ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 480 – 484 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

RTOS based Home Automation System using Android

Syed Anwaarullah[1], S.V. Altaf[2]
[1]PG Engineering Student at Lords Institute of Engineering and Technology, India, captanwaar@gmail.com
[2]Assoc Prof at Lords Institute of Engineering and Technology, India, svaltaf@hotmail.com

ABSTRACT

With the openness, flexibility and features that Android offers, it has been widely adopted in applications beyond just SmartPhones. This paper presents the design and implementation of a low cost yet compact and secure Android smart phone based home automation system. This design is based on the popular open sourced Arduino prototyping board where the sensors and electrical appliances are connected to the input/output ports of the board. In order to enhance the system responsiveness and to make it more dynamic, we've integrated a popular and open source RTOS, the scmRTOS, which has a very small footprint on the microcontroller. The controlling application which has been developed for Android devices can also be easily developed on other popular SmartPhone operating systems like Apple's iOS, Microsoft's WP7/8 and BlackBerry OS. Pattern based password protection is implemented to allow only authorized users to control the appliances. Another add-on included is the integration of Google's voice recognition feature that recognizes users' voice commands to control appliances.

Keywords: Android, RTOS, Home Automation, Voice Recognition

1. INTRODUCTION

With the promotion of Android as a Smart Phone Operating System by Google Inc, SmartPhones are becoming more and more popular around the world. Currently, Android has grown to more than 75% of Smart Phones/Tablets user base. This mass adoption of Smart Phones has fuelled a demand for applications both soft and hard. Today, SmartPhones are more than just Phones, they're now the main Human Interaction Devices and users thus want to control/accomplish most of their tasks from their Smart Phones rather than conventional ways. The many wireless protocols that come embedded on a SmartPhone has introduced a wireless lifestyle reliving people from the "wired" cable chaos. With the exception of few low cost tablets, Bluetooth can be found in almost all Android based devices which have been very popular over years for wireless data transmission with ease. Home automation systems are one of the major adopters of Bluetooth technology. Here, we propose a home automation system based on Bluetooth technology [1].

Some of the factors that influence the design of a home automation system include the scalability of the system, the ease of integrating new devices into the system and security. Also important is the ease of use and friendly user controlling interface. A cost effective system would qualify it for mass adoption.

Neng- Shiang Ling has presented an architecture for home automation [2] where the system is based on a dedicated network. This system depicted how to solve home automation problems at the software level and no hardware aspects were included. Yavuz and Hasan [3] presented a telephone and PIC based remote control system. Other studies such as those presented in [4] [5] give examples of web based home automation system.

Another PC based home automation system for appliances control was proposed by Sriskanthan [6]. However, the system cannot be controlled by a mobile/cell phone. R.Piyare proposed a Bluetooth based home automation system using cell phone [7]. However, this is a very basic system without advanced features like integration of RTOS, and also lacks onboard sensors like light and temperature that are used to intelligently control the home appliances without human intervention.

In this paper, we present the design and implementation of a low cost yet compact and secure Android smart phone based home automation system. This design is based on the popular open sourced Arduino prototyping board where the sensors and electrical appliances are connected to the input/output ports of the board. In order to enhance the system responsiveness and to make it more dynamic, we've integrated a popular and open source RTOS, the scmRTOS (Single Chip Microcontroller Real Time Operating System) [8]. Figure 1 shows the block diagram of the overall system's architecture.

This paper is organized as follows. Section II discusses the system's general architecture and also briefs out the hardware implementations. In section III, we list out the software development process. Finally, we conclude our major findings and outline our future work.

ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 480 – 484 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

