BLUETOOTH BASED HOME AUTOMATION SYSTEM USING CELL PHONE

R.Piyare, M.Tazil
Department of Electrical & Electronics Engineering
Fiji National University
PO Box 3722, Samabula, Suva, Fiji
rajeev.piyare@fnu.ac.fj

ABSTRACT

Technology is a never ending process. To be able to design a product using the current technology that will be beneficial to the lives of others is a huge contribution to the community. This paper presents the design and implementation of a low cost but yet flexible and secure cell phone based home automation system. The design is based on a stand alone Arduino BT board and the home appliances are connected to the input/ output ports of this board via relays. The communication between the cell phone and the Arduino BT board is wireless. This system is designed to be low cost and scalable allowing variety of devices to be controlled with minimum changes to its core. Password protection is being used to only allow authorised users from accessing the appliances at home.

1 INTRODUCTION

Wireless technologies are becoming more popular around the world and the consumers appreciate this wireless lifestyle which gives them relive of the well known "cable chaos" that tends to grow under their desk. Now with the embedded Bluetooth technology, digital devices form a network in which the appliances and devices can communicate with each other. Today, home automation is one of the major applications of Bluetooth technology. Operating over unlicensed, globally available frequency of 2.4GHz, it can link digital devices within a range of 10m to 100m at the speed of up to 3Mbps depending on the Bluetooth device class. With this capability of Bluetooth; we propose a home automation system based on Bluetooth technology [1].

There are few issues involved when designing a home automation system. The system should be scalable so that new devices can easily be integrated into it. It should provide a user- friendly interface on the host side, so that the devices can be easily setup, monitored and controlled. This interface should also provide some diagnostic services so that if there is any problem with the system, it can be tracked down. Moreover the overall system should be fast enough to realize the true power of wireless technology. Finally the system should be cost effective in order to justify its application in home automation.

Neng has presented and architecture for home automation [2] where the system was based on a dedicated network. This system only shows how to solve home automation problems at software level and no hardware aspects were considered. Yavuz and Hasan [3] presented a telephone and PIC based remote control system where pin- check algorithm has also been introduced. Also to remote control of home appliances such as oven, air conditioner and computer by telephones which offer easy usage has been investigated by [4][5]. Communication takes place via a dedicated telephone line not via a Bluetooth technology.

Other studies such as ones presented in [6][7] has examples of web based automation. However, they are not too feasible to be carried out as a low cost solution [8]. Lately Al-Ali and AL-Rousan [9] introduced a low cost Java- Based Home Automation System, without highlighting the low level details of the type of peripherals that can be attached.

Sriskanthan proposed a home automation system that can control home appliances from a PC using Bluetooth [10]. However, the system cannot be controlled by the cell phone.

In this paper we present a low cost secure cell phone based, flexible home automation system. Appliances at home are connected to the Arduino BT board. The communication between the cell phone and the Arduino BT board is wireless. Additional devices can be connected into the system with little modifications. Since the cell phone script is written in Python, it is portable and can run on any Symbian Operating System platform. Figure 1 shows the block diagram of the overall system's architecture.

This paper is organized as follows. In Section 2, the system's general architecture and hardware implementations are discussed. In Section 3 we then describe the system's software development. Finally we conclude our major findings and outline our future work.

978-1-61284-842-6/11/$26.00 ©2011 IEEE

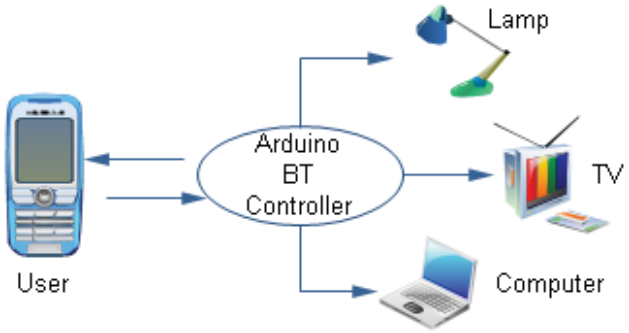

Figure 1. Block diagram of home automation system.

2 HARDWARE ARCHITECTURE AND IMPLEMENTATION

This home automation system consists of two main hardware components: the cell phone and the Arduino BT board. The cell phone hosts the Python script which enables the user to access the home appliances and also the control commands for the appliances. This Python script communicates with the Arduino BT board and sets up an ad-hoc communication protocol between the two devices, which allows controlling the behaviour of the Arduino BT board.

