outputs after every 5 minutes.



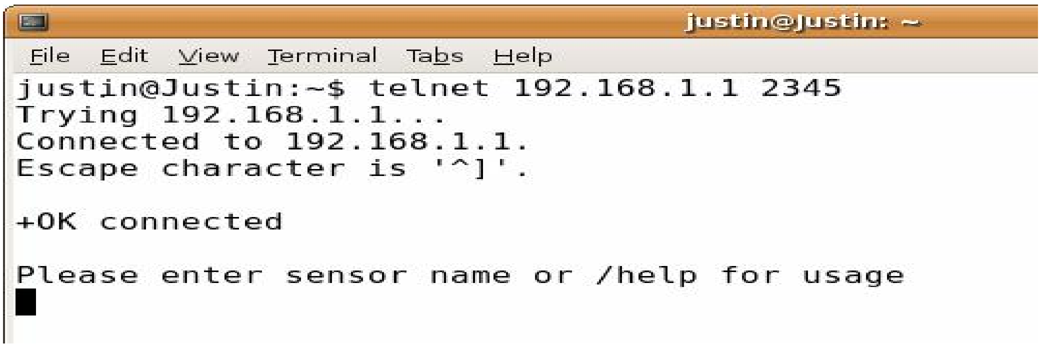
```
justin@Justin:~$ ssh root@192.168.1.1
root@192.168.1.1's password:


BusyBox v1.00 (2007.01.30-11:42+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

  _______                     ________        __
 |       |.-----.-----.-----.|  |  |  |.----.|  |_
 |   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
 |_______||   __|_____|__|__||________||__|  |____|
          |__| W I R E L E S S   F R E E D O M
 WHITE RUSSIAN (0.9) -------------------------------
  * 2 oz Vodka   Mix the Vodka and Kahlua together
  * 1 oz Kahlua  over ice, then float the cream or
  * 1/2oz cream  milk on the top.
 ---------------------------------------------------
root@OpenWrt:~# cd ..
root@OpenWrt:/# ./Home_MonitoringServer 2345 5
Sensor monitoring Server
```

Figure 5.5 Process of starting the Home Monitoring Server

Figure 5.6 illustrates how the user tries to establish a connection to the monitoring server through a telnet client



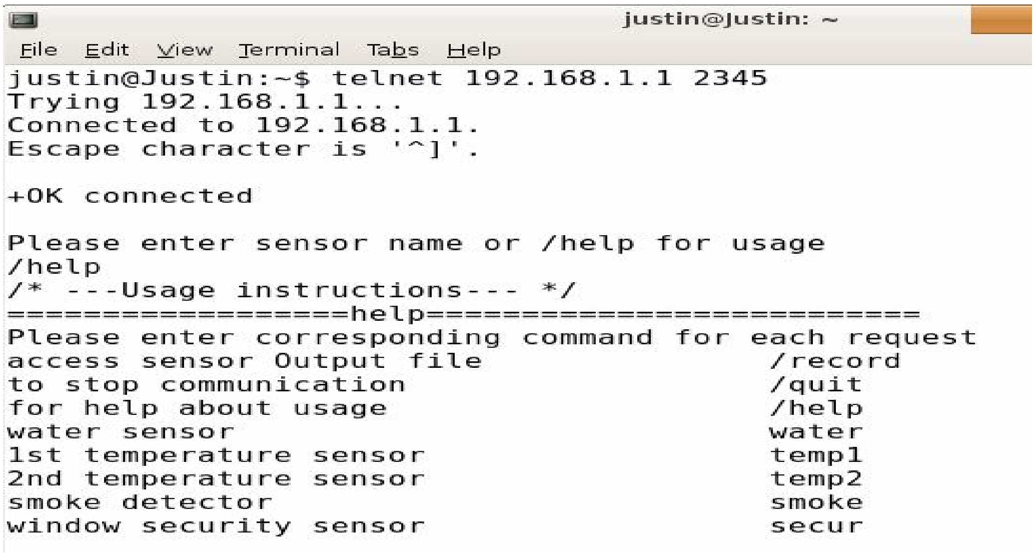
```
justin@Justin:~$ telnet 192.168.1.1 2345
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.

+OK connected

Please enter sensor name or /help for usage
```

Figure 5.6 Connection to the server through telnet

Figure 5.7 shows the usage instructions after the user has successfully established a connection to the server and thereafter requested for help.