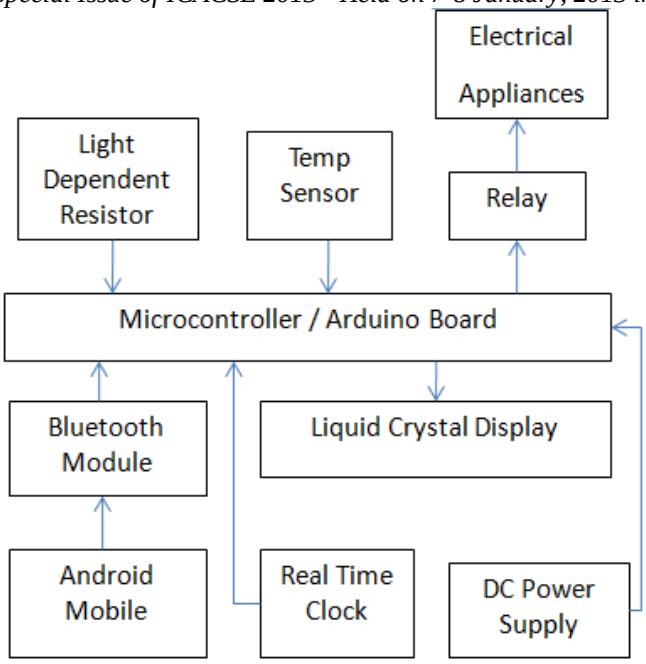

Figure 1: Block diagram of Home Automation System using Android

2. HARDWARE ARCHITECTURE AND IMPLEMENTATION

The key hardware components that make up the home automation system are the Smart Phone, the Microcontroller Board, the Bluetooth module connected to the Microcontroller board and relay boards that drive the electrical appliances. The other components that are also present include a 16x2 char LCD display, a Real Time Clock (DS1307), a Temperature sensor (LM35) and a Light Dependent Resistor.

2.1 Arduino Board

Arduino is an open source computing platform based on a simple input/output board and a development environment that implements the Processing Language. The Arduino board we use in our project is the popular Arduino Uno having an ATMega328P Microcontroller on board that comes with 32KB flash memory and 2KB SRAM. We needed a microcontroller with at least 1KB SRAM as we have to integrate the scmRTOS that requires a minimum of 512 bytes of SRAM. 2KB of the flash memory is consumed by the Arduino Bootloader. The ATMega328P microcontroller has an 8-bit CPU and has 14 Digital I/O pins and 6 Analog I/O pins.

2.2 Nexus One Smartphone

The Nexus One is Google's original flagship Smartphone manufactured by Taiwan's HTC Corporation. It currently runs the Android Gingerbread 2.3.7 version that was modified using the AOSP (Android Open Source Project) by the XDA Developer Community. The phone features Bluetooth 2.1 + EDR, 512MB of RAM and a 1GHz Snapdragon processor. The Bluetooth adapter present in the phone can be programmatically configured to send data commands to the Bluetooth module on the Arduino board that would in turn control the electrical appliances.

2.3 Bluetooth Module

The Bluetooth module allows us to wirelessly transmit and receive data. The module that we're using is based on the Bluetooth V2.0 protocol and is having a range of 10 meters operating at frequency of 2.4GHz with a maximum data exchange rate of 2.1Mbps.

2.4 Relay Board, RTC, LCD and LDR

The relay board that we're using comes with 3 relays that can each handle a load of up to 6A. The relays are driven using a ULN2003 high voltage high current Darlington array IC. The 16x2 char LCD display allows us to see the time (driver by the RTC, DS1307), the temperature (given by the LM35) and also the light intensity (extracted from the Light Dependent Resistor). The light intensity monitoring is used to switch on an emergency light if the intensity of light goes lower than the basic visible level. We've also implemented a mechanism where the entire system can be halted in case of an emergency by manually pressing a push button.

Different home appliances are connected to the digital ports of the Arduino board. We've used three lamps each of 10W running at 230V AC.

3. SOFTWARE DEVELOPMENT

scmRTOS allows us to run multiple tasks in parallel allowing for real time execution and context switching. scmRTOS, The Android application developed in Java inquires for available Bluetooth devices in the vicinity and connects to our module if it's available for sending commands. The Microcontroller on the Arduino is programmed in C++ through the Arduino IDE available freely.

3.1 scmRTOS (Single Chip Microcontroller OS

The real time operating system for Microcontroller, scmRTOS uses preemptive priority process scheduling. This RTOS supports up to 32 processes with each process having its own priority. This RTOS has been designed to be able to run with minimal RAM of 512 bytes. The system consists of three major parts: Kernel, Processes and Inter-process communication services. The kernel provides process management, process scheduling at program and interrupt

International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 480 – 484 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

level, inter-process communications support, system timer and extensions support.

scmRTOS is also available as an Arduino library with no changes to the Arduino core required. To use any library in Arduino, unzip the downloaded file and copy its contents to libraries folder inside the Arduino directory. You can check if the library is working by opening the Arduino IDE and going to Sketch → Import Library option and selecting the newly added library. If there is an example provided in the library, it'll show up under the libraries name in the File → Examples menu.