An off-the-shelf ready made Arduino BT is an 8-bit microcontroller board based on the ATmega168 and the Bluegiga WT11 Bluetooth module [11] is used. It supports wireless serial communication over Bluetooth. This board has 23 digital input and output ports, 16kB of flash memory, 10-bit analog to digital converter, pulse width modulator and extra hardware resources which makes it suitable for the required task. The Arduino BT board can be programmed wirelessly over the Bluetooth connection using the microcontroller's high-level interactive C language [11].

The Bluetooth antenna in our module picks up the packets sent from the cell phone. Subsequently, these packets containing the appliance status commands are pipelined through ATmega168 microcontroller and the designed analogue circuitry according to the definition of each output. Different home appliances are connected to the digital output ports of the Arduino BT board via relays to provide sufficiently high currents and voltage compatibility. For test purposes, 25W, 240V lamps have been used. Figure 2 shows the relay configuration for each device and Figure 3 depicts the Arduino BT board's communication with the home appliances.

Sending commands from software to turn ON/OFF a device may not guarantee the successful operation of the device as the device may be defective. To solve this problem, a feedback circuit has been designed and implemented to indicate the device's actual status after it receives the command (ON/OFF) from the cell phone. Once the command has been sent to turn ON a device, the feedback circuit senses the current and gives an output signal by turning ON a respective led on the switching circuitry indicating that the device is ON. Otherwise, the device is malfunctioning indicating that the command was not executed successfully.

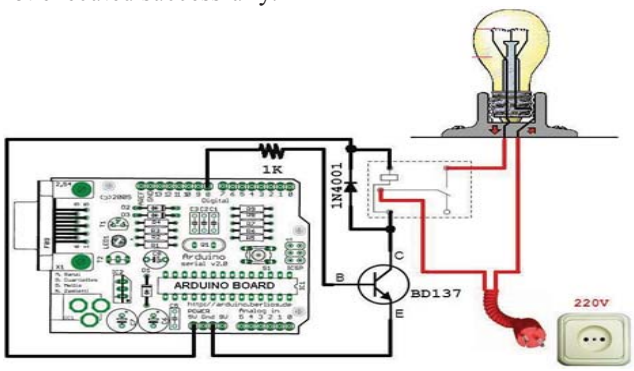

Figure 2. Block diagram of home automation system.

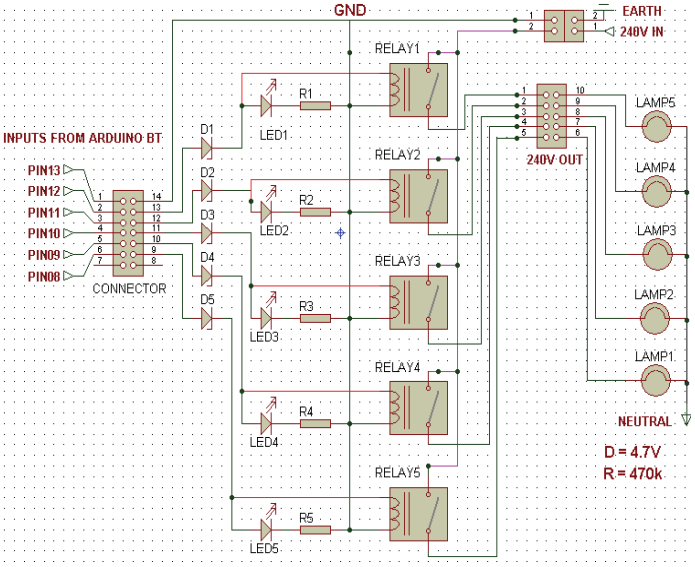

Figure 3. 5V-240V switching circuitry

3 SOFTWARE DEVELOPMENT

Bluetooth-compatible devices perform 'inquiries' to detect and find other Bluetooth enabled devices within the area. When performing an inquiry, an application must wait to about 10 seconds for a 95% chance of detecting every device. Not only does this process take time, it also consumes power. To minimize the need for an inquiry and hence saving time and power, Python allows an application to retrieve a list of devices that would probably be in an area without performing an inquiry.

3.1 The Program Flow chart

Upon the execution of the program, it first checks if Bluetooth is already enabled on the phone. If Bluetooth is enabled, the device and service discovery process will run. The software will check if there are already predefined devices stored in the phone's memory.

If they do exist, they will be listed down for the user to select one. The program then checks to see if the selected device is in range. It will then verify if the device is a Bluetooth transceiver (Arduino BT board). Now if there are no devices stored in memory, the program will search for Bluetooth-enabled devices within the area. Once discovered, these devices will be displayed on the screen and also stored in memory.

