IJCAT - International Journal of Computing and Technology
Volume 1, Issue 1, February 2014
www.IJCAT.org

Droid Ei – An Approach to Android Electronic Integration

[1]Abhishek Prabhakar, [2]AnuPaulson , [3]Mohamed Mansoor, [4]Smisha Pious T

[1, 2, 3, 4] Department Of Information Technology, Jyothi Engineering College, Cheruthuruthy, Thrissur, Kerala, India

Abstract - Droid Ei helps developers to remotely control their arduino devices using their smart phones. Droid Ei emulator must be installed in the android device which is connected to a arduino ADK Board. Developers can easily create the remote by adding sufficient buttons using drag and drop facilities. Knowledge in app development is not need. All they needed to burn the code into the arduino board. The main aim of Droid Ei is to boost development of android accesoory using open source electronic prototyping platform. Using Droidei anyone can create device which can able to control your TV, AC, robots or even your car through their smart phone.

***Keywords* - Arduino, Android Accessory, Electronic Prototyping.**

1. Introduction

Droid EI is an IDE for accessory development. An electric device that can be controlled over a smart phone is known as an accessory. Accessory can be controlled through USB Host or Bluetooth. Using droid electronic integration with smart phones made simpler. Beginners and experts can use Droid Ei as a perfect tool for developing and debugging accessories. Droid Ei act as an global remote for any electronic device. Droid Ei is free and open source. The projects in droid ei can be modified by anyone.

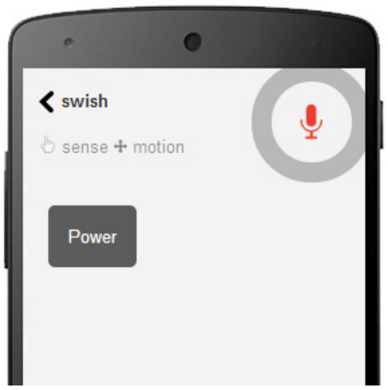

Fig. 1 Interface of Mobile Application

Droid Ei contains an inbuilt compiler which compiles the sketch and sends data to the connected device via bluetooth/USB. Droid Ei also support features like speech, gps, motion, touch to improve the user experience. Droid Ei is created with a simpler IDE which suits with beginners very well. The main of doing this project is to increase the accessory development in India and to boost electronic manufactures to bring droid ei powered device type, with subscripts and superscripts in a slightly smaller font size. This is acceptable.

2. Technique Used

2.1 Bluetooth Connectivity

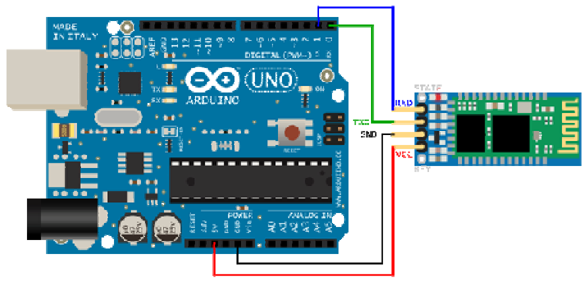

Fig. 2 Connecting a Bluetooth Shield with Arduino Board

The default pass key for most Bluetooth devices is 1234. If the pairing is successful, you will see a message.

3. Proposed System

Droid Ei is a platform for hardware manufacturers and hobbyists to use as a IDE for building accessories for Android. An accessory needs android app to communicate between the mobile and the device. The electronic device can be connected to the mobile via USB or Bluetooth. The device consists of a single board microcontroller like Arduino or Seeeduino. The board will receive data from mobile and process the circuit.

In electronics, prototyping means building an actual circuit to a theoretical design to verify that it works, and to provide a physical platform for debugging it if it does not. The prototype is often constructed using techniques such as wire wrap or using veroboard or breadboard, that create an electrically correct circuit, but one that is not physically identical to the final product. Open source tools exist to document electronic prototypes (especially the breadboard-based ones) and move forward toward production such as Fritzing and Arduino. A technician can build a prototype (and make additions and modifications) much more quickly with these techniques however it is much faster and usually cheaper to mass-produce custom printed circuit boards than these other kinds of prototype boards.