The RTOS common source code is located in eight files:

- scmRTOS.h – main header file, includes all system header file hierarchy
- OS_Kernel.h – basic definitions, declarations and OS kernel type definitions
- OS_Kernel.cpp – OS kernel declarations and definitions
- scmRTOS_defs.h – auxiliary declarations and macros
- OS_Services.h – intercommunication services fucntions
- userlib.h – auxiliary support library types
- userlib.cpp – auxiliary support library functions

Target source code is located in three files:

- OS_Target.h – target specific declarations and macros
- OS_Target_asm.ext2 – low level assembly code for context switch and OS start support
- OS_Target_cpp.cpp – stack frame prepare function, system timer interrupt service routine, idle process root function

Finally, the project part consists of three header files:

- scmRTOS_config..h – configuration macros and some type aliases
- scmRTOS_target_cfg.h – configuration code for project customization
- scmRTOS_extensions.h – extensions including control

The code for including the scmRTOS library and initializing it is shown below:

```
#include <scmRTOS.h> // Include the scmRTOS library
scmRTOS_PROCESS(0, 256, loop); // define processes
scmRTOS_PROCESS(1, 256, ldr);  // 1st arg is priority
scmRTOS_PROCESS(2, 256, temp); // 0 being lowest
scmRTOS_PROCESS(3, 256, relay); // 2nd arg -- stack size
scmRTOS_PROCESS(4, 50, reset); // 3rd – process name
```

Inside the **setup()** method of Arduino (that's called only once on boot up), we need to start the scmRTOS as: scmRTOS_START();
The processes are accordingly defined in the code. Given below is the sample code for **temp** process that displays the temperature on the LCD:

```
void temp(void)
{
   OS::sleep(10000);
   lcd.setCursor(10,1);
   tempC = analogRead(tempPin);
   tempC = (5.0*tempC*100.0)/1024.0;
   lcd.print(tempC);
   lcd.print('C');
   OS::sleep(5000);
}
```

3.2 Android Application

Android is a software stack for mobile devices that include and operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language. By providing an open development framework, Android offers developers the ability to build extremely rich and innovative applications. Developers have full access to the same framework APIs used by the core applications. Android includes a set of C/C++ libraries used by various components of the Android system. They include System C library, Media library, Surface Manager, LibWebCore, SGL, SQLite, FreeType and 3D libraries.

Android applications are written in Java programming language. The Android SDK compiles the code along with any data and resource files into an Android package, an archive file with an .apk file extension. All the code in a single .apk file is considered to be one application and is the file that Android powered devices use to install the application.

Once installed on a device, each Android application lives in its own security sandbox. Some important application fundamentals are:

- The Android operating system is a multi-user Linux system where each application is a different user
- By default, the system assigns each application a unique user ID. The systems sets permission for all the files in an application so that only the user ID assigned to that application can access them
- Each process has its own virtual machine, so an application's code runs in isolation from other applications
- Every application runs its own Linux process

ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 480 – 484 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

3.3 Bluetooth Development

The Android platform [9] includes support for the Bluetooth network stack, which allows a device to wirelessly exchange data with other Bluetooth devices. The application framework provides access to the Bluetooth functionality through the Android Bluetooth APIs.
These APIs let applications to wirelessly connect to other Bluetooth devices, enabling point-to-point and multipoint wireless features.

Using the Bluetooth APIs, an Android application can perform the following:

- Scan for other Bluetooth devices
- Query the local Bluetooth adapter for paired Bluetooth devices
- Establish RFCOMM channels
- Connect to other devices through service discovery
- Transfer data to and from devices
- Manage multiple connections

The program flow chart that scans, establishes a connection and then sends data to our Bluetooth enabled Arduino board is shown in Figure 2. As shown in the flowchart, we first get hold of the local Bluetooth adapter present in the smartphone and establish a connection with our external Bluetooth module that is connected to the Arduino board. Once the connection is successfully established, we then send command from our smartphone to the Arduino that are then used by the microcontroller to decide which appliances to switch ON/OFF.

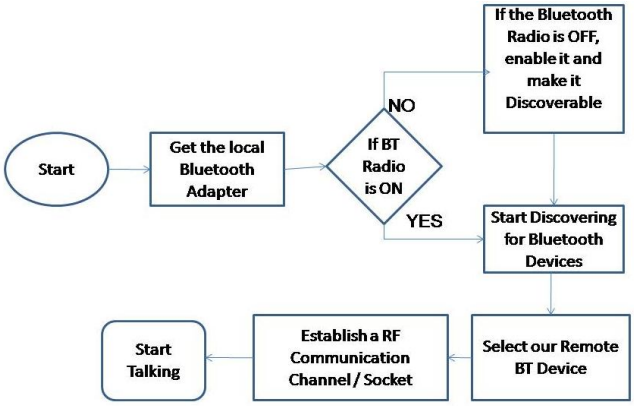

Figure 2: Program flowchart for establishing Bluetooth Connection

3.4 User Application Interface Screens

One of the important features of our system is to simplify the user interaction with our system by hiding all the process in the background while giving the user only the bare essentials. Figure 3 indicates some examples of our graphical user interface.