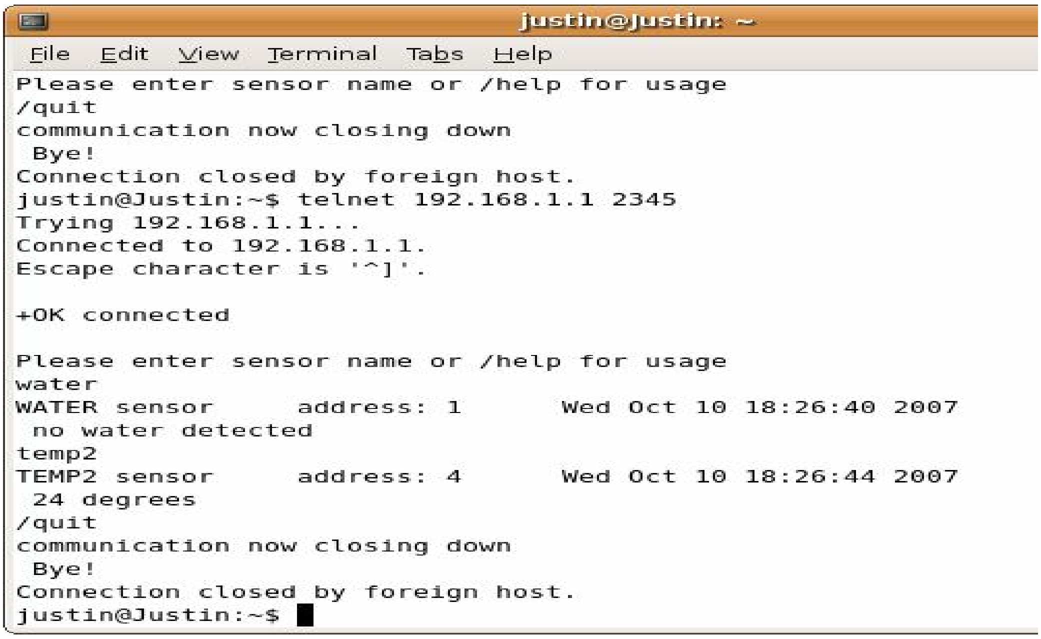
```
justin@Justin:~$ telnet 192.168.1.1 2345
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.

+OK connected

Please enter sensor name or /help for usage
/help
/* ---Usage instructions--- */
==================help============================
Please enter corresponding command for each request
access sensor Output file                 /record
to stop communication                     /quit
for help about usage                      /help
water sensor                              water
1st temperature sensor                    temp1
2nd temperature sensor                    temp2
smoke detector                            smoke
window security sensor                    secur
```

Figure 5.7 The help request

In Figure 5.8 the user enters water and temp2 sensor names to check for water leakage and temperature level respectively before terminating the communication.