Once it is confirmed that the device is indeed a transceiver, the software will store the unique addresses of all the controller modules connected to it, in this case Arduino BT. If the address of a controller module has not been saved, then it will be designated a number i.e. 'BTLAMP'. Otherwise, it will be given its saved name and will prompt the user to enter the pairing password for the Arduino BT board. Upon entering the correct password, the program stores all connected controller modules' names inside the phones' memory, then only the Main Menu user interface will be displayed.

The Main Menu displays three options: 'Options', 'List of Lamps', and 'Exit'. As shown in Figure 4. A 'List of Lamps' is a combination of one or more lights which have been preset to a certain status or state. These states are either ON or OFF. There are two options to choose from in the 'List of Lamps'' interface: they are either LAMP ON or LAMP OFF. When the certain instruction has been chosen, the software will send data to the Arduino BT transceiver, which in turn will send the data to the controller modules.

The 'List of Lamps' option in the Main Menu will display the entire controller modules saved in memory. The user can modify the lights' status from here. 'Options' will display instructions on how to use the software. Lastly, 'Exit' will let the user end the program.

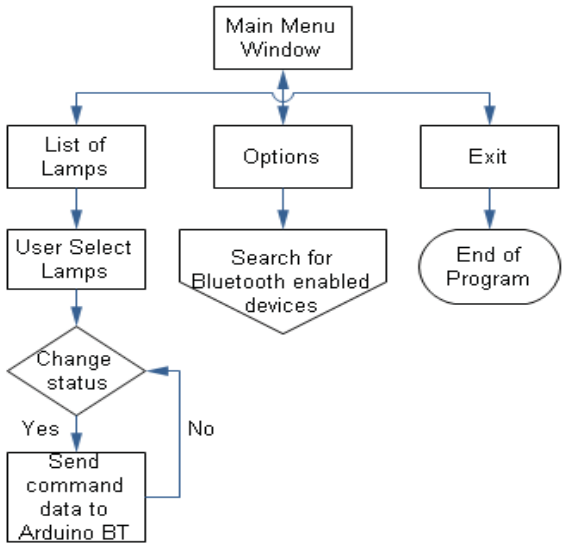

Figure 4. Program Flowchart for Main Menu Window of the GUI.

3.2 Stack Initialization Module

Bluetooth stack is a piece of software or firmware that manages and controls a Bluetooth device. The stack initialization process will involve a number of steps to get the Bluetooth device ready for wireless communication. This means setting several parameters such as serial port name, baud rate, connectable mode and discoverable mode. In this project, the open source software provided with the Arduino BT (Arduino Alpha) was used for the initialization. The code is shown below.

```
Serial.begin (115200);    // connect to the serial port
//configuring the Bluetooth module
Serial.println ("SET BT PAGEMODE 3 2000 1");
Serial.println ("SET BT NAME BTLAMP");
Serial.println ("SET BT ROLE 0 f 7d00");
Serial.println ("SET CONTROL ECHO 0");
Serial.println ("SET BT AUTH * 12345");
Serial.println ("SET CONTROL ESCAPE - 00 1");
```

3.3 Simple Device Discovery

Since sending inquiry signals consumes a lot of power, it is wise to interact with the local device (cell phone) first to retrieve the information of the pre-known and the cached Bluetooth devices. Pre-known devices are the devices that the mobile phone communicates regularly with, while cached devices are the devices that have been found via previous inquiries [12]. This educated guess may mean that no new inquiry signal has to be sent if the remote device's information is already known. As previously stated, this reduces the power consumption and hence saves battery. The following algorithm shows the approach taken to interact with the mobile phone Bluetooth stack and display the list of known devices:

```
def bt_socket_connect(target=''):
    if not target:
        (address, services) = socket.bt_discover()
    if len(services) > 1:
        choices = services.keys()
        choices.sort()
        choice = appuifw.popup_menu(
            [unicode(services[x])+": "+x for x in choices],
u'Choose port:')
        target = (address, services[choices[choice]])
    else:
        target = (address, services.values()[0])
    sock =
socket.socket(socket.AF_BT,socket.SOCK_STREAM)
    sock.connect(target)
    return sock
```

3.4 Communication Module

The Serial Port Profile (SPP) is the Bluetooth profile that realizes the RFCOMM connection between two devices. The RFCOMM protocol is an emulation of the RS-232

serial port connection of two devices over a wireless link. In this system, we have established an RFCOMM connection between the application on the mobile phone and the Arduino BT. Once the connection is established, binary streams can be exchanged between the two devices. After the RFCOMM connection has been made, the client will start sending the binary streams or the appliance status commands. ASCII commands are sent from the cell phone to Arduino BT, which are then converted to binary automatically by the Arduino BT. After the commands have been send from the cell phone, Arduino BT reads in the ASCII values through serial port and compares with the binary equivalent of the these values. Then it turns ON / OFF the respective lamps according to the commands received. Part of the code for reading commands sent from the cell to Arduino BT is given below.

```
void loop ()
{ val = Serial.read();// read the serial port
if (val == 65 ) //reads the value from the mobile phone
{ //if the value is 65(which is A in ASCII) turn the
LAMP1 ON
 digitalWrite(LAMP1, HIGH);//sets the LAMP1 ON
  delay(10);
}
```