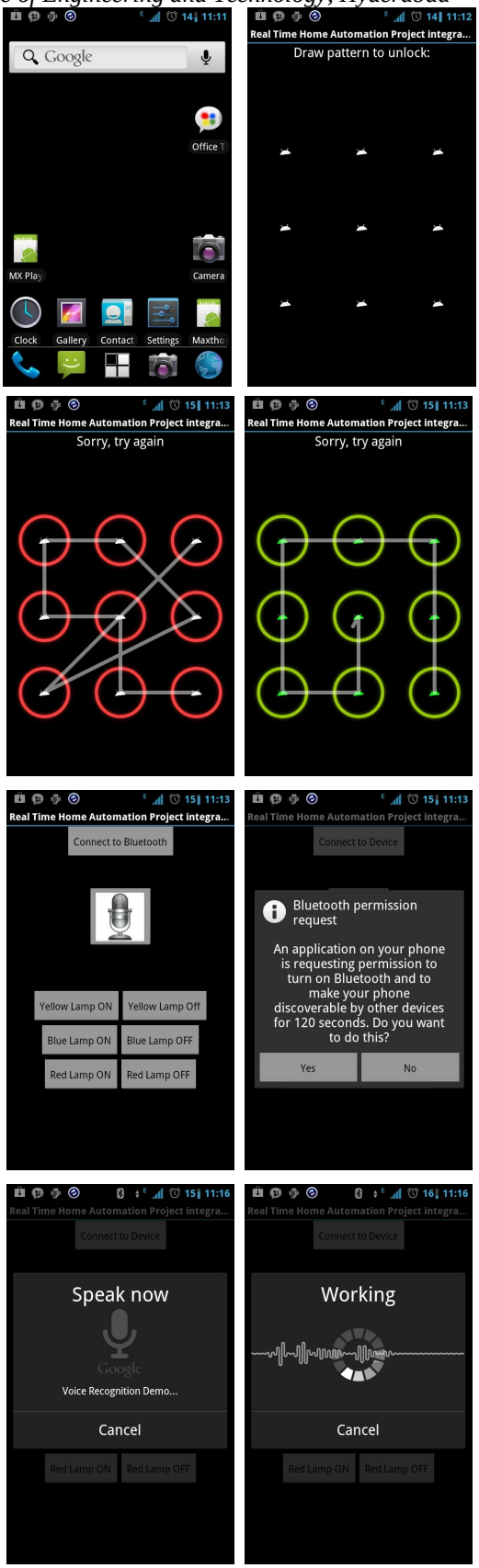

Figure 3: User Interface Screens on Android Application

International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 480 – 484 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

3.5 Final Setup

Figure 4 shows the prototype setup that was made by the authors:

Figure 4: Prototype working design

4. CONCLUSION

In this paper, we've introduced the design and implementation of a low cost smartphone based home automation system. This system can be easily manufactured on a large scale for mass adoption owing to its simplicity and ease of design. Another advantage of is that fact that application software is based on Android, which today has the largest smartphone base. With improvements in technology and the fact that Android is free and open source, cheap SmartPhones (as low as $80) can be used as the controller in our project, making the overall system cost affordable for mass adoption.

Further enhancements can be implemented on the system such as the integration of an intelligent controller that controls the various home devices based on various factors like humidity, temperature etc. Also, we can easily integrate Google's latest offering, Google Cloud Messaging to control our home systems from the Internet, thus making it possible to monitor our home appliances from anywhere in the world.

REFERENCES

1. 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.
2. 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.
3. 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.
4. N. Sriskanthan and Tan Karand. "**Bluetooth Based Home Automation System**". *Journal of Microprocessors and Microsystem*s, Vol. 26, pp.281-289, 2002.
5. R.Piyare, M.Tazil. "**Bluetooth based home automation system using cell phone**". *2011 IEEE 15th International Symposium on Consumer Electronics.*
6. Official page for scmRTOS development (http://scmrtos.sourceforge.net/ScmRTOS)
7. Official Android developer website: http://www.developer.android.com
8. The official Bluetooth website from Bluetooth SIG: http://www.bluetooth.com and http://www.bluetooth.org