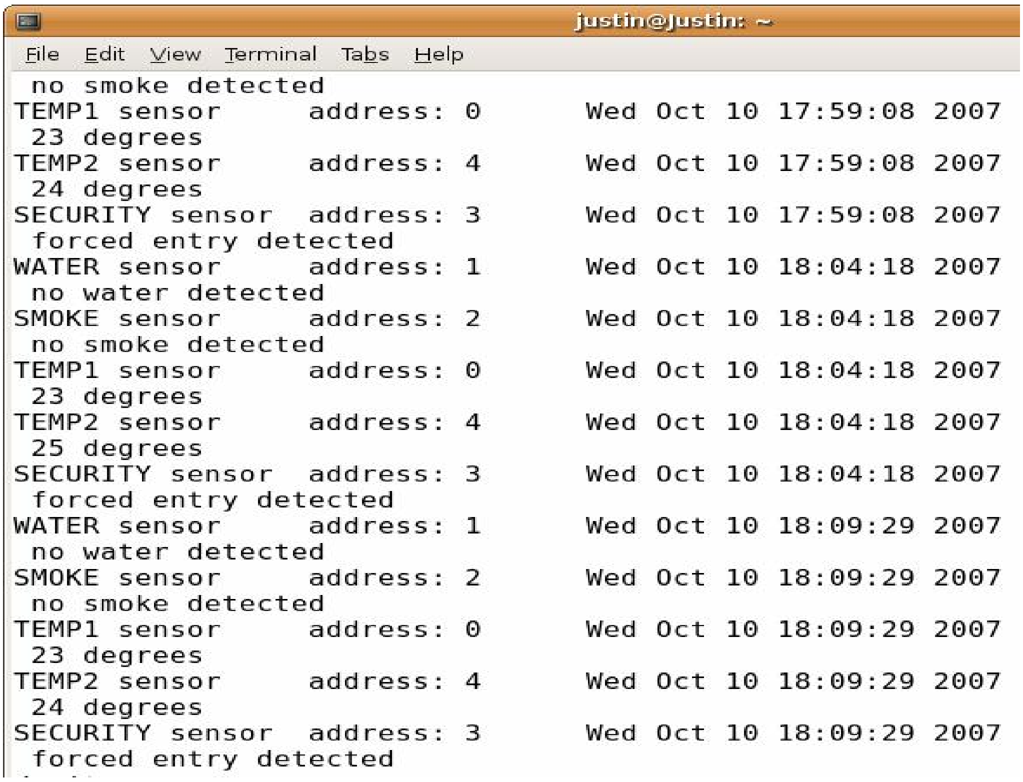
```
Please enter sensor name or /help for usage
/quit
communication now closing down
 Bye!
Connection closed by foreign host.
justin@Justin:~$ telnet 192.168.1.1 2345
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.

+OK connected

Please enter sensor name or /help for usage
water
WATER sensor     address: 1      Wed Oct 10 18:26:40 2007
 no water detected
temp2
TEMP2 sensor     address: 4      Wed Oct 10 18:26:44 2007
 24 degrees
/quit
communication now closing down
 Bye!
Connection closed by foreign host.
justin@Justin:~$ 
```

Figure 5.8 Water and Temperature sensor names

Figure 5.9 shows part of the textual database that the server maintains every after 5 minutes.

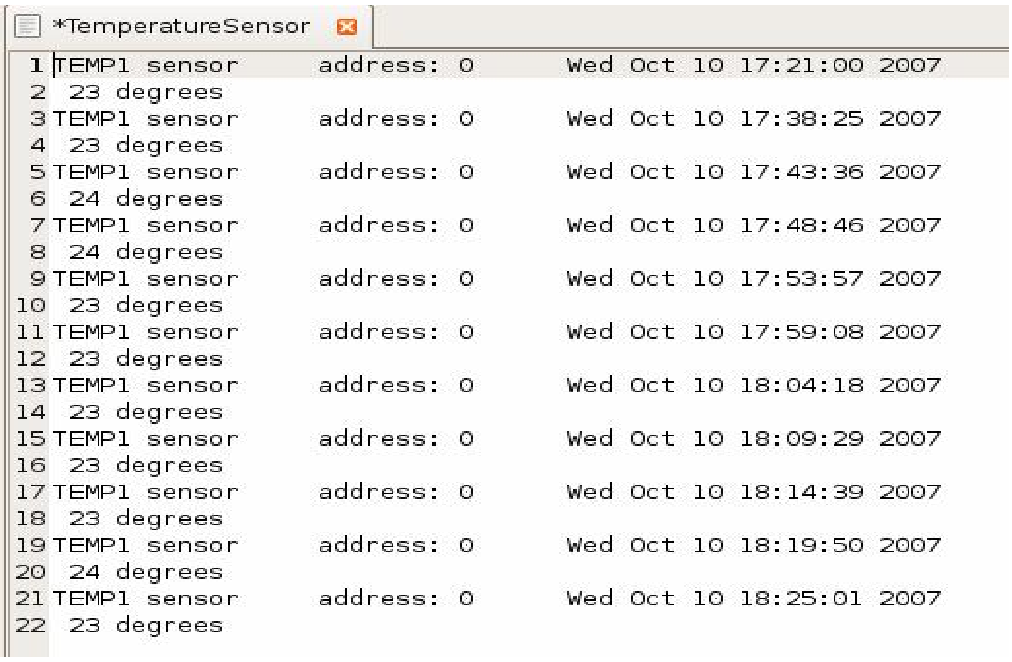


```
 no smoke detected
TEMP1 sensor     address: 0      Wed Oct 10 17:59:08 2007
 23 degrees
TEMP2 sensor     address: 4      Wed Oct 10 17:59:08 2007
 24 degrees
SECURITY sensor  address: 3      Wed Oct 10 17:59:08 2007
 forced entry detected
WATER sensor     address: 1      Wed Oct 10 18:04:18 2007
 no water detected
SMOKE sensor     address: 2      Wed Oct 10 18:04:18 2007
 no smoke detected
TEMP1 sensor     address: 0      Wed Oct 10 18:04:18 2007
 23 degrees
TEMP2 sensor     address: 4      Wed Oct 10 18:04:18 2007
 25 degrees
SECURITY sensor  address: 3      Wed Oct 10 18:04:18 2007
 forced entry detected
WATER sensor     address: 1      Wed Oct 10 18:09:29 2007
 no water detected
SMOKE sensor     address: 2      Wed Oct 10 18:09:29 2007
 no smoke detected
TEMP1 sensor     address: 0      Wed Oct 10 18:09:29 2007
 23 degrees
TEMP2 sensor     address: 4      Wed Oct 10 18:09:29 2007
 24 degrees
SECURITY sensor  address: 3      Wed Oct 10 18:09:29 2007
 forced entry detected
```