3.5 Graphical User Interface (GUI) Module

The most important feature of our application is to hide several processes from the user while allowing some degree of interaction with the application. By using the GUI package, we were able to customize the application to include a variety of user interface elements such as text boxes, choice groups, alert messages, lists and command buttons. Figure 5 illustrates some designs for the graphical user interface.

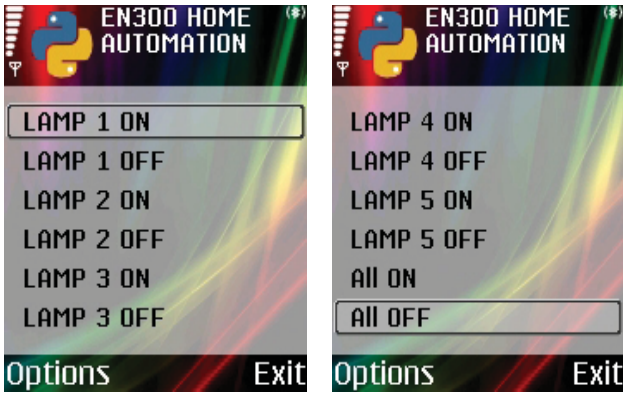

Figure 5. Cell Phone GUI for controlling the home appliances

4 CONCLUSION

In this paper we have introduced design and implementation of a low cost, flexible and wireless solution to the home automation. The system is secured for access from any user or intruder. The users are expected to acquire pairing password for the Arduino BT and the cell phone to access the home appliances. This adds a protection from unauthorized users. This system can be used as a test bed for any appliances that requires on-off switching applications without any internet connection.

The full functionality of the home automation system was tested and the wireless communication between the cell phone and Arduino BT was found to be limited to <50m in a concreted building and maximum of 100m range was reported to be applicable in an open range.

Right now the Symbian OS cell phones only support Python scripts. For future work it is recommended to develop the GUI application for the cell phone to be written in Java so that it can be supported by most of the cell phones available nowadays.

5 REFERENCES

[1] The official Bluetooth website from Bluetooth SIG: http://www.bluetooth.com

[2] Neng- Shiang Liang; Li-Chen Fu; Chao-Lin Wu. "An integrated, flexible, and Internet-based control architecture for home automation system in the internet era". *Proceedings ICRA `02. IEEE International Conference on Robotics and Automation*, Vol. 2, pp.1101-1106, 2002.

[3] E. Yavuz, B. Hasan, I. Serkan and K. Duygu. "Safe and Secure PIC Based Remote Control Application for Intelligent Home". *International Journal of Computer Science and Network Security*, Vol. 7, No. 5, May 2007.

[4] B. Koyuncu. "PC remote control of appliances by using telephone lines". *IEEE Transaction on Consumer Electronics*, Vol. 41, Issue 1, pp.201-209, 1995.

[5] S. Schneider, J. Swanson and Peng-Yung Woo. "Remote telephone control system". *IEEE Transaction on Consumer Electronics*, Vol.43, Issue 2, pp.103-111, 1997.

[6] K.Tan, T.Lee and C.Yee Soh. "Internet-Based Monitoring of Distributed Control Systems-An Undergraduate Experiment". *IEEE Transaction on Education*, Vol. 45, No. 2, May 2002.

[7] N. Swamy, O. Kuljaca and F. Lewis. "Internet-Based Educational Control Systems Lab Using Net-meeting". *IEEE Transaction on Education*, Vol. 45, No. 2, pp.145-151, May 2002.

[8] P. Lin and H. Broberg. "HVAC Applications". *IEEE Industry Applications Magazine*, pp.49-54, January 2002.

[9] A.R.Al-Ali and M. AL-Rousan. "Java-Based Home Automation System". *IEEE Transaction on Consumer Electronics*, Vol.50, No. 2, May 2004.

[10] N. Sriskanthan and Tan Karand. "Bluetooth Based Home Automation System". *Journal of Microprocessors and Microsystems*, Vol. 26, pp.281-289, 2002.

[11] Official Arduino BT website: http://www.arduino.cc/en/Guide/ArduinoBT

[12] Official Nokia forum website: http://library.forum.nokia.com