This is for the same reasons that writing a poem is fastest by hand for one or two, but faster by printing press if you need several thousand copies. The proliferation of quick-turn pcb fab companies and quick-turn PCB assembly houses has enabled the concepts of rapid prototyping to be applied to electronic circuit design. It is now possible, even with the smallest passive 8 components and largest fine-pitch packages, to have boards fabled and parts assembled in a matter of days.

The Android 3.1 platform (also back ported to Android 2.3.4) introduces Android Open Accessory support, which allows external USB hardware (an Android USB accessory) to interact with an Android-powered device in a special "accessory" mode. When an Android-powered device is in accessory mode, the connected accessory acts as the USB host (powers the bus and enumerates devices) and the Android-powered device acts as the USB device. Android USB accessories are specifically designed to attach to Android-powered devices and adhere to a simple protocol (Android accessory protocol) that allows them to detect Android-powered devices that support accessory mode. A single-board microcontroller is a microcontroller built onto a single printed circuit board. This board provides all of the circuitry necessary for a useful control task: microprocessor, I/O circuits, clock generator, RAM, stored program memory and any support ICs necessary. The intention is that the board is immediately useful to an application developer, without needing to spend time and effort in developing the controller hardware.

4. Literature Survey

4.1 Amarino-‘Arduino meets Android [1]

Normally Smartphone events are tightly coupled to your phone device itself. When your cell phone is ringing, your phone speaker plays a ringtone. When you get a new text message, your phone displays it on its screen. Wouldn't it be thrilling to make those phone events visible somewhere else, on your wearable, in your living room, on your robot, in your office or where ever you want it to occur or would you like to use your Smartphone sensors, like the accelerometer, light sensor, compass or your touch screen to control other devices. 'Android meets Arduino' is a toolkit, basically consisting of an Android application and an Arduino library which will help you to interface with your phone in a new dimension. Your Android phone can even be made to sense external factors like temperature, light intensity remotely. My paper basically deals with the concept of Arduino interfacing with android and its diverse applications in various fields. Arduino is an open source electronics prototyping platform (software and hardware) designed to below-cost and easy to learn. Arduino is well documented and there is a great community supporting it. Our paper provides detailed explanation about the data transfer and communication between android mobile and Arduino. Our paper also gives step by step explanation of our project of MULTICOLOURED LAMP CONTROLUSING AMARINO (prototype) using Bluetooth. Practically using AMARINO we can control even a small flying plane from our Android mobile and hence it’s a great device to be explored more.

The capabilities of smart phones are increasing day by day with loads of new features. But all such features are limited within the small box. Only a very few innovative techniques are available for smart phones to gain access and control external interfaces like hardware devices.

“Even the most powerful smart phones, with access to worldwide information network, still focus attention on a single box“- Mark Weiser.

For the smart phones to control a hardware device or to act as a heat or light sensor we need an external circuitry that communicates between the phone and the client device (like LED’s in our prototype). There are 3 frequently used i/o boards for these purpose Arduino as a design platform for a course on embedded systems and ask the question, is the Arduino platform suitable for teaching computer engineers and computer scientists an embedded system course with to examine this question, we describe a project based learning embedded system course that we have taught and identify which topics are covered in it compared to the IEEE/ACM recommendations. The major contention lies in the idea that students can access and use an open source community that is focused on getting things working as opposed to strictly looking at low-level technical aspects of embedded systems. Additionally, the presence of open source and reusable designs makes it difficult to identify what a student is doing. In our experience, using the Arduino exposes students to sufficient complexity and challenges for In a recent article at Make online, titled, “Why the Arduino Won and Why It’s Here to Stay” [1], the author describes the world of microcontroller development kits and how the Arduino [2] has captured the hearts of many non engineers. The question we pose in this paper is, should the Arduino and related open source projects be used as a address this question we are try to establish which concepts/outcomes do we expect from a undergraduate level course in embedded systems. We then describe a course we have taught on embedded systems using the Arduino Uno, and discuss how the course does or does not satisfy these various topics. We and that the device as a platform, though not perfect, has many benefits that help students build devices that would not, likely, be possible with other control platforms. This is mainly due to the Arduino community, which consists of not only traditional engineers and scientists, but has a large contingency of artists and hobbyists. The size of this community, the basic desire for users to get something working, and the open sharing of designs means students have access to a huge base of knowledge, that they can leverage to build their systems. We compare this with our experience in the previous year.