Figure 5.9 Part of the Textual database

5.2 Analysis of the Home monitoring application

Analysis of the implemented application is based on the reliability of the resulting artifact, cost and technology. To test the accuracy of the information obtained with the sensors, digital thermometer readings are used as a reference. From the TEMP1 sensor readings in Figure 5.10 the average reading is 23 degrees and the thermometer reading is 23.7 degrees. Since the difference is small, it therefore implies that the sensed temperature values obtained with the application are within the acceptable limits. Other conditions such as smoke and water are also detected appropriately. The application also provided a continuous correct operation for a period of more than 24 hours. From these two characteristics i.e. accurate measurements and continuous correct operation the application exhibits some degree of reliability.



```
TEMP1 sensor      address: 0      Wed Oct 10 17:21:00 2007
 23 degrees
TEMP1 sensor      address: 0      Wed Oct 10 17:38:25 2007
 23 degrees
TEMP1 sensor      address: 0      Wed Oct 10 17:43:36 2007
 24 degrees
TEMP1 sensor      address: 0      Wed Oct 10 17:48:46 2007
 24 degrees
TEMP1 sensor      address: 0      Wed Oct 10 17:53:57 2007
 23 degrees
TEMP1 sensor      address: 0      Wed Oct 10 17:59:08 2007
 23 degrees
TEMP1 sensor      address: 0      Wed Oct 10 18:04:18 2007
 23 degrees
TEMP1 sensor      address: 0      Wed Oct 10 18:09:29 2007
 23 degrees
TEMP1 sensor      address: 0      Wed Oct 10 18:14:39 2007
 23 degrees
TEMP1 sensor      address: 0      Wed Oct 10 18:19:50 2007
 24 degrees
TEMP1 sensor      address: 0      Wed Oct 10 18:25:01 2007
 23 degrees
```

Figure 5.10 Temperature sensor readings

Affordable components have been employed in the implementation of the application with the router being the expensive item that costs about €70. When combining the cost of the other components the total cost adds up to about €120 excluding the cost of labour. When compared with the already existing systems like the Black Box ServSensor that costs between €300 and €450 the application can be said to be affordable.

The application eliminates the need for maintaining a PC for the application since the controlling software can execute on the router. The executable code occupies 25332 bytes and the maximum amount of memory that can be occupied by the textual database (Outputfile.txt) before it is deleted is 80000 bytes. It therefore implies that more memory space is still available in the router for other applications. The minimum time interval within which the application can effectively keep sensor output record for the four sensing elements is after every 1 minute; it is takes an average of about 3 seconds to poll one sensor. It is possible through the telnet client to conveniently access the sensors' outputs from any location.