4.2 Arduino for Teaching Embedded Systems [2]

Creating models Project Based Learning (PBL) curricula (which is a version of Problem Based Learning) is becoming the norm for many engineering fields, business, and medicine to help deal with the above identified topics in preparing students for the real-world. PBL pedagogy centres learning around the activity of the student. An approach to preparing students to become industrial designers is to include design projects throughout the curriculum, hence PBL curriculum. The accreditation agency, ABET, among other entities, influenced engineering programs into including a major capstone around 1995 to 1997. For computer engineering curriculum, lab only courses slowly evolved to include both labs and final projects. The senior capstone has been studied to help understand how to prepare students for this culminating experience.

Embedded system courses it well into PBL curriculum. The question still remains, what are the specific topics that should be covered in such a course. The IEEE/ACM model computer engineering curriculum has an embedded system portion that consists of 11 units (7 core and 4 electives) with a total of 59 topics and 39 learning outcomes. The model does provide timelines, but a simple mathematical calculation leaves approximately 50 minutes per topic over a one-term 3 credit hour course to cover topics as complex as “DMA transfers” and “Memory system power consumption”. In other words, an embedded system is a very large subject matter for a single course. The reality is a graduate degree in embedded systems as describe in ARTIST in 2003] will, likely, cover all the topics in IEEE/ACM model over a number of courses, but the average computer engineer undergraduate will either need to extend their embedded system skills when in industry or they will never be involved in the field. Other scholars have proposed course curriculum for embedded systems and embedded programming. In all cases, the goal is to teach a common set of topics and to allow the student to interact with devices in the lab at both the software and hardware level. An embedded system course deals with the design and analysis of the software and hardware for a dedicated application. In this review, we have identified that there is no common set of topics to cover for such a course, and there are a number of approaches to teach students a subset of these concepts at various depth of coverage.

It’s intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments. The basic system consists of a microcontroller with various peripheral interfaces that is programmed by an existing software platform. This may sound vague, but the Arduino can come in a number of form factors. For example, the Arduino Uno consists of an ATmega328 microprocessor, a USB to serial chip, and an AC to DC power converter. The Uno can either be built by hand or can be bought premade from a seller such as sparkfun.com for approximately 35 USD. The Arduino software platform is written in Java and is based, mainly, on Processing (a language developed mainly for artists). The IDE is installed on a machine and then can program the UNO over the USB.

The base IDE includes a number of examples for blinking LEDs, making noises, etc. The Uno is only one type of Arduino kit and others exist such as the Nano (for compact use), the LilyPad (for wearable applications), and Fio (for wireless communication). There is not much literature relative to Arduinos being used in computer science and computer engineering to our knowledge with the exception of Buechley have described their experiences in building the LilyPad Arduino and using the device in workshops for K-12 based students. They noted how the device was successful in at-tracing females to participate in programming and hardware design.

Balogh has described their Arduino based robot platform that was used in some lectures in robotic control and embedded systems in their curriculum. In their work, they noted the increase in Arduino searches on the Internet as a key factor in deciding to use the Arduino. Taking a similar approach, Figure 1 shows the search trends for the terms arduino, x86, hc11, mips, and nios. What is remarkable is that Arduino searches are now roughly on par with x86 searches as of March 2011. This just illustrates the activity within the Arduino community. The major benefits for using Arduino in an educational setting that we have identified are:

- Ease of setup - plug and play.
- Many open source projects to look at.
- Works on Windows, Linux, and Mac.
- Low cost hardware - build or purchase prebuilt.
- Low cost software – free.
- Low maintenance cost – Destroyed microprocessors can be replaced for approximately 4 USD.
- Students can prototype quickly.
- Can be programmed in an a number of languages including C.

4.3 Using Android Mobile as a Universal Remote Control [3]

The infrared remote control has the performance of the high signal to noise ratio, strong anti-interference, reliable transmission of information, and untouchable, low power and cost, therefore it is widely used in home appliances more or less, even as the highlight on sale. In addition to utilizing in home appliances, it has been brought into the industrial control, the aerospace, the security and so on. But the formats of infrared remote control protocol used are different between the different companies' production, the consequence from this is that an infrared remote control device must be fit for the home appliances. Because of that, the paper presents a system for decoding based on the infrared coding format in now infrared remote control devices [1], which can replace the existing infrared remote control devices and control more home appliances remotely. The decoded codes are written in a file called as a config file. This config file is unique in the sense that it is different for different manufacturer as well as different products from the same manufacturer. IR remotes operate by modulating an infra red (IR) light source. When the IR light source is "on" it is actually turning itself on and off thousands of times per second, too fast for the human eye to follow. The rate at which this occurs is called the carrier frequency. This is done to provide a better transmission system and allow the overall IR system (transmitter and receiver) to operate in noisy (with respect to light) environments. It is important to understand that the IR receiver for a given remote is tuned to IR "carrier" frequency for that remote and will effectively not see IR signals sent on a different carrier frequency such as from other remotes. The human eye can never see an infrared transmission, so the concept of on and off is not with regards to visible light. Some equipment has a "telltale", a little red light that visibly flashes when the equipment receives IR signals. That is what we can see. The "information" is placed on the "carrier" using several different techniques.

The most common technique is Pulse Width Modulation. In Pulse Width Modulation the duration of the ON (carrier present, light flashing thousands of times per second), or off (no light at all coming out of the IR emitter) periods is made to vary. We first need to simplify the problem so that we don’t have deal with too many "Pulse widths". We can easily do this by representing the number in base 2, or binary. Therefore we only need to have two distinct pulse each manufacture decides the length of bit patterns he wishes to send to his equipment, and the meaning given to that number (or numbers) when they are received. The environment through which the IR signals are passing (the air) is noisy in a light sense. Bright sunlight, Fluorescent lights, all contributes to the noise.

Some manufacturers add additional "redundant" information such as sending the numbers twice to ensure that they get to the equipment correctly. Infrared Codes uses two modulation techniques: Pulse Width Modulation and Phase Modulation. II. IRCP III. DESIGN In order to control the home appliances one has to send the same Infrared code that is being sent through the remote. Our approach is to decode these codes and save them in a config file. Once the config file is prepared, this can be saved in the mobile’s SDRAM memory and can be internally called through the connection manager when the device is to be controlled. 89 Using Android Mobile as a Universal Remote Control A. Decoding Infrared Waves In order to decode the IR codes of household devices we used an Universal Remote Control, URC22B.

15 or one can use remotes of the device he is having. The Infrared codes are decoded with the help of Iguana’s Serial IR Transceiver which is connected to the PC serial port and Linux Infrared Remote Control (LIRC) software and are saved in a config file in .txt format. The Iguana’s Serial IR transceiver device connected to serial

port and the snapshot of the method to decode the infrared code respectively.

4.4 Reliable Real-Time Applications on Android OS [4]

The Android operating system (OS) is widely used within several types of embedded & mobile platforms, including mobile phones and tablets, and the industry is exploring the ability of Android within other embedded platforms, i.e., automotive or military, that require real-time guarantees and the ability to meet deadlines as a pre-requisite for reliable operation. In this paper, we present preliminary conclusions on Android's real-time behaviour based on experimental measurements performed on a commercially available Android platform. Index Terms – Android OS, Real time Software, OMAP Dalvik VM makes full use of Linux for memory management and multi-threading, which is intrinsic in the Java language. The Application Framework provides many higher-level services to applications in the form of Java classes. This will vary in its facilities from one implementation to another. Studies on the reliability of software focus on functional failures, and do not emphasize the time-related behaviour of systems that can also cause the software to fail. The ability to meet deadlines and time constraints is critical to embedded systems software (as in automotive or robotic applications) that mandate response to stimuli within prespecified real-time design specifications, and reliability considerations require a detailed evaluation of the ability of the system to meet these specifications.