6 Conclusions

The study has been able to achieve the objectives of employing an embedded operating system to implement an inexpensive home monitoring system with some selected applications. It supports remote and local accessibility making it a convenient application. The application is said to be reliable since it provides outputs that are within acceptable limits for ordinary applications and it is capable of running continuously for a long time without any interference.

Accuracies of the values obtained with sensor using long cable connection for sensed signal transmission such as temperature values can be improved by including a correction factor. The factor can be determined by investigating the average amount of signal variation per unit length. For convenience and efficient operation, the application should be designed to start once the router is powered. Very sensitive conditions that need special attention can be separately monitored by a special function in the software. The study shows that it is possible to utilize the router for other applications other than what it is normally meant for.

The study can be expanded further to include remote controlling of the connected devices. Better means of adding more sensors in a user friendly manner should also be explored to facilitate new sensor additions with minimal or no technical knowledge.

References

[1] Tiiu Koskela, Kaisa Vaananen-Vainio-Mattila, “Evolution towards smart environments: empirical evaluation of three user interfaces” Personal and Ubiquitous Computing, Volume 8, Issue 3-4, Pages 234-240, July 2004

[2] William Green, Diane Gyi, Roy Kalawsky, David Atkins, “Capturing user requirements for an intergrated home environment”, Proceedings of the third Nordic conference on Human-computer interaction NordiCHI ’04, Pages 255-258, ACM Press, October 2004

[3] Kwang Yeol Lee, Jae Weon Choi,”Remote-Controlled Home Automation System via Bluetooth Home Network” SICE Annual Conference Volume 3, Pages 2824-2829, August 2003

[4] Peter M Corcoran, Fedrenc Papai and Arpad Zoldi ,”User Interface Technologies For Home Appliances and Networks” IEEE Transactions on Consumer Electronics, Pages 679-685, August 1998

[5] Yi-Min Wang, “Managing the Aladdin Home Networking System: AN Experience Report”, IEEE Proceedings of the Third Workshop on Software Technologies for Future Embedded and Ubiquitous Systems SEUS, Pages 25-29, May 2005

[6] Ulrich Norbisrath and Christof Mosler, “The eHomeConfigurator Tool Suite”, 1st International Workshop on Pervasive Systems(PerSys 2006), Pages 1315-1324, Montpeller, France, October 2006

[7] A. R Al-Ali and M. Al-Rousan,”Java Based Home Automation System” IEEE Transactions on Consumer Electronics, Vol. 50, No. 2, Pages 499-504, May 2004

[8] Yong-Seok Kim, Hee-Sun Kim and Chang-Goo Lee, “The development of USB Home Control Network System” 8th International Conference on Control,Automation, Robotics and Vision, Kunming, Pages 289-293, China, December 2004

[9] Ikuo Keshi, Yumi Shiraishi, Hiroaki Niwamoto, Minoru Okada, and Heiichi Yamamoto,”Is Home Network Acceptable or Not?” IEEE International Symposium on Circuits and Systems ISCAS 2005, Pages 23-26, May 2005

[10] Zhefa Jiang,Sangok Kim, Kanghee Lee, Hyunchul Bae, and Sangwook Kim, “Security Service Framework for Home Network” Fourth Annual ACIS International Conference on Computer and Information Science, Pages 233-238, 2005

[11] A. Alheraish, “Design and Implementation of Home Automation System” IEEE Transactions on Consumer Electronics, Vol. 50, No. 4, Pages 1087-1092, July 2004

[12] Geon-Woo Kim, Do-Woo Kim, Jun-Beon Hwang, and Jong-Wook Han, “Considerations on Security Model of Home Network” The 8th Conference International on Advanced Communication Technology ICACT, Volume1, Pages 109-112, Korea, February 2006

[13] Nishi R., Morioka H., and Sakurai K., “Trends and Issues for Security of Home-Network Based on Power Line Communication” 19th Conference on Advanced Information Networking and Applications, Volume2, Pages 655-660, 2005

[14] Madjid Merabti, “Networked Appliances in Home Entertainment” Proceedings of the 2006 International Conference on Game Research and Development, ACM International Proceedings Series, Volume 223, Pages 288-293, 2006

[15] Diomidis D Spinellis, “The information furnace: consolidated home control” Personal and Ubiquitous Computing, Volume 7 Issue 1, Pages 53-69, May 2003

[16] Aleksic, Z. J. ”The Analysis of Transmission-Type Optical Smoke Detector Threshold Sensitivity to the High Rate Temperature Variations” IEEE Transactions on Instrumentation and Measurement, Volume 53, Issue 1, Pages 80-85, February 2004

[17] Clyde F. Coombs, Jnr. “Electronic Instrumentation Handbook” Third Edition, Chapter 6, section 6.2 and 6.2.1

[18] William Kleitz, “Digital Electronics a Practical Approach” 4th Edition, Page 552

[19] Alex Lennon, “Embedded Linux”, IEEE Review, Pages 33-37, May 2001

[20] Hojae Hyun, Kwangman Koh, Sunyoung Han, Moon Hae Kim, and Chung-Hyon Chang, “Lightweight Home Network Middleware supporting Mobility Management” Proceedings of International Symposium on Autonomous Decentralized Systems ISAD, Pages 435-442, April 2005

[21] Seongsoo Hong, “Embedded Linux Outlook in the PostPC Industry”, Sixth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing, Pages 37-40, May 2003

[22] Andre Przywara, Rudiger Kusch, and Dietrich Naunin, “Real-Time Operating Systems on Small Embedded Devices for Industrial Control and Communication” The 29th Annual Conference of the Industrial Electronics Society IECON, Volume 3, Pages 2047-2052, November 2003

[23] Edward A. Lee, “What’s Ahead for Embedded Software?” IEEE computer Volume 33, Issue 9, Pages 18-26, September 2000

[24] David Geer, “Survey: Embedded Linux Ahead of the Pack”, IEEE Distributed Systems Online, Volume 5, No 10, Pages 3-3, October 2004

[25] Gabriele Bruzzone, Massimo Caccia, Alessio Bertone and Gianfranco Ravera, “Standard Linux for embedded real-time manufacturing control systems”, 14th Mediterranean Conference on Control and Automation, 2006. MED -06”, Pages 1-6, June 2006

[26] Nakajima Tatsuo, Midori Sugaya, Oikawa S, “Operating System for Buliding Robust Embedded Systems” Proceedings of the 10th IEEE International Workshop on Object –Oriented Real-Time Dependable Systems (WORDS ‘05), Pages 211-218, February 2005

[29] Aleksic, Z. J. “The analysis of the transmission-type smoke detector threshold sensitivity to the high rate temperature variations” IEEE Transactions on Instrumentation and Measurement, Volume 53, Issue 1, Pages 80-85, February 2004 .

[27] OpenWRT Linux Distribution, http://openwrt.org (September 6, 2007)

[28] Qinma Kang, Hong He, Hongrun Wang, “Study on Embedded Web Server and Realization” 2006 1st Symposium on Pervasive Computing and Applications, Pages 675-678, August 2006

[29] Jose’ C. M. Delgado, Renato J. C. Nunes, “An Internet Application for Home Automation” 10th Mediterranean Electrotechnical Conference (MELECON 2000), Volume 1, Pages 298-301, May 2000

[30] Rick Downs, “Using 1-Wire I/O for Distributed System Monitoring” IEE WESCON/98, pages 161-168, 15-17th , September 1998

[31] 1-Wire File System http://www.owfs.org/ (September 6, 2007)

[32] National Semiconductor Corporation, “Precision Temperature Sensors”, datasheet, http://www.farnell.com/datasheets/64315.pdf (September 6, 2007)

[33] Premier Farnell plc 2004, “BC107/BC108 Series, Low Power Transistors”, datasheet, http://www.farnell.com/datasheets/85123.pdf (September 6, 2007)

[34] Analog Devices, “LC^{2}MOS Latchable 4-/8-Channel High Performance Analog Multiplexers”, data sheet, http://www.farnell.com/datasheets/68336.pdf (September 6, 2007)