The Android OS is an operating system primarily designed for mobile platforms by Google. It is an open source OS based on LINUX kernel (version 2.6) that enables developers to write applications primarily in Java with support for C/C++ as well [4]. Android is finding widespread acceptance in the mobile and portable computing market, and this study examines, for the first time, its performance & reliability in more demanding embedded real-time applications. An Android architecture an Android system is a stack of software components. At the bottom of the stack is Linux (kernel version 2.6). This provides basic system functionality like process and memory management and security. Also, the kernel handles all the things such as network interface and a vast array of device drivers, which make it easy to interface to peripheral hardware. On top of Linux is a set of libraries, including bionic (the Google libc), media support for audio and video, graphics (OpenGL ES), support for browsers (Webkit), and a lightweight database, SQLite [4]. A key component of an Android system is the runtime engine – the Dalvik Virtual Machine (VM). It was designed specifically for Android and is optimized in two ways. It is designed to be instantiated multiple times – each application has its own private copy running in a Linux process. Android OS Software Architecture [4]. Android OS in Real-Time Embedded Applications we use the automotive application as an example of the type of reliable embedded software applications that are being investigated in the context of the use of Android.

In the typical automotive application, there are different services (Control Class: drive control, braking; Safety Class: seatbelts, airbags; Infotainment Class: multimedia, climate control, communication services, etc.), that usually provide their own user interfaces. This might overwhelm and distract the typical driver restricting the user from exploiting the full capabilities of these devices. With all these features bundled together on a single platform, the unpredictability in response time of these simultaneously executing and interacting applications may cause the software to fail, resulting in unreliable operation. For instance, if the driver were using his GPS navigation while driving, and a higher priority phone call is received causing the GPS application to be de-scheduled for a long time, the GPS application might miss out on updating the driver on some turn that he should have taken, or if the time to respond to a phone call were too long, the call would be missed. Additional safety considerations come into play if the navigation system or the braking systems were also controlled by the Android OS, in the near future.

5. Conclusions

Droid Ei is the world first Android Accessory Development Environment which make possible to integrate any electronic device with an Android mobile. Developers can easily create their UI using the drag and Drop feature. Droid Ei also support voice, gps, and motion control. Droid Ei also supported in cross mobile platform.

References

[1] R. BalaVishnu, P. Gowtham Raj,"Amarino-Arduino meets Android", Kongu Engineering College-Perundurai, Amritha University – Coimbatore.

[2] Peter Jamieson, "Arduino for Teaching Embedded Systems".Miami University, Oxford, OH, 45056.

[3] Gaurav Chitranshi, Madhvi Gaur, " Using Android Mobile as a Universal Remote Control" Dept. of Electronics and Communication Engineering Amity University Noida, India.

[4] Bhupinder S. Mongia, “Reliable Real-Time Applications on Android” Electrical and Computer Engineering Electrical and Computer Engineering,Georgia Tech Georgia Tech , Atlanta.

[5] J.R. Barry,Wireless Infrared Communications. Boston: Kluwer, 1994.

[6] RFID: The state of Radio Frequency Identification (RFID), 2005. White Paper IEEE - USA White Paper.

[7] US Department of Commerce, 2005. Radio Frequency Identification: Opportunities and challenges in Implementation.

[8] M. Gaber et al., “An Analytical Study of Central and In-Network Data Processing For Wireless Sensor Networks,”in Information Processing Letters, Vol. 110, Iss. 2,10.1016/j.ipl.2009.10.008