[35] Texas Instruments, “HEX BUFFERS/DRIVERS with open-collector High-voltage outputs”, datasheet, http://focus.ti.com/lit/ds/symlink/sn7417.pdf (September 6, 2007)

[36] The Konnex Association http://www.knx.org/ (September 6, 2007)

[37] National Semiconductor Corporation, “ADC0801/ADC0802/ADC0803/ ADC0804/ADC0805 8-Bit µP Compatible A/D Converters”, datasheet, http://www.farnell.com/datasheets/55746.pdf (September 6, 2007)

[38] Takeshi Saito, Ichiro Tomoda, Yoshiaki Takabake, Junko Ami and Keiichi Teramoto, “Home Gateway Architecture and its Implementation” IEEE Transactions on Consumer Electronics, Volume 46, Issue 4, Pages 1161 -1166, November 2000

[39] Intersil Corporation, “CMOS Universal Aynchronous Reciever Transmitter (UART) “, datasheet, http://www.intersil.com/data/fn/fn2956.pdf (September 6, 2007)

[40] Philips Semiconductors, “14- Stage binary ripple counter with oscillator”, datasheet, http://www.farnell.com/datasheets/43324.pdf (September 6, 2007)

[41] HC49 Crystals, datasheet, http://www.farnell.com/datasheets/64644.pdf (September 6, 2007)

[42] Sipex Corporation, “Sipex SP3222E/SP3232E” data sheet, http://www.sipex.com/Files/DataSheets/sp3222_3232e.pdf (September 6, 2007)

[43] W. Richard Stevens, Stephen A Rago, “Advanced Programming in the UNIX Environment”, second edition, Pages 631-668

[44] Marc J Rochkind, “Advanced Unix Programming”, second edition, Pages 203-247

[45] Texas Instruments, “SN74LS08 Quadruple 2-Input Positive –And Gates” datasheet, http://www.farnell.com/datasheets/62972.pdf (September 6, 2007)

[46] Texas Instruments, “SN74LS14 HEX SCHMITT TRIGGER INVERTERS”, datasheet, http://focus.ti.com/lit/ds/symlink/sn74ls14.pdf (September 6, 2007)

[47] STMICROELECTRONICS, “L7800 SERIES POSITIVE VOLTAGE REGULATORS”, datasheet, http://www.farnell.com/datasheets/63524.pdf (October 7, 2007)

[48] Philips Semiconductors, “High-speed diodes 1N4148; 1N4448”, datasheet, http://www.nxp.com/acrobat_download/datasheets/1N4148_1N4448_5.pdf (October 7, 2007)

[49] WRTrouters http://www.wrtrouters.com/guides/dualserialport/ (October 8, 2007)

[50] Linksys WRT54GL, http://wiki.openwrt.org/OpenWrtDocs/Hardware/Linksys/WRT54GL (October 8, 2007)

[51] OpenWrt SDK http://downloads.openwrt.org/whiterussian/0.9/ (October 8, 2007)

[52] X-10 Communications Protocol and Power Line Interface, http://www.x10pro.com/pro/pdf/technote.pdf (October 8, 2007)

Appendix

```
/* ====function for addressing and reading sensor outputs==== */
/*============================================================*/

char * readSensor( int fdrs, char *addr ){
      unsigned char buf1[256];
      int ret, i;

       //address the selected sensor twice and read twice
      for(i = 0; i < 2; i++){
            // flush both I/O before any transactions
            tcflush(fdrs, TCIOFLUSH);

             //address the selected sensor
            write(fdrs, addr, strlen(addr));
            memset(buf1, '\0', sizeof(buf1));

            //read output of the addressed sensor
            ret = read(fdrs, buf1, sizeof(buf1));//
            if(ret < 0){
                  perror("Failure to read serial contents\n");
                  exit(1);
                   }
            //wait before repeating the 2nd time
            sleep(1);
            }

      memset(buf2,'\0', sizeof(buf2));
      strcpy(buf2,buf1);

      //return buffer contents to the calling function
      //buf2 is declared as a global variable
      return (buf2);
 }
```